Introduzione alle reti

2004.03.09

Sergio Berdondini Luca Diciotti
Alessio Frusciante Daniele Masini

09 Marzo 2004

- Pagine totali: 69 -

Indice

Elenco delle figure

Capitolo 1

Introduzione

Se due sistemi vogliono comunicare tra loro, devono attuare dei meccanismi in grado di portare un insieme di informazioni dall'uno all'altro senza alcuna perdita. Due sistemi non comunicano direttamente tra loro ma lo fanno attraverso le loro *interfacce di rete*, ovvero un'interfaccia di rete di un sistema comunica con un'interfaccia di rete di un altro sistema.

Un'applicazione a sua volta utilizzerà un'interfaccia di rete allo scopo di comunicare con un'altra interfaccia di rete. Un sistema può essere dotato di una o più interfacce di rete, quindi se due sistemi comunicano tra loro è perché lo fanno le loro relative interfacce di rete. È sempre bene tenere a mente il fatto che l'elemento di riconoscimento del mittente e del destinatario dell'informazione da comunicare è l'interfaccia di rete e non il sistema.

La modellizzazione della comunicazione via rete è stata effettuata quando i sistemi già utilizzavano la rete per comunicare e tutte le modellizzazioni che sono state fatte suppongono di suddividere l'accesso alle funzionalità della comunicazione a vari livelli di astrazione a partire da quello più basso che è quello dei segnali coinvolti nell'effettiva comunicazione (la cui natura e tipologia dipendono dalla natura del mezzo di trasmissione) fino a quello più elevato che deve "vedere" l'applicazione che vuole comunicare. L'applicazione passerà i dati da inviare al livello sottostante che si preoccuperà di processarli correttamente e quindi li passerà al livello a sua volta sottostante e così via fino ad arrivare al livello più basso in cui i segnali logici saranno trasformati in segnali elettrici o elettromagnetici e quindi inviati sulla rete. Allo stesso modo, l'interfaccia di rete che riceve i dati processerà le informazioni arrivate man mano ad un livello sempre più elevato fino ad arrivare all'applicazione. Ogni livello passa le informazioni al suo adiacente tramite un insieme di funzioni che costituiscono l'*interfaccia* di comunicazione tra un livello e l'altro.

1.1 Il modello ISO/OSI

Il modello che è stato la pietra miliare nella definizione di un'interfaccia di rete, ma di cui non esiste nessuna implementazione pratica, è il modello OSI (Open System Interconnection) proposto da ISO che si suddivide nei 7 livelli di figura 1.1

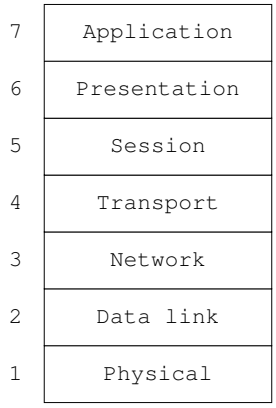

Figura 1.1: Il modello ISO/OSI

Il livello fisico (**1. Physical**) è sotto il dominio dell'ingegneria più che dell'informatica. Sulla base del mezzo di trasmissione utilizzato vengono individuate la natura e le caratteristiche del segnale da utilizzare per la trasmissione dei dati (modulazione/demodulazione, conversione digitale/analogica e viceversa, ...). Questo livello si preoccupa di trasformare i dati da inviare negli appositi segnali elettrici o elettromagnetici che vengono fisicamente inviati sulla rete e di convertire i segnali ricevuti negli equivalenti dati digitali.

Il livello di collegamento (**2. Data link**) gestisce la metodologia di acceso al mezzo trasmissivo. Stabilisce se la trasmissione può essere iniziata o meno (per esempio controlla se c'è già qualche altro sistema che sta già impegnando il mezzo di comunicazione). Inoltre si preoccupa che le informazioni arrivino a destinazione in maniera corretta, aggiungendo alle informazioni da trasmettere dei codici di controllo di basso livello.

Il livello di rete (**3. Network**) ha il compito di attuare la strategia di instradamento per far giungere i dati a destinazione. Marca i dati con l'identificativo del mittente e del destinatario. Ogni interfaccia di comunicazione di rete che riceve dei dati confronta l'identificativo del destinatario con il proprio, in modo da riconoscere se i dati sono destinati a lei. In caso affermativo passa i dati al livello superiore, altrimenti scarta il messaggio, ignorandolo[1].

[1] Una scheda di rete può essere anche impostata in modalità promiscua, il che le consente

Il livello di trasporto (**4. Transport**) si preoccupa di scomporre lo stream di dati da inviare in pacchetti più piccoli per occupare la rete in maniera intelligente (senza renderla inaccessibile agli altri sistemi magari per qualche minuto se la quantità di dati da trasmettere è elevata). Viceversa si deve anche preoccupare di ricostruire lo stream di dati a partire dai pacchetti ricevuti nel corretto ordine con cui sono stati inviati. Quindi in ogni pacchetto saranno inserite informazioni addizionali per identificare l'ordine di invio di un pacchetto rispetto agli altri. Si preoccupa inoltre di gestire errori di comunicazione (inserimento e controllo di checksum) ed implementa i meccanismi necessari per richiedere quello che eventualmente è andato perduto nella comunicazione.

Il livello di sessione (**5. Session**) ha la responsabilità di gestire la comunicazione, ovvero implementa il protocollo al livello più elevato: si preoccupa di assicurarsi che l'interfaccia di comunicazione di rete a cui si vogliono spedire i dati sia pronta a riceverli (inviando un segnale tipo un ping) ed alla fine della comunicazione manda un segnale di chiusura (tipo un altro ping) in modo che il destinatario sappia che la comunicazione è terminata e possa effettuare le operazioni opportune (liberazione di risorse di memoria dedicate alla comunicazione, ...).

Il livello di presentazione (**6. Presentation**) ha la responsabilità di gestire la la conversione dei dati da inviare secondo lo standard di codifica del sistema di destinazione. I sistemi che devono comunicare possono essere diversi (per esempio *Windows* e *Unix*), quindi i dati potrebbero avere codifiche diverse (es. ASCII e EBCDIC). A questo livello avviene la conversione nella codifica opportuna. In realtà non esistono implementazioni di questo livello, quindi la codifica deve essere fatta a livello di applicazione.

Il livello di applicazione (**7. Application**) è appunto il livello da cui le applicazioni "vedono" la rete, ovvero il livello più astratto. Un'applicazione deve soltanto indicare quali informazioni inviare ed a chi inviarle.

L'applicazione che desidera inviare un certo stream di dati ad un'altra applicazione presente su di un altro sistema, connesso in rete tramite un'opportuna interfaccia di rete, invierà lo stream dal livello 7 al livello 6 e per lei il gioco è fatto (non si preoccupa di ciò che avviene dopo) L'applicazione "vede" soltanto l'interfaccia con il livello 6 e nient'altro. In realtà lo stream da inviare viene processato ad ogni livello dal 6 fino all'1 che si preoccupa effettivamente di inviarlo sulla rete. Dall'altra parte, l'interfaccia di rete ricevente riceve le informazioni trasmesse e queste passeranno dal livello 1 al livello 7 dopo opportuni controlli e ricostruzioni, per fornire così lo stream inviato all'applicazione ricevente.

Sarebbe bello avere la modularità ad ogni livello, ovvero sostituire un'implementazione di un livello con un'altra, senza che gli altri livelli se ne accor-

---

di far passare ai livelli superiori tutti i pacchetti che arrivano al suo connettore, anche se non destinati a lei (questa modalità è utilizzata dagli *sniffer*).

gano. Ma ciò dipende fortemente dall'implementazione dei livelli e quindi è praticamente impossibile. Comunque, in genere la modularità c'è a livello 2 il che significa che si può sostituire l'implementazione dei livelli 1 e 2 con un'altra senza che i livelli superiori se ne accorgano. Ovviamente si può sostituire l'implementazione dei livelli 3-6 con un'altra senza che i livelli sottostanti se ne accorgano, ma se ne accorge l'applicazione che dovrà usare un'interfaccia di comunicazione con il livello 6 diversa.

1.2 Lo stack TCP/IP

Per le reti ci sono stati vari standard di protocolli di comunicazione che ovviamente hanno portato a problemi di interconnessione tra reti diverse. Quello che oggigiorno prevale (e sarà destinato a farlo sempre di più) è il TCP/IP (che sarebbe meglio chiamare *internet suite*), una suite di protocolli nata con internet (anni '70) ovvero per le reti geografiche, poco affidabili rispetto alle LAN, ma che ha buone prestazioni anche su LAN (sebbene su LAN esistano dei protocolli più efficienti in termini di rapporto tra i codici di controllo e le effettive informazioni da trasmettere). Lo stack TCP/IP è composto dai 5 livelli di figura 1.2.

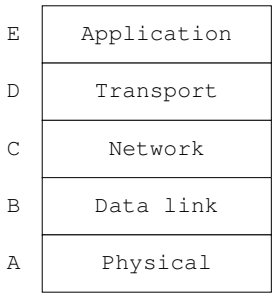

Figura 1.2: Il modello TCP/IP

I livelli A, B e C si mappano esattamente sui livelli 1, 2 e 3 del modello OSI, mentre il livello D ingloba i livelli 4 e 5 del modello OSI ed il livello E ingloba i livelli 6 e 7 del modello OSI.

L'unità logica dei dati che transitano ad un determinato livello assume un nome diverso che dipende appunto dal livello. Ogni unità di informazione che transita a livello B è detta *frame*, ogni unità che transita a livello C è detta *datagram* ed ogni unità che transita a livello D è detta *packet*.

Gli standard che specificano i protocolli dello stack TCP/IP sono in generale pubblicati attraverso il meccanismo delle Request For Comments (RFC) [2]. Una RFC è un documento che viene presentato alla Internet Engineering Task Force (IETF), che viene pubblicato per ricevere commenti. Una RFC che aspira a divenire uno standard passa attraverso 3 stadi, detti

livelli di maturità. Il primo livello è Proposed Standard, il secondo è Draft Standard, il terzo è Standard.

Lo stato dei protocolli è controllabile in un documento chiamato "Internet Official Protocol Standards" [22].

1.3 Altri stack di protocolli

Esistono altri stack di protocolli, in buona parte sviluppati prima dell'esplosione di internet e quindi del TCP/IP. Ad esempio, IBM ha sviluppato SNA, Novell usa SPX/IPX e Xerox aveva sviluppato alla fine degli anni settanta XNS. Tutti questi stack hanno qualche parentela con il modello ISO/OSI, o lo hanno in parte ispirato.

Capitolo 2

Il livello fisico e di collegamento

Un *protocollo* è un insieme di regole di comunicazione relative ad un determinato livello del modello di interfaccia di rete, tra sistemi diversi (v. figura 2.1). Un protocollo è quindi una logica che fa comunicare due interfacce di rete a livelli diversi. Se ogni livello fa correttamente il proprio dovere, si può pensare che la comunicazione avvenga tra le due interfacce di rete al livello che ci interessa, sebbene la comunicazione avvenga in realtà sempre al livello più basso (1).

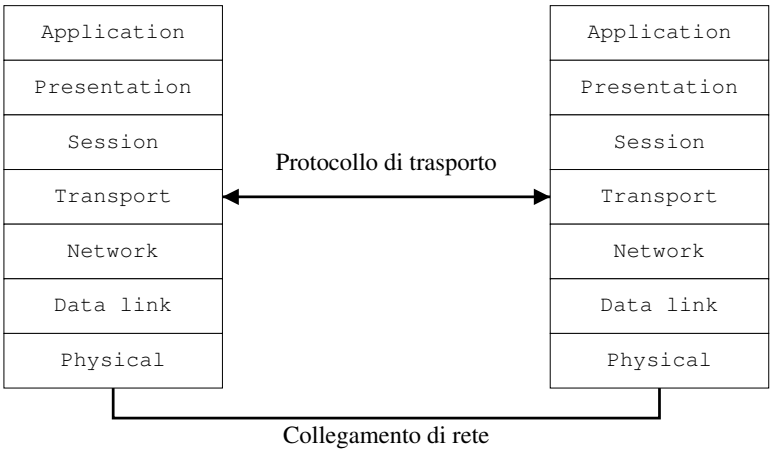

Figura 2.1: Esempio di protocolli

Una *scheda di rete* è l'implementazione di un'interfaccia di rete. A bordo della scheda vi è la circuiteria necessaria per l'implementazione del livello fisico (*transceiver*) e gran parte del livello di collegamento. Sebbene i protocolli di collegamento non siano necessariamente collegati a quelli del livello fisico, di fatto ogni scheda di rete implementa entrambi i livelli (a meno del

driver per la gestione del livello di collegamento). Ecco quindi perché si è soliti parlare di schede di rete Ethernet o schede di rete token ring.

Vediamo adesso alcune caratteristiche dei livelli più bassi dello stack ISO/OSI, ossia il livello fisico e il livello data link. I mezzi trasmissivi più diffusi sono i cavi coassiali, i doppini intrecciati UTP (Unshielded Twisted Pair) o STP (Shielded Twisted Pair), le fibre ottiche. Esistono comunque reti che si basano sulla trasmissione di onde radio o su mezzi ancora più esotici.

La topologia fisica di una rete è la conformazione delle linee di trasmissione, ossia la geometria dei cavi da una stazione all'altra. La topologia logica è invece il percorso che compiono i dati. In generale, non è detto che la topologia fisica e quella logica coincidano. Vediamo le topologie più diffuse:

Mesh Si tratta di una famiglia di topologie, più che di una topologia sola, ma in generale, ogni stazione è connessa direttamente con più stazioni, al limite tutte le altre. Viene utilizzata per i computer paralleli.

Bus Le stazioni sono connesse tutte ad un solo cavo.

Anello Le stazioni sono connesse tutte ad un cavo che si richiude ad anello.

Stella Le stazioni sono connesse ad un concentratore (hub) e comunicano attraverso di esso.

Poiché, a parte nel caso di mesh, i canali di comunicazione tra stazioni sono molto minori del numero di macchine, è necessario stabilire come viene gestito l'accesso al canale, ossia il processo chiamato CAM (Channel Access Method). I metodi principali sono due: la contesa e il token passing (ci sarebbe anche il polling, ma non credo esistano più reti che funzionano così).

Nel modello a contesa, le stazioni cercano di accedere al canale arbitrariamente e poi negoziano tra loro chi ha la precedenza. Lo svantaggio principale è che se il canale è molto trafficato la fase di contesa può essere lunghissima.

Nel modello a token passing, sul canale è sempre presente un messaggio chiamato token, che può essere in due stati: libero o occupato. Se ad una macchina arriva un token libero, può parlare (e deve cambiare lo stato ad occupato). Quando al destinatario arriva il messaggio, legge il messaggio e libera il token.

Il tipo di commutazione di una rete indica invece come vengono gestite le trasmissioni tra una stazione e l'altra. I due tipi principali sono:

Commutazione di circuito Tra la stazione A e la stazione B viene riservato un circuito, ossia un tratto di canale che le collega. Solo A e B

possono utilizzarlo, finché non si conclude la chiamata. Questo è il sistema con cui funziona la rete telefonica fissa classica (PSTN). Lo svantaggio più evidente è che la linea è occupata anche se A e B non stanno parlando.

Commutazione di pacchetto La stazione A condivide un canale con molte altre stazioni. Ogni stazione può inviare un messaggio avente lunghezza massima fissata. Chiaramente ci deve essere una fase in cui i messaggi più lunghi vengono suddivisi in spezzoni. Ogni stazione invia a turno un pacchetto, quindi si realizza una sorta di multiplexing. Lo svantaggio più grande è dato dal fatto che la banda a disposizione di una stazione dipende dal traffico sulla rete e non è facilmente predicibile.

