Capitolo 32

Nozioni elementari sulle reti

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

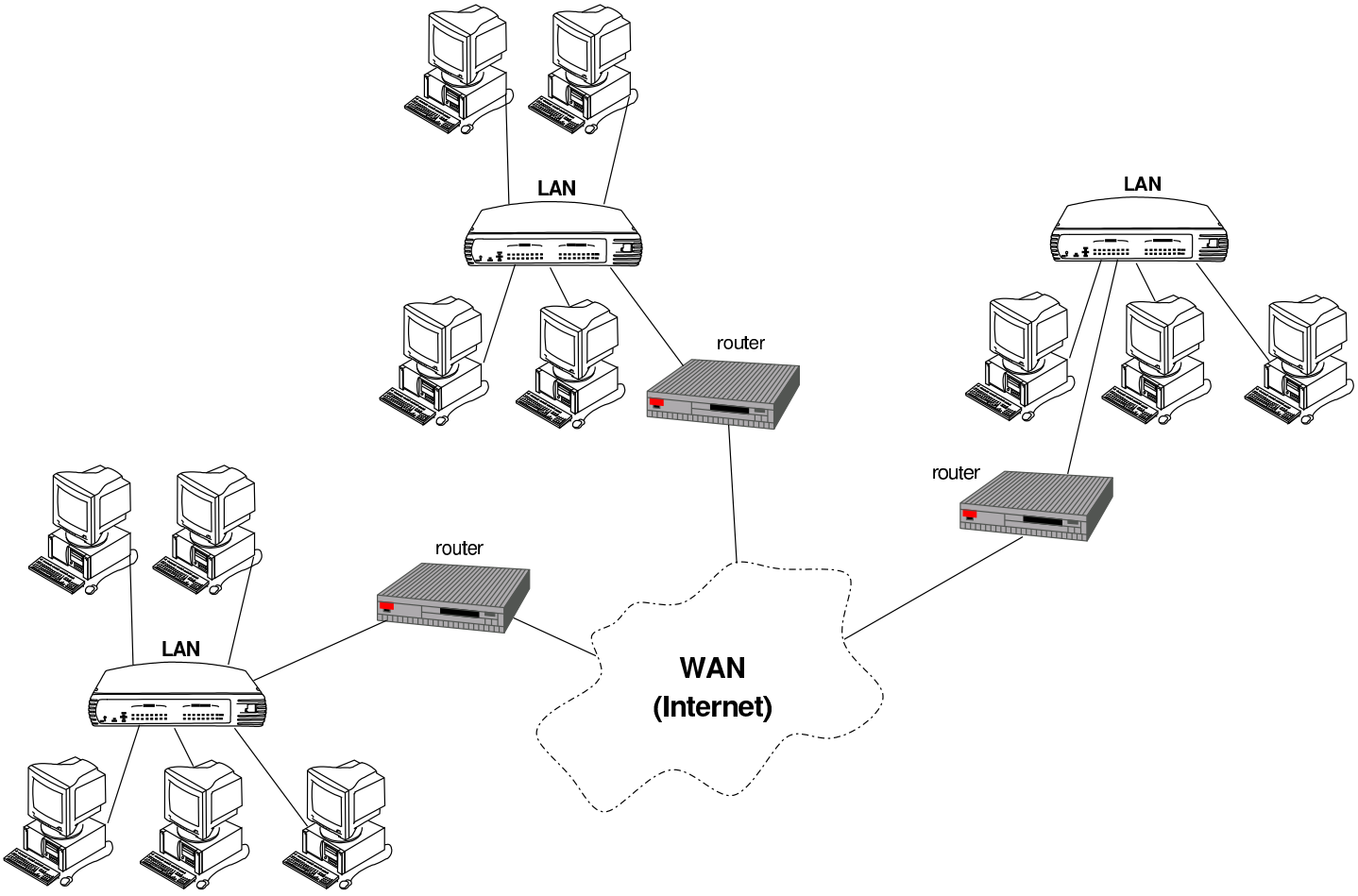
«

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.



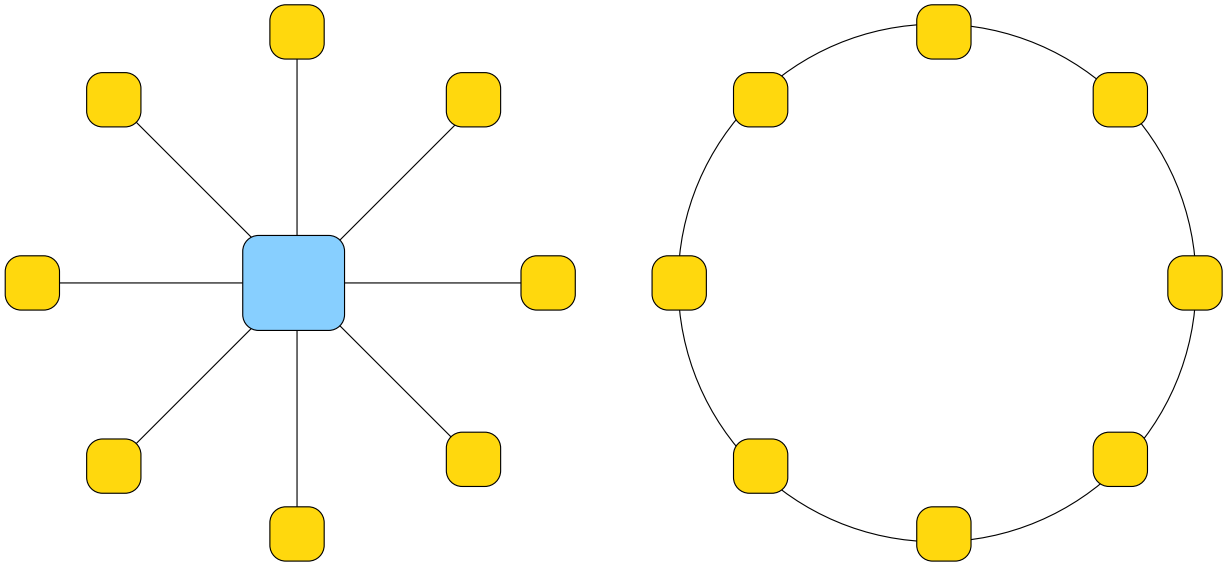
32.1.2 Topologia

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 col-

legamento ***punto-punto***, o *point-to-point*, in cui sono collegati solo due nodi di rete.

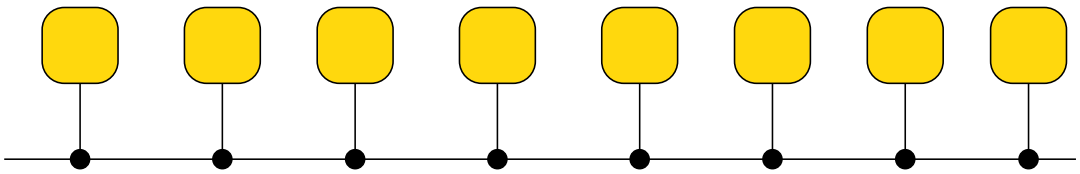
Figura 32.2. A sinistra, topologia a stella; a destra, topologia ad anello.



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.

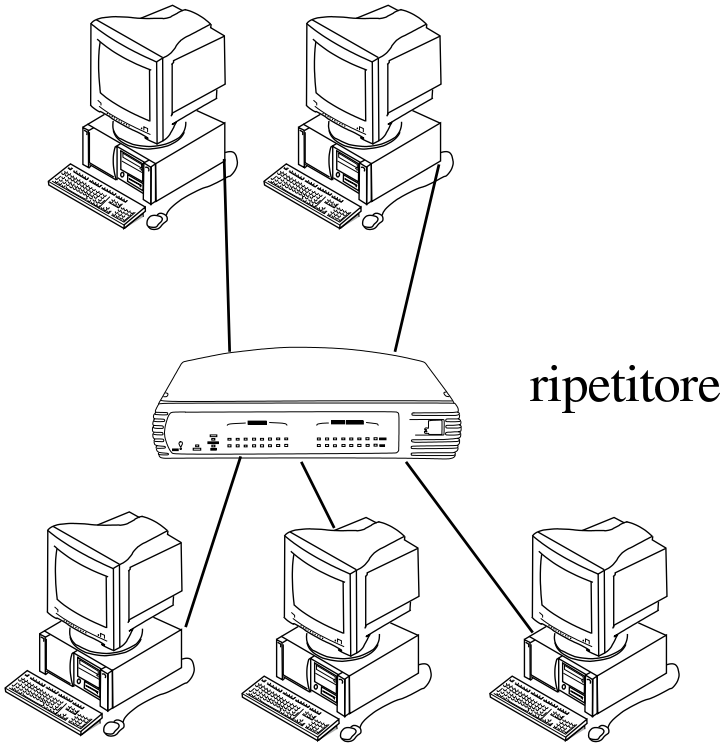
Figura 32.3. Topologia a bus.



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

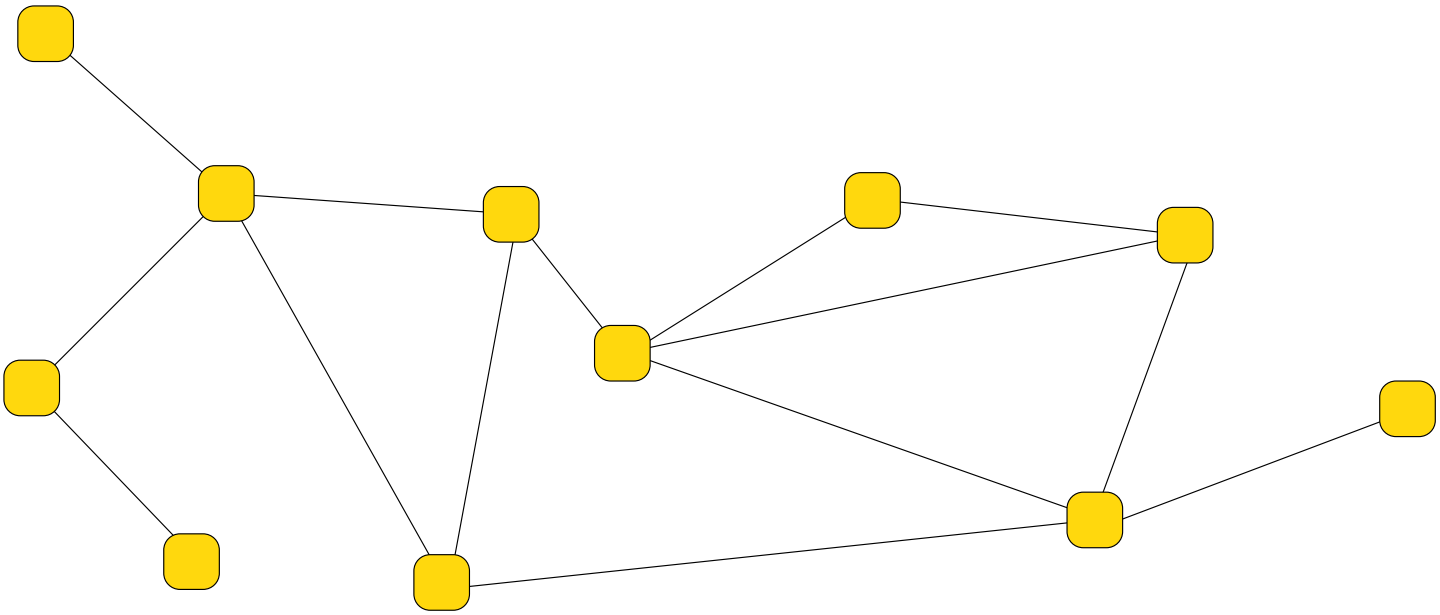
dell'amministratore di rete.

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.



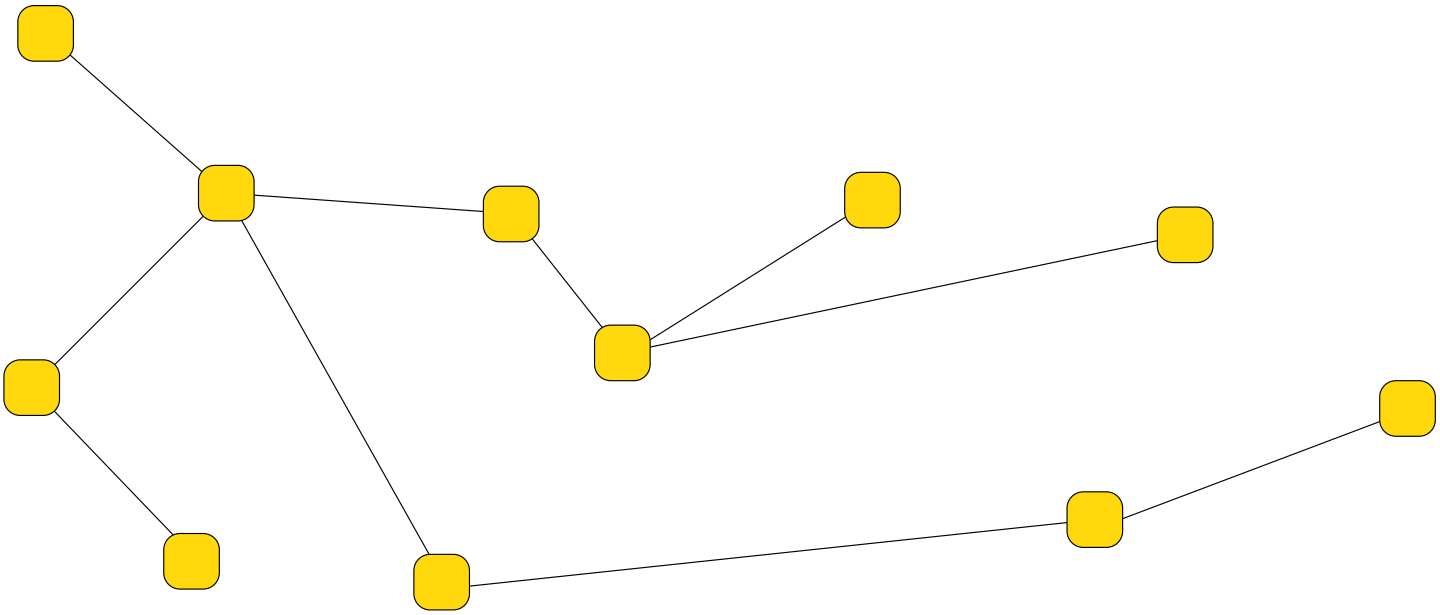
I tre tipi elementari di topologia si possono integrare tra di loro in strutture più complesse; in particolare, quando ci possono essere più percorsi alternativi per raggiungere un certo nodo, si ha normalmente una rete a maglia.

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

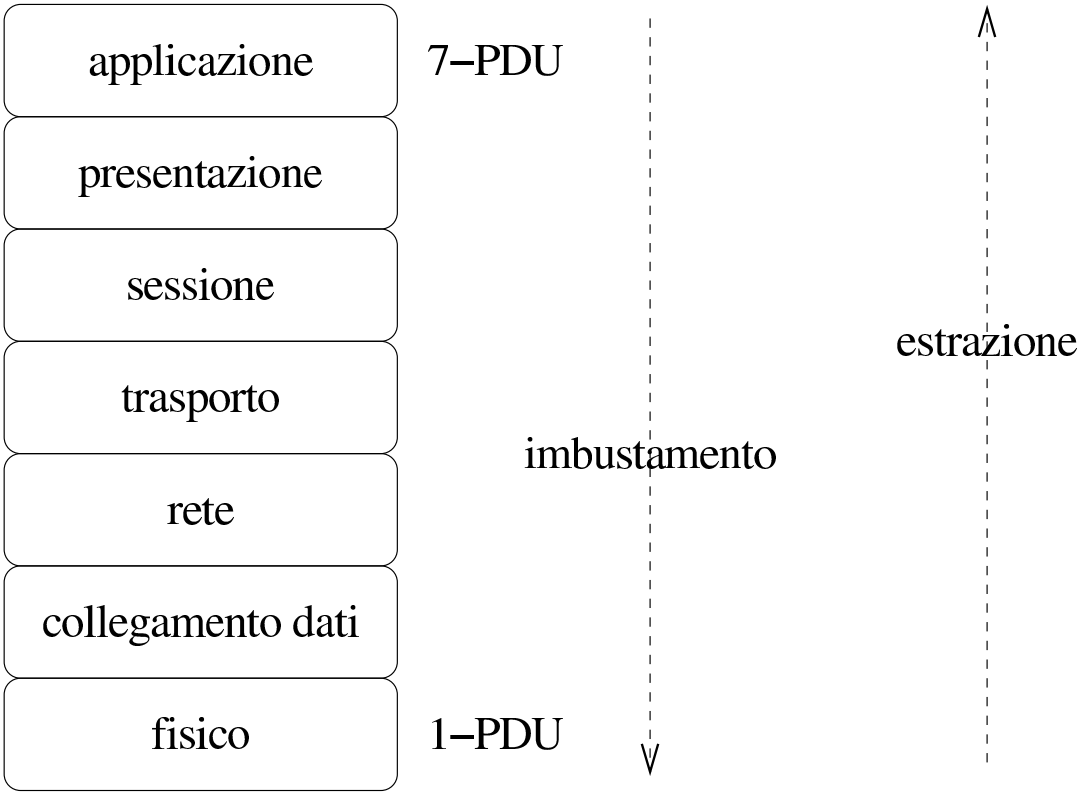
«

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 preferi-

bilmente 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.



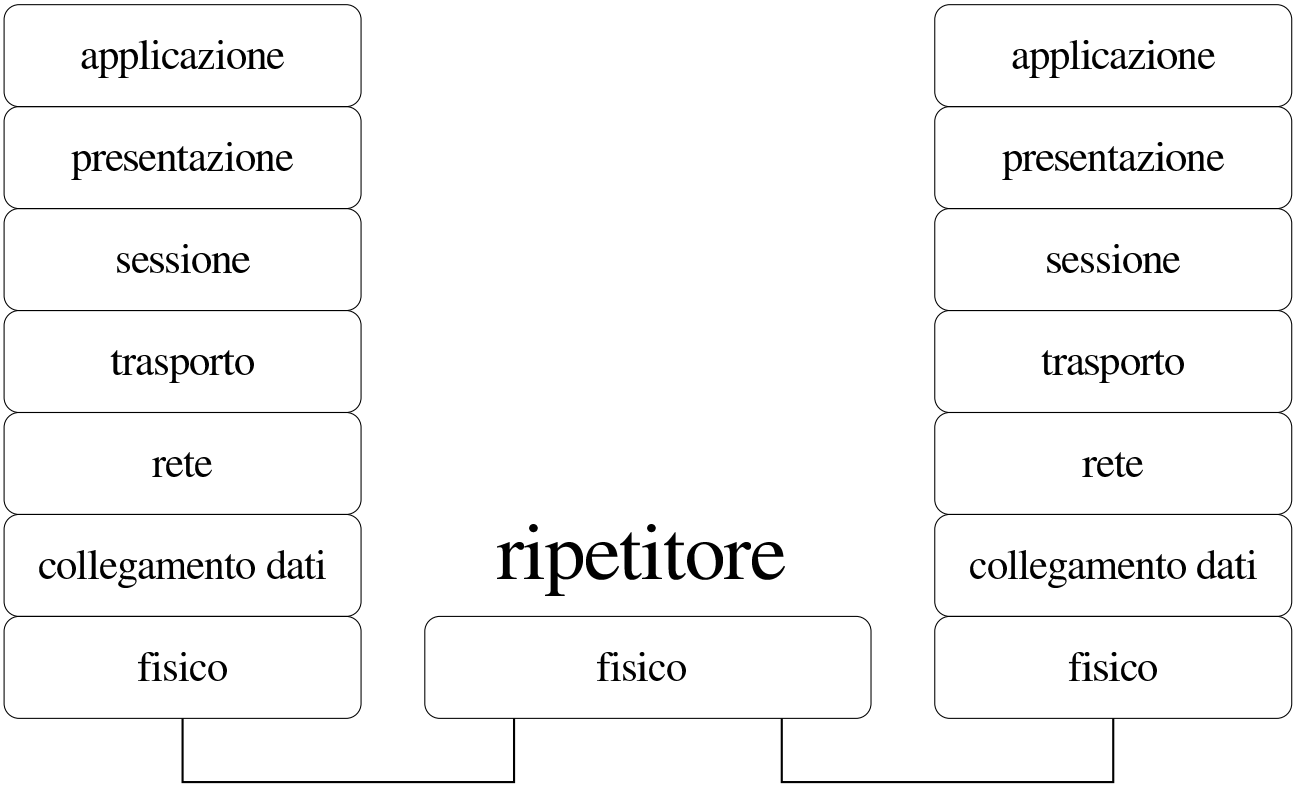
Nel passaggio da un livello a quello inferiore, l'imbustamento implica un aumento delle dimensioni del pacchetto, ovvero del PDU.[1] Ma oltre a questo, a certi livelli, può essere introdotta la frammentazione e la ricomposizione dei pacchetti, a seconda delle esigenze di questi.

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

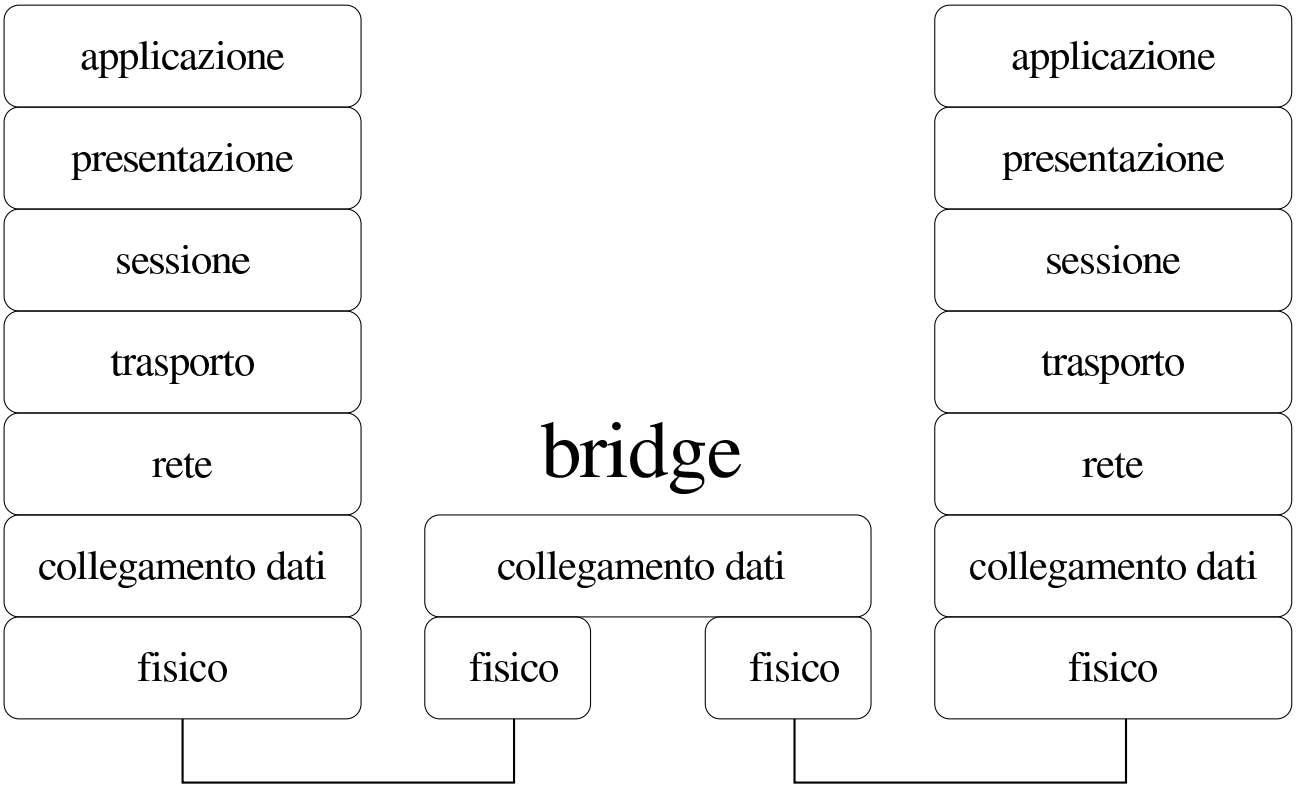
è in grado di determinare gli indirizzi dei nodi connessi nelle varie reti, in modo da trasferire solo i pacchetti che necessitano questo attraversamento.

