# Appunti di Architettura Reti

Antonio Strippoli

Basati su:
- Le slide del prof. Osvaldo Gervasi
- Reti di Calcolatori (A. Tanenbaum)

Revisore:
Alessio Amatucci

*"Degli appunti per domarli, degli appunti per trovarli,*
*degli appunti per ghermirli e nel buio incatenarli"*

# Obiettivi

L'obiettivo principale dei presenti appunti era la preparazione personale alla disciplina "Architettura Reti" insegnata dal prof. Osvaldo Gervasi dell'Università di Perugia.

Non conoscendo nulla riguardo l'architettura di una rete ed i suoi protocolli, il materiale reso disponibile dal docente (slides) si è rivelato insufficiente per un ripasso o uno studio nei casi in cui non sono stato presente a lezione. Anche altri appunti resi gentilmente disponibili da altri studenti non si sono rivelati chiari in molti punti.

Per cui, l'obiettivo di questi appunti è fornire una buona visione d'insieme di ogni argomento presente nel programma 2019/20 di Architettura Reti, tenendo a mente che il lettore potrebbe non aver mai letto nulla riguardo la suddetta materia e fornendo quindi anche definizioni basilari e spiegazioni base.

Ove opportuno, alcuni argomenti sono presentati in forma anche leggermente più estesa solamente a scopo di chiarimento. Tuttavia, altri argomenti sono particolarmente dettagliati e non tutti i dettagli sono ovviamente affrontati durante le lezioni. In questi casi, si è deciso di limitarsi ad una visione d'insieme più che sufficiente per la preparazione all'esame, ma che potrebbe non risultare esaustiva per gli studenti più curiosi. Questi argomenti sono singolarmente segnalati in ogni paragrafo qualora si volesse approfondire.

Inoltre, è bene segnalare che non viene trattato l'intero programma dell'anno 2019/2020, in quanto sono stati tralasciati i seguenti argomenti:

- Topologie di reti;
- Livello fisico/data link OSI (mezzi trasmissivi, ethernet, metodi di accesso al bus…);
- Frame Relay e ATM;
- VPN.

Si noti che è comunque possibile contribuire alla repository dedicata su GitHub (https://github.com/CoffeeStraw/Appunti-Architettura-Reti) per ultimare e tenere aggiornati i presenti appunti, modificando il file .docx con cui sono stati scritti e rigenerando il .pdf utilizzato per la distribuzione.

# Indice

**1 Definizioni Generali ..... 1**
1.1 Rete ..... 1
1.2 Internet ..... 1
1.3 Intranet ..... 1
1.4 Extranet ..... 1
1.5 ISP ..... 1
1.6 Protocollo ..... 1
1.7 Modello client/server ..... 1
1.8 Gateway ..... 2
1.9 LAN ..... 2
1.10 MAN ..... 2
1.11 WAN ..... 2
1.12 Servizio connection oriented ..... 2
1.13 Servizio connectionless ..... 2
1.14 Servizio best effort ..... 3
1.15 Quality of Service ..... 3
1.16 Unicasting, Multicasting e Broadcasting ..... 3
1.17 RFC ..... 3
**2 Le Architetture di Rete ..... 4**
2.1 Il modello di riferimento ISO/OSI ..... 5
2.2 Il modello di riferimento TCP/IP ..... 6
**3 Internet Protocol (IP) ..... 6**
3.1 Datagram IP ..... 6
*3.1.1 IPv4 ..... 7*
*3.1.2 Subnetting ..... 11*
*3.1.3 Supernetting ..... 11*
*3.1.4 Piano di indirizzamento IP ..... 11*
*3.1.5 IPv6 ..... 12*
3.2 Configurazione IP ..... 16
3.3 Internet multicasting ..... 16
*3.3.1 IGMP ..... 16*

**4 Protocolli di controllo Internet ..... 17**
4.1 ICMP ..... 17
4.2 ARP ..... 18
4.3 RARP ..... 18
**5 Routing ..... 19**
5.1 Algoritmi di routing ..... 19
*5.1.1 Routing table ..... 20*
*5.1.2 Routing in una internetwork ..... 21*
5.2 Classificazione di algoritmi adattivi ..... 21
*5.2.1 Algoritmi di tipo distance-vector (vettore-distanza) ..... 21*
*5.2.2 Algoritmi di tipo link-state (stato dei collegamenti) ..... 22*
5.3 OSPF ..... 22
5.4 RIP ..... 24
5.5 BGP ..... 24
**6 Protocolli di trasporto Internet ..... 26**
6.1 UDP ..... 26
6.2 TCP ..... 27
**7 Servizi di Rete ..... 30**
7.1 Telnet ..... 30
7.2 Comandi r ..... 30
7.3 FTP ..... 31
7.4 SSH ..... 32
7.5 DNS ..... 33
*7.5.1 Lo spazio dei nomi DNS ..... 33*
*7.5.2 Risoluzione ..... 34*
*7.5.3 BIND ..... 34*
7.6 NIS ..... 35
7.7 NFS ..... 35
7.8 SNMP ..... 36
7.9 DHCP ..... 37
7.10 NAT ..... 37

**8 Posta Elettronica ..... 38**
8.1 RFC822 ..... 39
*8.1.1 MIME ..... 39*
8.2 Il trasferimento dei messaggi: SMTP ..... 40
8.3 POP3 ..... 41
8.4 IMAP ..... 41
8.5 Posta elettronica privata ..... 42
*8.5.1 PGP ..... 42*
**9 News ..... 42**
9.1 NNTP ..... 42
**10 World Wide Web ..... 43**
10.1 HTTP ..... 43
10.2 Pagine Web dinamiche ..... 44
**11 Sicurezza delle reti ..... 45**
11.1 SSL ..... 47
11.2 Firewall ..... 48
*11.2.1 Packet Filter ..... 48*

# 1 Definizioni Generali

## 1.1 Rete

Una **rete** è un insieme di dispositivi collegati in modo da permettere lo scambio di dati e la comunicazione tra più utenti. I dati vengono trasferiti sotto forma di **pacchetti** (a volte detti **PDU**, **Packet Data Unit**).