2.1 Reti locali

2.1.1 Ethernet

La tecnologia più diffusa per reti locali è senz'altro **Ethernet**. Questa ha una modalità di accesso alla rete a contesa, nota come CSMA/CD (Carrier Sense Multiple Access / Collision Detection). Chiunque può impegnare il mezzo trasmissivo per iniziare una comunicazione se nessun altro sistema sta già comunicando (Multiple Access), non c'è alcun impedimento fisico, ma c'è soltanto un controllo logico (Carrier Sense). Dopo aver impegnato il mezzo di comunicazione si deve essere sicuri che nessun altro abbia deciso di impegnarlo quasi in contemporanea. Il protocollo allora si basa sull'ascolto di eventuali collisioni (Collision Detection). Se viene rilevata una collisione (due tentativi di comunicazione quasi contemporanea), la comunicazione viene interrotta per un tempo variabile casuale e quindi si riprova (il sistema che riparte per prima ha il diritto di comunicare). Questo approccio si porta dietro l'indeterminazione del tempo di trasmissione: non sarà possibile determinare a priori il tempo necessario per scambiarsi dei dati. Inoltre, a causa delle collisioni, c'è un limite al numero dei sistemi interconnessi con questo protocollo.

Ethernet II è lo standard di fatto utilizzato ad oggi per internet (che viene privilegiato rispetto allo standard IEEE 802.3, sebbene tutte le schede Ethernet gestiscano entrambi gli standard, leggermente diversi)[1]. Nella specifica Ethernet II i frame hanno la struttura in figura 2.2. Ogni frame è preceduto da un segnale, detto preambolo, che è una sequenza di bit che serve a sincronizzare il ricevitore.

[1] Affinché un'interfaccia di rete sia compatibile ad internet, deve essere in grado di trasmettere pacchetti Ethernet II ed in grado di ricevere pacchetti Ethernet II e IEEE 802.3.

Figura 2.2: Frame Ethernet II

All'inizio del frame sono presenti gli indirizzi del destinatario e del mittente. Gli indirizzi Ethernet sono a 48 bit, generalmente rappresentati come sei coppie di numeri esadecimali. Ogni scheda di rete ha il suo indirizzo, fornito dal produttore, che è univoco (non esistono due schede Ethernet al mondo con lo stesso indirizzo). La topologia fisica a bus fa sì che tutte le macchine ricevano tutti i pacchetti. Nel caso che l'indirizzo di destinazione non coincida con il proprio, il frame viene scartato e non viene passato ai livelli superiori. L'unica eccezione a questo comportamento è data da una modalità particolare della scheda, detta modo promiscuo, in cui si ascoltano tutti i frame. C'è un indirizzo particolare, FF:FF:FF:FF:FF:FF, che è il broadcast; nel caso che venga posto come indirizzo di destinazione, tutte le macchine si riterranno destinatarie.

Il campo tipo è un codice che indica quale protocollo è incapsulato dentro il frame. Questo serve in quanto il driver della scheda di rete dovrà passare il pacchetto al gestore dell'opportuno protocollo (demultiplexing). I tipi più importanti per lo stack TCP/IP sono 0800 (IP), 0806 (ARP), 8035 (RARP). Abbiamo poi un campo dati, che può contenere da 46 a 1500 byte e infine un campo FCS (Frame Check Sequence), che è una verifica sull'integrità dei dati. In generale, se si hanno errori di FCS significa che una scheda di rete non funziona bene. Si noti che la minima lunghezza di un frame Ethernet è 64 byte. Un frame più corto di 64 byte è detto runt e non è considerato valido.

2.1.2 Token ring

Il protocollo **token ring** è stato sviluppato da IBM e si trova principalmente all'interno dei suoi sistemi. È stato poi standardizzato da IEEE nella specifica 802.5. Si basa su un modello di accesso di tipo token passing, su una topologia logica ad anello ed una fisica a stella. Soltanto il sistema che ha ricevuto il token vuoto può iniziare una comunicazione inviando il token ed i dati da trasmettere al sistema adiacente, il quale riceverà il pacchetto, controllerà se i dati sono per lui ed in caso affermativo li preleverà rispedendo poi il token al sistema successivo. In caso negativo ripasserà il token più i dati al sistema successivo. I frame token ring sono di tre tipi diversi: il token, il frame di abort e il frame di dati/comandi.

Il throughput delle reti token ring può essere 4 Mb/s, 16 Mb/s o 100 Mb/s. La tecnologia token ring è in generale più costosa della Ethernet.

2.2 Wide Area Network

2.2.1 Protocolli punto-punto

I protocolli punto-punto servono per connettere tra loro due macchine, quindi non fanno a rigore parte dei protocolli di rete. In genere vengono usati per le connessioni dial-up attraverso modem. Esistono tre protocolli di questo tipo: **SLIP** (Serial Line Internet Protocol) [23], **CSLIP** (Compressed Serial Line Internet Protocol) [7] e **PPP** (Point to Point Protocol) [24]. Tali protocolli sono nati per implementare il livello di collegamento quando non esiste la scheda di rete vera e propria, simulandola tramite un collegamento via modem.

Lo SLIP è banale: mette un carattere di controllo in testa ed in coda al pacchetto da inviare e quindi controlla che nel pacchetto non vi siano altre occorrenze del carattere utilizzato per delimitare l'inizio e la fine del frame. Se vi sono occorrenze di tale carattere, ogni occorrenza viene fatta precedere dal carattere ESC e quindi tutte le occorrenze del carattere ESC vengono a loro volta raddoppiate. Quindi invia il frame sulla linea seriale. Ha alcuni difetti, tra cui:

- Ognuno dei due nodi deve conoscere a priori il proprio indirizzo IP.
- Trasporta solo il protocollo IP.

Il CSLIP è analogo allo SLIP ma è più efficiente perché se non è cambiato niente dall'ultimo frame, non ritrasmette l'header.

Il PPP risolve i problemi che aveva lo SLIP. È composto di due protocolli:

- Link Control Protocol (LCP), che stabilisce e configura la connessione. Ciò permette di negoziare varie opzioni tra i due nodi, tra cui ad esempio lo scambio di password. In particolare [25] sono previsti il Password Authentication Protocol (PAP) e il Challenge-Handshake Authentication Protocol (CHAP).
- Network Control Protocol (NCP), è una famiglia di protocolli (ci sarà l'NCP per l'IP, per AppleTalk, etc.), che specifica come viene trasportato il protocollo di livello superiore.

In ambiente *Windows* tutte le macchine possono avere client SLIP e PPP. Su *Windows NT Server* c'è il RAS che è un server PPP. Su *Linux* c'è anche un server SLIP.

2.3 Maximum Transmission Unit

La Maximum Transmission Unit di una rete è la lunghezza del campo dati nel protocollo relativo. Questa grandezza limita superiormente la lunghezza dei pacchetti del protocollo superiore. Alcune MTU sono[2]:

Token Ring 16 Mbit/s	17914
Token Ring 4 Mbit/s	4464
Ethernet	1500
PPP	¡ 1500

[2]La RFC relativa al il Path MTU discovery è la 1191.

Capitolo 3

I dispositivi di rete

I dispositivi di rete sono tutte le apparecchiature che rendono possibile la comunicazione in rete. Di seguito è riportato un elenco dei dispositivi di rete più comuni.

3.1 Hub

Un **hub** è un sistema che lavora a livello 1 e permette di centralizzare il cablaggio della rete. Quando la topologia fisica di una rete è a stella, il centro stella è in genere costituito da un hub. Dipendentemente dalla qualità dell'hub, questo può presentare anche una porta AUI che è uno standard per l'interfacciamento tra il livello 1 ed il livello 2. Pertanto a tale porta può essere collegato qualunque transceiver che permette di adattare l'hub a qualunque tipo di cablaggio di rete si desidera: connettore BNC, RJ, fibra ottica, ...

Alcuni tipi di hub per le reti Ethernet sfruttano un meccanismo, chiamato autonegoziazione, tramite il quale possono capire a che velocità può trasmettere una periferica trasmessa ad una loro porta e quindi gestire sia schede di rete a 10 Mb/s che a 100 Mb/s. Questi dispositivi sono detti switching hub.

3.2 Bridge

Un **bridge** è un sistema che lavora a livello 2 ed ha il compito di connettere tra loro due segmenti di rete. Ascolta i messaggi che provengono dalla rete A, scarta quelli non validi (ad esempio i runt), legge l'indirizzo di destinazione e, li inoltra solo se sono diretti sulla rete B. Lo stesso fa per i messaggi che provengono dalla rete A. In questa maniera il traffico nelle due reti rimane separato, per quanto possibile. Nel caso di Ethernet si dice che si hanno due domini di collisione separati.

I bridge Ethernet sono detti trasparenti, in quanto le due reti funzionano come una sola e le macchine non hanno bisogno di sapere che il bridge è presente. Infatti il bridge riesce a capire quali macchine sono sulla rete A e quali sulla rete B con un meccanismo chiamato backward learning. Quando la macchina 1 manda un messaggio, il bridge sente su quale porta arriva e quindi capisce che la macchina 1 sta sulla rete A. Dopo sufficiente tempo, ha una mappa completa della disposizione delle stazioni.

Un bridge può anche connettere due reti con tecnologie diverse (ad esempio Ethernet e token ring). In questo caso si parla di translational bridge, in quanto il bridge deve anche tradurre i pacchetti nel formato adatto.

3.3 Switch

Uno **switch** è un sistema che lavora a livello 2, simile ad un hub ma è dotato di intelligenza per ottimizzare la comunicazione sulla rete: memorizza in una sua cache il MAC address delle interfacce connesse ad ogni connettore in modo da far diminuire drasticamente le collisioni, in quanto un pacchetto inviato da un'interfaccia verso un'altra interesserà soltanto il connettore del destinatario, gli altri non si accorgeranno neanche della comunicazione tra i due[1].

3.4 Router

Un **router** è un sistema che svolge il compito di instradare i messaggi ricevuti, verso un'altra rete in modo da farli arrivare a destinazione. Essenzialmente è un sistema che lavora a livello 3 tra due o più reti diverse, in modo da riconoscere chi è il destinatario del pacchetto (e questo avviene proprio a livello 3, tramite l'indirizzo IP) e quindi instradarlo verso la rete più opportuna. La rete più opportuna può essere riconosciuta attraverso una logica descritta da una routing table che informa il sistema di inviare certi indirizzi su una certa rete e certi altri su di un'altra.

Un router ben configurato non dovrebbe routare il traffico broadcast (in modo da non sovraccaricare la rete) tranne quelli diretti ad una specifica applicazione (porta).

3.5 Gateway

Un **gateway** è un sistema che lavora a livello 7 ed ha il compito di effettuare operazioni più complesse di un router che sono quelle relative alla conversione

[1] Uno sniffer che gira su di una macchina collegata ad una LAN tramite switch riesce a tracciare soltanto il traffico broadcast (per tracciare tutte le comunicazioni si può comunque configurare lo switch in modalità hub).

di protocollo di sessione/trasporto. Per esempio potrebbe capitare di inviare un pacchetto da una rete secondo il protocollo TCP/IP ad un'interfaccia che sta su una rete che utilizza un altro protocollo. A questo punto, per far comunicare le due reti è necessario che vi sia un gateway che, oltre a instradare correttamente i pacchetti, effettui anche la necessaria conversione di protocollo.

Capitolo 4

Livello di rete

Come già accennato in precedenza, il protocollo per la comunicazione in rete che si sta diffondendo sempre più, grazie anche a internet, è il TCP/IP o meglio l'internet suite. Alla base di questo insieme di protocolli c'è l'Internet Protocol, abbreviato in IP. Attualmente è in uso la versione 4, ma è stata approntata una nuova versione, la numero 6, che per ora non ha un largo uso.

4.1 Le classi di indirizzi IP

Ogni interfaccia di rete è identificata, secondo IPv4, da un numero a 32 bit detto *indirizzo IP* che ovviamente deve essere univoco per ogni interfaccia presente su una determinata rete[1]. La rappresentazione di un indirizzo IP avviene in genere suddividendo il numero in notazione binaria in gruppi da 8 bit e rappresentando i valori decimali delle relative valori binari, ognuno separato dall'altro da un punto. Tale notazione è detta *dotted notation*. Per esempio l'indirizzo IP 12345678_{H} è rappresentato da 18.52.86.120 in dotted notation, infatti $18 = 12_{\mathrm{H}}$, $52 = 34_{\mathrm{H}}$, $86 = 56_{\mathrm{H}}$ e $120 = 78_{\mathrm{H}}$.

Ogni indirizzo IP è costituito da due parti: la prima (più significativa) è detta *parte rete* e la seconda (meno significativa) è detta *parte interfaccia*.

Gli indirizzi IP sono suddivisi in classi. La suddivisione è nata dall'esigenza di assegnare indirizzi IP univoci per le macchine connesse permanentemente ad internet (rete geografica mondiale). Tali classi permettono di fornire una gerarchia al valore che rappresenta un indirizzo IP. Per l'assegnamento dell'indirizzo IP per una macchina collegata permanentemente ad internet c'è l'organismo IANA (Internet Assigned Numbers Authority) che si preoccupa di garantirne l'univocità della parte rete (la parte interfaccia è a cura dell'amministratore dei singoli sistemi). In pratica, quando si richiede

[1] L'indirizzo IP è relativo ad una scheda di rete, non ad una macchina. Una macchina può montare anche più di una scheda di rete, ed in questo caso è detta multihomed.

un indirizzo alla IANA, viene assegnata un'intera rete, che poi verrà gestita dall'amministratore.

4.1.1 Classe A

Regola. L'indirizzo IP ha il bit 31 (MSB) posto a 0. I 24 bit meno significativi sono utilizzati per rappresentare la parte interfaccia, mentre i rimanenti 8 sono utilizzati per rappresentare la parte rete.

Quindi esisteranno al massimo $2^7 = 128$ reti di classe A (in quanto il bit più significativo è fissato a 0) ad ognuna delle quali potranno essere collegate al massimo $2^{24} = 16777216$ interfacce di rete. Tali valori saranno quindi compresi tra 0.0.0.0 e 127.255.255.255.

4.1.2 Classe B

Regola. L'indirizzo IP ha il bit 31 (MSB) posto ad 1 ed il bit 30 posto a 0. I 16 bit meno significativi sono utilizzati per rappresentare la parte interfaccia, mentre i rimanenti 16 sono utilizzati per rappresentare la parte rete.

Quindi esisteranno al massimo $2^{14} = 16384$ reti di classe B (in quanto i due bit più significativi sono fissati a 10) ad ognuna delle quali potranno essere collegate al massimo $2^{16} = 65536$ interfacce di rete. Tali valori saranno quindi compresi tra 128.0.0.0 e 191.255.255.255.

4.1.3 Classe C

Regola. L'indirizzo IP ha i bit 31 (MSB) e 30 posti ad 1 ed il bit 29 posto a 0. Gli 8 bit meno significativi sono utilizzati per rappresentare la parte interfaccia, mentre i rimanenti 24 sono utilizzati per rappresentare la parte rete.

Quindi esisteranno al massimo $2^{21} = 2097152$ reti di classe C (in quanto i tre bit più significativi sono fissati a 110) ad ognuna delle quali potranno essere collegate al massimo $2^8 = 256$ interfacce di rete. Tali valori saranno quindi compresi tra 192.0.0.0 e 223.255.255.255.