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.

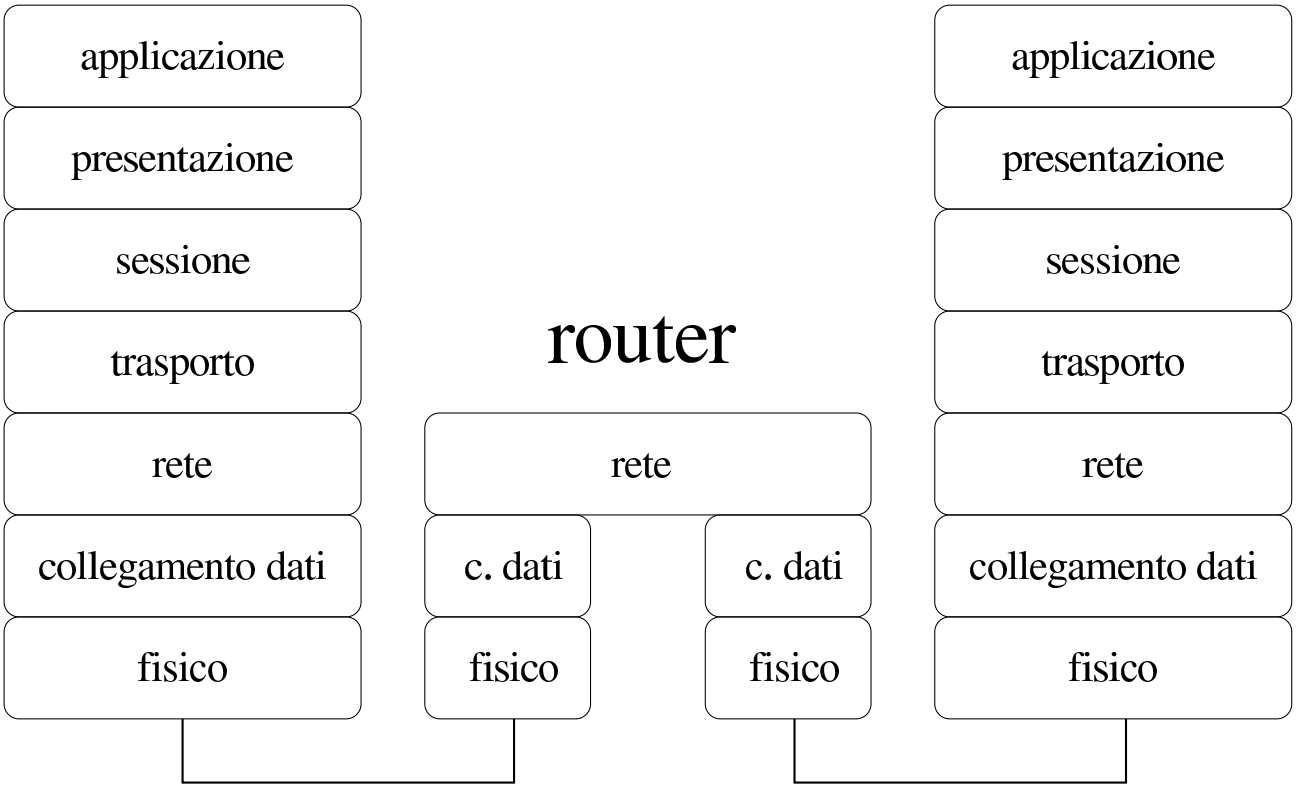
Figura 32.10. Il bridge trasferisce PDU di secondo livello; in pratica trasferisce tutti i tipi di pacchetto riferiti al tipo di rete fisica a cui è connesso.



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.



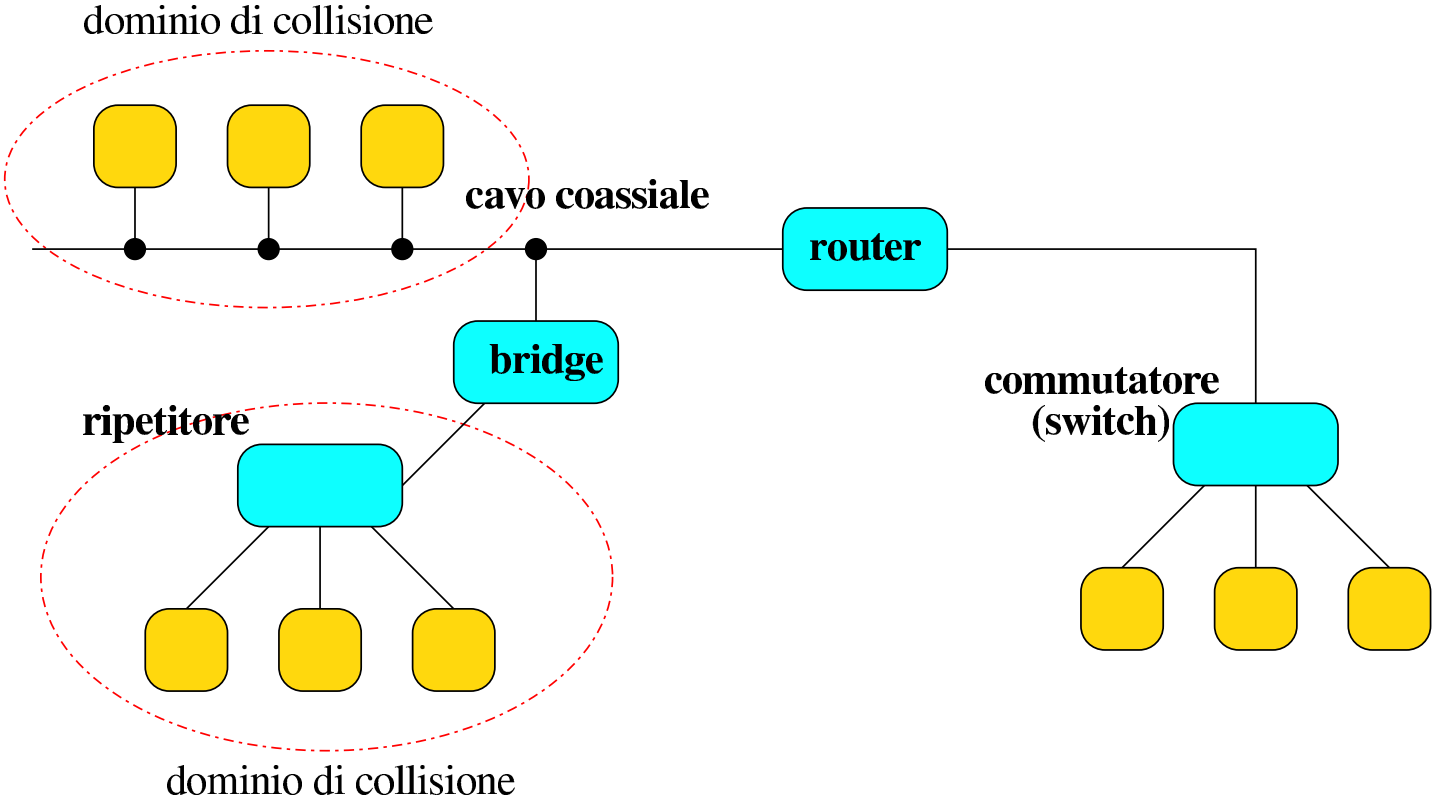
L'instradamento dei pacchetti attraverso le reti connesse al router avviene in base a una tabella di instradamento che può anche essere determinata in modo dinamico, in presenza di connessioni ridondanti, come già accennato per il caso dei bridge.

32.3.1 Topologia relativa al livello di astrazione

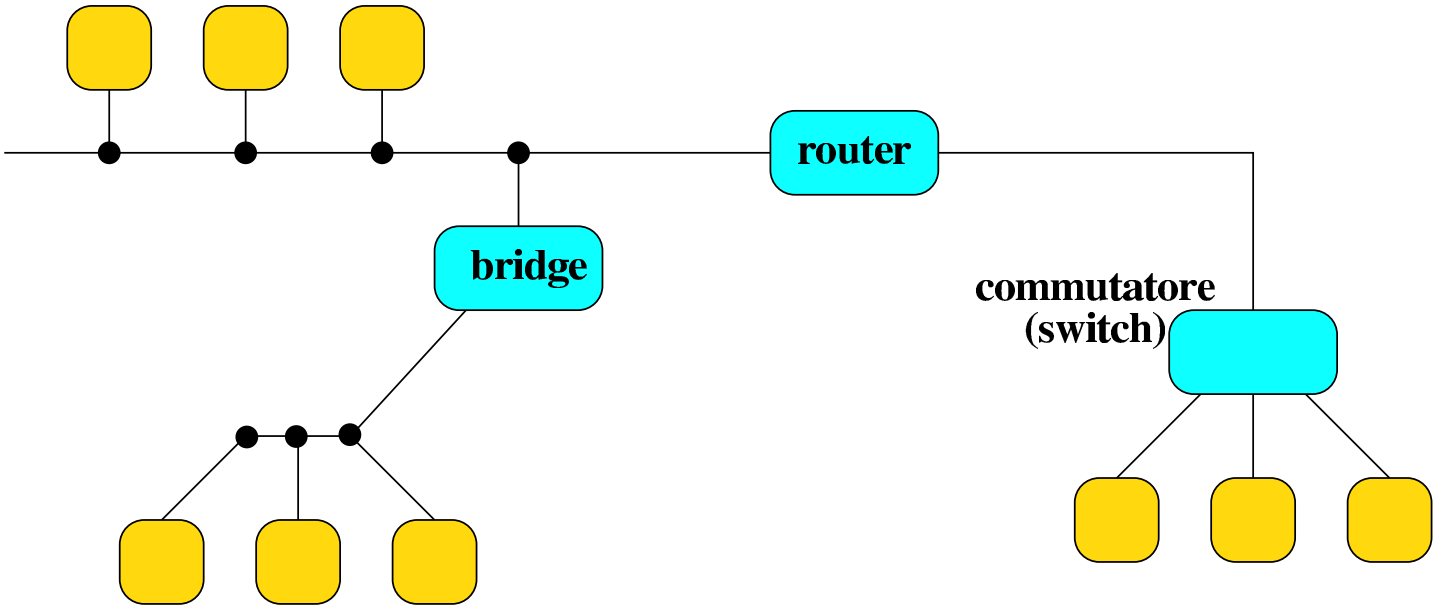
La topologia di rete può essere considerata al livello fisico, oppure a un livello più alto secondo il modello ISO-OSI. In pratica, quando ci si eleva a un livello superiore, alcuni componenti della rete «scompaiono», perché non vengono più considerati.

A titolo di esempio viene mostrato uno schema in cui appaiono dei nodi, collegati in vari modi tra di loro. Inizialmente, lo schema vie-

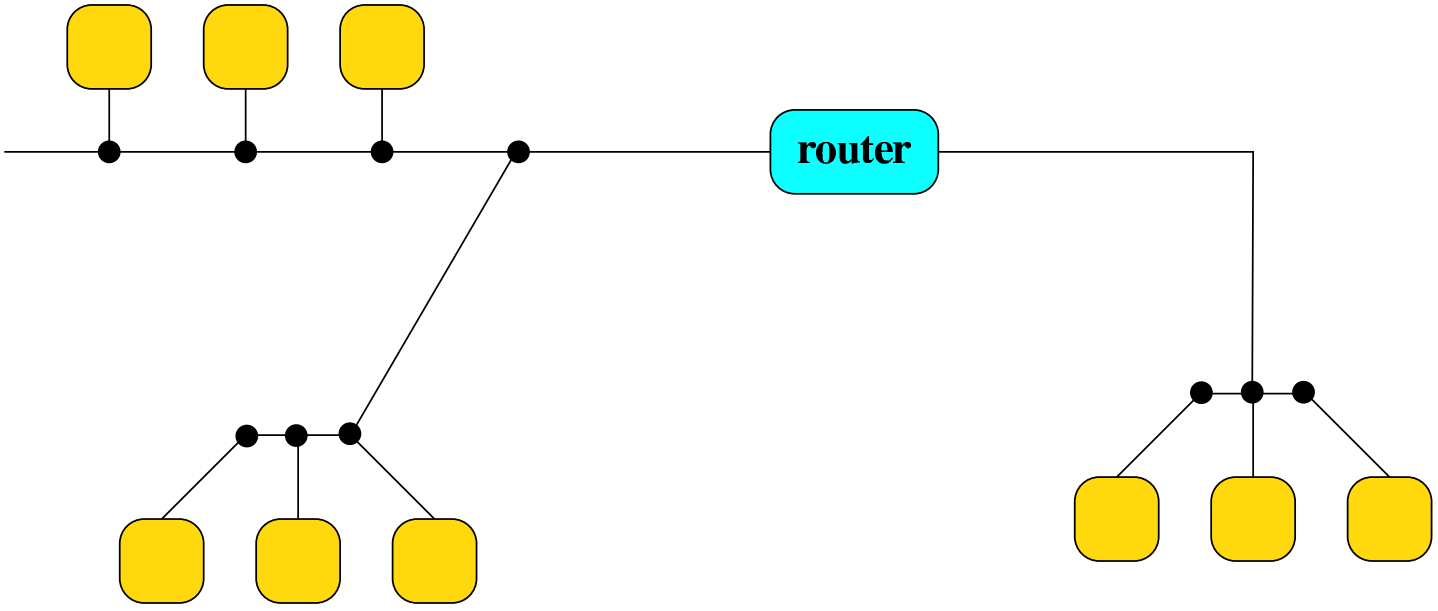
ne considerato al livello fisico, così si possono individuare anche i domini di collisione, ovvero i gruppi di nodi che possono creare accavallamenti di trasmissioni tra di loro:



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.

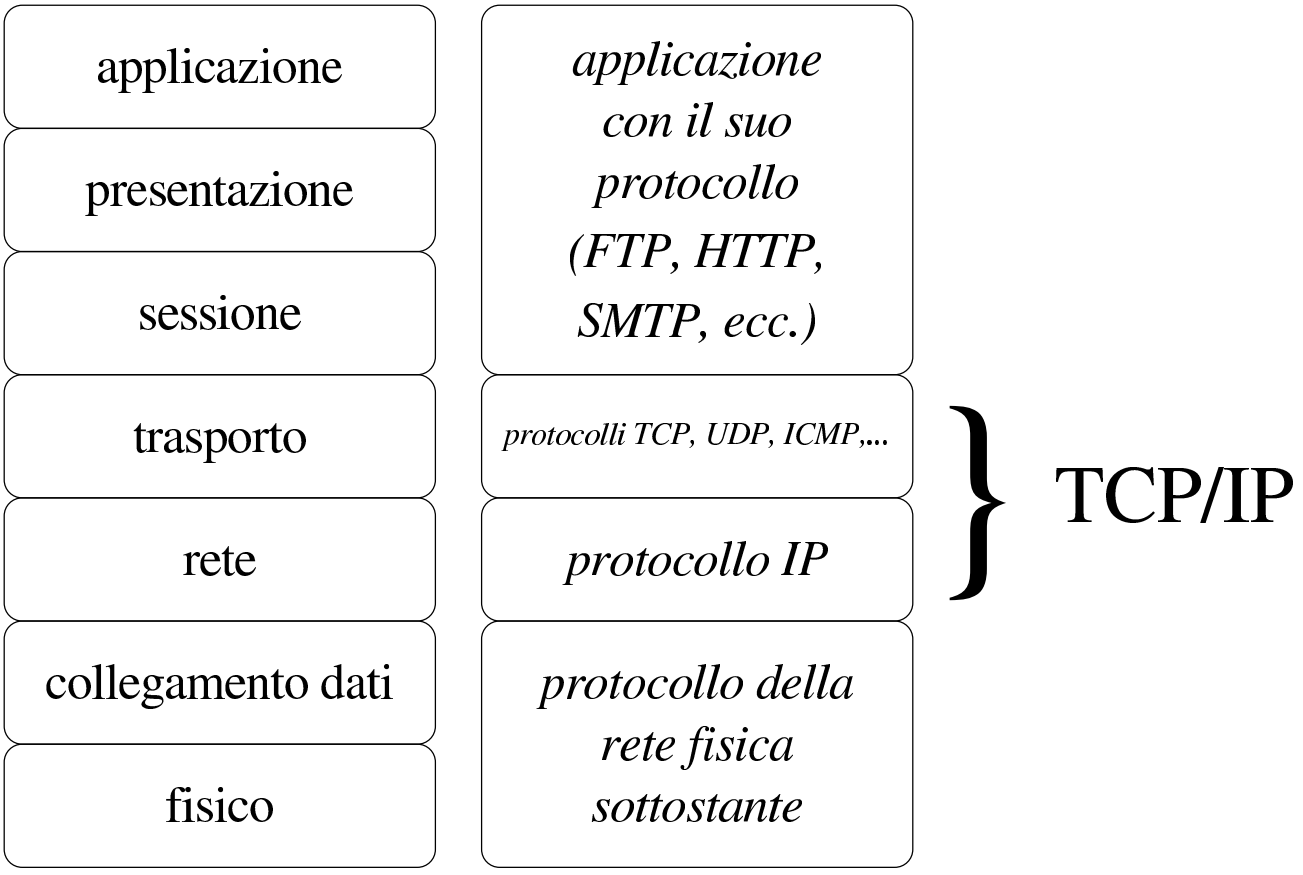
Tabella 32.15. Modello ISO-OSI di suddivisione delle competenze di un sistema TCP/IP.

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.
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

Rete	IP iniziale	IP finale	Broadcast
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 . sss hhhhh

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 . ssss hhhh

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

Rete	IP iniziale	IP finale	Broadcast
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 . sssss hhh

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
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

Rete	IP iniziale	IP finale	Broadcast
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.

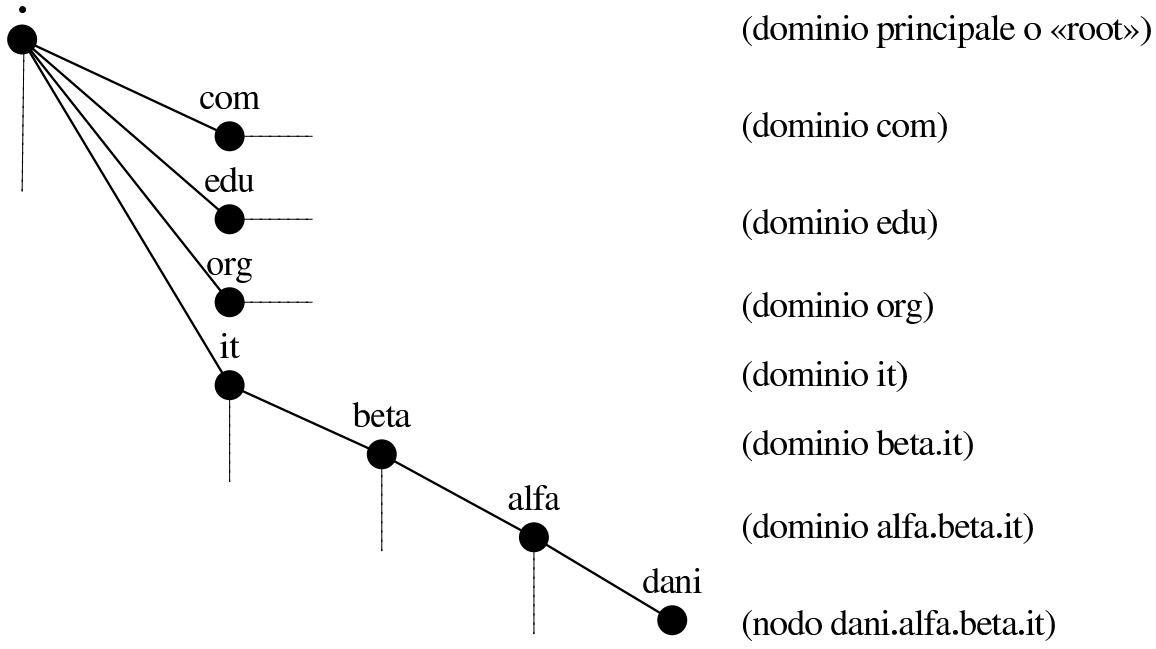
Indirizzo	Maschera	Descrizione
10.255.0.0	255.255.0.0	Si fa una suddivisione in sottoreti, come nell'esempio precedente, ma si rischia di interferire con l'indirizzo broadcast della rete complessiva, che sarebbe lo stesso.
172.16.0.0	255.255.0.0	Si fa una suddivisione in sottoreti e questa sottorete 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.

Indirizzo	Maschera	Descrizione
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.