Esistono inoltre anche le classi D ed E ma non servono per l'identificazione di interfacce di rete, ma servono per il *multicast*, una tecnica di invio di informazioni ad un gruppo di interfacce di rete in maniera simultanea (ad esempio trasmissioni video su internet). Per quello che è stato presentato sin ora, sarebbe necessario inviare la stessa informazione tante volte quante sono le interfacce di rete destinatarie. La tecnica multicast ovvia questo problema, permettendo di inviare una sola volta (o qualcuna di più per essere sicuri di non perdere pacchetti durante il percorso a causa del rumore) le informazioni per tutti i destinatari. Gli indirizzi IP multicast sono indirizzi di cui l'interfaccia di rete si impossessa temporaneamente e per un numero

finito di trasmissioni e quindi li rilascia. Tale tipo di trasmissione arriva fino a livello C per tutte le interfacce coinvolte: scegliendo indirizzi IP specifici, riservati a questo scopo, non si corre il rischio di far lavorare anche le interfacce di rete non coinvolte nella comunicazione.

4.2 Indirizzi IP riservati

All'interno dell'insieme dei valori degli indirizzi IP da 0.0.0.0 a 223.255.255.255, esistono dei particolari valori che sono riservati e quindi non utilizzabili come indirizzi IP per internet. Tali valori sono riservati per reti LAN che utilizzano il protocollo TCP/IP ma non sono direttamente connesse ad internet (*indirizzi privati*). Di tali indirizzi ce ne sono per ogni classe:

10.X.X.X	(classe A)
172.16.X.X, ..., 172.31.X.X	(classe B)
192.168.X.X	(classe C)

Sono da considerarsi riservati anche gli indirizzi 127.X.X.X (rete di *loopback*). Inviando dati ad uno di tali indirizzi IP, i dati transiteranno dal livello E al livello C e quindi, riconosciuto il particolare indirizzo di destinazione (loop back) a questo livello, i dati ritorneranno fino al livello E e saranno trattati dall'applicazione come appena ricevuti. Questa serie di indirizzi è stata dedicata per la diagnostica dei protocolli dal livello C al livello E.

Inoltre non sono considerati indirizzi IP validi quelli che hanno tutti i bit a 0 o ad 1 nella parte rete o interfaccia. Per esempio non sono indirizzi IP validi i seguenti: 0.12.123.43, 15.0.0.0, 17.255.255.255, 131.10.0.0, 192.167.3.255. Gli indirizzi IP con tutti i bit della parte macchina impostati ad 1 sono chiamati directed broadcast, ovvero utilizzati quando si deve inviare un messaggio a tutte le interfacce di una determinata rete: per esempio il messaggio con destinatario 192.167.58.255 sarà considerato inviato a tutte le interfacce presenti sulla rete 192.167.58.X. Gli indirizzi IP con tutti i bit della parte interfaccia a 0 sono utilizzati per identificare la rete nel suo complesso: per esempio l'indirizzo IP 192.167.58.0 identificherà la rete 192.167.58.X.

È possibile visualizzare l'indirizzo IP relativo alla propria interfaccia di rete dalla console di un sistema *Windows NT*, tramite il comando `ipconfig`.

Sotto *Linux*, invece è presente il comando `ifconfig`, che permette sia di configurare un'interfaccia che di visualizzarne lo stato attuale.

4.3 Internet Protocol

L'Internet Protocol (IP) [16] è un servizio di consegna dei pacchetti inaffidabile, best-effort, senza connessione. Inaffidabile significa che non è garantita

la consegna: un pacchetto può essere perso, duplicato o consegnato fuori sequenza, ma né il mittente né il destinatario saranno informati se avverranno questi eventi. Best-effort si riferisce al fatto che i pacchetti non vengono scartati o duplicati se le condizioni al contorno lo permettono. In pratica la responsabilità di queste inefficienze non è dovuta all'IP ma, ad esempio a malfunzionamenti del livello 2 o all'esaurimento di risorse. Senza connessione significa che ogni pacchetto è trattato indipendentemente dagli altri.

Il protocollo IP prevede l'invio di pacchetti, chiamati datagram, il cui header ha la forma in figura 4.1.

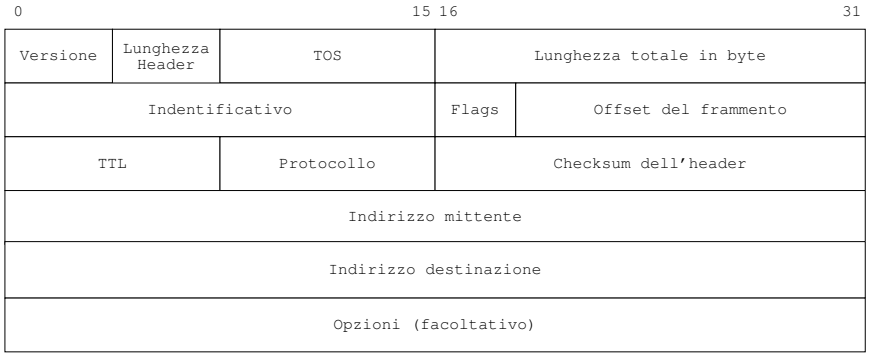

Figura 4.1: Header di un datagram IP

In generale le opzioni non sono presenti, quindi l'header di un datagram può essere considerato lungo 20 byte. Non spiegheremo il significato di tutti i campi, ma si noti che nel pacchetto è presente l'indirizzo IP del mittente e l'indirizzo IP del destinatario. Come viene inviato un pacchetto da una macchina all'altra? Si pone il problema dell'instradamento dei pacchetti, che non era presente ad esempio nell'Ethernet, in quanto i pacchetti raggiungevano comunque tutte le macchine sulla rete locale.

Se il destinatario si trova sulla stessa rete locale del mittente, basterà inviarlo al livello 2, che si occuperà di effettuare l'inoltro. Nel caso che il destinatario non si trovi sulla stessa rete, la strategia dell'IP è quella di inviare il pacchetto ad un'altra macchina, chiamata router, che ha almeno due interfacce di rete, sperando che sappia come trovare il destinatario. Un host avrà uno o più router di default, ossia macchine che si occuperanno di instradare tutti i pacchetti di cui l'host non conosce l'ubicazione precisa. In generale un computer avrà una tabella di instradamento (routing table), che specifica che cosa fare per ogni destinatario possibile, tramite delle regole del tipo: se la destinazione è XXX.XXX.XXX.XXX invia il pacchetto a YYY.YYY.YYY.YYY. Le operazioni che un host esegue sono:

1. Cercare nella tabella se è presente l'indirizzo completo di destinazione. Se è presente, eseguire la regola.

2. Cercare se è presente la rete di destinazione di cui l'indirizzo fa parte. Se è presente, eseguire la regola.

3. Cercare la default route. Se è presente eseguire la regola.

Se nessun passo funziona, la macchina non è in grado di instradare il datagram.

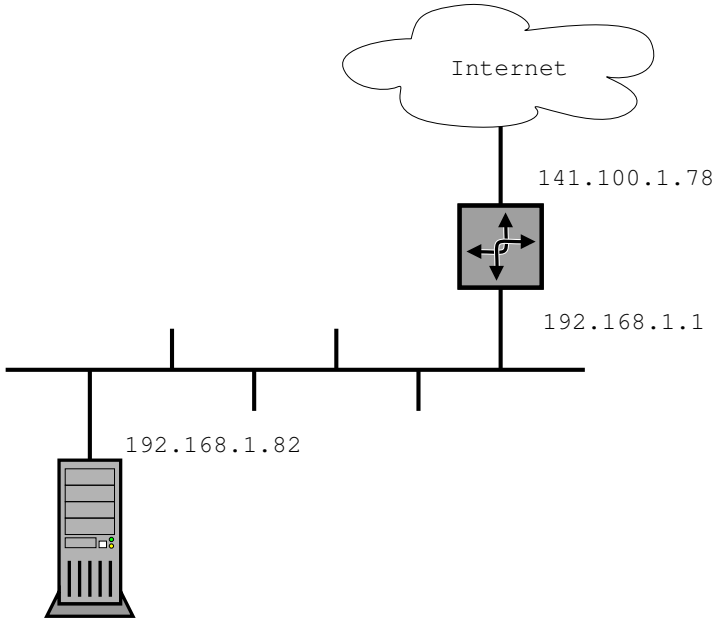

Figura 4.2: Instradamento

Sotto *Linux* la tabella si visualizza con il comando `route`, mentre sotto *Windows NT* con `route PRINT`. Vediamo un esempio di configurazione (figura 4.2): una macchina che ha indirizzo IP 192.168.1.200, il cui router di default sia 192.168.1.1. Sul lato internet, il router ha l'indirizzo 141.100.1.78. La tabella di instradamento dell'host sarà

```
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.200   *               255.255.255.255 UH    0      0        0 eth0
192.168.1.0     *               255.255.255.0   U     0      0        0 eth0
127.0.0.0       *               255.0.0.0       U     0      0        0 lo
default         192.168.1.1     0.0.0.0         UG    0      0        0 eth0
```

Spieghiamo alcuni campi:

- Destination. È l'indirizzo IP di destinazione del pacchetto, che può essere un indirizzo di un host, l'indirizzo di una rete (quindi tutti i pacchetti indirizzati ad un host di questa rete soddisferanno questa

regola), oppure default, ossia tutti i pacchetti che non soddisfano le regole precedenti.

- Gateway. Nel caso che ci sia bisogno di un router per instradare alcuni pacchetti, questo campo contiene l'indirizzo del router. Ovviamente tale macchina deve essere raggiungibile attraverso altre regole.
- Flags. In questo campo appaiono i flag U, G, H (e anche altri, ma possiamo tralasciarli). Il flag U specifica che il collegamento è su (UP). Il flag H specifica che la destinazione è un host (se non è presente significa che la destinazione è una rete). Il flag G specifica che la destinazione va raggiunta attraverso un router (il cui indirizzo è specificato nel campo gateway).
- Iface. Interfaccia sulla quale inviare il pacchetto.

Le tabelle di instradamento di un router possono essere molto più complesse di quelle di un host. Inoltre esiste la possibilità di aggiornare automaticamente la tabella tramite dei protocolli di routing dinamico (RIP, OSPF, BGP, ...). Le problematiche legate a questi protocolli esulano dagli scopi di questo testo.

4.4 Subnetting

Abbiamo visto che ogni indirizzo può essere suddiviso net id e host id, a seconda della classe a cui l'indirizzo appartiene. Adesso è però richiesto che ogni host supporti il subnetting [10]. Alle due suddivisioni precedenti si aggiunge il subnet id, che è una parte dell'indirizzo che specifica, all'interno di una rete, a quale sottorete si appartiene. In pratica, dopo aver ottenuto l'attribuzione di una rete dall'InterNIC, si ha la possibilità di suddividere tale rete in più sottoreti, proprio come se fossero distinte. Ad esempio, se ci viene data la rete di classe B 139.201.0.0, possiamo creare delle sottoreti specificando delle suddivisioni nello spazio a nostra disposizione, secondo lo schema di figura 4.3.

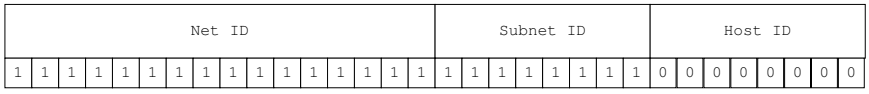

Figura 4.3: Maschera di sottorete

In questo caso abbiamo 254 sottoreti, con 254 host ciascuna. In questo modo, un router esterno non ha bisogno di conoscere i dettagli della nostra suddivisione e instrada tutto al router della rete 139.201.0.0.

Per fare in modo che l'host 139.201.1.1 veda solo la sottorete 139.201.1.0 e non tutta la rete 139.201.0.0, si utilizza la maschera di sottorete (subnet mask). Si specifica cioè quali bit dell'indirizzo costituiscono net id e subnet id, mettendoli ad 1, mentre i bit che specificano l'host id sono a 0.

Per comodità, la maschera di sottorete viene spesso rappresentata in notazione dotted decimal. In questo caso è 255.255.255.0. È possibile anche specificare una maschera non allineata al byte. In questo caso la notazione dotted decimal è meno intuitiva.

Adesso possiamo anche capire il significato del campo Genmask nella tabella di instradamento vista in precedenza: rappresenta la netmask della rete.

4.5 Classless Inter-Domain Routing

Successivamente si è pensato di svincolare gli indirizzi dal concetto di classe, per due motivi principali:

- Molti enti avevano bisogno di più di 254 indirizzi, ma meno di 65534. Tipicamente si facevano assegnare una rete di classe B e ne usavano un sottoinsieme. Ciò ha portato al rapido diminuire delle reti classe B.
- Le tabelle di instradamento dei router delle backbone erano diventate enormi e difficili da mantenere, in quanto contenevano un numero enorme di indirizzi di rete (di cui molti erano della stessa organizzazione).

Il Classless Inter-Domain Routing (CIDR) [6] si basa sul concetto di prefisso, che indica quanti bit identificano il net id. Quindi, invece di avere reti di classe A, B e C, abbiamo reti con prefisso 1, 2, etc. In questo modo è possibile creare una rete con 510 indirizzi ($2^9 - 2$), semplicemente scegliendo il prefisso 23. Contemporaneamente si riducono le tabelle di instradamento, perché questa rete occupa un solo record nella tabella, invece che due nel caso fosse stata realizzata con due reti di classe C.

All'interno di una rete è poi possibile effettuare un subnetting come spiegato precedentemente, il CIDR riguarda solamente i router esterni e non ha nessuna influenza sull'architettura della rete vista dall'interno.

4.6 Address Resolution Protocol

L'Address Resolution Protocol (ARP) [14], è utilizzato per richieste di MAC address delle interfacce di rete.

Le schede di rete realizzano l'intero livello A e parte del livello B, in quanto esiste del software (driver della scheda di rete) che lavora sul PC a

livello B per far funzionare la comunicazione tra la scheda ed il resto dello stack. Questo significa che inviando un'informazione sulla rete destinata ad una determinata interfaccia, è evidente che tutte le interfacce connesse alla rete la raccoglieranno. La maggior parte di esse, confrontando l'identificativo del destinatario con il proprio, scarteranno tutto il pacchetto ricevuto (perché non diretto a loro) e solo una riconoscerà il suo identificativo con quello del destinatario e elaborerà il pacchetto inviato. Ma, poiché il riconoscimento dell'interfaccia di destinazione è legato ad un indirizzo IP che è configurato e gestito dal software a bordo del sistema ma non sulla scheda di rete, significa che per il riconoscimento del destinatario è necessario far girare il software sul sistema, ovvero il microprocessore del sistema deve impiegare alcuni cicli di CPU per identificare se il pacchetto è diretto a lui o meno. Dunque, inviando un pacchetto sulla rete, si fanno lavorare tutte le CPU dei sistemi presenti sulla rete. Questo ovviamente va a diminuire le performance dei vari sistemi. Un rimedio a tale problema è rappresentato dal protocollo di rete ARP. Ogni scheda di rete prodotta con protocollo di collegamento Ethernet viene marchiata con un identificativo univoco (a livello mondiale) detto *MAC address* a 48 bit, che è un codice piatto (non ha una gerarchia come per l'indirizzo IP). Dunque, dal livello superiore, il livello di rete (C) riceve l'ordine di inviare delle informazioni ad un determinato indirizzo IP. Il software al livello C controlla innanzitutto se ha nella propria cache il MAC address relativo all'indirizzo IP da raggiungere. In caso negativo tramite il protocollo ARP, viene inviata una richiesta di MAC address all'interfaccia richiesta. Il pacchetto di richiesta inviato conterrà il MAC address del mittente, l'indirizzo IP del destinatario e come MAC destinatario il broadcast. Tale pacchetto, immesso nella rete, farà lavorare tutti i microprocessori dei sistemi che si trovano sulla rete, in quanto il software a livello C di ogni sistema controllerà se la richiesta di MAC address è indirizzata alla sua interfaccia o meno. Tutti i sistemi ignoreranno tale richiesta tranne l'unico che ha l'interfaccia richiesta. Tale sistema preleverà il MAC address del mittente dal pacchetto e invierà al mittente il suo MAC address. Da questo momento gli altri sistemi non sprecheranno più tempo di CPU in quanto a partire dalla risposta di MAC address le due interfacce si invieranno pacchetti con mittente e destinatario espressi sotto forma di MAC address oltre che di indirizzo IP. In questo modo la logica presente sulla scheda di rete è in grado di riconoscere direttamente se un pacchetto è destinato a lei o meno senza dover più coinvolgere il microprocessore del sistema (il MAC address è mantenuto nella cache del software di livello C per circa 10 minuti per evitare il fatto che in un sistema possa venir sostituita un'interfaccia di rete ma mantenuto lo stesso indirizzo IP: se così non fosse, non sarebbe più possibile trovare tale interfaccia di rete a partire dal MAC address). Se però il destinatario non viene rintracciato sulla rete il software a livello C controlla se è impostato un indirizzo IP di router. In caso negativo viene generato un messaggio di errore perché l'interfaccia richiesta

non è stata trovata. In caso affermativo viene inviata la richiesta, tramite il protocollo ARP, del MAC address del router. Quindi, ricevutolo, verrà inviato il pacchetto con l'indirizzo IP del destinatario e con il MAC address del router. In questo modo il router riceverà il pacchetto ma si accorgerà che non è diretto a lui e quindi provvederà ad instradarlo correttamente per farlo giungere a destinazione.