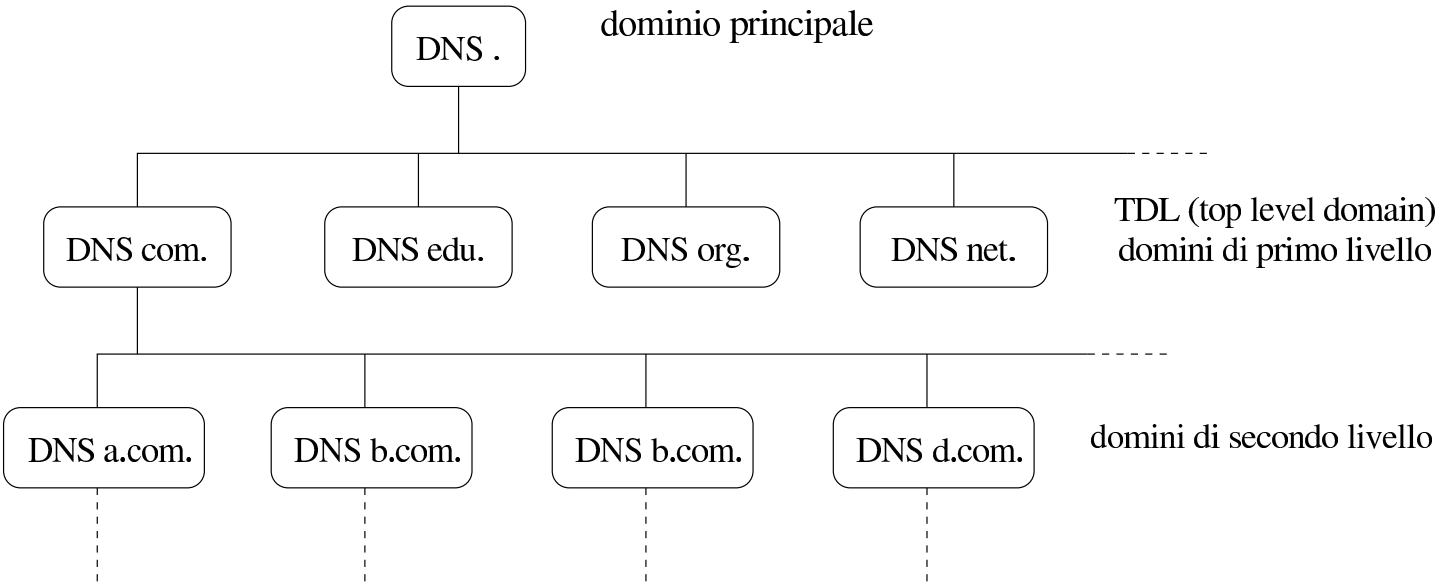
Spesso, all'interno della propria rete locale, è possibile identificare un nodo attraverso il solo nome finale (a sinistra), senza la parte iniziale del dominio di appartenenza. Per esempio, se la rete in cui si opera corrisponde al dominio *brot.dg*, il nodo *roggen* viene inteso essere *roggen.brot.dg*.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

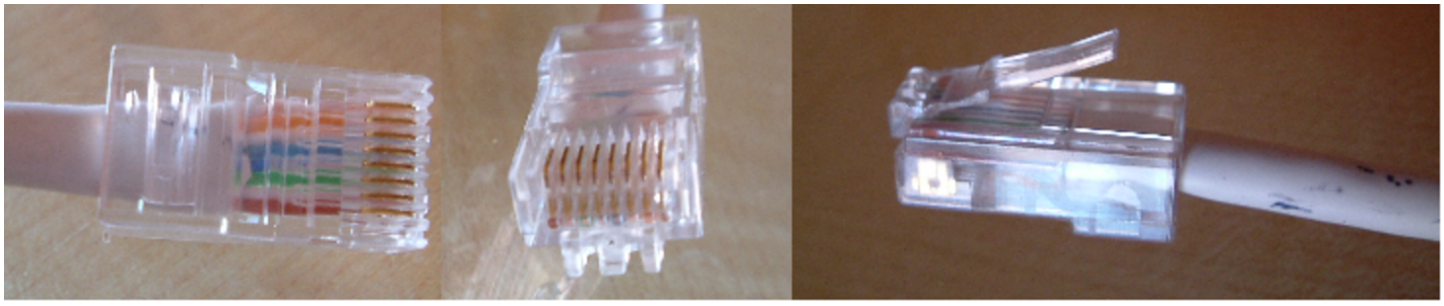


Figura 32.45. Componente per il raccordo dei collegamenti UTP, costituito generalmente da un commutatore di pacchetto (*switch*).

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.

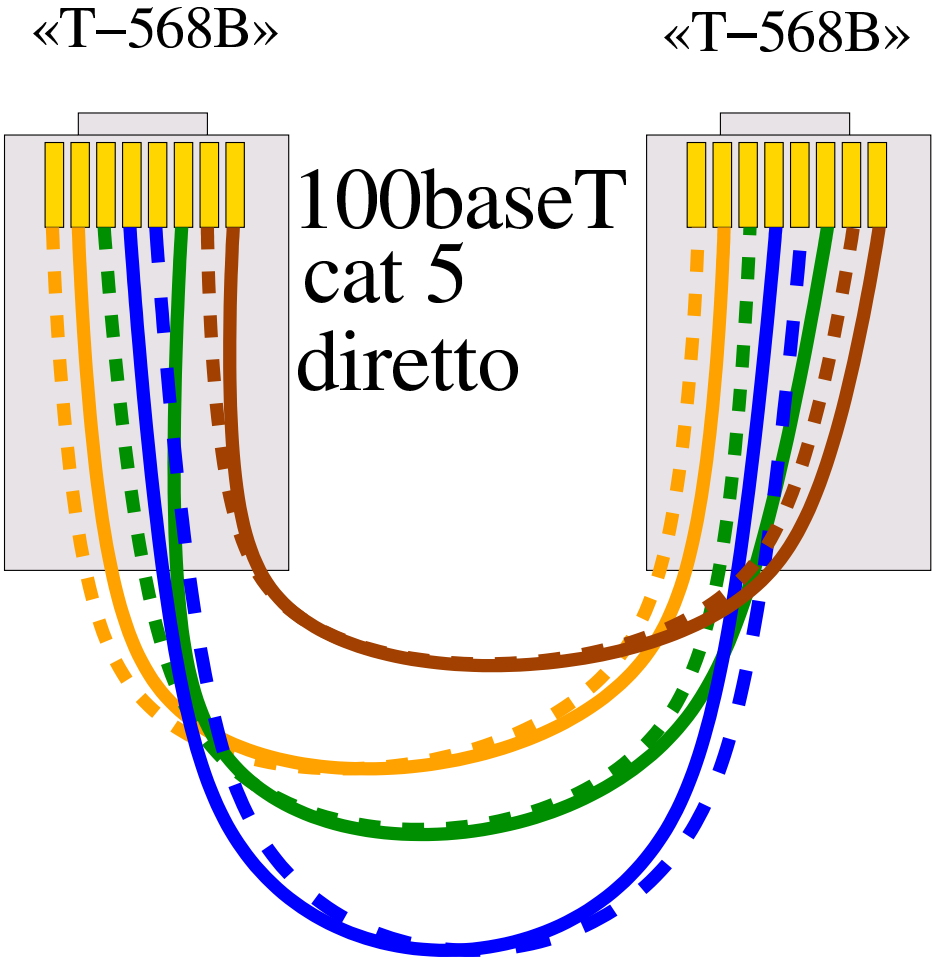
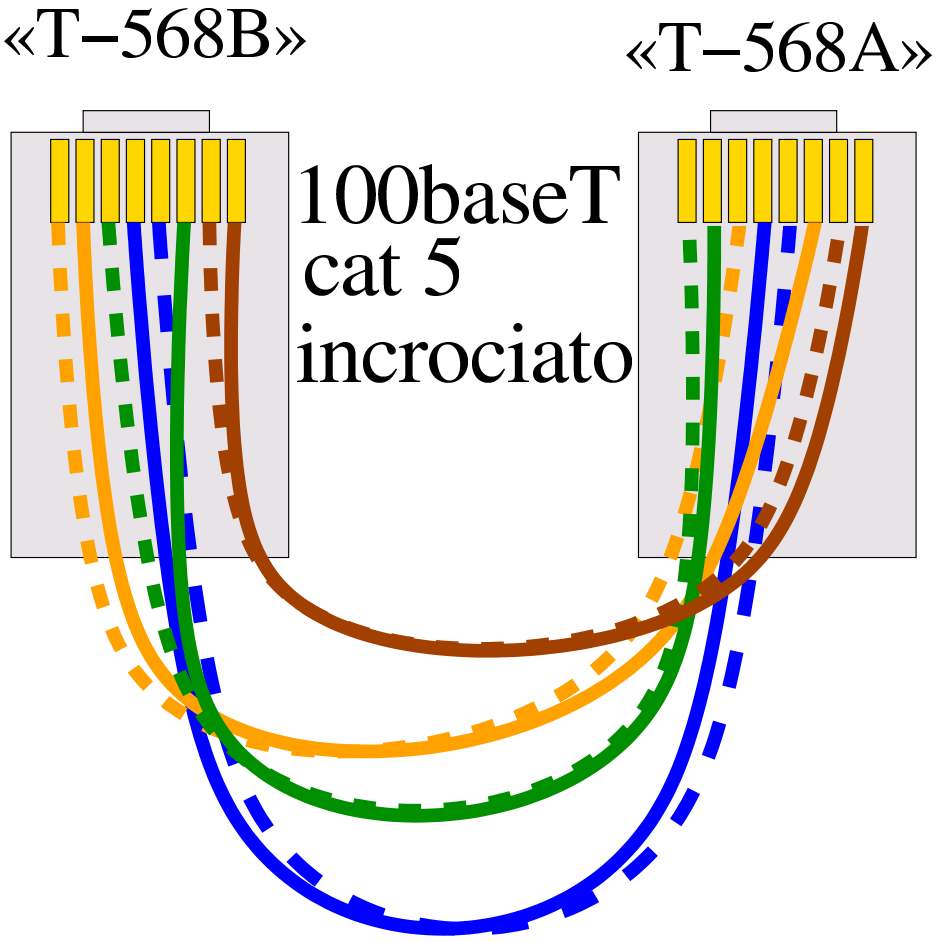


Figura 32.47. Cavo 100/1000baseT categoria 5 o 6 incrociato. Le coppie 1-2:3-6, 3-6:1-2, 4-5:7-8 e 7-8:4-5 sono ritorte.

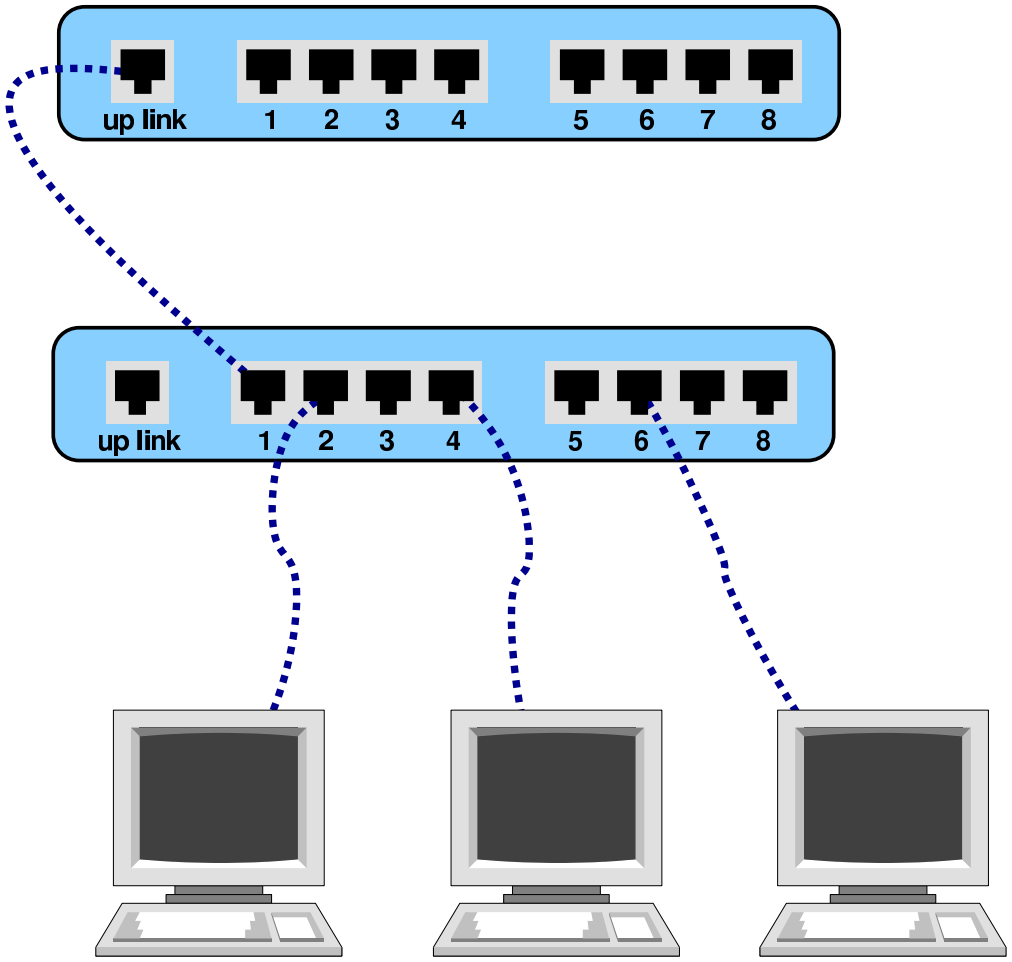


32.5.3.1 Commutatori di pacchetto e porte «up link»

«

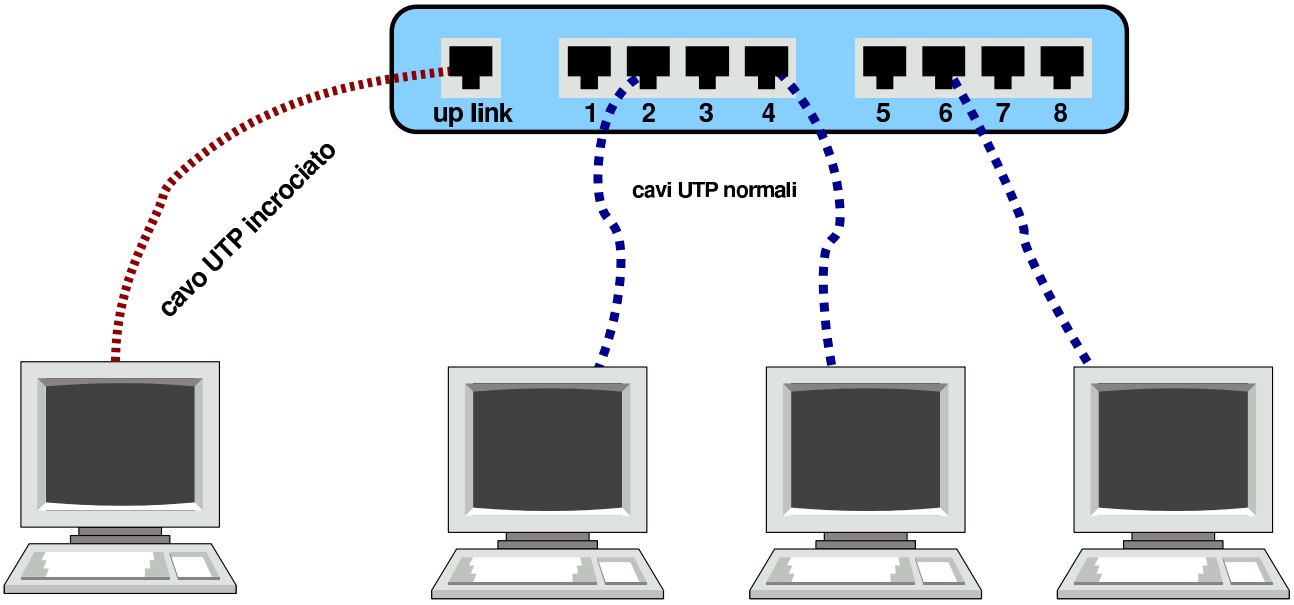
I commutatori di pacchetto più vecchi dispongono di un certo numero di porte «normali» e di una porta aggiuntiva, denominata *up link*. Questa porta speciale serve a collegare più commutatori di pacchetto assieme, come si può vedere nella figura 32.48.

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.

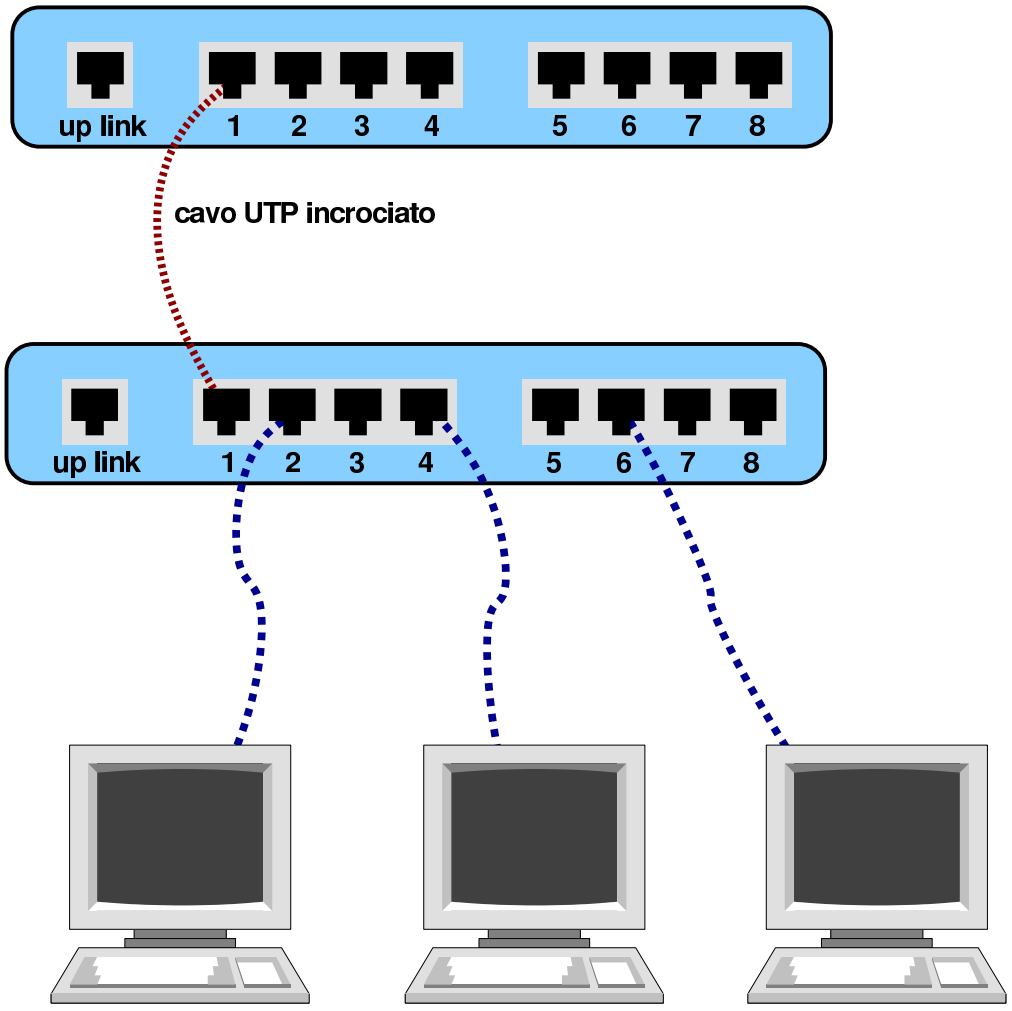
Figura 32.49. Utilizzo della porta *up link* per un collegamento attraverso cavo incrociato.



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.

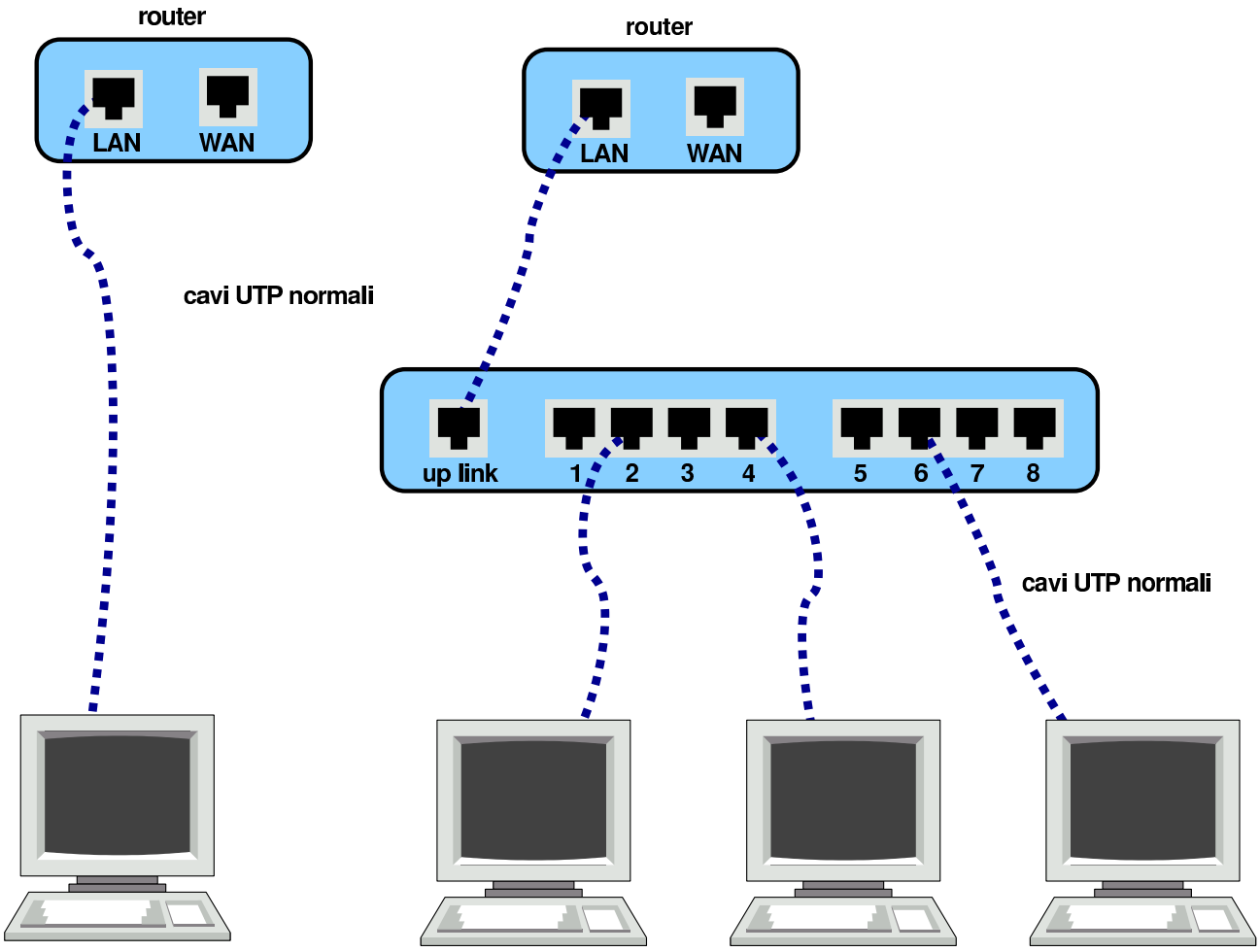
Figura 32.50. Collegamento tra due commutatori di pacchetto, usando solo le porte normali, con un cavo incrociato.



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.

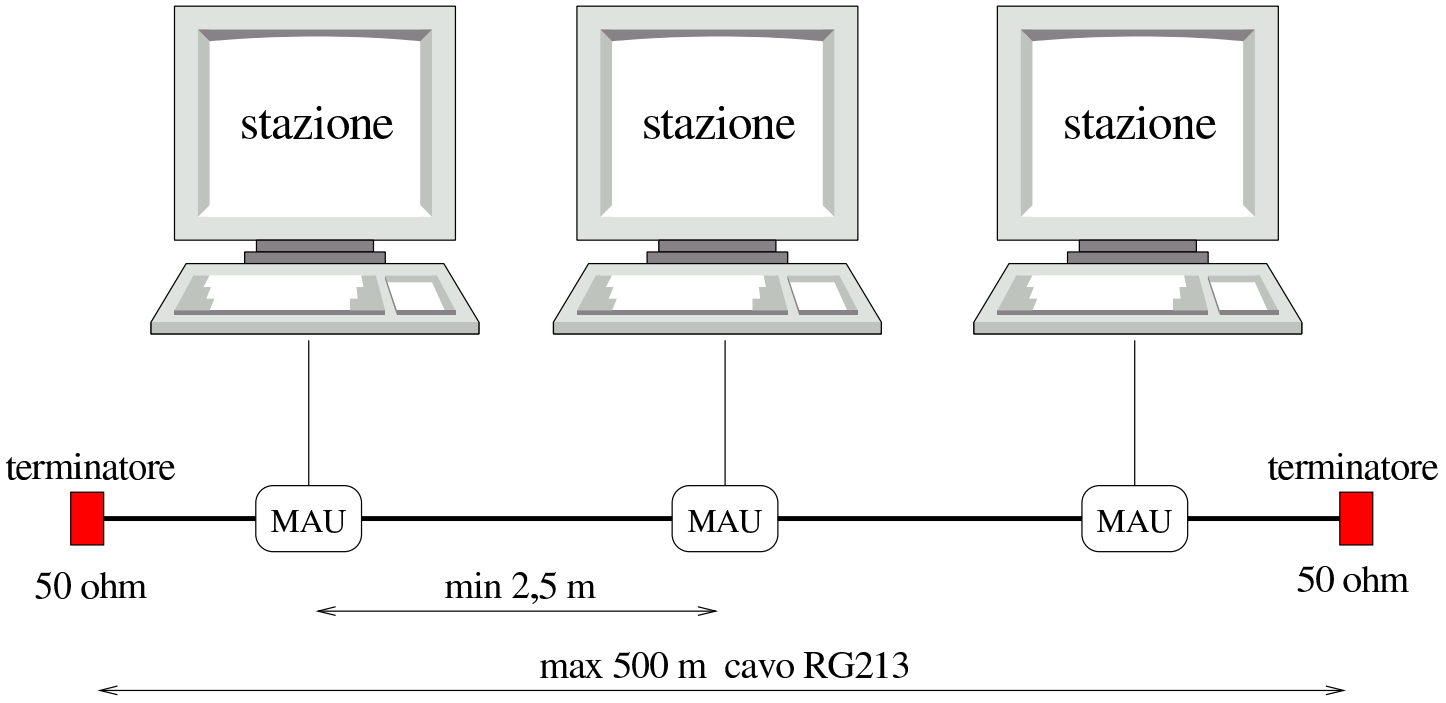
32.6.2.1 10base5 senza ripetitori

«

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.



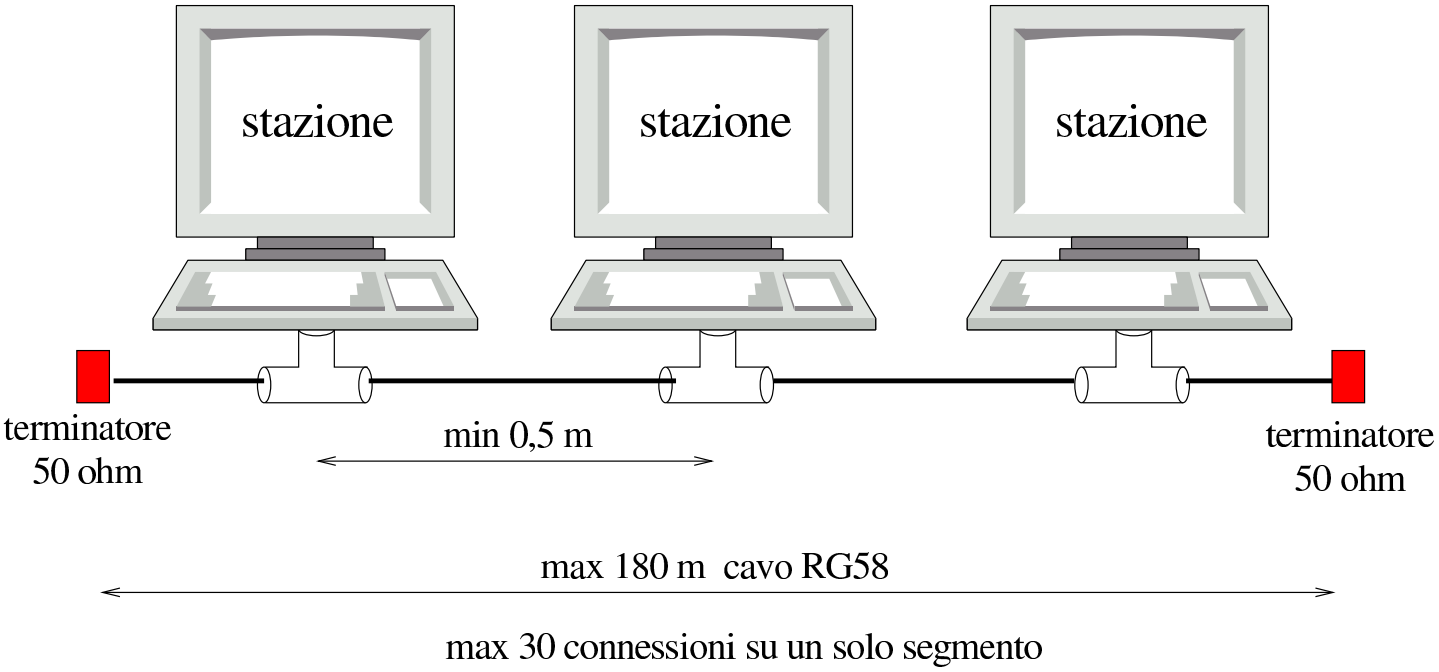
Come si può intuire, se il tratto di cavo coassiale non è continuo, ma ottenuto dalla giunzione di più pezzi, la lunghezza massima deve essere diminuita.

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.



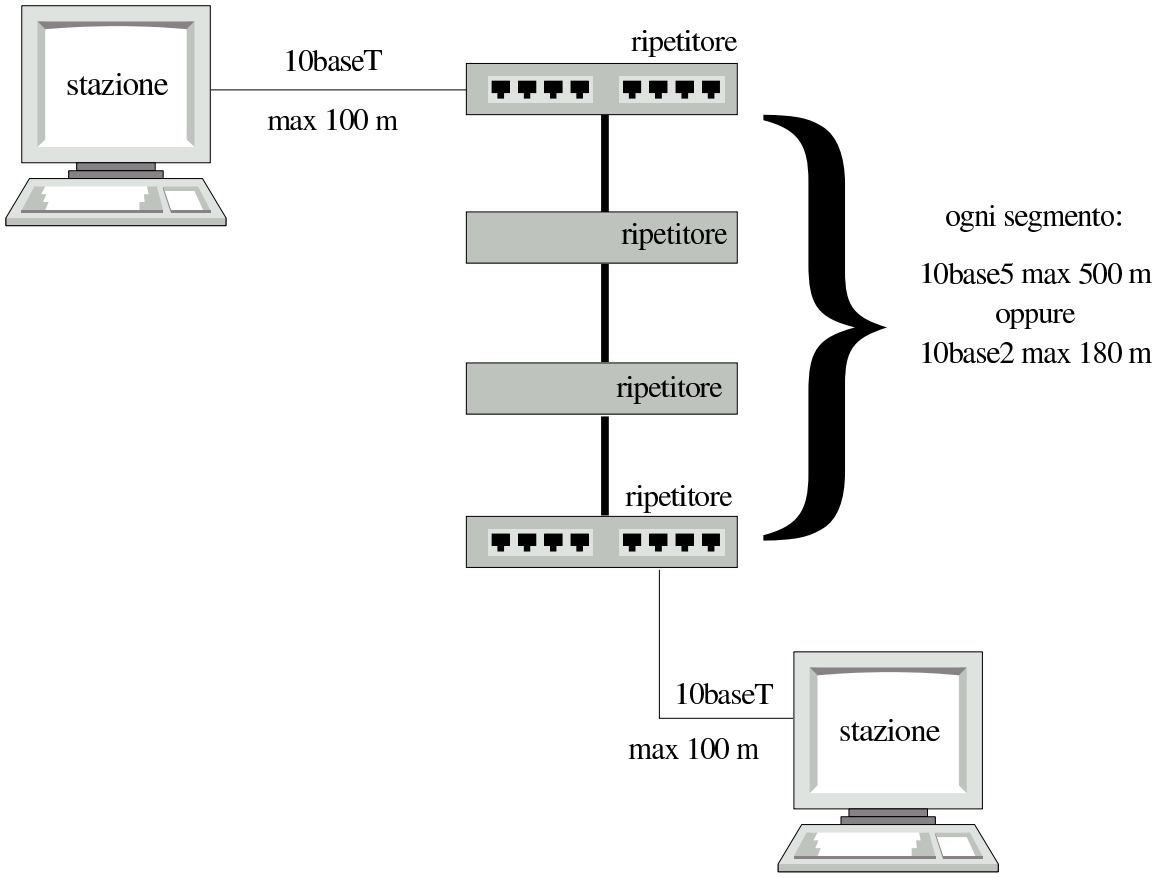
32.6.2.3 10baseT

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).

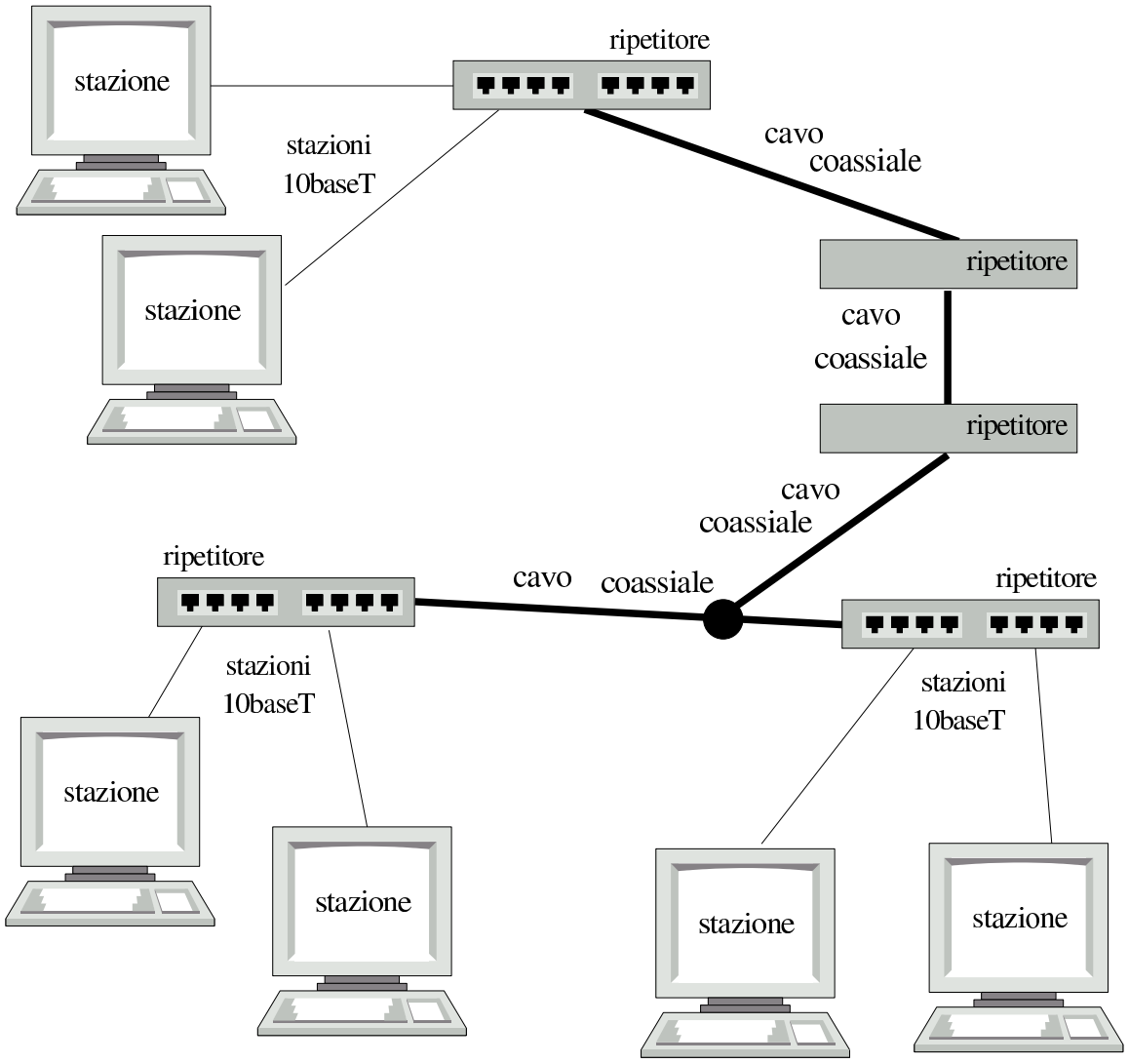
Figura 32.57. Esempio di configurazione massima con quattro ripetitori, tre segmenti coassiali e due segmenti 10baseT.



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 potreb-

bero prevedere l'impostazione della porta come «normale» o «bidirezionale». Se si può scegliere, è opportuno che la porta sia normale.

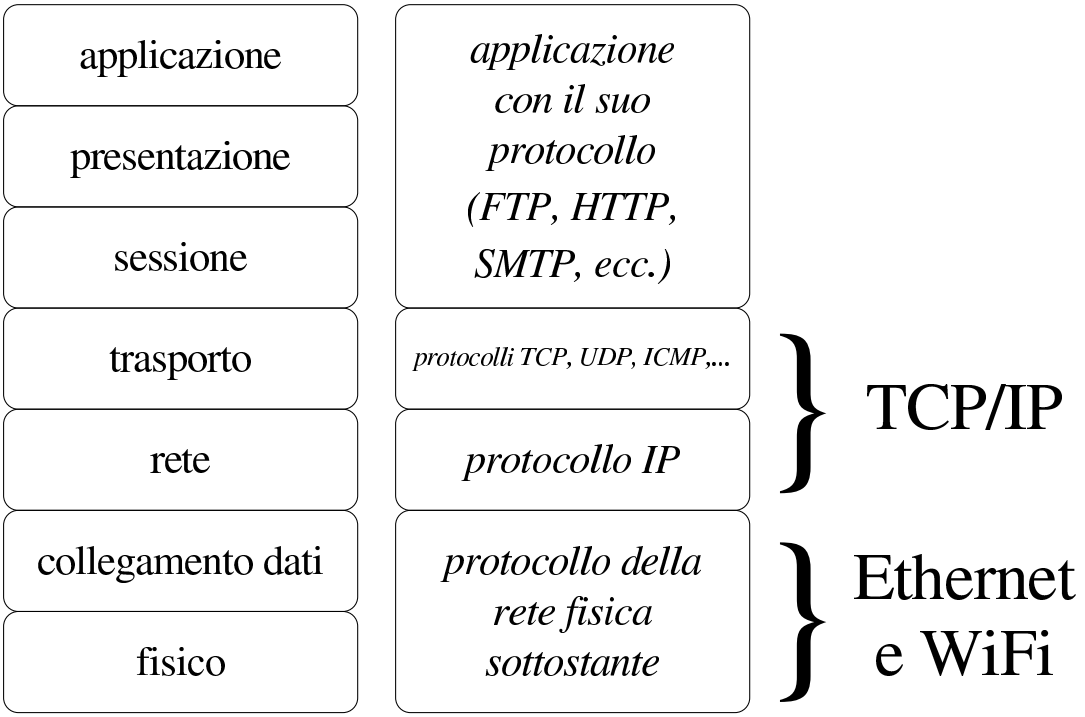
A questo punto si pone il problema del riconoscimento della porta da parte del kernel. Se il file principale del kernel incorpora la gestione del protocollo PLIP, l'interfaccia dovrebbe essere individuata automaticamente e in modo corretto (riguardo alla sua configurazione effettiva). Eventualmente si 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.

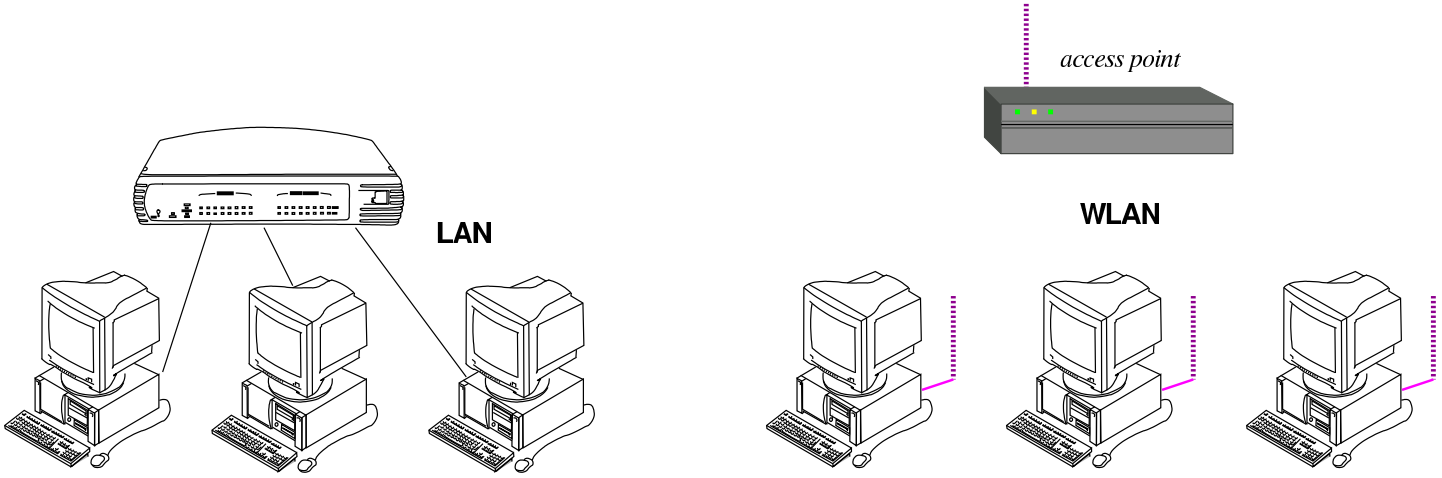
Figura 32.60. Collocazione dei componenti di una rete WiFi nel modello ISO-OSI.



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.

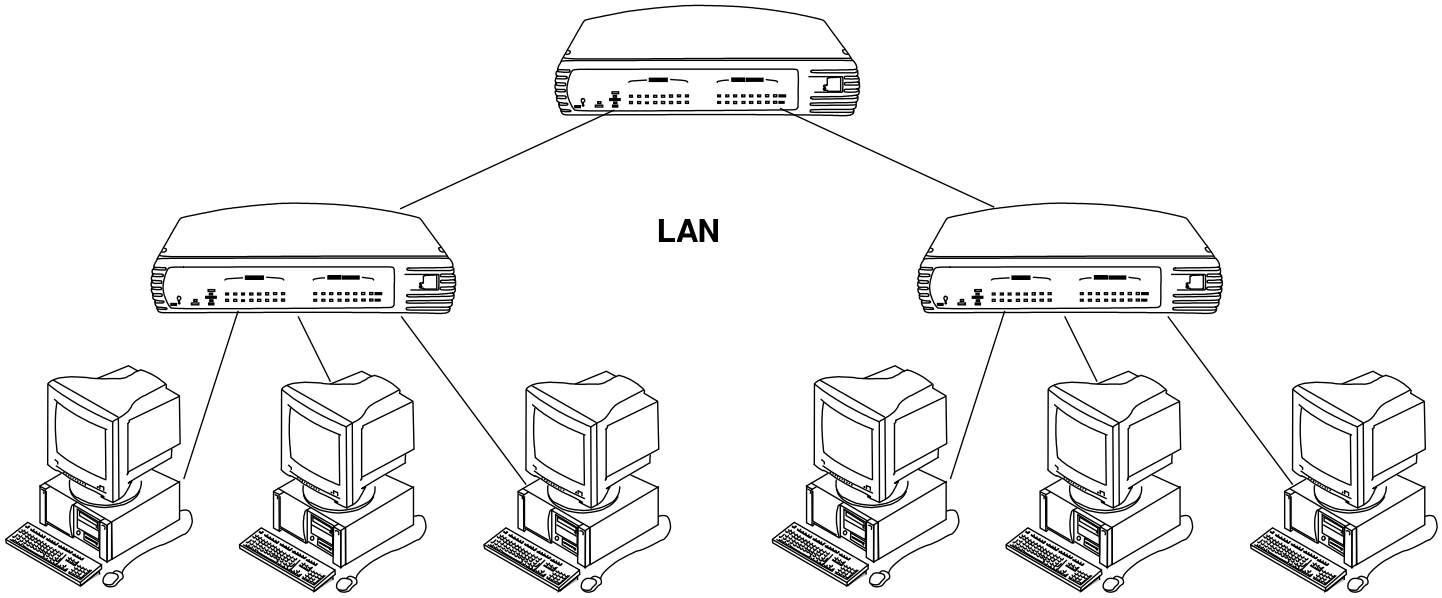
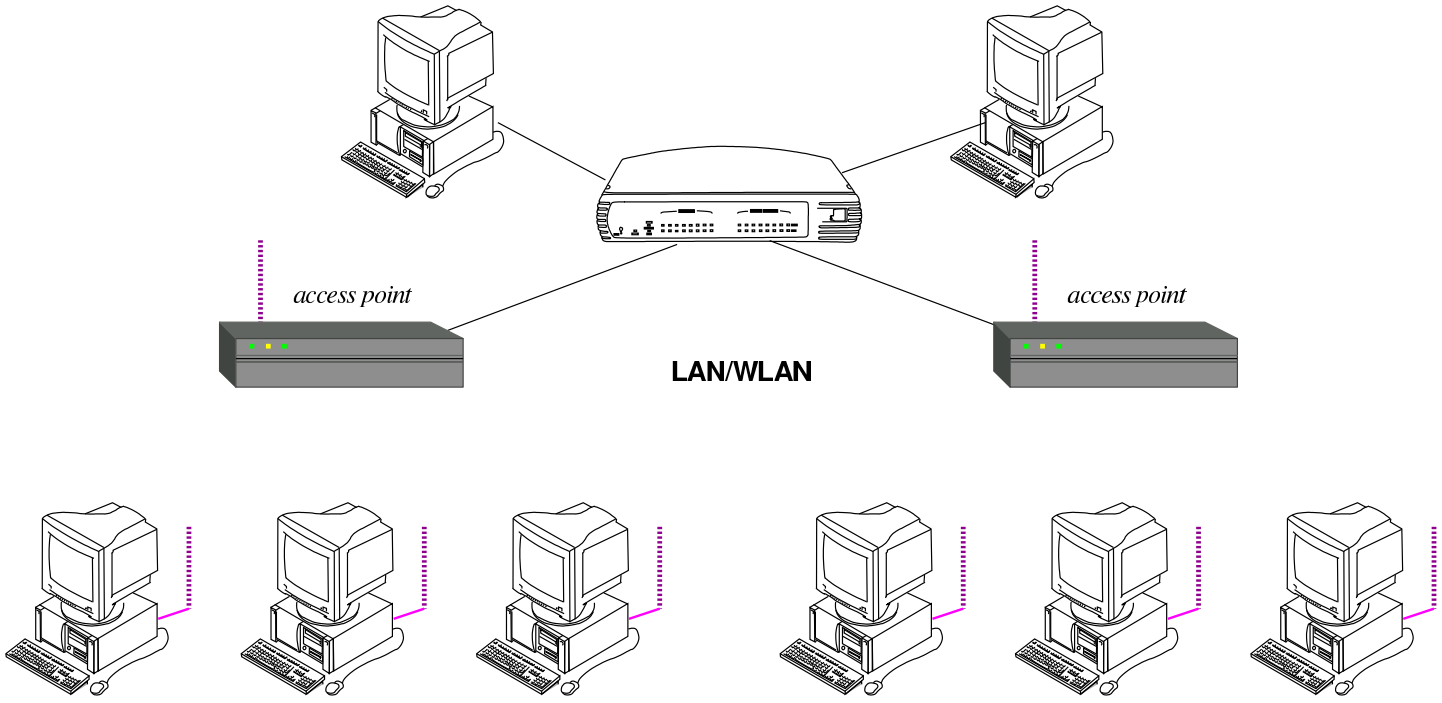


Figura 32.63. Una rete locale mista, con cablaggio e vari punti di accesso via radio.