È possibile visualizzare la cache ARP relativa alla propria interfaccia di rete dalla console di un sistema *Linux* o *Windows NT*, tramite il comando `arp -a`.

4.7 Internet Control Message Protocol

L'Internet Control Message Protocol (ICMP) [17] è un protocollo associato all'IP che serve per inviare messaggi di controllo, ad esempio per notificare errori al mittente di un pacchetto. In realtà, anche se logicamente appartiene al livello di rete, viene incapsulato in un datagram IP.

I messaggi ICMP possono essere query, oppure segnalare errori. Un tipico messaggio di query è `Echo request`, che invia un pacchetto ad un host, chiedendo di rispondere. Il destinatario, se attivo, risponderà con un altro messaggio di query, `Echo reply`. Questi due messaggi sono alla base del programma `ping`, disponibile su tutte le piattaforme. Un messaggio di errore, ad esempio, viene generato se l'indirizzo IP di destinazione di un datagram non viene trovato, perché la macchina è sconnessa, il router che si accorge di questo fatto, manderà il messaggio
`Host unreachable`
così il mittente potrà gestire l'errore.

4.8 Ancora sull'IP

L'IP è stato progettato per essere indipendente dalla rete fisica utilizzata. Abbiamo visto, però, che al cambiare della rete fisica, l'MTU può essere diverso, quindi se si fosse voluto trasmettere un solo datagram per ogni frame, si sarebbe dovuto limitare la dimensione di un datagram alla minima MTU disponibile. Questo, però, avrebbe reso inefficiente la trasmissione su reti con MTU grande, in quanto la lunghezza dell'header in un pacchetto aggiunge un overhead fisso.

La scelta dei progettisti dell'IP è stata diversa: la dimensione di un pacchetto è decisa dallo stack TCP/IP e nel caso che si debba attraversare una rete con MTU piccola, si suddividono i datagram in pezzi più piccoli. Tali pezzi sono detti frammenti. Abbiamo visto che nell'header dell'IP è presente un campo offset dei frammenti, che permette di specificare in quale ordine vanno giustapposti per ricostruire il pacchetto originale.

Il riassemblaggio dei frammenti non avviene appena la MTU ridiventa più alta, ma all'host di destinazione. Questa strategia è stata scelta per semplificare al massimo la gestione nei router. L'host di destinazione ha un timeout: se non riceve tutti i frammenti di un datagram entro un certo tempo, scarta i pezzi ricevuti. Ciò significa che se fallisce la trasmissione di un frammento, tutto il datagram viene scartato.

Il campo TTL (Time To Live) rappresenta la durata di vita del datagram in internet. Ogni volta che un datagram passa attraverso un router, il TTL viene decrementato del massimo tra 1 e il numero di secondi che sono trascorsi tra il momento in cui è stato ricevuto e quello in cui è stato instradato (quindi generalmente è 1). Quando il TTL raggiunge lo 0, il router lo scarta e invia un messaggio ICMP: `Time exceeded`. Questo è utile per risolvere eventuali loop a cui potrebbe andare incontro un pacchetto su internet.

Il programma `traceroute` utilizza il TTL per determinare il cammino percorso da un host ad un altro. Invia un pacchetto con TTL=1, quindi il primo router lo scarta e invia il messaggio `Time exceeded`. Abbiamo in questo modo ottenuto l'indirizzo del primo router. Inviando un pacchetto con TTL via via crescente, otteniamo il cammino fatto (supponendo che nel frattempo non siano cambiate le tabelle di instradamento).

Capitolo 5

Livello di trasporto

Esistono due protocolli a questo livello: **TCP** (Transmission Control Protocol) e **UDP** (User Datagram Protocol). Abbiamo visto che a livello IP, un datagram viene inviato da una macchina all'altra, ma non c'è modo di specificare l'applicativo che deve riceverlo. A livello di trasporto esiste un'astrazione, chiamata porta, che stabilisce quale processo è il destinatario di un certo pacchetto. Alcuni numeri di porte sono assegnati dalla IANA, nella sezione "Well known port numbers" del documento Assigned Numbers (RFC 1700). Una copia della sezione è presente nel file `/etc/services` sotto *Linux*.

5.1 User Datagram Protocol

L'User Datagram Protocol (UDP) [15] è un protocollo non affidabile e senza connessione. Sostanzialmente l'unica caratteristica in più rispetto all'IP è la presenza della porta. Il formato dell'header dei pacchetti UDP è mostrato in figura 5.1:

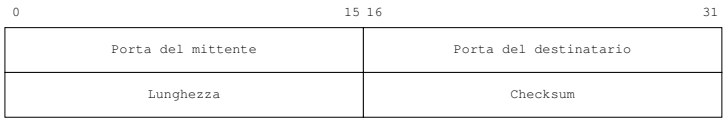

Figura 5.1: Header di un pacchetto UDP

5.2 Transmission Control Protocol

Il TCP [18] è un protocollo affidabile, con connessione, fatto per gestire grosse quantità di dati. Il TCP verifica che i dati arrivino correttamente a destinazione e si preoccupa di gestire la richiesta di reinvio delle informazio-

ni qualora queste non fossero arrivate correttamente a destinazione. È un protocollo a doppia conferma.

I pacchetti TCP vengono chiamati segmenti. Il formato dell'header di un segmento è mostrato in figura 5.2:

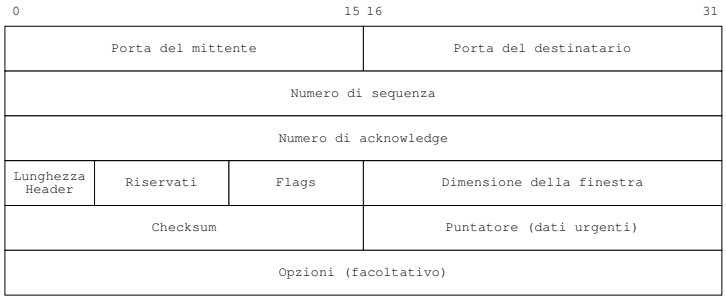

Figura 5.2: Header di un pacchetto TCP

Il TCP vede i dati da trasmettere come stream, ossia la sequenza di byte originaria è passata direttamente al destinatario così come è, senza ulteriori elaborazioni. Inoltre tali stream non sono strutturati (non ci sono confini tra record, o cose del genere).

Quando un client vuole aprire una sessione di comunicazione, deve innanzitutto stabilire una connessione. Una volta stabilita la connessione, può cominciare lo scambio dei dati. Ciò implica che, per uno scambio di dati breve ma frequente, il TCP non è il protocollo più indicato.

Per garantire l'affidabilità il TCP associa ad ogni segmento un numero sequenziale, in modo da poter ricostruire l'ordine dei segmenti. All'inizio della connessione il client invia una richiesta di apertura con un flag chiamato SYN, che serve a comunicare il numero sequenziale di inizio dal lato del client. Il server che la riceve risponde con un messaggio in cui sono attivi i flag ACK (ossia: ho ricevuto la richiesta di sincronizzazione) più un SYN (ossia: la mia sequenza parte da qui). Il client risponde a sua volta con un ACK. Questa procedura è detta three-way handshake.

A questo punto la sessione di comunicazione si può ritenere aperta. Il client invierà i suoi dati al server, segmento dopo segmento, ed il server ogni tanto gli manderà un ACK per confermargli qual è l'ultimo segmento correttamente ricevuto in sequenza. Esiste un parametro detto *window* che rappresenta il massimo numero di packet inviabili prima di ricevere un ACK di conferma (tale valore può variare dinamicamente tramite negoziazione tra i due sistemi che stanno comunicando). Se il client, dopo aver inviato un numero di packet pari a window non riceve un ACK, dopo un timeout riparte a rispedire il packet successivo all'ultimo confermato dal destinatario. Se anche dopo un certo numero di tentativi di ritrasmissione non si riceve

nessun ACK, allora viene generato un messaggio di errore di trasmissione. Un esempio di diagramma di trasmissione è quello di figura 5.3.

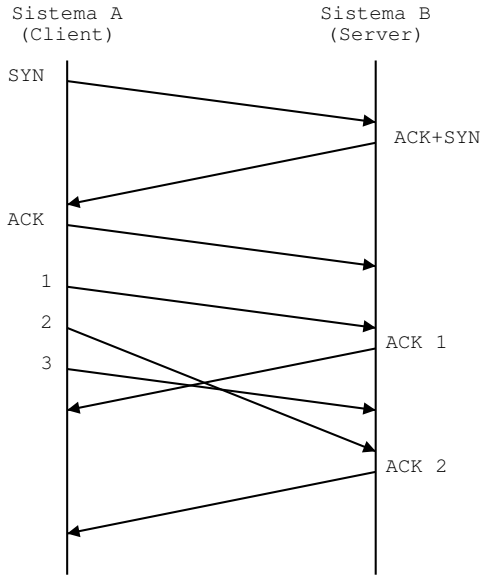

Figura 5.3: Esempio di comunicazione secondo il protocollo TCP

Per terminare una trasmissione c'è un procedimento simile al three-way handshake. Siccome il TCP è full-duplex, ossia la connessione è bidirezionale, essa può essere chiusa anche in una sola direzione (si parla in questo caso di half-close). La chiusura completa di una connessione consta di due half-close. Un lato della connessione, ad esempio il client, notifica all'altro la volonta' di chiudere la connessione inviando un messaggio con il flag FIN attivo. Il server risponde con un pacchetto con ACK, e successivamente invia anche un FIN, per indicare che anche lui termina la connessione. Il client conferma la ricezione rispondendo con un ACK.

Capitolo 6

Livello di applicazione

L'internet suite (ovvero lo stack TCP/IP), fornisce un sacco di protocolli a questo livello: **HTTP** (HyperText Transfer Protocol), **HTTPS** (HyperText Transfer Protocol Secure), **FTP** (File Transfer Protocol), **SMTP** (Simple Mail Transfer Protocol), **POP3** (Post Office Protocol 3), **NNTP** (Network News Transfer Protocol), **SNMP** (Simple Network Management Protocol), **DNS** (Data Name Service), **LDAP** (Light [weight] Directory Access Protocol), **DHCP** (Dynamic Host Configuration Protocol), **TELNET** (Network Terminal Protocol), **GOPHER**, ...

6.1 Telnet

Il protocollo telnet [20] permette di effettuare il login su una macchina remota. Nel mondo UNIX questo era già implementato tramite il protocollo rlogin [8]. Telnet è invece pensato per operare tra sistemi operativi diversi. All'inizio c'è una negoziazione di alcune opzioni tra le due parti, ad esempio il tipo di emulazione di terminale. Alla base di telnet c'è il concetto di Network Virtual Terminal (NVT), un dispositivo virtuale sul quale il server e i client mappano i propri terminali. Successivamente viene chiesto l'user id e una password. Siccome la password viene inviata in chiaro, non è consigliabile l'uso di telnet per motivi di sicurezza.

L'interesse di telnet è dovuto, oltre all'importanza di avere un login remoto, anche al fatto che altri protocolli (FTP, SMTP) usano gli NVT per trasmettere e ricevere i comandi.

6.2 HyperText Transfer Protocol

L'HyperText Transfer Protocol (HTTP) [5] è quasi esclusivamente basato sul comando GET. Per default il server ascolta sulla porta 80. In genere viene aperta una sessione di comunicazione per ogni GET. Oltre al GET esiste il comando POST che serve per far inviare i dati dal client verso il

server. Non è previsto alcun meccanismo di autenticazione del client. Questo è lasciato all'implementazione dell'applicazione.

Il protocollo HTTPS è analogo all'HTTP ma utilizza il protocollo SSL (Secure Socket Layer) per le comunicazioni riservate (criptate). È il browser (client) che deve prevedere e supportare il protocollo SSL.

6.3 File Transfer Protocol

Il File Transfer Protocol (FTP) [21] è utilizzato per il trasferimento di file. Viene richiesta l'autenticazione del client con uno username e una password. Le informazioni transitano sulla rete tutte in chiaro, anche la password, presentando ovvi problemi di sicurezza. In genere c'è sempre una coppia di sessioni (porte) per lo scambio delle informazioni: una per lo scambio dei comandi (porta 21) e l'altra per lo scambio dei dati veri e propri. I modi in cui la trasmissione dei dati può cominciare sono due:

- Attivo. Il client invia al server la porta su cui vuole ricevere i dati. Il server li invia dalla porta 20 e, quando sono terminati, chiude la connessione. Se il client vuole ricevere altri dati aprirà un'altra connessione. In ogni caso la connessione per i comandi rimane aperta.
- Passivo. Il client informa il server che vuole una connessione passiva. Il server invia al client un numero di porta effimera (non assegnata) e il client apre una connessione con questa porta. Tutte e due le connessioni sono quindi iniziate dal client.

Perché questi due metodi? Storicamente il secondo serviva al trasferimento tra due host comandato da un terzo host. Adesso il modo passivo serve se si è dietro un firewall.

Se il server è configurato in modo da permettere a chiunque di accedere al sito, in genere il client può collegarsi con username *anonymous* e password qualunque (le buone maniere prevedono di usare il proprio indirizzo e-mail come password).

6.4 Simple Mail Transfer Protocol

Il Simple Mail Transfer Protocol (SMTP) [19] è usato per l'invio di posta elettronica. È sostanzialmente anonimo e libero. È un protocollo a terminale virtuale, ovvero non ha caratteri di controllo particolari, ma ha delle stringhe che rappresentano dei comandi (es. SEND, ...). Fino a poco tempo fa i server SMTP permettevano a chiunque di collegarsi a loro per inviare delle e-mail. Chiunque poteva connettersi al server della Casa Bianca ed inviare e-mail spacciandosi per il presidente degli Stati Uniti. Adesso, anche per proteggersi dallo spamming, i server SMTP ben configurati non consentono

il relaying della posta, ovvero non inviano la posta ad indirizzi e-mail fuori dal loro dominio a meno che il mittente non appartenga al loro dominio.

6.5 Post Office Protocol

Il Post Office Protocol, versione 3 (POP3) [11] è usato per lo scaricamento della posta elettronica dal server al client. Il server SMTP deve liberarsi subito della posta (ha dei tempi molto stringenti) e la passa al server POP3 che la esamina e la smista per mailbox. Inoltre richiede l'autentificazione del client con uno username ed una password. È anch'esso un protocollo a terminale virtuale.

6.6 Network News Transfer Protocol

Il Network News Transfer Protocol (NNTP) [9] è utilizzato per la gestione dei newsgroup.

6.7 Dynamic Host Configuration Protocol

Il Dynamic Host Configuration Protocol (DHCP) [3] è utilizzato per l'assegnamento automatico degli indirizzi IP alle interfacce di rete. La macchina con il DHCP client, una volta accesa, manda una richiesta di indirizzo IP (Discover) in broadcast IP e MAC, specificando come indirizzo IP del mittente 0.0.0.0 e MAC address il proprio. Il server DHCP risponde in broadcast o meno (dipende dai flag impostati nel Discover) inviando un'offerta di indirizzo IP (e non solo, ma vengono inviati tutti i parametri di configurazione dell'interfaccia di rete). Il client (che riconosce che l'offerta è rivolta a lui in quanto contiene un identificatore che lui aveva inviato nella propria Demand) può accettare l'offerta inviando un pacchetto di Request. Il server DHCP, ricevuto il Request, invia un pacchetto di ACK di conferma. Inoltre il client controlla anche dopo la conferma che non vi siano altre interfacce di rete sulla stessa rete con lo stesso indirizzo IP per mezzo di un messaggio broadcast di richiesta MAC address (tipo ARP). A questo punto il client è sicuro di poter utilizzare tale indirizzo IP. In genere gli indirizzi dinamici sono concessi a tempo (per un certo numero di giorni) e c'è tutta una procedura di richiesta automatica di un nuovo indirizzo IP.

6.8 Network File System

Il Network File System (NFS) è un metodo per condividere i file su una rete locale, sviluppato dalla Sun Microsystems.

6.9 Simple Network Management Protocol

Il Simple Network Management Protocol (SNMP) si basa sull'UDP ed è usato per la gestione degli apparati di rete (macchine, hub, switch, ...) e per l'analisi del traffico. È in grado di impostare al volo indirizzi IP (anche se la versione Microsoft presente in *Windows NT Server* non fa alcuna operazione di SET, ovvero si limita all'analisi di quello che succede ma non può modificare niente). Programmi che usano a fondo tale protocollo sono molto costosi perché permettono di amministrare una rete in maniera remota e di intervenire su di un problema prima ancora che l'utente lo segnali.

6.10 Gopher

Il protocollo GOPHER [1], ormai in disuso, è un'interfaccia alternativa al web, una specie di gestione gerarchica dei siti (come directories).

6.11 Lightweight Directory Access Protocol

Il Lightweight Directory Access Protocol (LDAP) [26] rappresenta il protocollo del futuro. È utilizzato per le pagine bianche di internet (elenchi e-mail, tipo elenchi telefonici).

6.12 La risoluzione degli indirizzi

Ogni interfaccia di rete è identificata da un indirizzo IP, ma per meglio ricordarla è possibile associarle un nome mnemonico detto **host name**. Da questo deriva il fatto che in qualche modo, dall'host name dovrà essere possibile ricavare il relativo indirizzo IP. Ciò va sotto il nome di tecnica di *risoluzione degli indirizzi*. L'host name ha la necessità di essere univoco in tutto il mondo nel caso si tratti di una macchina collegata ad internet. Quindi i nomi delle macchine vengono gestiti secondo una gerarchia ad albero. Dalla radice (root) discendono i domini di livello zero (es. edu, mil, gov, com, it, uk, de, ...). Da ognuno di questi domini discendono i domini di primo livello (es. microsoft.com, esaote.com, elea.it, ...). Da questi possono discendere i domini di secondo livello e così via, fino ad arrivare all'host name dell'interfaccia di rete desiderata. In questo modo è sufficiente assicurarsi che nell'ultimo livello di domino considerato non vi sia un'altra interfaccia con lo stesso host name per essere sicuri che in tutto il mondo non esiste un'altra interfaccia con lo stesso host name, o meglio, non esiste nessun'altra interfaccia con lo stesso **FQDN** (Fully Qualified Domain Name) che è quello formato dal nome dell'interfaccia "visto" dalla radice dei domini che è quello formato a partire dall'host name risalendo verso la radice e concatenando i nomi dei domini che si incontrano, separati da un punto.

La corrispondenza host name ed il relativo indirizzo IP viene mantenuta all'interno del file ASCII `hosts` che si trova nella directory `\Winnnt\System32\drivers\etc`, per i sistemi *Windows NT*, sulla macchina sulla quale risiede l'interfaccia di rete considerata. Quindi, quando l'applicazione passa al livello D l'informazione da trasmettere e l'host name dell'interfaccia di rete a cui trasmetterla, questo è il primo posto dove viene ricercato l'host name per ricavare il suo indirizzo IP (perché il livello C poi avrà bisogno dell'indirizzo IP). Il problema di avere la relazione host name e indirizzo IP su ogni sistema collegato in rete è che se viene cambiato l'indirizzo IP di un'interfaccia di rete di un altro sistema, bisogna aggiornare tutti i file `hosts` presenti su tutti i sistemi collegati in rete, affinché il tutto funzioni! Ecco che viene in aiuto il DNS che sposta il meccanismo di risoluzione degli indirizzi ad una tecnica client/server. Se non esiste nessun riferimento all'host name del destinatario nel file `hosts` e se è stato impostato l'indirizzo IP del server DNS di default, viene fatta una richiesta di risoluzione dell'indirizzo all'interfaccia di rete con tale indirizzo IP. Un server DNS ha un file (tipo il file `hosts`) in cui vengono mantenuti l'host name ed il relativo indirizzo IP di ogni interfaccia di rete di sua competenza, ovvero che si trovano gerarchicamente sotto il suo dominio. Infatti per ogni dominio esiste un server DNS che tiene conto delle macchine raggiungibili nel suo dominio. Quando la richiesta di risoluzione dell'indirizzo arriva ad un server DNS, questo controlla il dominio di appartenenza dell'FQDN specificato. Se il dominio è quello di sua competenza, guarda nella sua tabella se riesce a trovare il corrispondente indirizzo IP. In caso affermativo lo ritorna all'interfaccia di rete che ne ha fatto richiesta, altrimenti le invia un messaggio che la informa del fatto che il nome specificato non esiste. Nel caso in cui il dominio di appartenenza dell'FQDN sia diverso da quello di competenza del server DNS, il server DNS invia la richiesta al server DNS di root, il quale si preoccuperà a sua volta di trovare l'indirizzo IP del server DNS relativo al dominio di livello zero a cui appartiene l'FQDN specificato. Quindi ritorna tale indirizzo IP al server DNS che ne ha fatto richiesta. A questo punto il server DNS inoltra un'altra richiesta al server DNS di primo livello (con l'indirizzo IP fornitogli dal server DNS di root). Il server DNS di primo livello controllerà che il dominio di primo livello specificato dall'FQDN inviato sia di sua competenza. In caso affermativo ricercherà nella sua lista l'indirizzo IP del server DNS di secondo livello e lo passerà al server DNS che ne ha fatto richiesta. E così via fino ad arrivare al server DNS di competenza che sarà in grado di inviare l'indirizzo IP dell'interfaccia di rete specificata nel caso che esista, altrimenti manderà un messaggio di errore. Il server DNS quindi reinvierà la risposta all'interfaccia che ne ha fatto richiesta. Inoltre ogni server DNS a cui è stata fatta una richiesta, metterà il risultato (se con esito positivo) nella sua cache (in cui permarrà per circa 60 minuti) per velocizzare i tempi di ricerca in caso di una successiva richiesta analoga.

L'insieme dei server DNS forma così un database distribuito su internet[1].

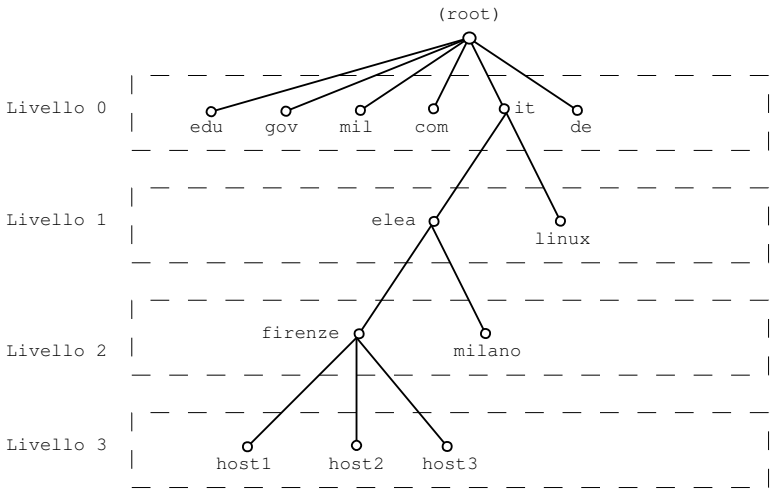

Figura 6.1: Esempio di gerarchia di server DNS

Per esempio, si supponga che nell'esempio illustrato in figura 6.1 l'interfaccia di rete *host1.firenze.elea.it* richeda la risoluzione dell'indirizzo relativo a *www.microsoft.it*. Il server DNS di livello 2 riconosce che l'indirizzo da risolvere non è sotto il suo dominio (*firenze.elea.it*) e quindi inoltra la richiesta di risoluzione al server DNS di root che controlla se a livello 0 esiste il dominio di livello 0 per cui è stata fatta richiesta di risoluzione di indirizzo (*it*). In caso affermativo ritorna il suo indirizzo al richiedente (il server DNS *firenze.elea.it*). A questo punto il server DNS *firenze.elea.it* fa un'altra richiesta di risoluzione dell'indirizzo al server *it* che controlla se a livello 1 esiste il dominio di livello 1 per cui è stata fatta richiesta di risoluzione di indirizzo (*microsoft.it*). In caso affermativo ritorna il suo indirizzo al richiedente (il server DNS *firenze.elea.it*). A questo punto il server DNS *firenze.elea.it* fa un'altra richiesta di risoluzione dell'indirizzo al server *microsoft.it* che controlla se a livello 2 esiste l'interfaccia di rete per cui è stata fatta richiesta di risoluzione di indirizzo (*www.microsoft.it*). In caso affermativo ritorna il suo indirizzo al richiedente (il server DNS *firenze.elea.it*). Il server DNS *firenze.elea.it* ritorna quindi l'indirizzo IP cercato all'interfaccia di rete che ne ha fatto richiesta (*host1.firenze.elea.it*).

Analogamente a quanto illustrato precedentemente si può ricercare l'host name di un'interfaccia di rete a partire dal suo indirizzo IP (*reverse lookup*).

In generale è possibile specificare più indirizzi IP relativi a server DNS. Vengono usati in sequenza. Se il primo server DNS non risponde, viene

[1]Per ogni dominio devono esistere sempre almeno due server DNS, in modo da non far sparire dei domini da internet nel caso in cui il server DNS muoia improvvisamente; inoltre ciò è utile anche per la suddivisione del carico di lavoro.

provato a chiamare il secondo e così via finché non se ne trova uno che risponde.

Inoltre è possibile specificare una lista di ordine di ricerca per i suffissi dei domini (Domain Suffix Search Order). Quando il protocollo TCP/IP chiede al server DNS la risoluzione di un host name, questo viene automaticamente espanso a FQDN con il nome del dominio corrente. Se il nome non viene risolto, viene riprovato con i suffissi specificati nella lista con il loro ordine.

In *Linux*, tutte queste informazioni risiedono nel file `/etc/resolv.conf`. Un esempio di tale file potrebbe essere:

```
nameserver 192.168.1.1 nameserver 192.168.2.4 domain esaote.com search gnu.org
```

In questo caso abbiamo due server DNS, poi abbiamo un dominio di default in cui chiamare le macchine con il solo hostname, e infine, se una macchina non è presente in questo dominio, cerchiamo lo stesso hostname in gnu.org. Il riferimento per queste opzioni è `resolver(5)`.

L'implementazione TCP/IP di Microsoft permette di risolvere anche host name che si trovano sulla stessa rete dell'interfaccia richiedente, anche se non si trovano nel file `hosts` e non è specificato nessun indirizzo IP di un server DNS. La risoluzione dell'host name avviene tramite una richiesta broadcast particolare (implementata da Microsoft) a cui risponde l'interfaccia di rete ricercata, inviando il suo indirizzo IP al richiedente.

È possibile inoltrare delle query di risoluzione indirizzi direttamente al server DNS desiderato dalla console, tramite il comando `nslookup`.

6.13 Interfacciamento LAN/internet

Il problema è quello di collegare una LAN ad internet. La cosa più semplice è quella di assegnare un'indirizzo IP valido per internet ad ogni macchina sulla LAN, in modo tale che questa possa tranquillamente andare su internet. Il problema è che così facendo la macchina è vista da internet, ovvero se la macchina viene utilizzata per lo sviluppo e magari condivide qualche risorsa (hard disk, stampante, ...), anche il mondo internet protrà usufruire di tali condivisioni, quindi bisogna stare molto attenti a come è configurata ogni singola macchina. Ci sono altri modi più intelligenti di interfacciare una LAN ad internet di seguito esposti.

6.13.1 Il proxy server

Una soluzione all'interfacciamento di una LAN ad internet può essere attraverso l'uso di un **proxy server**. Si predispone una macchina con due schede di rete di cui una connessa ad internet e l'altra connessa alla LAN. Per client (browser) è possibile specificare il proxy server che gestisce la comunicazione tra la macchina della LAN ed internet. Inoltre il proxy server può gestirsi una sua cache di pagine html in modo da velocizzarne l'accesso

agli utenti della LAN. Il protocollo HTTP prevede di lavorare attraverso un proxy server. La comunicazione avviene in maniera diversa che non a farla direttamente su internet senza proxy server. Il proxy server può inoltre filtrare le connessioni per certe macchine sulla LAN. Un proxy server non fa routing. Un proxy server è utilizzabile soltanto da applicazioni che ne prevedono l'uso.

Microsoft propone la soluzione del *proxy client*: su ogni macchina sulla LAN viene sostituita la dll relativa all'interfaccia *WinSock* con uno stub che verifica, per ogni chiamata a tale interfaccia, se la richiesta è fatta ad un indirizzo che non fa parte della LAN. In caso affermativo chiama il proxy server dicendogli di inoltrare la richiesta all'indirizzo specificato, altrimenti chiama la vecchia dll opportunamente rinominata.

6.13.2 Il masquerading

Un'altra soluzione all'interfacciamento di una LAN ad internet può essere attraverso l'uso di un server **NAT** [4] (Network Address Translation), ovvero usando la tecnica del **masquerading**. Si predispone una macchina con due schede di rete di cui una connessa ad internet e l'altra connessa alla LAN. Ogni macchina sulla LAN ha configurato come default router il server NAT che effettua una conversione di indirizzi ed un routing. Ogni pacchetto che arriva al server NAT da una macchina che sta sulla LAN viene modificato cambiandogli l'indirizzo IP di origine con il proprio (relativo all'interfaccia che va su internet). Il server NAT deve inoltre tenersi traccia da quale macchina è stata fatta la richiesta. Quindi invia il pacchetto su internet. Quando gli ritorna la risposta, deve cambiare l'indirizzo IP del destinatario con quello della macchina sulla LAN che aveva fatto la richiesta di cui ha tenuto traccia.

6.13.3 I firewall

Un firewall, come fa intuire la parola, è un meccanismo per limitare gli accessi alle macchine in una rete locale dall'esterno. Ad esempio, si può volere che l'accesso ad un server web sia ristretto alla rete locale e non sia visibile dall'esterno.

Esistono due tipi di firewall: packet filtering e stateful inspection. I firewall di tipo packet filtering lavorano generalmente a livello IP, analizzano ciascun pacchetto in ingresso e in uscita e decidono se inoltrarlo o scartarlo a seconda di alcune regole. I firewall di tipo stateful inspection sono invece in grado di seguire gli stati dei protocolli (ad esempio FTP), rendendosi conto se le operazioni sono effettuate secondo le regole specificate.