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

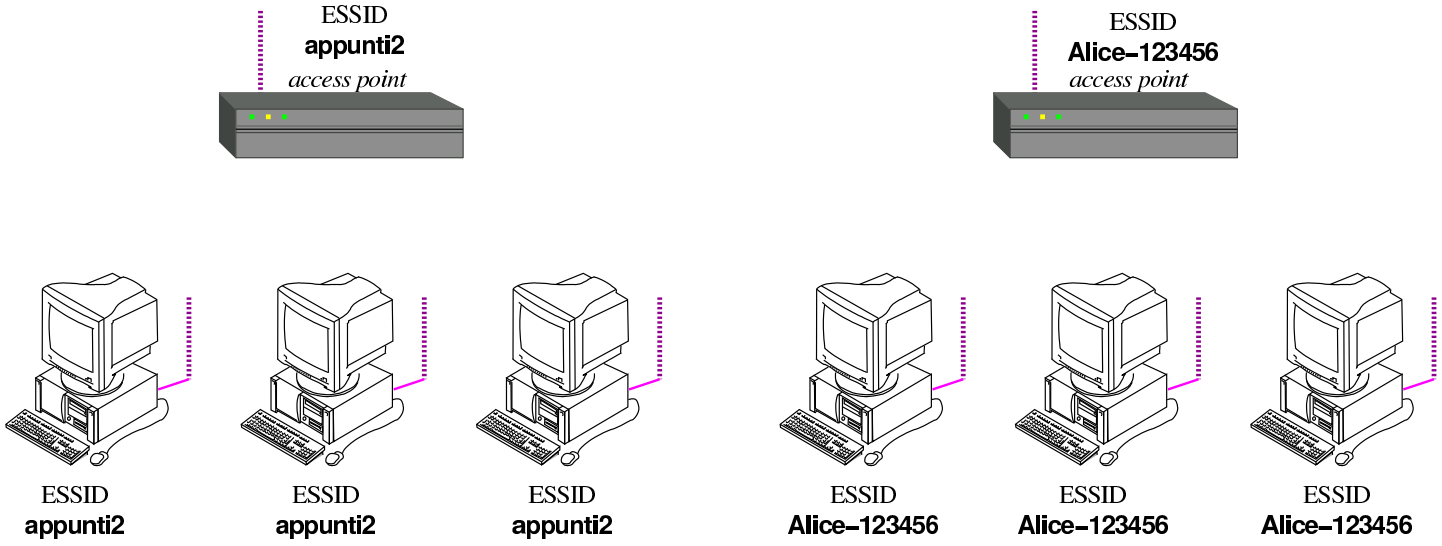
32.7.4 ESSID: extended service set id

«

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 chiun-

que 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.

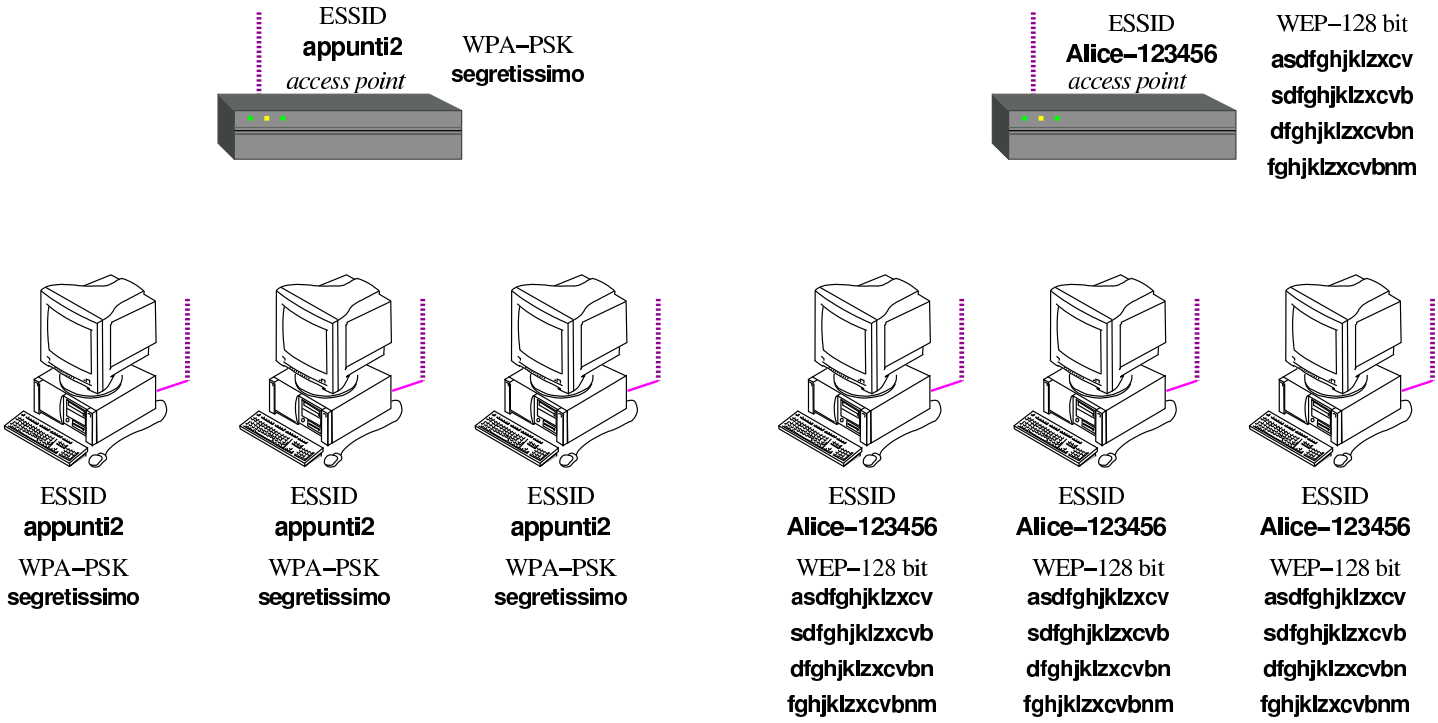
Il sistema crittografico WEP richiede la definizione di una chiave segreta che deve avere una dimensione esatta. Per la precisione, esistono varie versioni nella lunghezza di tale chiave, ma non sono ammissibili dimensioni intermedie. Tra i vari tipi di crittografica WEP, quelli usati comunemente sono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

Tipo di crittografia WEP	Lunghezza della chiave in bit	Lunghezza della chiave in byte	Lunghezza della chiave in cifre esadecimali
??	128 bit	16 byte	32 cifre
??	152 bit	19 byte	28 cifre
??	232 bit	29 byte	58 cifre
??	256 bit	32 byte	64 cifre

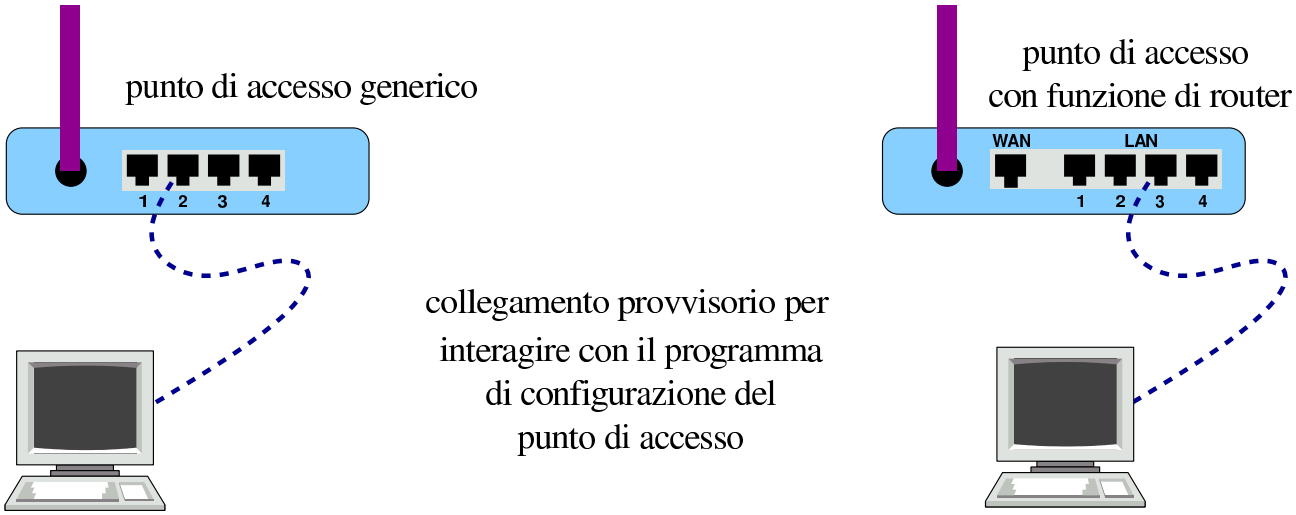
Figura 32.68. Due WLAN con configurazione indipendente del nome ESSID e del sistema crittografico.



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

Voce di configurazione	Valore attribuito
`mode`	AP (*Access point*)
`band`	2.4 GHz, b+g
`ESSID`	appunti2
`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).

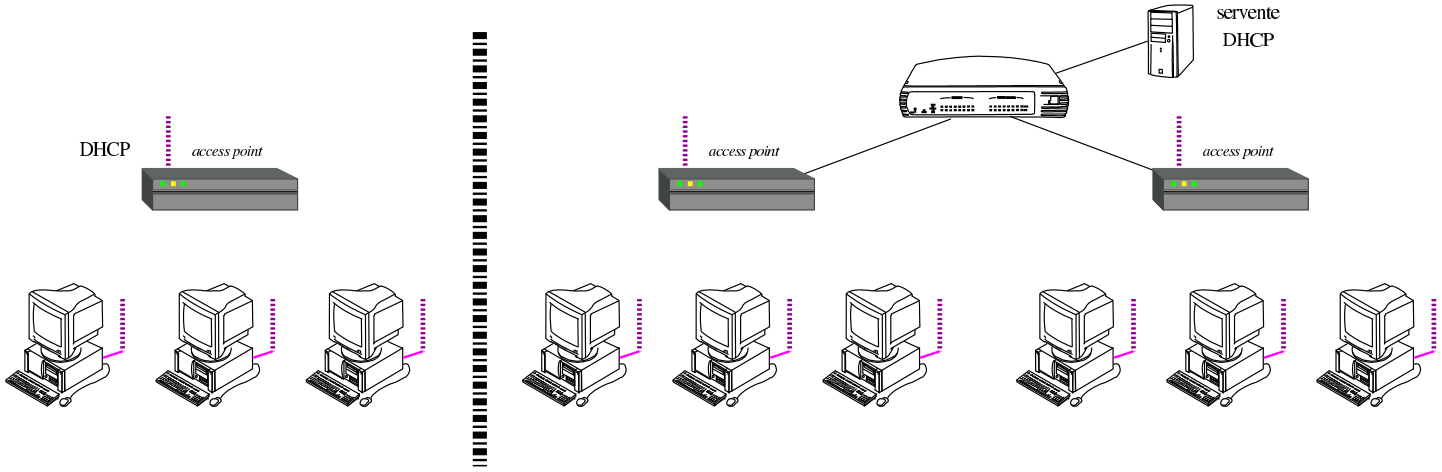
32.7.7 Configurazione automatica dei nodi periferici

«

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*).

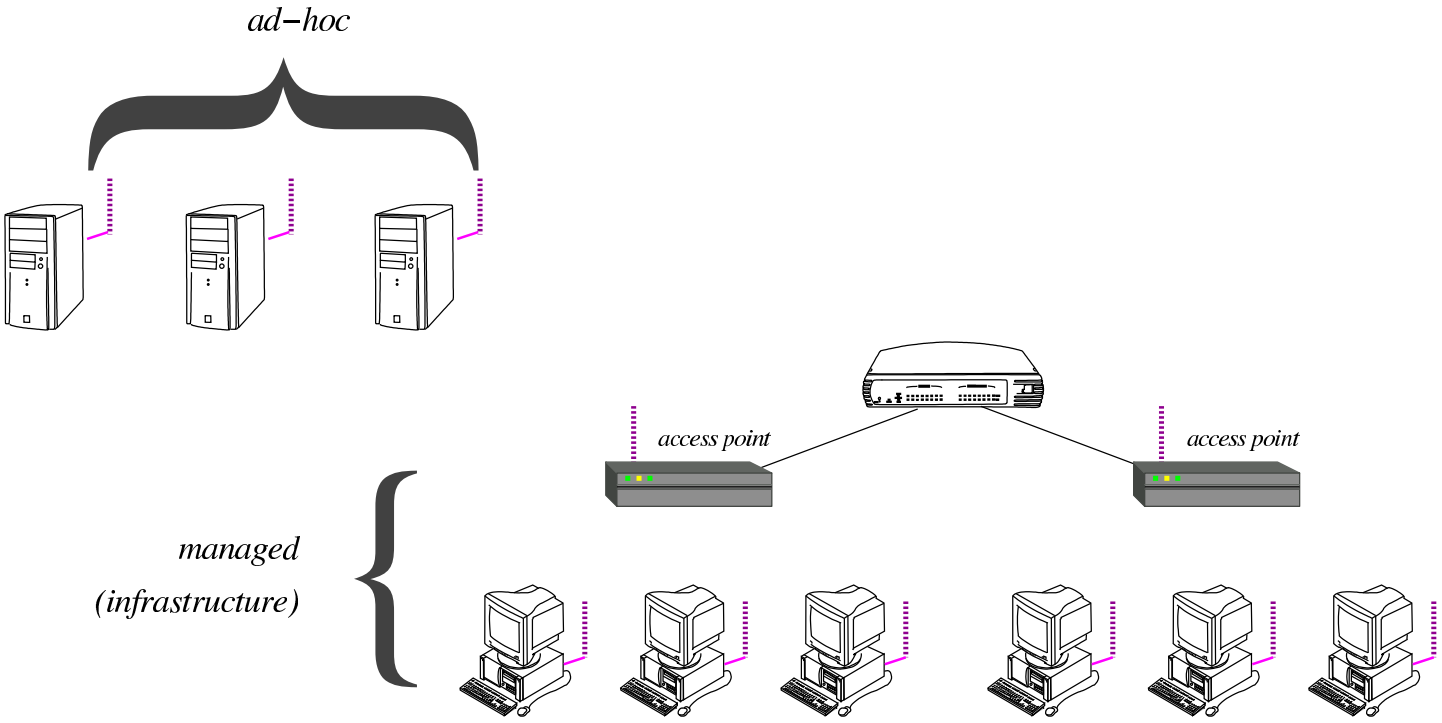


Tabella 32.74. Alcune modalità di funzionamento di un componente WiFi.

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 no-

me 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.

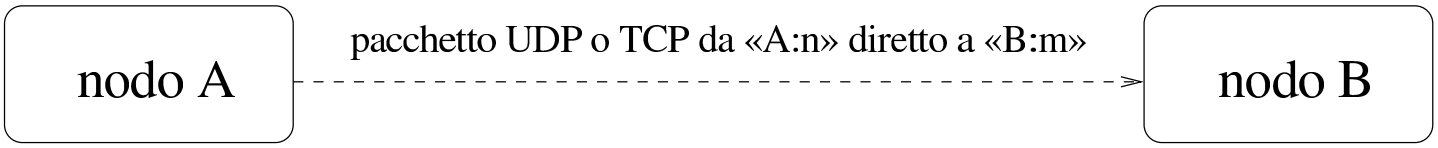
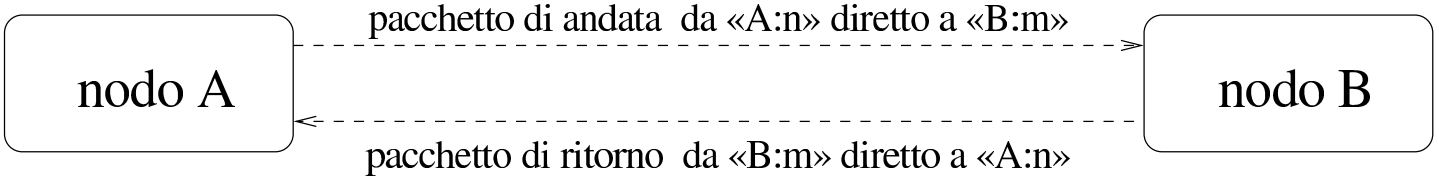
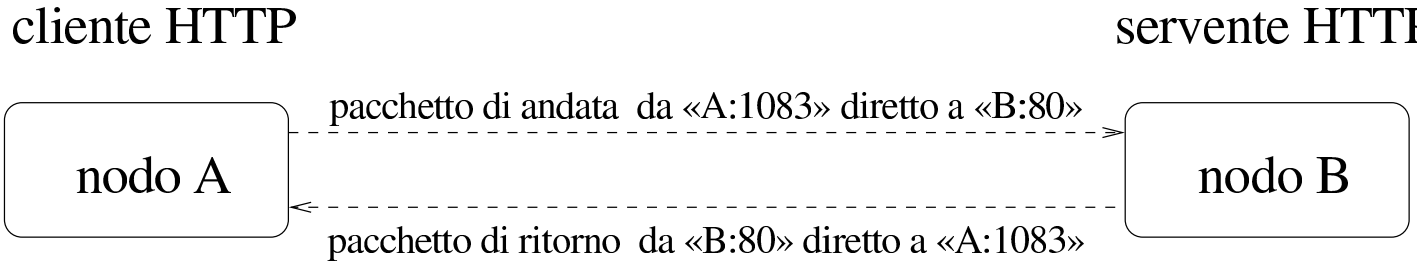


Figura 32.104. Andata e ritorno per le connessioni che prevedono l'uso delle porte: «***n***» è la porta usata nel nodo «A»; «***m***» è la porta usata nel nodo «B».



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			

Tipo	Codice	Nome	Chi lo utilizza
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			

Tipo	Codice	Nome	Chi lo utilizza
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	

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;

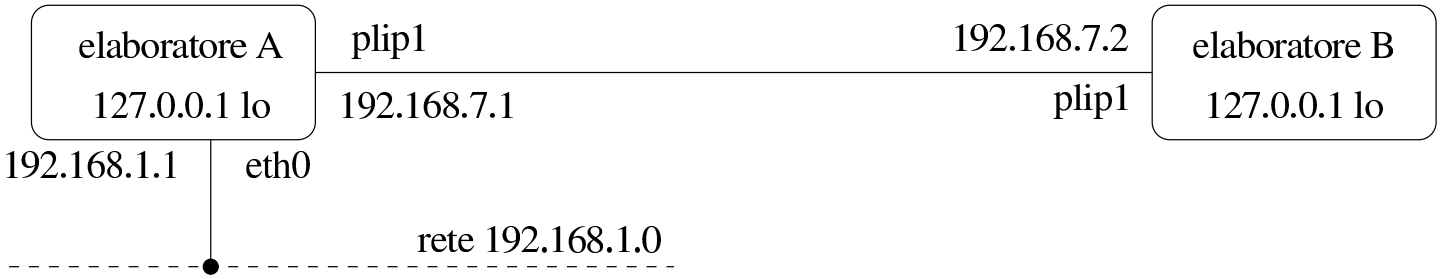
- un'interfaccia di rete di un elaboratore può comunicare con un'interfaccia di un altro elaboratore solo se queste sono fisicamente connesse alla stessa rete;
- un'interfaccia di rete di un elaboratore può comunicare con un'interfaccia di un altro elaboratore solo se gli indirizzi di queste interfacce 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).

Opzione	Descrizione
`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.

Opzione	Descrizione
`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

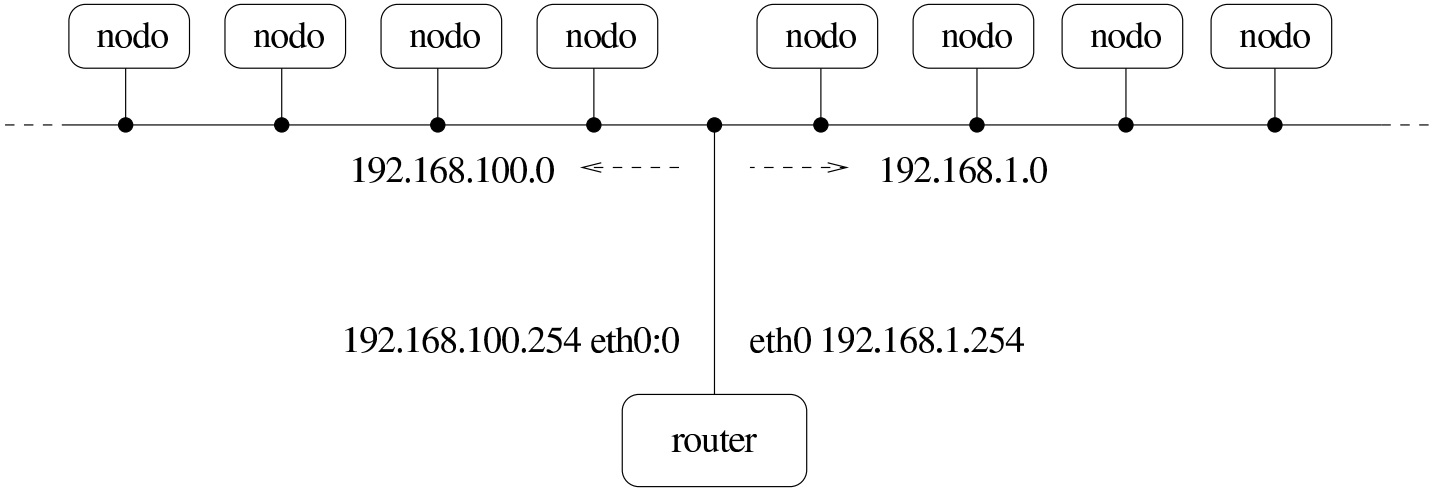
È possibile attribuire a ogni interfaccia di rete più di un indirizzo IPv4. Ciò si ottiene definendo delle interfacce virtuali, riferite a quel-

le 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]
```

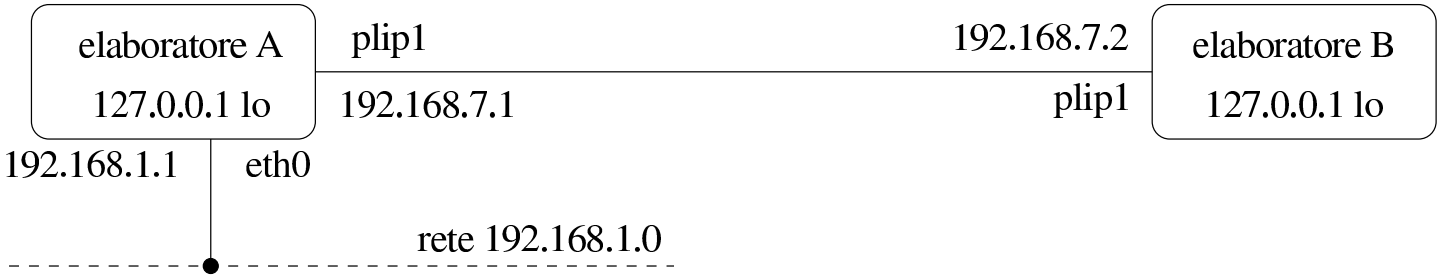
32.10 IPv4: instradamento locale

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 instra-

damento 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]
```

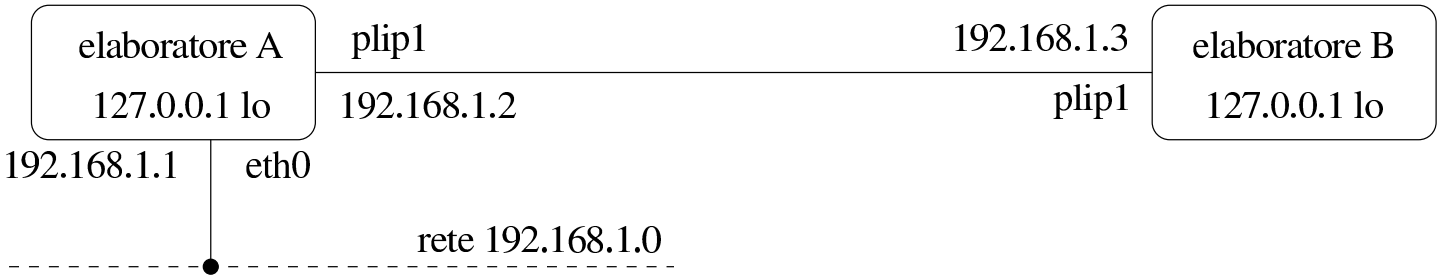
Oppure:

```
$ 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;

Modello	Descrizione

`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.

Opzione	Descrizione
`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.

Opzione	Descrizione
`reject`	Permette di impedire l'utilizzo di un instradamento.
`[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.

Nome	Descrizione
`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 si-

stema 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.
`-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.

Opzione	Descrizione
`-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.

Opzione	Descrizione
`-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).
`-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:

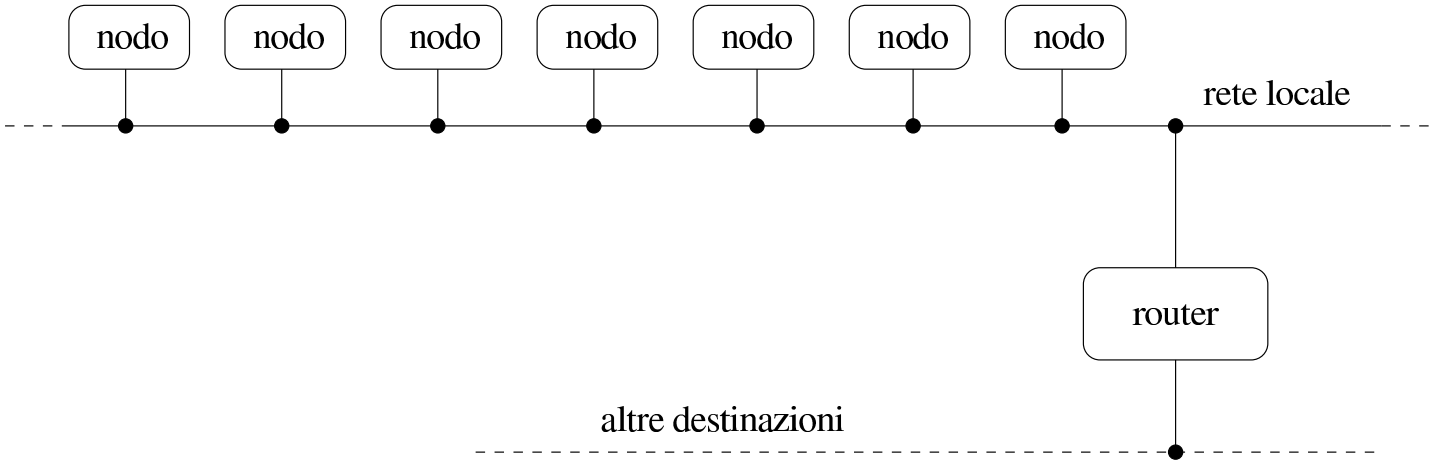
```
192.168.1.2 00:01:02:03:04:05
192.168.1.3 00:14:02:23:07:1c
192.168.1.4 00:00:03:2d:00:0b
```

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.

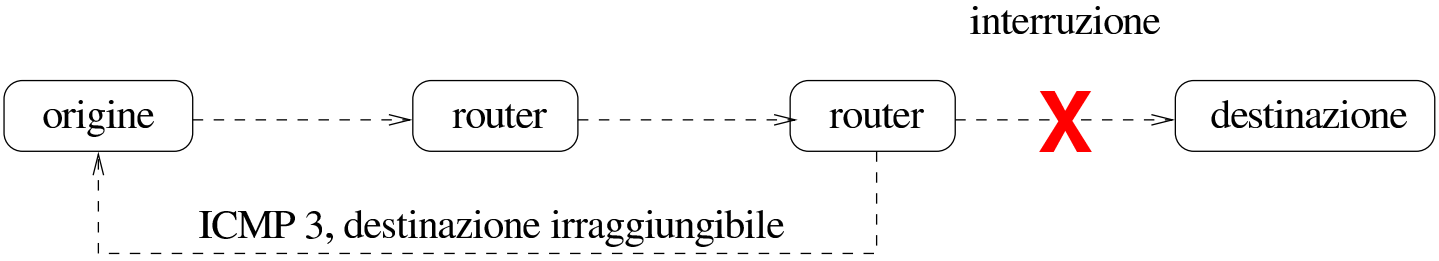
32.11.1 Destinazione irraggiungibile

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

«

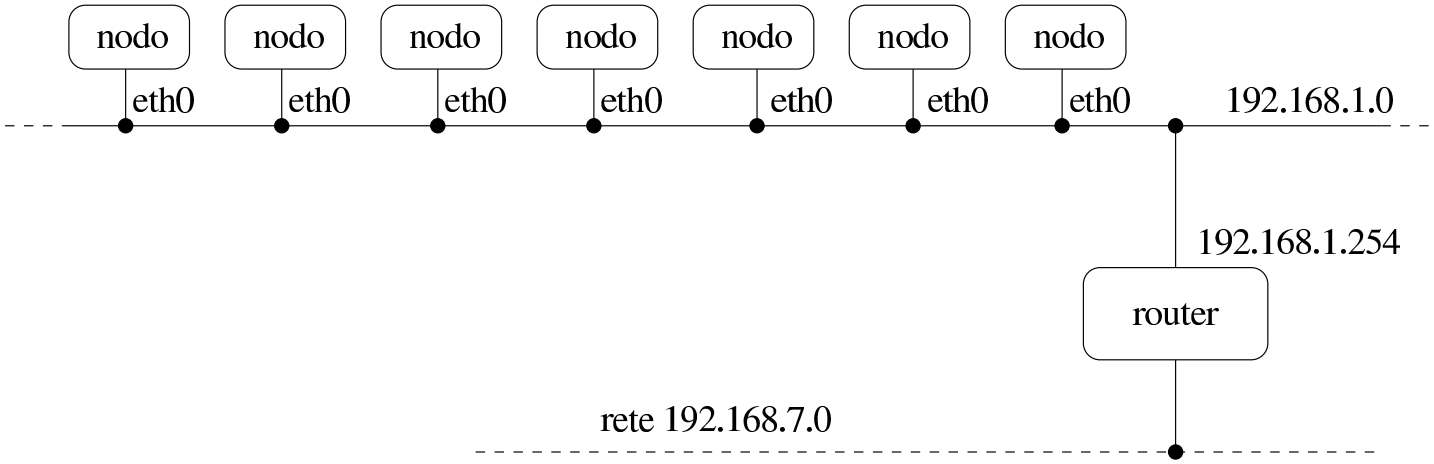
Una rete locale potrebbe essere articolata in sottoreti in modo da evitare di sovraffollare di traffico un’unica rete. Per fare in modo che le sottoreti possano comunicare tra loro in caso di necessità, si devono utilizzare i router che funzionano come ponti tra una sottorete e un’altra.

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]
```

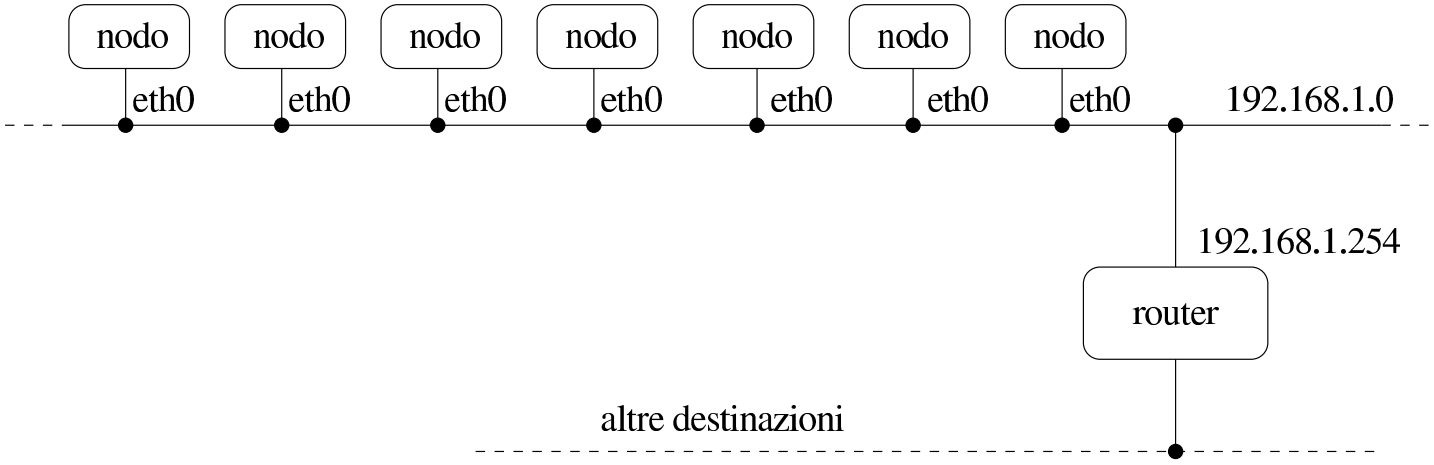
```
Kernel IP routing table
Destination   Gateway   Genmask    Flags Metric Ref    Use Iface
0.0.0.0       0.0.0.0   0.0.0.0    U     0      0        1 eth0
```

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:

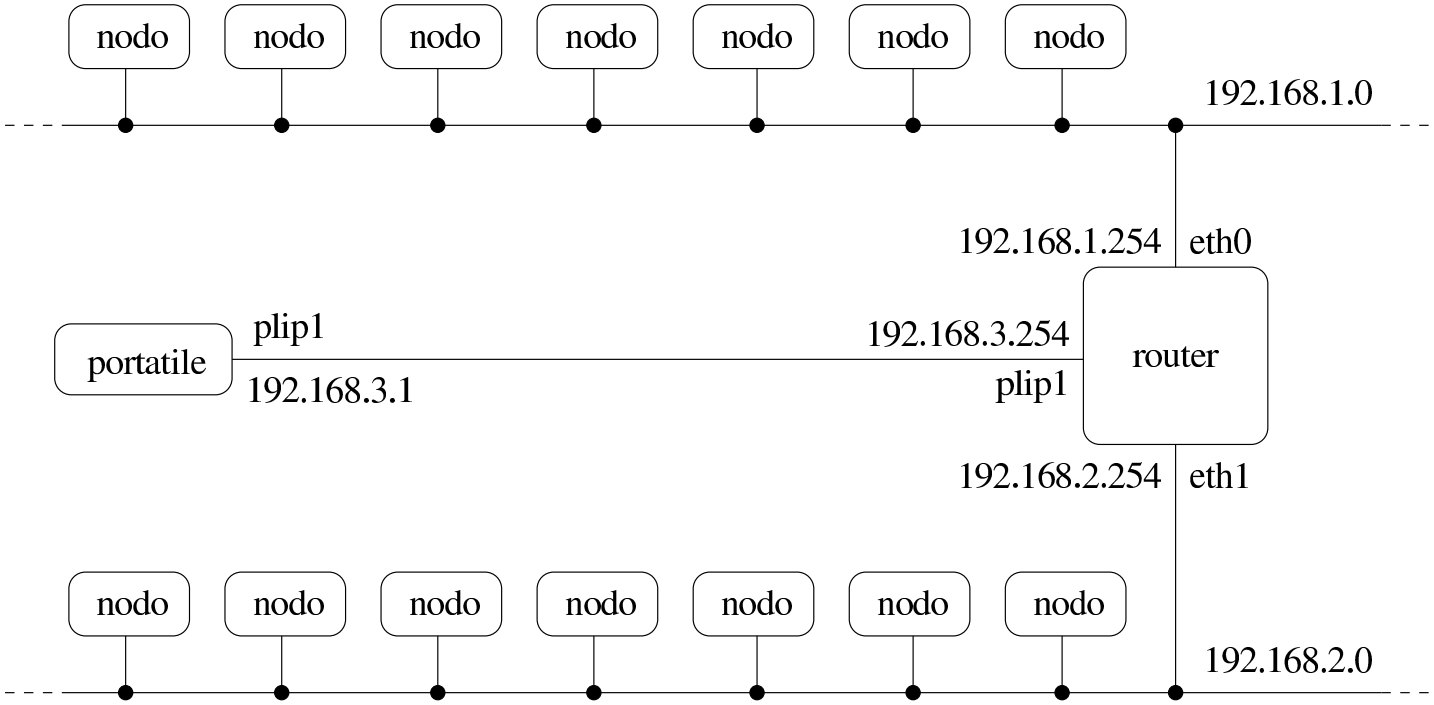
```
# echo 1 > /proc/sys/net/ipv4/ip_forward[Invio]
```

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

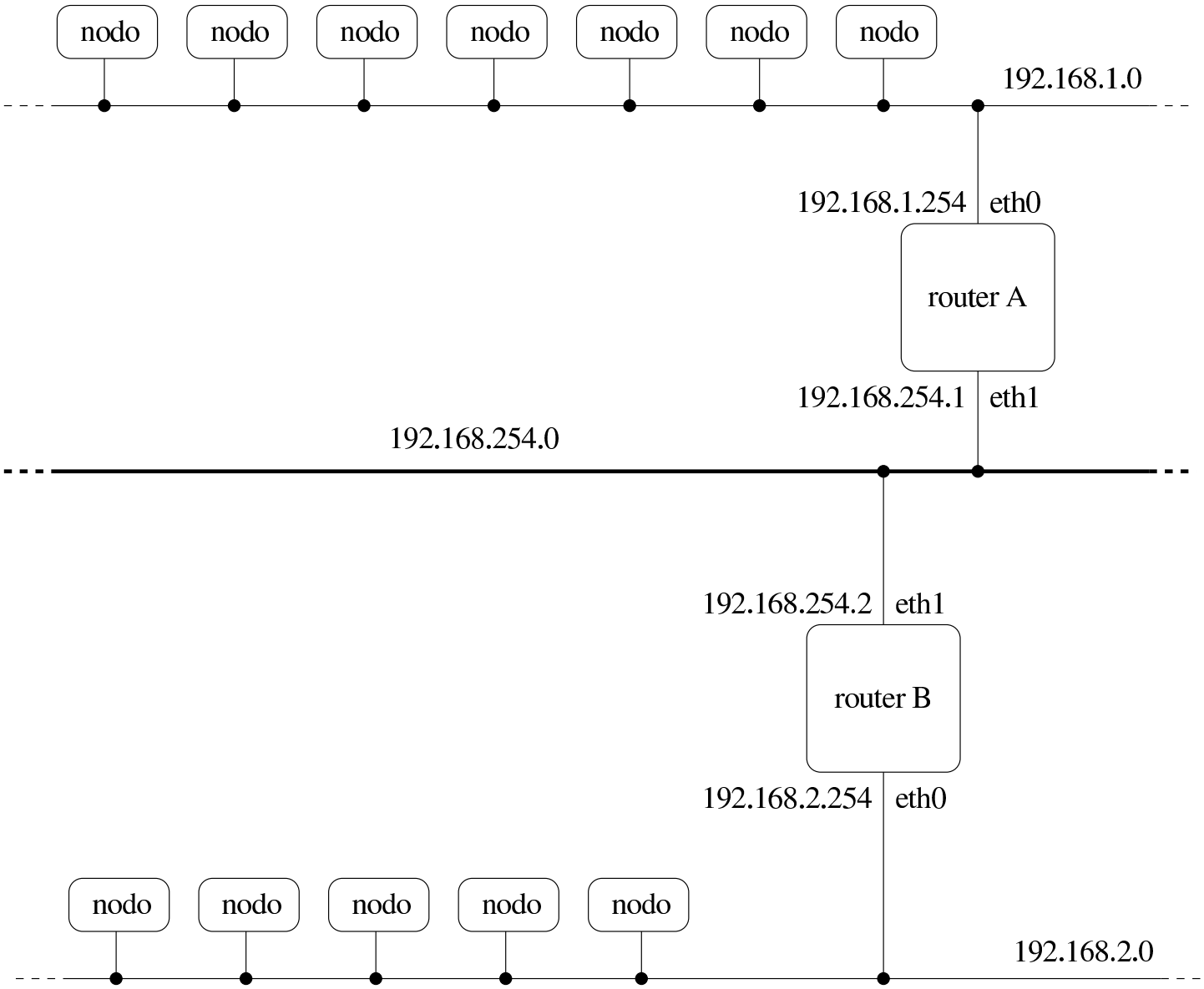
«

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.

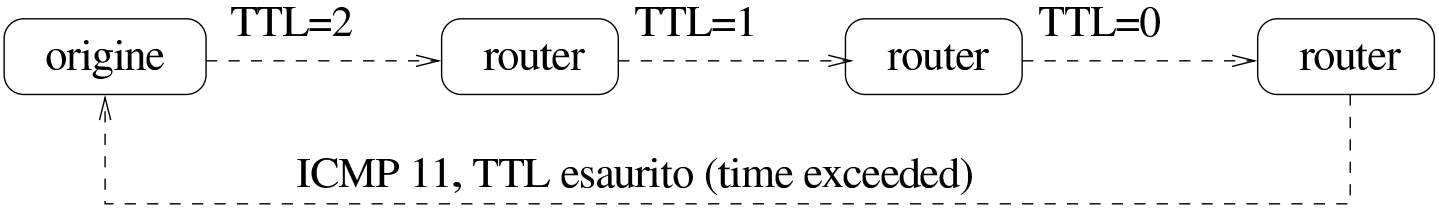
```
# route add -net 192.168.2.0 netmask 255.255.255.0 ↩
↪       dev eth0 [Invio][24]

# 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.

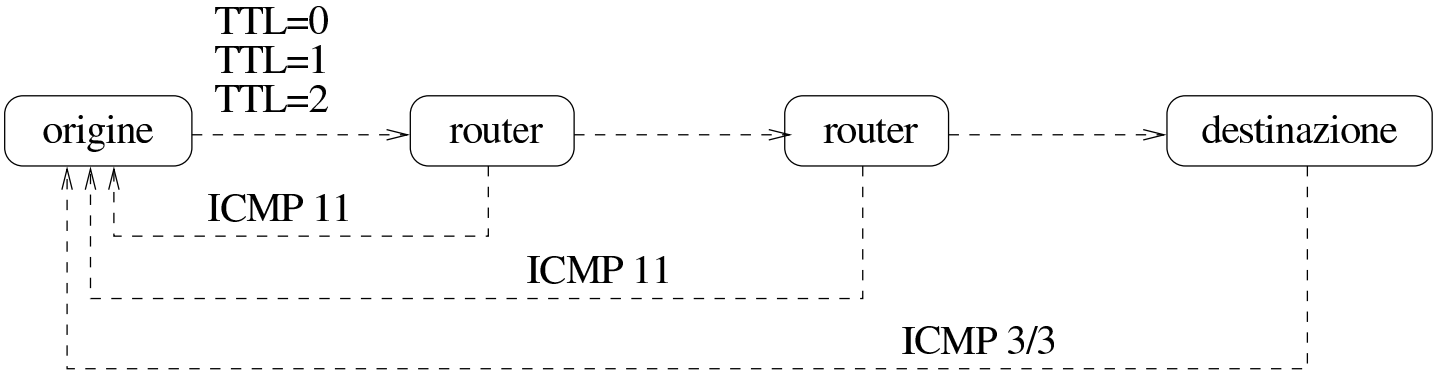
Per verificare quale sia il percorso utilizzato effettivamente dai pacchetti per raggiungere una destinazione, si utilizza Traceroute,[26] a cui corrisponde l'eseguibile '**traceroute**' che si usa con la sintassi seguente:

```
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.

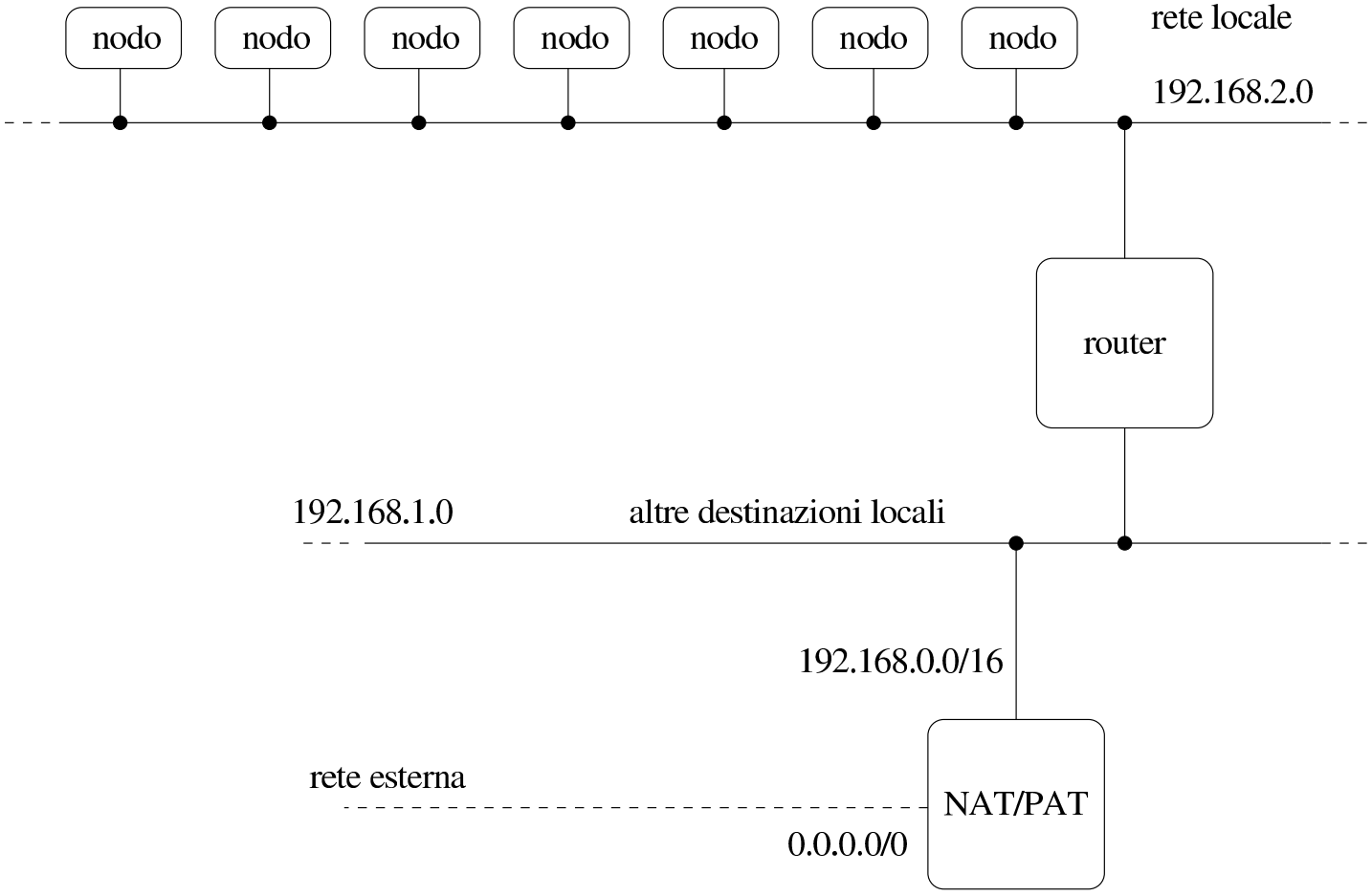
Tabella 32.167. Alcune opzioni per l'eseguibile '**traceroute**'.