Capitolo 7

Sviluppo di applicazioni

Le norme sul protocollo TCP/IP, specificano il funzionamento generale del protocollo, ma non danno delle specifiche stringenti né sull'interfaccia, né sull'implementazione dello stesso. Quindi sono state sviluppate diverse interfacce per questo protocollo (ogni produttore ne ha sviluppata una propria). Microsoft ha sviluppato l'interfaccia *WinSock* che deriva dall'interfaccia *Sockets* del mondo *Unix* ma è ampliata con funzioni proprietarie. Le versioni di *WinSock* sono la 1.0 (16 bit), la 1.1 (16/32 bit) e la 2.0 (32 bit).

Innanzitutto si pone il problema di come identificare l'applicazione che deve ricevere le informazioni inviate dal mittente, visto che su di un sistema multitasking possono girare contemporaneamente varie applicazioni. Per il protocollo TCP/IP è stato introdotto il concetto di **porta**. Ogni sessione di comunicazione avviene attraverso una specifica porta identificata da un valore numerico piatto su 16 bit, ovvero il mittente deve specificare il numero di porta a cui vuole inviare il pacchetto[1]. Esiste un ente a livello mondiale che definisce dei numeri di porta per applicazioni standard: ad esempio la porta 80 è quella standard usata dai server web (HTTP), le porte 20 e 21 sono usate dai server FTP, la porta 25 dai server SMTP, la porta 110 dai server POP3, ... Questi numeri di porta sono detti *well known port numbers*. Per applicazioni private in genere è bene utilizzare dei numeri di porta compresi tra 1024 e 5000, in quanto vi sono degli well known port number sotto il valore 1024 e sopra il valore 5000, o consultare la RFC relativa (1060). Nel file `\Winnnt\System32\drivers\etc\Services` c'è un elenco dei numeri di porta utilizzati da altre applicazioni che girano su un sistema *Windows NT*. Per ogni porta ci può essere associata soltanto un'applicazione (in realtà le estensioni Microsoft permettono di avere anche più di un'applicazione sulla stessa porta).

Si definisce **socket** la coppia $\langle indirizzoIP, porta \rangle$. Si chiama **asso-**

[1] In effetti, la necessità effettiva di conoscere a priori la porta di destinazione è limitata al client: il client deve conoscere, prima di connettersi, il numero di porta del server; il server risponderà su una porta che il client gli fornisce automaticamente.

ciazione la terna $\langle protocollo, socketmittente, socketdestinatario \rangle$. In ogni istante non possono esistere contemporaneamente due associazioni uguali.

7.1 Schema di un'applicazione

7.1.1 Applicazione client

La prima cosa di cui si preoccupa un client è di creare un'associazione per poter iniziare la comunicazione. Questo viene fatto utilizzando la funzione `socket()` che fa allocare al sistema operativo una struttura che tiene conto dell'associazione e la riempie soltanto con il codice (numero) relativo al protocollo utilizzato. Tale funzione ritorna un handle[2] alla struttura creata.

Il client procede poi con la chiamata alla funzione `connect()` in cui specifica l'indirizzo IP e la porta dell'applicazione con cui desidera comunicare. Questo fornisce le informazioni relative al socket di destinazione che vengono scritte nella struttura dati dell'associazione e le informazioni relative al socket mittente vengono automaticamente gestite dal sistema operativo (che assegna un numero di porta libero).

A questo punto il client può inviare (`send()`) e ricevere (`recv()`) dati. Al termine della comunicazione deve chiudere la sessione con la chiamata alla funzione `shutdown()` e quindi può terminare l'associazione chiamando la funzione `closesocket()`.

Un'applicazione che usi il protocollo UDP, anziché TCP, non avrebbe bisogno di chiamare la `connect()` poiché il protocollo UDP non prevede l'apertura di una sessione di comunicazione. La chiamata a `connect()` può comunque essere effettuata, per poter utilizzare in seguito le funzioni `send()` e `recv()` invece di `sendto()` e `recvfrom()` (che hanno un parametro in più rispetto alle precedenti), ma non vi è alcuna connessione.

7.1.2 Applicazione server

Il server in genere è un'applicazione che sta in ascolto se qualche client vuole comunicare con lui ed in caso affermativo, attiva una connessione con il client.

La prima chiamata sarà sempre alla funzione `socket()` per far allocare al sistema operativo una struttura che tiene conto dell'associazione, riempiendola soltanto con il codice (numero) relativo al protocollo utilizzato.

A questo punto il server deve impostare il numero di porta utilizzato che non può essere assegnato automaticamente dal sistema operativo, altrimenti i client non ne sarebbero a conoscenza e quindi non potrebbero comunicare

[2]Un handle non è un puntatore, ma un riferimento ad una tabella del sistema operativo in cui ci sono dei puntatori a strutture dati; vengono utilizzati handle anziché puntatori per questioni di sicurezza dei dati e per il fatto che gli spazi di indirizzamento del gestore dei dati e dell'applicazione sono diversi.

con tale server. Per far ciò il server chiama la funzione `bind()` specificando l'indirizzo IP dell'interfaccia di rete ed il numero di porta sulla quale è in ascolto. Queste informazioni costituiscono il socket mittente e tali valori vengono inseriti nella struttura relativa all'associazione.

Il server deve poi comunicare al sistema operativo di prepararlo all'ascolto, fornendogli una coda di richieste in attesa. Questo lo fa chiamando la funzione `listen()`. Il numero massimo di richieste in attesa (specificato dal server) è il numero di richieste di connessione che possono arrivare al server mentre lui ne sta già servendo una (ovvero che arrivano durante il periodo di tempo che intercorre dal momento in cui il server esce dall'`accept()` finché non vi ritorna).

A questo punto il server può effettivamente mettersi in ascolto di richieste di connessione chiamando la funzione `accept()`. È da notare il fatto che a questo punto la struttura dati relativa all'associazione non è ancora completamente riempita (valorizzata), infatti mancano le informazioni relative al socket destinatario. Quando gli arriva una richiesta di connessione da parte di un client, la funzione `accept()` ritorna e fornisce anche le indicazioni relative al socket che desidera stabilire una connessione. Tali valori costituiscono l'elemento mancante per riempire la struttura dati relativa all'associazione. Il server può quindi decidere di accettare la connessione passando la gestione relativa ad essa ad un altro thread, e ritornare quindi in ascolto di altre eventuali richieste di connessioni. In questo frattempo potrebbero essere arrivate altre richieste di connessioni da parte di altri client ed a questo serve la coda delle richieste in attesa (impostata tramite la `listen()`). In tale coda ci saranno bufferizzate tali eventuali richieste (tutto ciò viene gestito automaticamente).

Il thread del server che si preoccupa di gestire una connessione può inviare (`send()`) e ricevere (`recv()`) dati. Al termine della comunicazione deve chiudere la sessione con la chiamata alla funzione `shutdown()` e quindi può terminare l'associazione chiamando la funzione `closesocket()`.

7.1.3 Estensioni Microsoft

Ogni applicazione che desidera utilizzare l'interfaccia *WinSock* deve effettuare almeno una chiamata alla funzione `WSAStartup()` per inizializzare interfaccia e quando ha finito di utilizzarla deve chiamare la funzione `WSACleanup()`. L'importante è che queste funzioni vengano chiamate lo stesso numero di volte perché al loro interno hanno una gestione del tipo reference counting per l'utilizzo della dll necessaria per poter utilizzare l'interfaccia *WinSock* (`ws2_32.dll` per *WinSock* 2.0, `wsock32.dll` per *WinSock* 1.1 e `winsock.dll` per *WinSock* 1.0).

La gestione dei codici di errore ritornati dalle varie funzioni è effettuata dalla funzione proprietaria `WSAGetLastError()`.

7.1.4 Esempio di codice (Server TCP WinSock)

Quello riportato di seguito è un esempio di codice per un'applicazione server TCP sulla porta 3000.

```
#include <winsock2.h>
#include <windows.h>
#include <stdio.h>
#include <conio.h>

#define PORT_NUMBER 3000
#define EXPECTED_MSG_SIZE 200

void Errore()
{
    printf("\nErrore: %d",WSAGetLastError());
    WSACleanup();
    exit(1);
}

void main()
{
    WORD l_RequiredVersion;
    WSADATA l_wsaData;
    int l_Err;
    SOCKET l_AssociationHandle;
    SOCKET l_AcceptedAssociationHandle;
    struct sockaddr_in l_Socket;
    struct sockaddr_in l_AcceptSocket;
    struct hostent * l_HostEntity;
    char l_HostName[200];
    char l_IPAddress[18];
    unsigned int l_addr;
    unsigned int l_AcceptSocketLen;
    int l_TotReceivedBytes;
    char l_RecvMsg[EXPECTED_MSG_SIZE];
    int l_ReceivedBytes;

    l_RequiredVersion = MAKEWORD(2, 0);

    if (WSAStartup(l_RequiredVersion, &l_wsaData) != 0)
        Errore();

    /* controllo versione disponibile (se necessario) */
    /*
    if (LOBYTE(l_wsaData.wVersion) != 0
       || HIBYTE(l_wsaData.wVersion) != 2)
        Errore();
    */
```

```
/* creazione struttura dati relativa all'associazione */
l_AssociationHandle = socket(AF_INET,SOCK_STREAM,
                             IPPROTO_TCP);
if (l_AssociationHandle == INVALID_SOCKET)
    Errore();

/* protocollo */
l_Socket.sin_family = AF_INET;

/* porta */
l_Socket.sin_port = htons(PORT_NUMBER);


/* ---------------------------------------
   Ricerca dell'indirizzo IP dell'interfaccia di rete
   sulla quale effettuare il servizio
   --------------------------------------- */

if (gethostname(l_HostName, 200) != 0)
    Errore();

l_HostEntity = gethostbyname(l_HostName);
if (l_HostEntity == NULL)
    Errore();

/* ------------------------------------ */


memcpy(&(l_Socket.sin_addr.s_addr),
       l_HostEntity->h_addr_list[0],
       l_HostEntity->h_length);

if (bind(l_AssociationHandle,
         (struct sockaddr *)&l_Socket,
         sizeof(l_Socket)) == SOCKET_ERROR)
    Errore();

if (listen(l_AssociationHandle, 10) == SOCKET_ERROR)
    Errore();

l_AcceptSocketLen = sizeof(l_AcceptSocket);

/* Accetta un tentativo di connessione.BLOCCANTE */
l_AcceptedAssociationHandle =
   accept(l_AssociationHandle,
          (struct sockaddr *)&l_AcceptSocket,
          (int *)&l_AcceptSocketLen);

if (l_AcceptedAssociationHandle == INVALID_SOCKET)
```

```
        Errore();

    /* prima operazione: ricezione dati dal client */

    l_TotReceivedBytes = 0;

    while (l_TotReceivedBytes < EXPECTED_MSG_SIZE)
    {
        l_ReceivedBytes =
           recv(l_AcceptedAssociationHandle,
                &l_RecvMsg[l_TotReceivedBytes],
                EXPECTED_MSG_SIZE - l_TotReceivedBytes,
                0);
        if (l_ReceivedBytes == SOCKET_ERROR)
        {
            Errore();
        }
        else if (l_ReceivedBytes == 0)
        {
            /* Shutdown from the client */
            break;
        }
        else
        {
            l_TotReceivedBytes += l_ReceivedBytes;
        }
    } /* end while */

    ...

    /* chiusura dell'associazione. */
    closesocket(l_AssociationHandle);

    WSACleanup();
}
```

Innanzitutto va precisato il fatto che la riga di codice[3]

```
memcpy(&(l_Socket.sin_addr.s_addr),
       l_HostEntity->h_addr_list[0],
       l_HostEntity->h_length);
```

è equivalente alla riga

```
l_Socket.sin_addr =
    *((LPIN_ADDR)*l_HostEntity->h_addr_list[0]);
```

Esistono altri modi in cui può essere trovato l'indirizzo IP dell'interfaccia di rete sulla quale effettuare il servizio. Si può utilizzare, ad esempio, la funzione `gethostbyaddr()`

[3]Il secondo parametro della `memcpy()` può essere anche soltanto `l_HostEntity->h_addr` poiché nel file `winsock2.h` è posto `#define h_addr h_addr_list[0]`.

```
l_addr = inet_addr("127.0.0.1");
if (l_addr == INADDR_NONE)
    Errore();

l_HostEntity = gethostbyaddr((char *)&l_addr,4,AF_INET);
if (l_HostEntity == NULL)
    Errore();
```

oppure specificare il fatto che qualunque interfaccia di rete installata sulla macchina è utilizzata dall'applicazione server (i client possono connettersi a qualunque interfaccia di rete installata sulla macchina per comunicare col server)[4]

```
l_Socket.sin_addr.s_addr = INADDR_ANY;
```

oppure si può utilizzare un indirizzo IP di loop back

```
l_Socket.sin_addr.s_addr = inet_addr("127.0.0.1");
```

Con la funzione `bind()` si riempie la struttura dati relativa all'associazione soltanto per quanto riguarda il socket mittente (quello di destinazione non è ancora conosciuto dal server). Quindi, il server viene posto in attesa di connessioni tramite la chiamata alla funzione `accept()`, che ritorna[5] quando un client si connette al server. A questo punto viene fornito al server una nuova struttura dati relativa ad un'associazione vera e propria (tutti i campi sono riempiti: il protocollo ed il socket del mittente sono copiati dalla precedente ed il socket del destinatario è riempito dal framework con il socket relativo al client) che può essere utilizzata per operazioni di ricezione ed invio dati.

In generale la prima operazione effettuata da un server sarà quella di attendere una richiesta, ovvero il server si pone in ricezione. A tal proposito va fatta una precisazione: nonostante il fatto che la funzione `recv()`, che pone l'applicazione in ricezione, sia bloccante (per default), essa non ritorna necessariamente quando ha finito di leggere i dati inviati dal mittente, ma può ritornare anche dopo aver letto soltanto una parte di tali dati. È compito dell'applicazione ricevente accertarsi del fatto che la `recv()` abbia effettivamente ricevuto tutto il messaggio inviato dal mittente. A tale scopo è necessario conoscere la lunghezza del messaggio inviato dal mittente, ovvero è necessario che il ricevente conosca a priori la lunghezza del messaggio che si aspetta. Quindi, se il ricevente si accorge che la `recv()` è ritornata senza aver ricevuto tutto il messaggio inviato dal mittente, deve richiamare la stessa `recv()` finché non si riceve completamente il messaggio inviato (ecco il motivo del ciclo `while`).

[4] La macro `INADDR_ANY` è definita uguale a 0, quindi rappresenta qualunque interfaccia di rete.

[5] Per default la funzione `accept()` è bloccante, ma c'è comunque la possibilità di impostarla in modalità non bloccante tramite la funzione `ioctlsocket()` (lo stesso vale per le altre funzioni bloccanti come la `connect()`, la `send()`, la `recv()`, ...).