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;

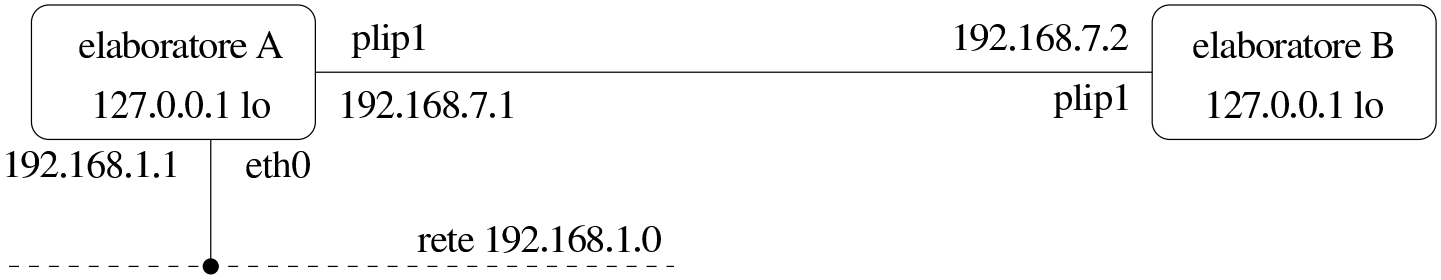
Parola chiave	Descrizione
`address`	si fa riferimento a un indirizzo IPv4 o IPv6, in base al contesto;
`neighbour`	fa riferimento alla tabella ARP (IPv4) o NDISC (IPv6);
`route`	interviene nella tabella degli instradamenti;
`rule`	fa riferimento a una regola nella politica degli instradamenti;
`maddress`	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	--
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.

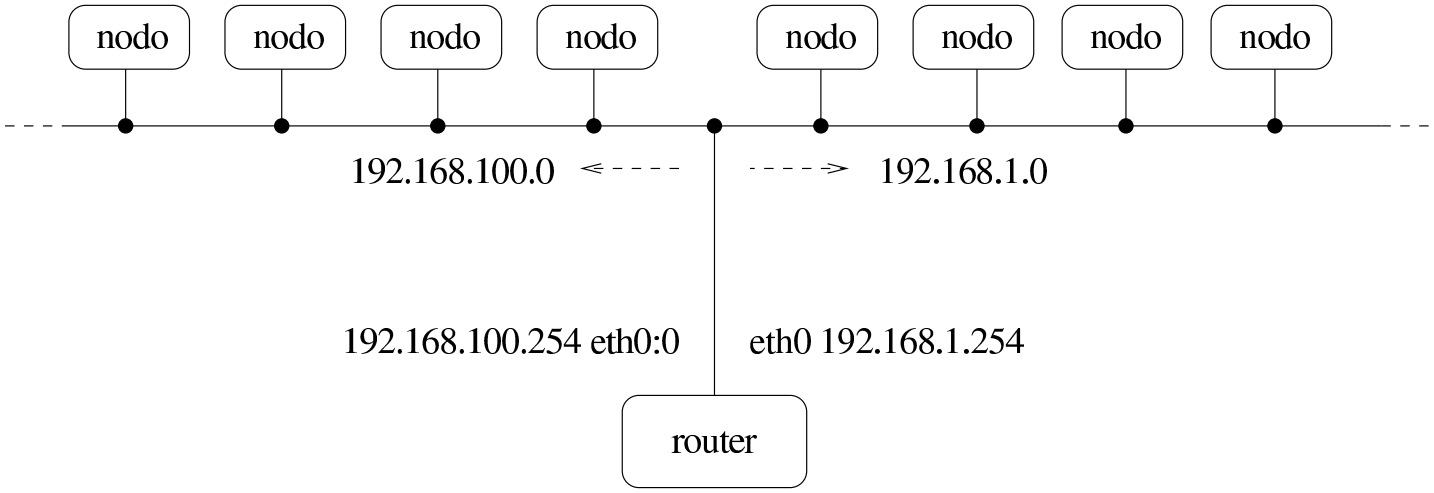
32.13.3 Indirizzi multipli per una stessa interfaccia di rete

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]`

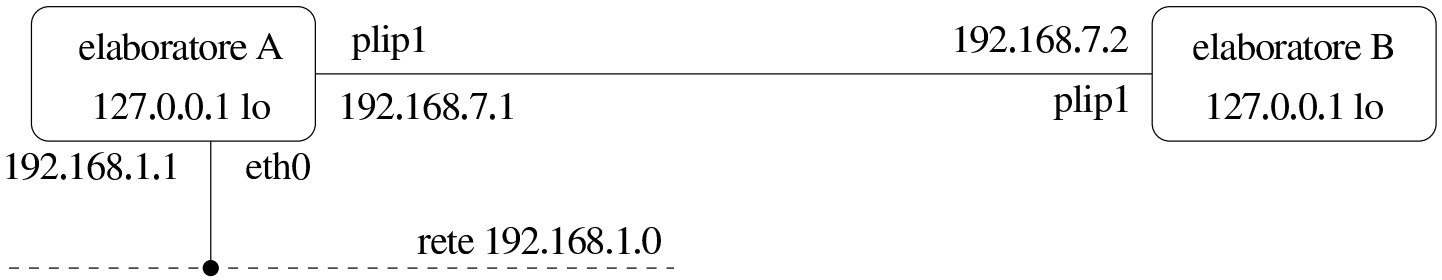
Cancella tutte le voci riferite all'interfaccia di rete '**eth0**', contenute nella tabella ARP.

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:

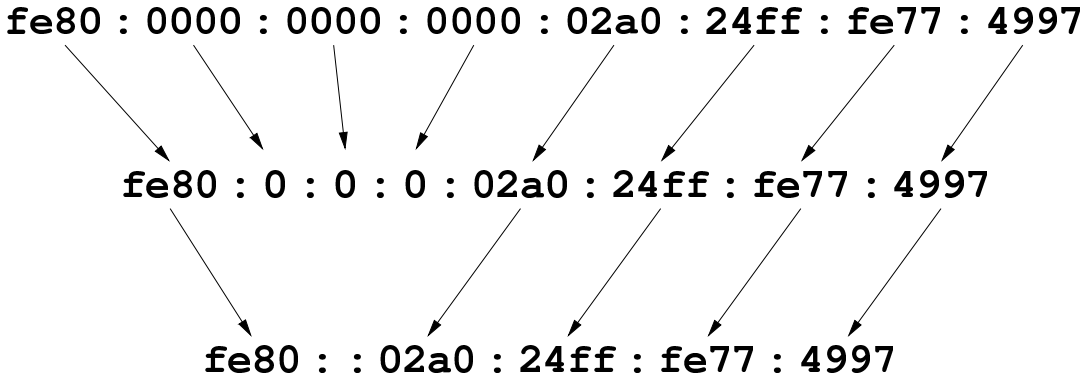
```
fe80:0:0:0:2a0:24ff:fe77:4997
```

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.

Prefisso binario	Prefisso esadecimale	Allocazione
$1111\ 10_2$	$F8_{16}..FB_{16}$	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.

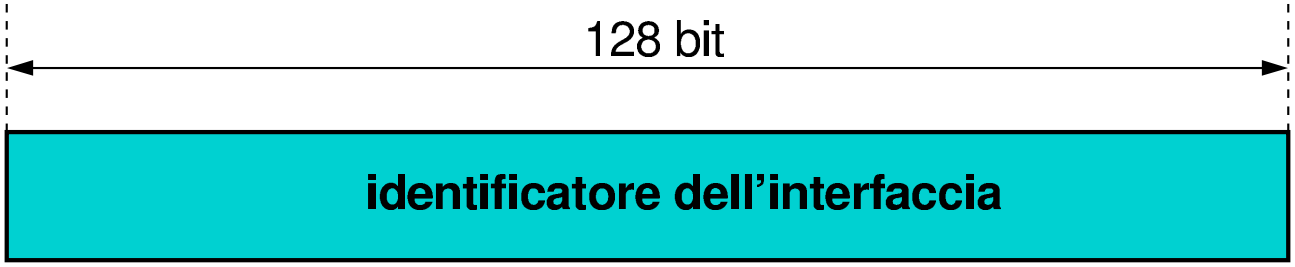
32.14.5 Indirizzi unicast

«

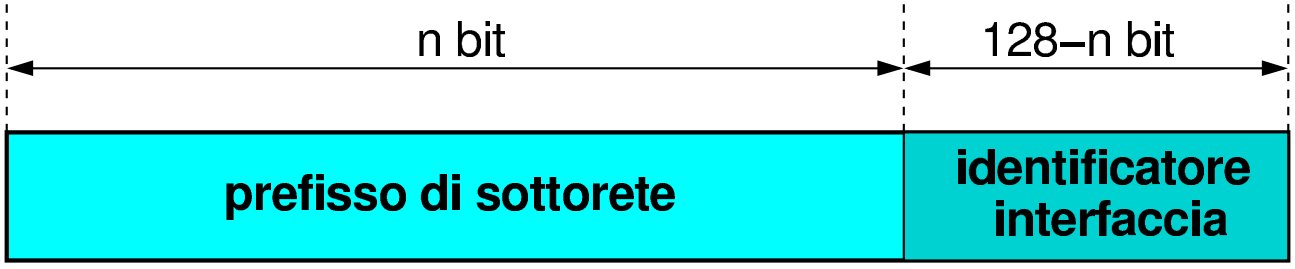
Si è accennato al fatto che tutti gli indirizzi, tranne quelli che iniziano per FF_{16}, sono di tipo unicast (e poi eventualmente tra questi si possono definire degli indirizzi anycast).

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 consentireb-

be 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:

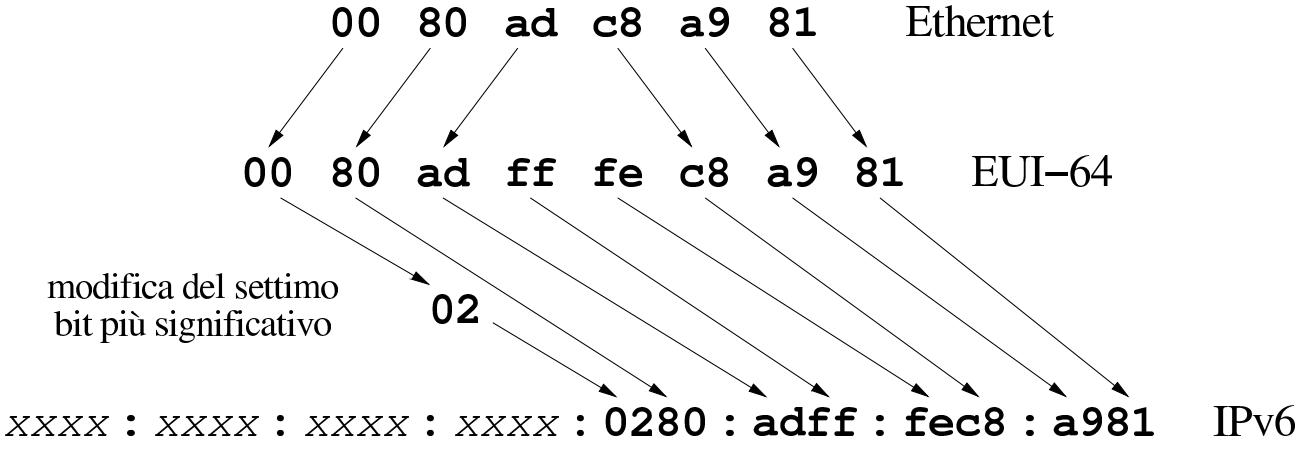
```
00-80-ad-ff-fe-c8-a9-81
```

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.

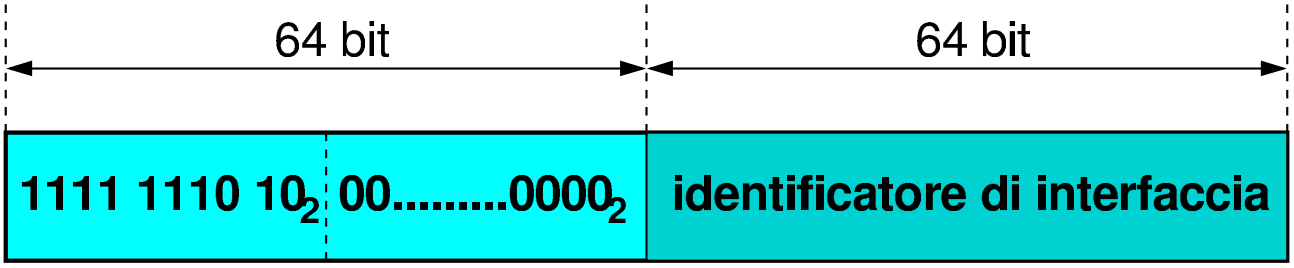
In pratica, un pacchetto destinato a questo indirizzo non deve uscire al di fuori del nodo (nella rete fisica esterna); inoltre, un pacchetto destinato a un altro nodo non può indicare come mittente questo indirizzo.

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**':

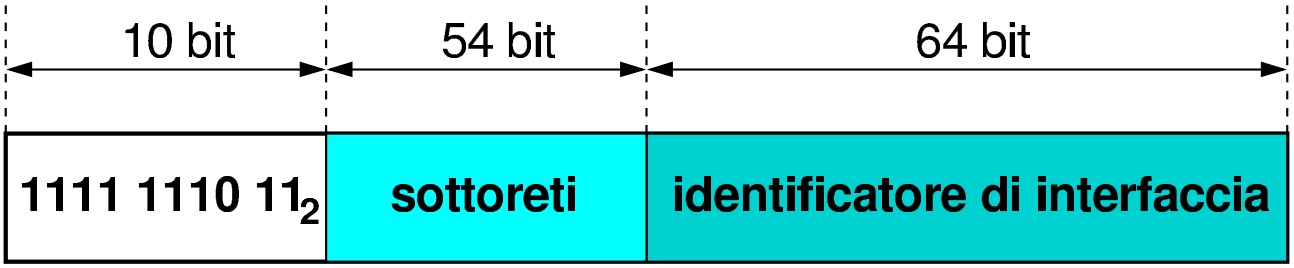
```
eth0      Link encap:Ethernet  HWaddr 00:80:AD:C8:A9:81
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::280:adff:fec8:a981/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 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:



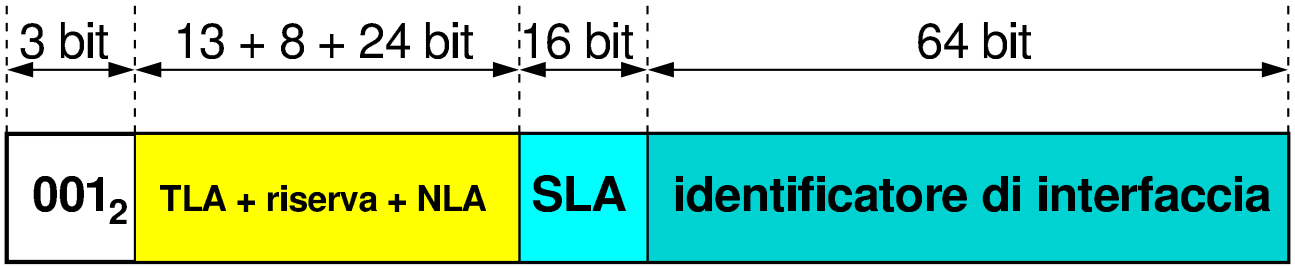
I primi 10 bit servono a definire il formato dell'indirizzo, stabilendo che si tratta del tipo site-local; lo spazio tra l'undicesimo e il 64-esimo bit può essere utilizzato per strutturare gli indirizzi in sottoreti, in base alle esigenze del sito. La seconda metà dell'indirizzo viene riservata per 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:

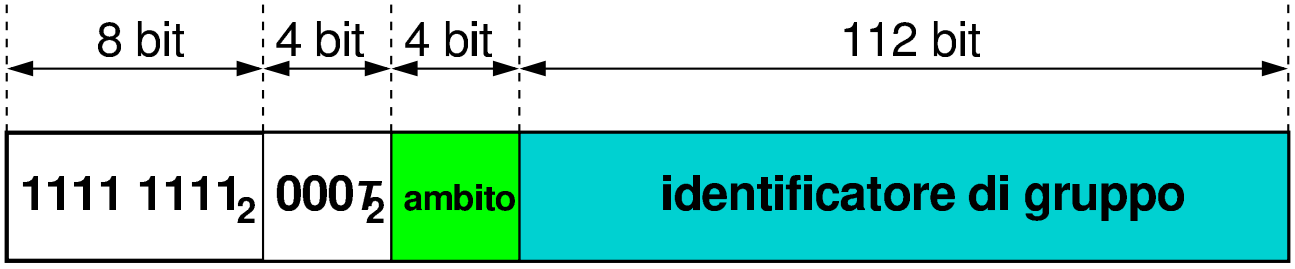


Dopo il prefisso di formato seguono 45 bit suddivisi in: un identificatore del primo livello di aggregazione (*Top level aggregation*), uno spazio di riserva e un identificatore successivo (*Next level aggregation*). Subito dopo seguono altri 16 bit la cui gestione dovrebbe essere affidata a un solo 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 me-

no 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.

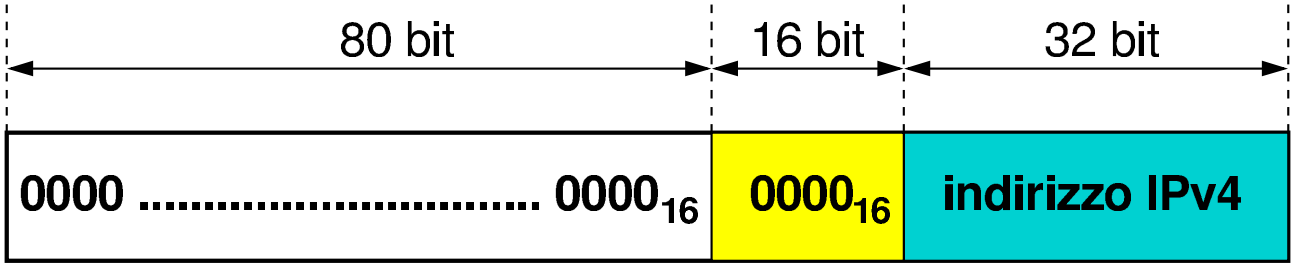
L'indirizzo anycast più comune è quello che serve a raggiungere simultaneamente tutti i router nell'ambito link-local. Si tratta precisamente del *Subnet router anycast address*, il quale si ottiene azzerando la parte di indirizzo che segue il prefisso. Per esempio, in una rete 3ffe:ffff:1:2:3:4:5:6/64, 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:

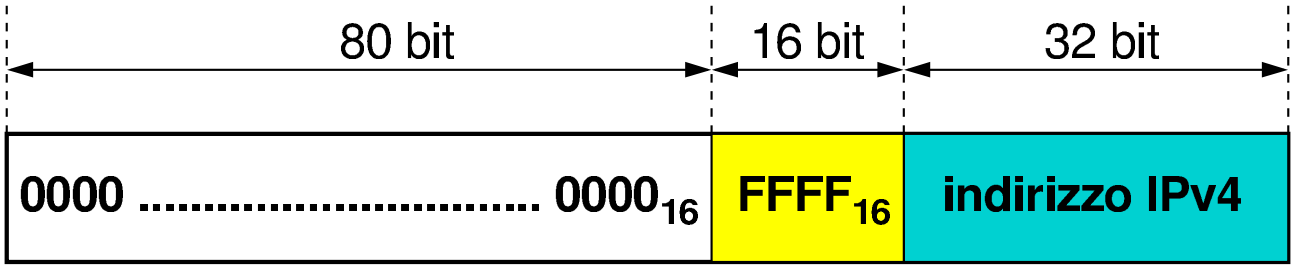


In presenza di indirizzi di questo tipo, è ammessa una notazione speciale, in cui la parte finale dell'indirizzo si indica secondo le convenzioni di IPv4. Nel caso di 192.168.1.1, si scrive:

```
::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:



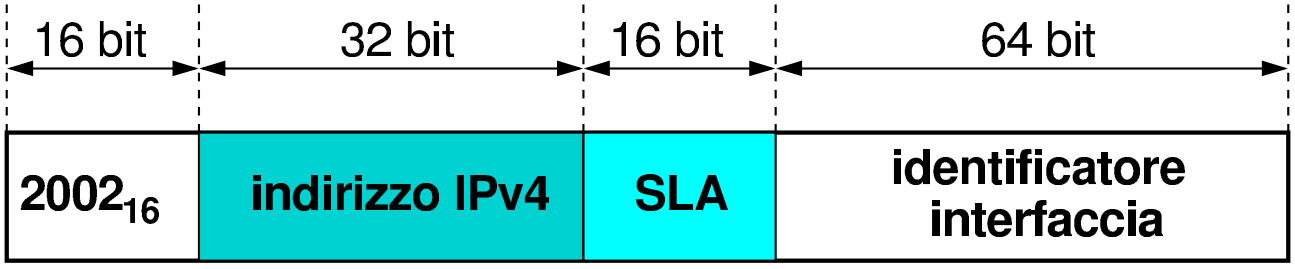
Anche in questo caso, in presenza di tali indirizzi è ammessa una notazione semplifica derivante da IPv4. Nel caso di 192.168.1.1, si scrive:

```
::ffff:192.168.1.1
```