7.1.5 Esempio di codice (Client TCP WinSock)

Quello riportato di seguito è un esempio di codice per un'applicazione client TCP sulla porta 3000. È interessante notare il fatto che il client deve conoscere a priori la porta alla quale connettersi per comunicare col server.

```
#include <winsock2.h>
#include <windows.h>
#include <stdio.h>
#include <conio.h>

#define SRV_HOST_NAME "SrvHost"
#define SRV_PORT_NUMBER 3000
#define MSG_SIZE 200

void Errore()
{
    printf("\nErrore: %d",WSAGetLastError());
    WSACleanup();
    exit(1);
}

void main()
{
    WORD l_RequiredVersion;
    WSADATA l_wsaData;
    SOCKET l_AssociationHandle;
    SOCKET l_AcceptedAssociationHandle;
    struct sockaddr_in l_DestSocket;
    struct hostent * l_HostEntity;
    char l_IPAddress[18];
    char l_Msg[MSG_SIZE];
    int l_SentBytes;

    strcpy(l_Msg,"luca");

    l_RequiredVersion = MAKEWORD(2, 0);

    if (WSAStartup(l_RequiredVersion, &l_wsaData) != 0)
        Errore();

    /* controllo versione disponibile (se necessario) */
    /*
    if (LOBYTE(l_wsaData.wVersion) != 0
       || HIBYTE(l_wsaData.wVersion) != 2)
        Errore();
    */

    /* creazione struttura dati relativa all'associazione */
    l_AssociationHandle = socket(AF_INET,SOCK_STREAM,
```

```
                                IPPROTO_TCP);
    if (l_AssociationHandle == INVALID_SOCKET)
        Errore();

    /* protocollo */
    l_DestSocket.sin_family = AF_INET;

    /* porta */
    l_DestSocket.sin_port = htons(SRV_PORT_NUMBER);

    l_HostEntity = gethostbyname(SRV_HOST_NAME);
    if (l_HostEntity == NULL)
        Errore();

    memcpy((char FAR *)&(l_DestSocket.sin_addr),
           l_HostEntity->h_addr,
           l_HostEntity->h_length);

    if (connect(l_AssociationHandle,
                (PSOCKADDR)&l_DestSocket,
                sizeof(l_DestSocket)) == SOCKET_ERROR)
        Errore();

    l_SentBytes = send(l_AssociationHandle, l_Msg, MSG_SIZE, 0);
    if (l_SentBytes == SOCKET_ERROR)
        Errore();

    ...

    /* terminazione della sessione */
    shutdown(l_AssociationHandle, 0x02);

    /* chiusura dell'associazione */
    closesocket(l_AssociationHandle);

    WSACleanup();
}
```

Innanzitutto è interessante notare il fatto che il client deve conoscere a priori l'host name dell'interfaccia utilizzata dal server e la porta da questo utilizzata per la comunicazione con i client. Queste informazioni sono necessarie all'applicazione client!

Con la funzione `connect()` il client effettua un tentativo di connessione al server. È da notare il fatto che il client ha riempito la struttura dati relativa all'associazione soltanto per quello che riguarda il protocollo ed il socket del destinatario, i dati relativi al socket del mittente vengono auto-

maticamente riempiti dal framework tramite la `connect()`[6]. La funzione ritorna a connessione stabilita o perché si è verificato un errore sulla rete. Quindi si può iniziare la fase di invio e ricezione dei dati.

In generale la prima operazione effettuata da un client sarà quella di inviare una richiesta al server. Questo lo si fa per mezzo della funzione `send()` che ritorna quando ha inviato l'intero messaggio (oppure ha riscontrato un errore sulla rete).

[6]Niente comunque impedisce di specificarli volontariamente tramite una precedente chiamata alla funzione `bind()`.

Capitolo 8

Altri protocolli usati da Microsoft

8.1 Il protocollo NetBEUI

Nel 1984 IBM sviluppò un'API per i PC Intel chiamata NetBIOS, che implementava delle semplici funzioni per connettersi ad un computer remoto e condividere delle risorse. L'anno seguente rilascio il NetBEUI (NetBIOS Extended User Interface), un protocollo basato su NetBIOS che funzionava solo su reti locali. Vista la crescita del TCP/IP, si pensò di implementare NetBIOS su TCP/IP (NBT) [12, 13].

NetBEUI è un protocollo a livello di sessione. Questo protocollo non è molto efficiente (e non è possibile, vista la sua struttura, aumentarne l'efficienza su reti con protocollo di rete Ethernet) poiché effettua un sacco di comunicazioni broadcast. Funziona solo per LAN e non è routabile (cioè non è in grado di attraversare un router). È molto semplice rispetto al TCP/IP e non ha bisogno di alcuna configurazione.

L'identificativo dell'interfaccia di rete è il *NetBIOS name* composto da 16 byte (15 + 1). È un valore piatto (senza gerarchia) ed è composto da 15 caratteri alfanumerici (ASCII) che corrispondono al Computer Name dei sistemi *Windows* ed un byte, chiamato resource type, che indica il tipo di servizio offerto.

In assenza di un NetBIOS NameServer (NBNS), esiste un meccanismo di contesa, con il quale si evita che due macchine abbiano lo stesso nome. Se è presente un NBNS, l'autorità per la distribuzione dei nomi è invece centralizzata. Analogamente, la risoluzione dei nomi può avvenire in due modi: senza NBNS si manda un broadcast e si attende che la macchina risponda con il suo indirizzo IP. Se c'è un NBNS si fa una query a quest'ultimo (non ci sono broadcast).

Quando si aggiorna il Computer Name su una macchina *Windows*, viene automaticamente aggiornato il file `hosts` assegnando all'host name relativo

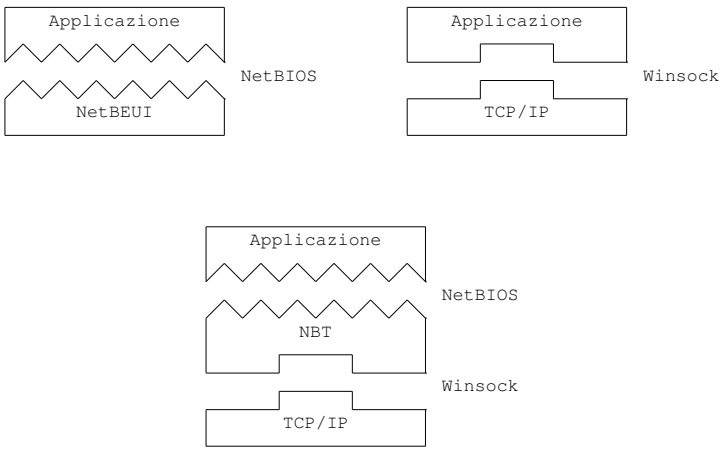

Figura 8.1: TCP/IP, NetBEUI e NBT

al TCP/IP dell'interfaccia di rete installata nel sistema, lo stesso valore del Computer Name.

Tutte le applicazioni Microsoft in genere utilizzano questo protocollo ma nel 1987 la IAB ha sviluppato l'NBT (NetBIOS over TCP/IP). In questo modo tutte le applicazioni scritte per NetBEUI possono usufruire del protocollo TCP/IP in modo del tutto trasparente (visto che lo strato NBT viene caricato automaticamente all'occorrenza). Quindi non c'è più bisogno di avere installato il protocollo NetBEUI sui PC *Windows*. Per NBT viene utilizzato il file `lmhosts` che è l'analogo del file `hosts` per il TCP/IP.

Il server WINS (Windows Internet Name Services) è per NBT l'analogo del server DNS per TCP/IP.

8.2 Server Message Block

Il protocollo Server Message Block (SMB) è un'estensione dell'NBT. In generale, in una configurazione di un sistema *Windows* si hanno i servizi Server e Workstation che non sono altro che il server ed il client del protocollo **SMB** (protocollo che permette la condivisione dei filesystem remoti). Come binding si trova che sotto Workstation, ad esempio, ci sono nell'ordine WINS Client (TCP/IP) (leggere NBT) e sotto NetBEUI Protocol. Questo indica che il client SMB prima prova con NBT e nel caso non funzioni (perché il server sul sistema remoto non ha il protocollo TCP/IP) viene provato direttamente con NetBEUI.

Capitolo 9

Licenza

Il presente documento, come riportato nella pagina precedente all'indice, è coperto dalla licenza GNU FDL (GNU Free Documentation License), della quale si riporta una copia della versione originale (in lingua inglese) ed una copia di una versione non ufficiale in lingua italiana, utile per meglio comprendere il senso della licenza stessa.

9.1 GNU Free Documentation License

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software

does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The **"Document"**, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as **"you"**. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A **"Modified Version"** of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A **"Secondary Section"** is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The **"Invariant Sections"** are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The **"Cover Texts"** are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A **"Transparent"** copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been

arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called **"Opaque"**.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The **"Title Page"** means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section **"Entitled XYZ"** means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as **"Acknowledgements"**, **"Dedications"**, **"Endorsements"**, or **"History"**.) To **"Preserve the Title"** of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to

the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include

the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See `http://www.gnu.org/copyleft/`.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

9.2 GNU Free Documentation License (italiano)

Quella presentata in questa sezione è una traduzione italiana non ufficiale della GNU Free Documentation License (Licenza per Documentazione Libera GNU). Non è pubblicata dalla Free Software Foundation e non ha valore legale nell'esprimere i termini di distribuzione delle opere che la utilizzano. Solo la versione originale in inglese della licenza (riportata in sez. 9.1) ha valore legale. La presente traduzione è riportata soltanto nella speranza che possa aiutare le persone di lingua italiana a capire meglio il significato della licenza.

Versione 1.2, November 2002

Preambolo

Lo scopo di questa licenza è di rendere un manuale, un testo o altri funzionali ed utili documenti "liberi" nel senso di assicurare a tutti la libertà effettiva di copiarli e redistribuirli, con o senza modifiche, a fini di lucro o meno. In secondo luogo questa licenza prevede per autori ed editori il modo per ottenere il giusto riconoscimento del proprio lavoro, preservandoli dall'essere considerati responsabili per modifiche apportate da altri.

Questa licenza è un "copyleft": ciò vuol dire che i lavori che derivano dal documento originale devono essere ugualmente liberi. È il complemento alla Licenza Pubblica Generale GNU, che è una licenza di tipo "copyleft" pensata per il software libero.

Abbiamo progettato questa licenza al fine di applicarla alla documentazione del software libero, perché il software libero ha bisogno di documentazione libera: un programma libero dovrebbe accompagnarsi a manuali che forniscano la stessa libertà del software. Ma questa licenza non è limitata alla documentazione del software; può essere utilizzata per ogni testo che tratti un qualsiasi argomento e al di là dell'avvenuta pubblicazione cartacea. Raccomandiamo principalmente questa licenza per opere che abbiano fini didattici o per manuali di consultazione.

1. APPLICABILITÀ E DEFINIZIONI

Questa licenza si applica a qualsiasi manuale o altra opera, in qualasiasi formato, che contenga una nota messa dal detentore del copyright che dica che si può distribuire nei termini di questa licenza. Tale nota assicura una licenza d'uso dell'opera, riconosciuta in tutto il mondo, libera da royalty, di durata illimitata, secondo le condizioni qui elencate. Con il termine **"documento"**, in seguito ci si riferisce a qualsiasi manuale o opera alla quale si applica questa licenza. Ogni fruitore è un destinatario della licenza e viene indicato con il termine **"voi"**. La licenza viene automaticamente accettata se si copia, modifica o distribuisce il documento in modo tale da richiedere il permesso relativo alle leggi sul copyright.

Una **"versione modificata"** di un documento è ogni opera contenente il documento stesso o parte di esso, sia riprodotto alla lettera che con modifiche, oppure traduzioni in un'altra lingua.

Una **"sezione secondaria"** è un'appendice cui si fa riferimento o una premessa del documento e riguarda esclusivamente il rapporto dell'editore

o dell'autore del documento con l'argomento generale del documento stesso (o argomenti affini) e non contiene nulla che possa essere compreso nell'argomento principale (per esempio, se il documento è in parte un manuale di matematica, una sezione secondaria non può contenere spiegazioni di matematica). Il rapporto con l'argomento può essere un tema collegato storicamente con il soggetto principale o con soggetti affini, o essere costituito da argomentazioni legali, commerciali, filosofiche, etiche o politiche pertinenti.

Le **"sezioni non modificabili"** sono alcune sezioni secondarie i cui titoli sono esplicitamente dichiarati essere sezioni non modificabili, nella nota che indica che il documento è realizzato sotto questa licenza. Se una sezione non corrisponde alla definizione di sezione secondaria sopra espressa, non è permesso designarla come non modificabile. Il documento può non contenere sezioni non modificabili. Se il documento non identifica nessuna sezione non modificabile significa che non ne contiene alcuna.

I **"testi copertina"** sono dei brevi brani di testo che sono elencati nella nota che indica che il documento è realizzato sotto questa licenza. Un testo di fronte copertina può contenere al più 5 parole, ed un testo di retro copertina può contenere al massimo 25 parole.

Una copia **"trasparente"** del documento indica una copia leggibile da un calcolatore, codificata in un formato le cui specifiche sono disponibili pubblicamente (preferibile per effettuare revisioni dei documenti), i cui contenuti possono essere visti e modificati direttamente, ora e in futuro, con generici editor di testi o (per immagini composte da pixel) con generici editor di immagini o (per i disegni) con qualche editor di disegni ampiamente diffuso, e la copia deve essere adatta al trattamento per la formattazione o per la conversione in una varietà di formati atti alla successiva formattazione. Una copia fatta in un altro formato di file trasparente il cui markup (o la cui assenza di markup) sia stato effettuato per intralciare o scoraggiare modifiche future da parte dei lettori non è trasparente. Un formato di immagine non è trasparente se è utilizzato per rappresentare qualsiasi porzione sostanziale di testo. Una copia che non è trasparente è detta **"opaca"**.

Esempi di formati adatti per copie trasparenti sono l'ASCII puro senza markup, il formato di input per Texinfo, il formato di input per LaTeX SGML o XML accoppiati ad una DTD pubblica e disponibile, e semplice HTML, PostScript o PDF conforme agli standard e progettato per essere modificato manualmente. Esempi di formati immagini trasparenti sono PNG, XCF e JPG. I formati opachi sono formati proprietari che possono essere letti e modificati solo con word processor proprietari, SGML o XML per cui non è in genere disponibile la DTD o gli strumenti per il trattamento, e HTML, PostScript o PDF generato automaticamente da qualche word processor per il solo output.

La **"pagina del titolo"** di un libro stampato indica la pagina del titolo stessa, più qualche pagina seguente per quanto necessario a contenere in modo leggibile, il materiale che la licenza prevede che compaia nella pagina

del titolo. Per opere in formati in cui non sia contemplata esplicitamente la pagina del titolo, con "pagina del titolo" si intende il testo prossimo al titolo dell'opera, precedente l'inizio del corpo del testo.

Una sezione **"intitolata XYZ"** indica una parte del documento il cui titolo è esattamente XYZ o contiene XYZ in parentesi che segue il testo XYZ tradotto in un altra lingua (qui XYZ sta per uno specifico nome di sezione menzionato di seguito, come **"Ringraziamenti"**, **"Dediche"**, **"Approvazioni"** o **"Storia"**.) Per **"conservare il titolo"** di tale sezione, qualora si modificasse il documento, si intende che il titolo deve rimanere "intitolato XYZ" in accordo con tale definizione.

Il documento può includere dinieghi di garanzie (Warranty Disclaimers) in prossimità della nota che attesta l'applicazione di questa licenza al documento. Questi dinieghi di garanzie sono considerati parte della licenza, ma soltanto per quello che riguarda il diniego di garanzie: qualunque altra implicazione che essi possono avere è nulla e non ha effetto sull'applicazione della licenza.

2. COPIE LETTERALI

Si può copiare e distribuire il documento con l'ausilio di qualsiasi mezzo, per fini di lucro e non, fornendo per tutte le copie questa licenza, le note sul copyright e l'avviso che questa licenza si applica al documento, e che non si aggiungono altre condizioni al di fuori di quelle della licenza stessa. Non si possono usare misure tecniche per impedire o controllare la lettura o la produzione di copie successive alle copie che si producono o distribuiscono. Però si possono ricavare compensi per le copie fornite. Se si distribuiscono un numero sufficientemente elevato di copie si devono seguire anche le condizioni della sezione 3.

Si possono anche prestare copie e con le stesse condizioni sopra menzionate possono essere utilizzate in pubblico.

3. COPIARE IN NOTEVOLI QUANTITÀ

Se si pubblicano a mezzo stampa (o si copiano su altri supporti che in genere sono provvisti di copertine) più di 100 copie del documento, e la nota della licenza indica che esistono uno o più testi copertina, si devono includere nelle copie, in modo chiaro e leggibile, tutti i testi copertina indicati: il testo della prima di copertina in prima di copertina e il testo di quarta di copertina in quarta di copertina. Ambedue devono identificare l'editore che pubblica il documento. La prima di copertina deve presentare il titolo completo con tutte le parole che lo compongono egualmente visibili ed evidenti. Si può aggiungere altro materiale alle copertine. Il copiare con modifiche limitate alle sole copertine, purché si preservino il titolo e le altre condizioni viste in precedenza, è considerato alla stregua di copiare alla lettera.

Se il testo richiesto per le copertine è troppo voluminoso per essere riprodotto in modo leggibile, se ne può mettere una prima parte per quanto ragionevolmente può stare in copertina, e continuare nelle pagine immediatamente seguenti.

Se si pubblicano o distribuiscono copie opache del documento in numero superiore a 100, si deve anche includere una copia trasparente leggibile da un calcolatore per ogni copia o menzionare per ogni copia opaca un indirizzo di una rete di calcolatori pubblicamente accessibile in cui vi sia una copia trasparente completa del documento, spogliato di materiale aggiuntivo, e a cui si possa accedere anonimamente e gratuitamente per scaricare il documento usando i protocolli standard e pubblici generalmente usati. Se si adotta l'ultima opzione, si deve prestare la giusta attenzione, nel momento in cui si inizia la distribuzione in quantità elevata di copie opache, ad assicurarsi che la copia trasparente rimanga accessibile all'indirizzo stabilito fino ad almeno un anno di distanza dall'ultima distribuzione (direttamente o attraverso rivenditori) di quell'edizione al pubblico.

È caldamente consigliato, benché non obbligatorio, contattare l'autore del documento prima di distribuirne un numero considerevole di copie, per metterlo in grado di fornire una versione aggiornata del documento.

4. MODIFICHE

Si possono copiare e distribuire versioni modificate del documento rispettando le condizioni delle precedenti sezioni 2 e 3, purché la versione modificata sia realizzata seguendo scrupolosamente questa stessa licenza, con la versione modificata che svolga il ruolo del documento, così da estendere la licenza sulla distribuzione e la modifica a chiunque ne possieda una copia. Inoltre nelle versioni modificate si deve:

A. Usare nella pagina del titolo (e nelle copertine se ce ne sono) un titolo diverso da quello del documento, e da quelli di versioni precedenti (che devono essere elencati nella sezione storia del documento ove presenti). Si può usare lo stesso titolo di una versione precedente se l'editore di quella versione originale ne ha dato il permesso.

B. Elencare nella pagina del titolo, come autori, una o più persone o gruppi responsabili in qualità di autori delle modifiche nella versione modificata, insieme ad almeno cinque fra i principali autori del documento (tutti gli autori principali se sono meno di cinque) a meno che essi non permettano esplicitamente di non rispettare questo requisisto.

C. Dichiarare nella pagina del titolo il nome dell'editore della versione modificata in qualità di editore.

D. Conservare tutte le note sul copyright del documento originale.

E. Aggiungere un'appropriata licenza per le modifiche di seguito alle altre licenze sui copyright.

F. Includere immediatamente dopo la nota di copyright, un avviso di licenza che dia pubblicamente il permesso di usare la versione modificata nei termini di questa licenza, nella forma mostrata nell'addendum alla fine di questo testo.

G. Preservare in questo avviso di licenza l'intera lista di sezioni non modificabili e testi copertina richieste come previsto dalla licenza del documento.

H. Includere una copia non modificata di questa licenza.

I. Conservare la sezione intitolata "Storia", conserver il suo titolo, e aggiungere a questa un elemento che riporti al minimo il titolo, l'anno, i nuovi autori, e gli editori della versione modificata come figurano nella pagina del titolo. Se non ci sono sezioni intitolate "Storia" nel documento, createne una che riporti il titolo, gli autori, gli editori del documento come figurano nella pagina del titolo, quindi aggiungete un elemento che descriva la versione modificata come detto in precedenza.

J. Conservare l'indirizzo in rete riportato nel documento, se c'è, al fine del pubblico accesso ad una copia trasparente, e possibilmente l'indirizzo in rete per le precedenti versioni su cui ci si è basati. Questi possono essere collocati nella sezione "Storia". Si può omettere un indirizzo di rete per un'opera pubblicata almeno quattro anni prima del documento stesso, o se l'originario editore della versione cui ci si riferisce ne dà il permesso.

K. Per ogni sezione intitolata "Ringraziamenti" o "Dediche", si conservino il titolo, il senso, il tono della sezione stessa.

L. Si conservino inalterate le sezioni non modificabili del documento, nei propri testi e nei propri titoli. I numeri della sezione o equivalenti non sono considerati parte del titolo della sezione.

M. Si cancelli ogni sezione intitolata "Approvazioni". Solo questa sezione può non essere inclusa nella versione modificata.

N. Non si modifichi il titolo di sezioni esistenti come "Approvazioni" o per creare confusione con i titoli di sezioni non modificabili.

O. Si conservi qualsiasi diniego di garanzia.

Se la versione modificata comprende nuove sezioni di primaria importanza o appendici che ricadono in sezioni secondarie, e non contengono materiale

copiato dal documento, si ha facoltà di rendere non modificabili quante sezioni si voglia. Per fare ciò si aggiunga il loro titolo alla lista delle sezioni immutabili nella nota di copyright della versione modificata. Questi titoli devono essere diversi dai titoli di ogni altra sezione.

Si può aggiungere una sezione intitolata "Approvazioni", a patto che non contenga altro che le approvazioni alla versione modificata prodotte da vari soggetti–per esempio, affermazioni di revisione o che il testo è stato approvato da una organizzazione come la definizione normativa di uno standard.

Si può aggiungere un brano fino a cinque parole come Testo Copertina, e un brano fino a 25 parole come Testo di Retro Copertina, alla fine dell'elenco dei Testi Copertina nella versione modificata. Solamente un brano del Testo Copertina e uno del Testo di Retro Copertina possono essere aggiunti (anche con adattamenti) da ciascuna persona o organizzazione. Se il documento include già un testo copertina per la stessa copertina, precedentemente aggiunto o adattato da voi o dalla stessa organizzazione nel nome della quale si agisce, non se ne può aggiungere un altro, ma si può rimpiazzare il vecchio ottenendo l'esplicita autorizzazione dall'editore precedente che aveva aggiunto il testo copertina.

L'autore/i e l'editore/i del documento non ottengono da questa licenza il permesso di usare i propri nomi per pubblicizzare la versione modificata o rivendicare l'approvazione di ogni versione modificata.

5. UNIONE DI DOCUMENTI

Si può unire il documento con altri realizzati sotto questa licenza, seguendo i termini definiti nella precedente sezione 4 per le versioni modificate, a patto che si includa l'insieme di tutte le Sezioni Invarianti di tutti i documenti originali, senza modifiche, e si elenchino tutte come Sezioni Invarianti della sintesi di documenti nella nota relativa alla licenza, e che si preservino tutti i relativi dinieghi di garanzie.

Nella sintesi è necessaria una sola copia di questa licenza, e multiple sezioni invarianti possono essere rimpiazzate da una singola copia se identiche. Se ci sono multiple Sezioni Invarianti con lo stesso nome ma contenuti differenti, si renda unico il titolo di ciascuna sezione aggiungendovi alla fine e fra parentesi, il nome dell'autore o editore della sezione, se noti, o altrimenti un numero distintivo. Si facciano gli stessi aggiustamenti ai titoli delle sezioni nell'elenco delle Sezioni Invarianti nella nota di copiright della sintesi.

Nella sintesi si devono unire le varie sezioni intitolate "Storia" nei vari documenti originali di partenza per formare un'unica sezione intitolata "Storia"; allo stesso modo si proceda con le sezioni intitolate "Ringraziamenti", e "Dediche". Si devono eliminare tutte le sezioni intitolate "Approvazioni".

6. RACCOLTE DI DOCUMENTI

Si può produrre una raccolta che consista del documento e di altri realizzati sotto questa licenza; e rimpiazzare le singole copie di questa licenza nei vari documenti con una sola inclusa nella raccolta, solamente se si seguono le regole fissate da questa licenza per le copie alla lettera come se si applicassero a ciascun documento.

Si può estrarre un singolo documento da una raccolta e distribuirlo individualmente sotto questa licenza, solo se si inserisce una copia di questa licenza nel documento estratto e se si seguono tutte le altre regole fissate da questa licenza per le copie alla lettera del documento.

7. RACCOGLIERE INSIEME A LAVORI INDIPENDENTI

Una raccolta del documento o sue derivazioni con altri documenti o lavori separati o indipendenti, all'interno di o a formare un archivio o un supporto per la distribuzione, è detta "aggregato" se il copyright per l'intera raccolta non è utilizzato per limitare i diritti legali degli utenti della raccolta oltre a quelli che ogni lavoro individuale permette. Quando il documento è incluso in un aggregato, questa licenza non si applica agli altri lavori in esso contenuti, qualora non siano però loro stessi lavori derivati dal documento.

Se le esigenze del Testo Copertina della sezione 3 sono applicabili a queste copie del documento allora, se il documento è inferiore alla metà dell'intero aggregato i Testi Copertina del documento possono essere posti in copertine che delimitano solo il documento all'interno dell'aggregato, o l'equivalente elettronico delle copertine se il documento è in formato elettronico. Altrimenti devono apparire sulla copertina stampata relativa all'intero aggregato.

8. TRADUZIONI

La traduzione è considerata un tipo di modifica, e di conseguenza si possono distribuire traduzioni del documento seguendo i termini della sezione 4. Rimpiazzare sezioni non modificabili con traduzioni richiede un particolare permesso da parte dei detentori del diritto d'autore, ma si possono includere traduzioni di una o più sezioni non modificabili in aggiunta alle versioni originali di queste sezioni immutabili. Si può fornire una traduzione della presente licenza, e tutte le note ad essa relative presenti nel documento, ed ogni diniego di garanzia, a patto che si includa anche l'originale versione inglese di questa licenza e le versioni originali delle note relative e dei dinieghi di garanzia. In caso di discordanza fra la traduzione e la versione originale di questa licenza o nota o diniego, la versione originale prevale sempre.

Se una sezione del documento è intitolata "Riconoscimenti", "Dediche" o "Storia", il requisito (sezione 4) di conservarne il titolo (sezione 1) richiederà di modificarne l'attuale titolo.

9. TERMINI

Non si può applicare un'altra licenza al documento, copiarlo, modificarlo, o distribuirlo al di fuori dei termini espressamente previsti da questa licenza. Ogni altro tentativo di applicare un'altra licenza al documento, copiarlo, modificarlo, o distribuirlo è deprecato e pone fine automaticamente ai diritti previsti da questa licenza. Comunque, per quanti abbiano ricevuto copie o abbiano diritti coperti da questa licenza, essi non ne cessano se si rimane perfettamente coerenti con quanto previsto dalla stessa.

10. REVISIONI FUTURE DI QUESTA LICENZA

La Free Software Foundation può pubblicare nuove versioni rivedute della Licenza per Documentazione Libera GNU volta per volta. Qualche nuova versione potrebbe essere simile nello spirito alla versione attuale ma differire in dettagli per affrontare nuovi problemi e concetti. Si veda `http://www.gnu.org/copyleft`.

Ad ogni versione della licenza viene dato un numero che distingue la versione stessa. Se il documento specifica che si riferisce ad una versione particolare della licenza contraddistinta dal numero o "ogni versione successiva", si ha la possibilità di seguire termini e condizioni sia della versione specificata che di ogni versione successiva pubblicata (non come bozza) dalla Free Software Foundation. Se il documento non specifica un numero di versione particolare di questa licenza, si può scegliere ogni versione pubblicata (non come bozza) dalla Free Software Foundation.

Bibliografia

[1] F. Anklesaria, M. McCahill, P. Lindner, D. Johnson, D. Torrey, and B. Albert. RFC 1436: The Internet Gopher Protocol (a distributed document search and retrieval protocol), March 1993. Status: INFORMATIONAL.

[2] S. Bradner. RFC 2026: The Internet standards process — revision 3, October 1996. Status: BEST CURRENT PRACTICE.

[3] R. Droms. RFC 2131: Dynamic host configuration protocol, March 1997. Status: DRAFT STANDARD.

[4] K. Egevang and P. Francis. RFC 1631: The IP network address translator (NAT), May 1994. Status: INFORMATIONAL.

[5] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. RFC 2616: Hypertext Transfer Protocol – HTTP/1.1, June 1999. Status: INFORMATIONAL.

[6] V. Fuller, T. Li, J. Yu, and K. Varadhan. RFC 1519: Classless inter-domain routing (CIDR): an address assignment and aggregation strategy, September 1993. Status: PROPOSED STANDARD.

[7] V. Jacobson. RFC 1144: Compressing TCP/IP headers for low-speed serial links, February 1990. Status: PROPOSED STANDARD.

[8] B. Kantor. RFC 1282: BSD Rlogin, December 1991. Status: INFORMATIONAL.

[9] B. Kantor and P. Lapsley. RFC 977: Network news transfer protocol: A proposed standard for the stream-based transmission of news, February 1986. Status: PROPOSED STANDARD.

[10] J. C. Mogul and J. Postel. RFC 950: Internet Standard Subnetting Procedure, August 1985. Status: STANDARD.

[11] J. Myers and M. Rose. RFC 1939: Post Office Protocol — version 3, May 1996. Status: STANDARD.

[12] NetBIOS Working Group. RFC 1001: Protocol standard for a netBIOS service on a TCP/UDP transport: Concepts and methods, March 1987. Status: STANDARD.

[13] NetBIOS Working Group. RFC 1002: Protocol standard for a netBIOS service on a TCP/UDP transport: Detailed specifications, March 1987. Status: STANDARD.

[14] D. C. Plummer. RFC 826: Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware, November 1982. Status: STANDARD.

[15] J. Postel. RFC 768: User datagram protocol, August 1980. Status: STANDARD.

[16] J. Postel. RFC 791: Internet Protocol, September 1981. Status: STANDARD.

[17] J. Postel. RFC 792: Internet Control Message Protocol, September 1981. Status: STANDARD.

[18] J. Postel. RFC 793: Transmission control protocol, September 1981. Status: STANDARD.

[19] J. Postel. RFC 821: Simple mail transfer protocol, August 1982. Status: STANDARD.

[20] J. Postel and J. K. Reynolds. RFC 854: Telnet Protocol specification, May 1983. Status: STANDARD.

[21] J. Postel and J. K. Reynolds. RFC 959: File transfer protocol, October 1985. Status: STANDARD.

[22] J. Reynolds and R. Braden. RFC 2600: INTERNET OFFICIAL PROTOCOL STANDARDS, March 2000. Status: STANDARD.

[23] J. L. Romkey. RFC 1055: Nonstandard for transmission of IP datagrams over serial lines: SLIP, June 1988. Status: STANDARD.

[24] W. Simpson. RFC 1661: The point-to-point protocol (PPP), July 1994. Status: STANDARD.

[25] W. Simpson. RFC 1994: PPP challenge handshake authentication protocol (CHAP), August 1996. Status: DRAFT STANDARD.

[26] M. Wahl, T. Howes, and S. Kille. RFC 2251: Lightweight Directory Access Protocol (v3), December 1997. Status: PROPOSED STANDARD.