32.14.8.3 6to4

«

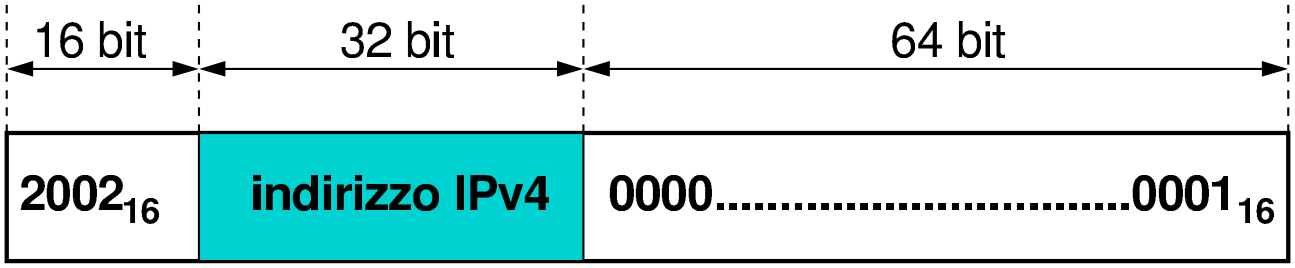
Il documento RFC 3056 *Connection of IPv6 Domains via IPv4 Clouds* descrive un insieme di indirizzi IPv6, di tipo unicast globale aggregabile, contenenti un indirizzo IPv4. Si fa riferimento a questo tipo di indirizzo e al meccanismo che ne sta dietro con la sigla 6to4. Semplificando le cose, l'indirizzo 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
```

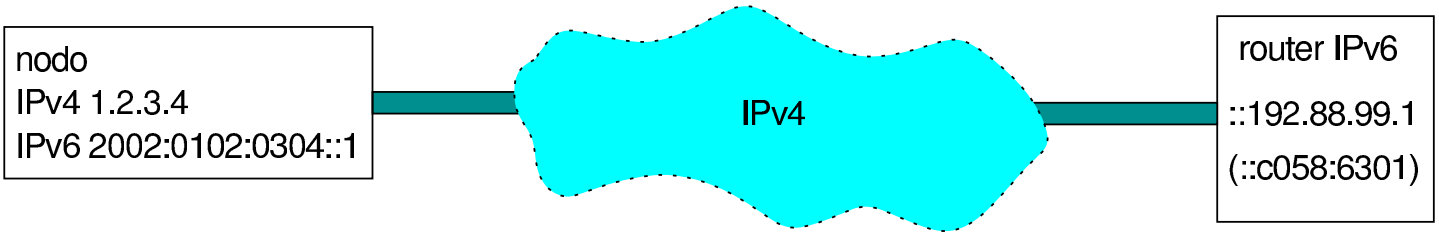
32.14.9 Tunnel 6to4

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 particola-

ri; 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:

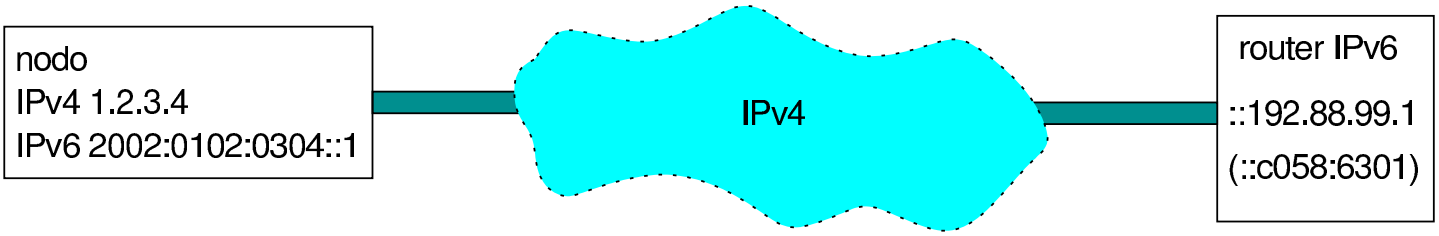
```
# echo 1 > /proc/sys/net/ipv6/conf/interfaccia/accept_ra [Invio]
```

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: cancellan-

do l'instradamento; disattivando l'interfaccia virtuale del tunnel; eliminando il tunnel. Ecco come:

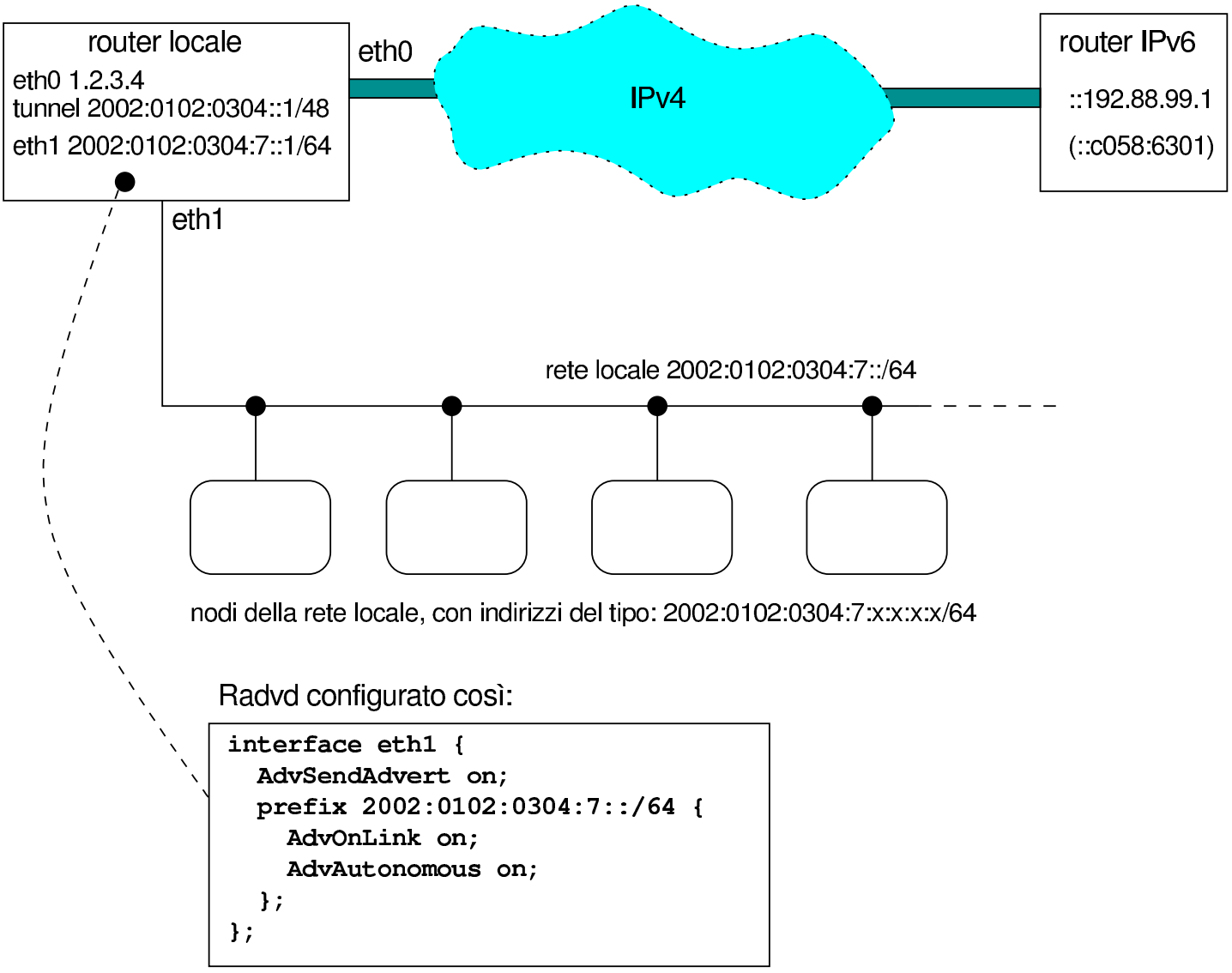
```
# ip -6 route flush dev t6to4 [Invio]

# 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,

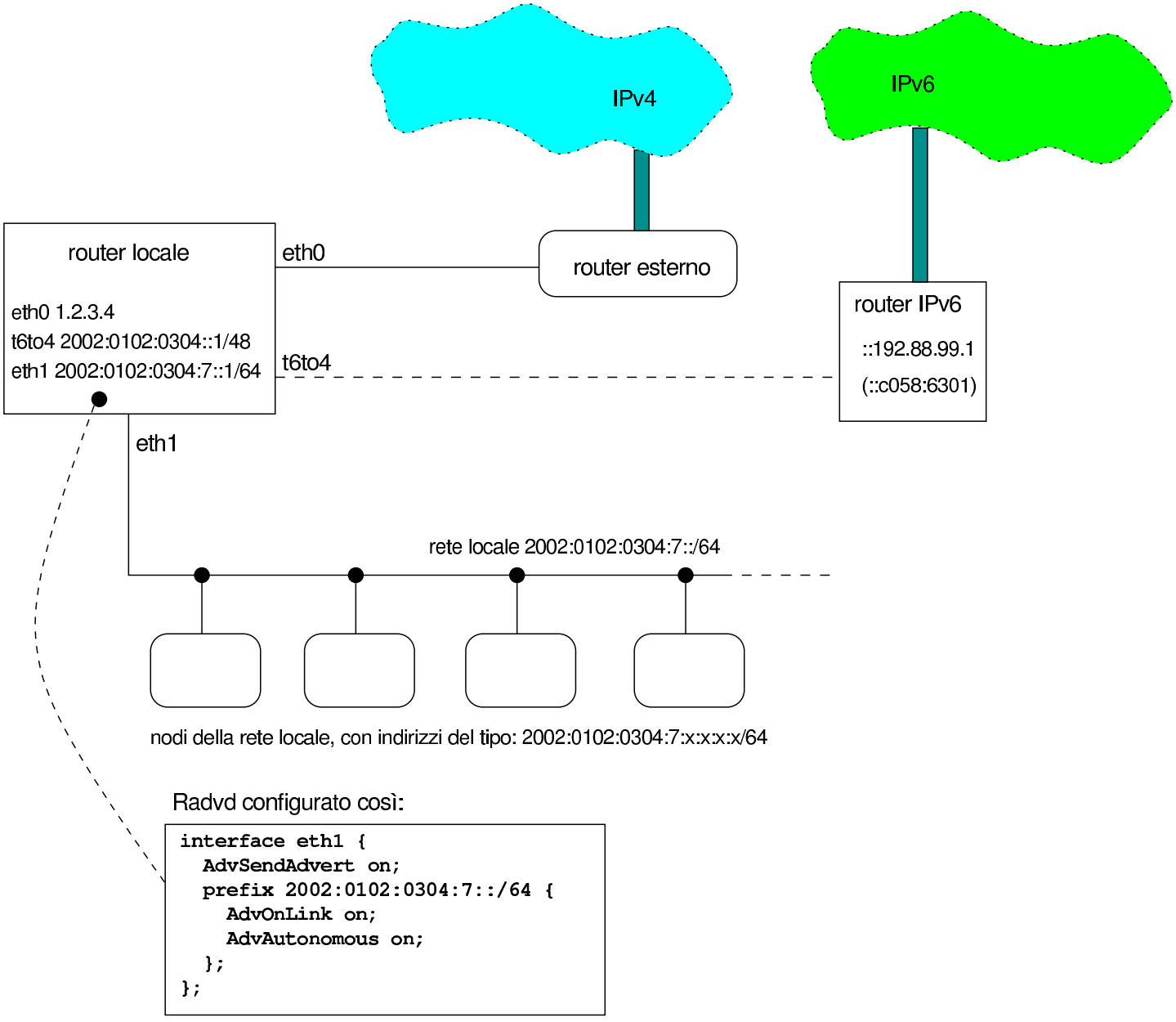
potrebbe impedire a Radvd di assegnare all'interfaccia '**eth1**' del router locale un indirizzo appartenente alla rete 2002:0102:0304:7:∗; inoltre, il fatto stesso che il nodo sia un router, impedisce l'attribuzione automatica dell'indirizzo (si veda la nota nel riquadro 32.245). Pertanto, è bene intervenire 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]
```

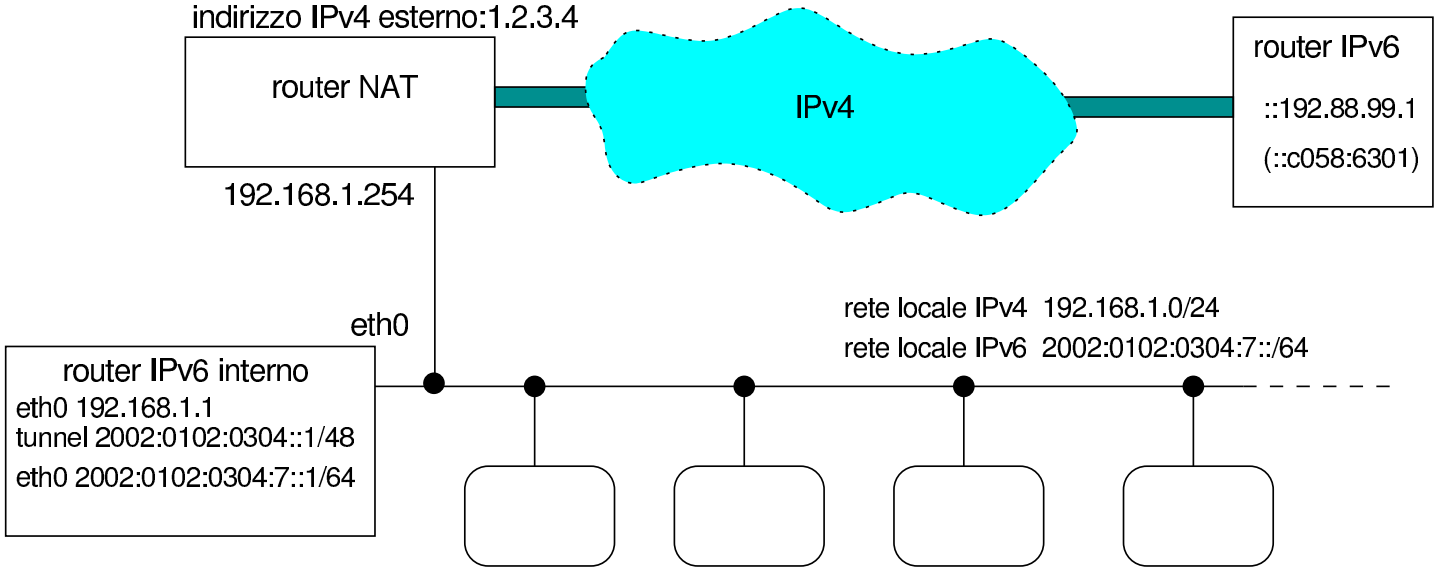
Figura 32.253. Situazione virtuale dopo la configurazione e l'instradamento attraverso il tunnel.



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.

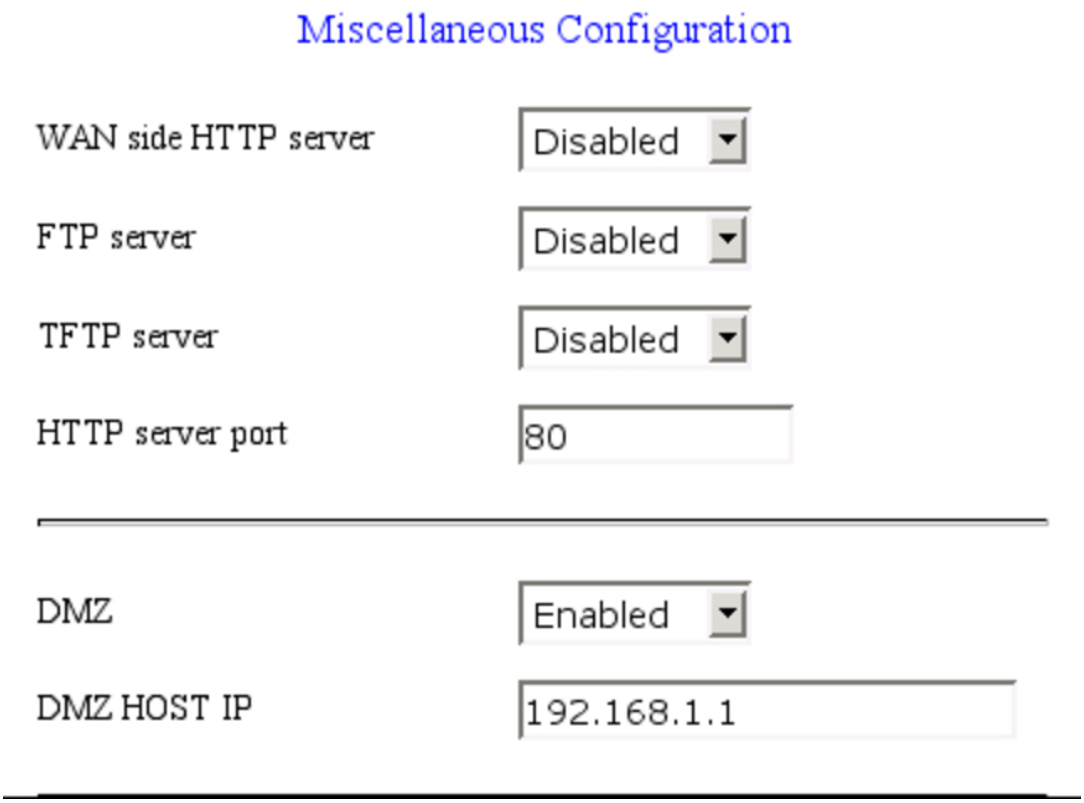
Prima di poter spiegare come si realizza il tunnel in questo caso, è necessario comprendere come si comporta il router IPv4. I pacchetti del tunnel hanno il numero di protocollo 41, come si può leggere nel file '`/etc/protocols`' di un sistema Unix comune:

```
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, per-

tanto 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 appe-

na 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
#
```

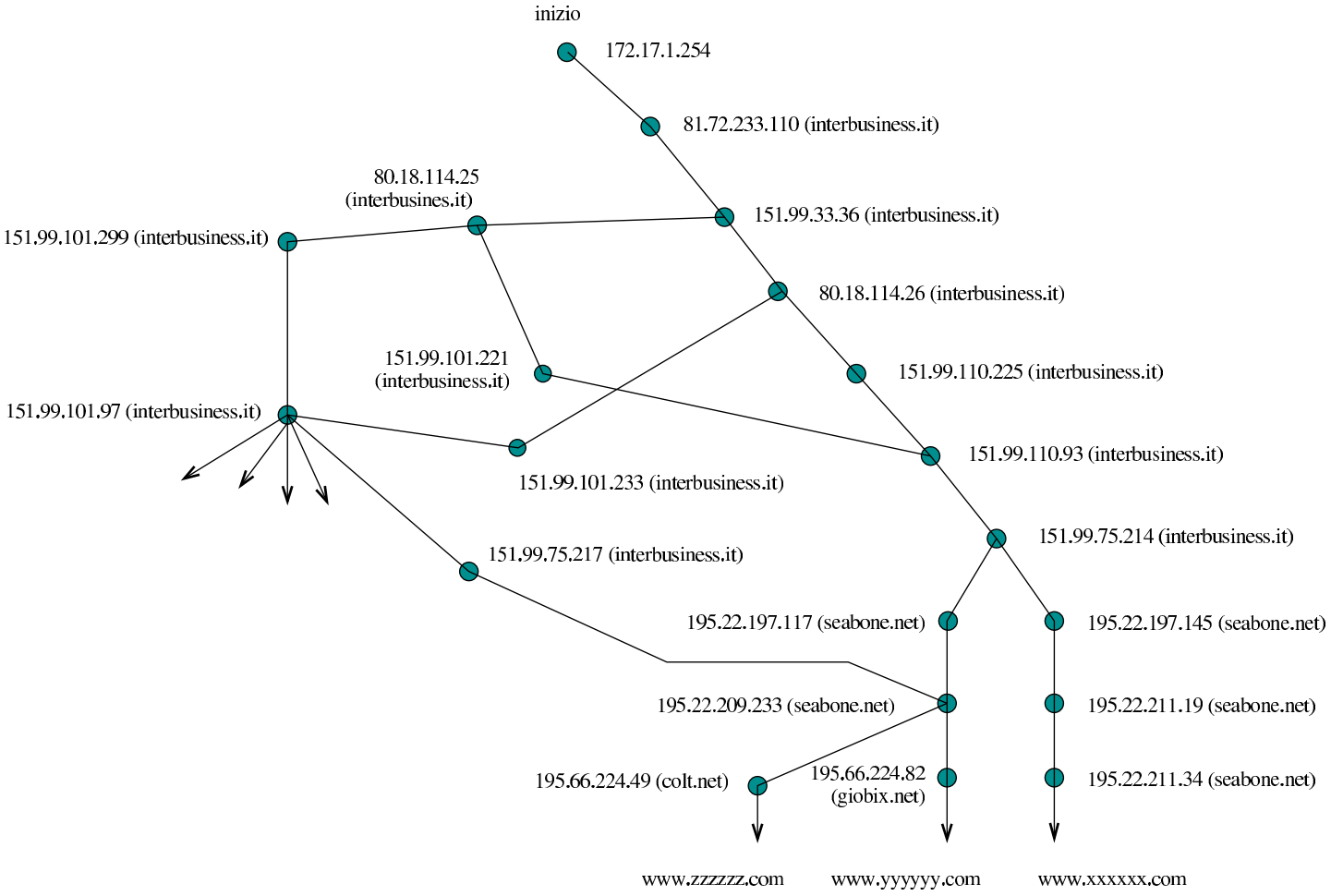
```
iptables  -t filter -A INPUT -p tcp  -s 0/0 -m state \
          --state ESTABLISHED,RELATED -j ACCEPT
iptables  -t filter -A INPUT -p udp  -s 0/0 -m state \
          --state ESTABLISHED -j ACCEPT
#
ip6tables -t filter -A INPUT -m state \
          --state ESTABLISHED,RELATED -j ACCEPT
```

32.16 Esercitazioni

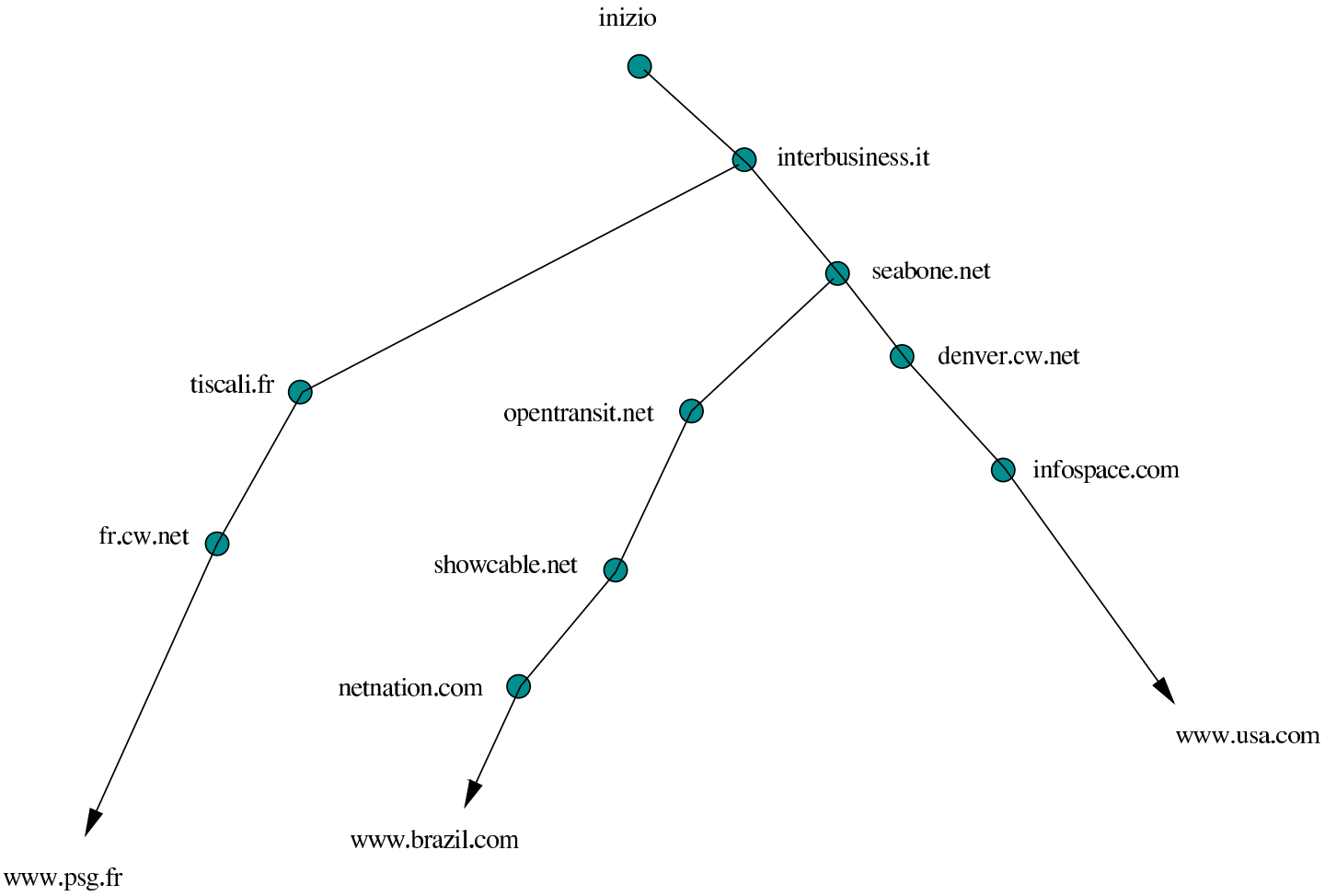
«

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.