Si distingue da un **sistema distribuito**, che invece appare ai propri utenti come un singolo sistema coerente.

## 1.2 Internet

**Internet (o Internetwork)** è l'interconnessione di reti di varia natura che consente lo scambio di informazione rappresentata in forma digitale, cioè come sequenza di cifre binarie (bit).

Si può dire che **un computer è su Internet** se esegue la pila di protocolli TCP/IP, ha un indirizzo IP, e può spedire pacchetti IP a tutti gli altri computer su Internet (questi argomenti saranno trattati nei capitoli successivi).

## 1.3 Intranet

**Intranet** è un sistema telematico di collegamento effettuato con le stesse modalità di Internet, ma riservato a un circuito chiuso di utenti (all'interno di aziende, di strutture pubbliche, di organizzazioni di ricerca ecc.).

## 1.4 Extranet

L'**Extranet** è una Intranet estesa ad alcuni soggetti non operanti nella stessa rete (p.e. clienti, fornitori, consulenti).

## 1.5 ISP

Un **ISP (Internet Service Provider)** indica un'organizzazione o un'infrastruttura che mette a disposizione dei servizi inerenti a Internet per degli utenti, come la posta elettronica o l'accesso al World Wide Web.

## 1.6 Protocollo

Informalmente, un **protocollo** è un accordo, tra le parti che comunicano, sul modo in cui deve procedere la comunicazione. Violare il protocollo rende la comunicazione più difficile, se non del tutto impossibile.

Formalmente, un **protocollo** è un insieme di regole che definisce l'interazione tra sistemi.

## 1.7 Modello client/server

Il **modello client/server** è un modello in cui sono presenti due entità (client e server), nel quale il client si connette al server mediante una rete, per la fruizione di un certo servizio (come la condivisione dati).

## 1.8 Gateway

Un gateway è un dispositivo di rete (generalmente un router) che collega due reti eterogenee. Il suo scopo principale è quello di veicolare i pacchetti di rete all'esterno di una rete locale.

## 1.9 LAN

Le **LAN (Local Area Network)** sono reti private installate all'interno di un singolo edificio o campus, con dimensione fino a qualche Km, ed hanno come scopo la condivisione di risorse (come stampanti) e lo scambio di informazioni.

## 1.10 MAN

Una **MAN (Metropolitan Area Network)** è una rete che copre un'intera città e collega più LAN geograficamente vicine. Di solito si tratta di singole filiali di un'azienda, che vengono connesse ad una MAN attraverso l'affitto di linee dedicate.

## 1.11 WAN

Una **WAN (Wide Area Network)** è una rete che copre un'area geograficamente estesa, spesso una nazione o un continente. Il numero di reti locali o singoli computer che si possono connettere ad una singola WAN è teoricamente illimitato.

## 1.12 Servizio connection oriented

Un servizio **connection oriented (orientato alla connessione)** è un servizio di rete in cui l'utente che lo vuole usare deve stabilire una connessione (mediante la creazione di un circuito, che sia fisico o virtuale), usarla e quindi rilasciarla. Nella maggior parte dei casi, l'ordine dei bit inviati è conservato e arrivano nella sequenza con cui sono stati trasmessi, che rende questa categoria di servizi **affidabile**.

Un'analogia utile per capire, è quella tra il servizio connection oriented e il **sistema telefonico**. Per parlare con qualcuno si deve prendere il telefono, comporre il numero, parlare e poi riagganciare.

## 1.13 Servizio connectionless

Un servizio **connectionless (senza connessione)** si contrappone al servizio con connessione e non viene quindi stabilita una connessione. I dati vengono instradati in maniera indipendente l'uno dall'altro, senza verificare che il destinatario sia raggiungibile e senza controllare che i dati arrivino nell'ordine desiderato, che rende questa categoria di servizi **inaffidabile**.

Un'analogia utile per capire, è quella tra il servizio connectionless e la **posta**. Ogni messaggio (lettera) trasporta l'indirizzo completo del destinatario ed è instradato attraverso il sistema postale in modo indipendente dagli altri. Normalmente, quando si mandano due messaggi alla stessa destinazione, il primo inviato è anche il primo ad arrivare; ma è possibile che incontri un ritardo, e quindi arrivi dopo il secondo.

## 1.14 Servizio best effort

Un servizio **best effort** (**massimo impegno**, interpretato "come va, va") è un servizio inaffidabile che non offre alcuna garanzia di consegna dei pacchetti (alcuni di essi possono perdersi e avere bisogno di essere ritrasmessi, o andare fuori sequenza) né di controllo di errore, di flusso e di congestione.

## 1.15 Quality of Service

Il termine **Quality of Service** (abbreviato **QoS**) è utilizzato per indicare i parametri usati per caratterizzare la qualità del servizio offerto da una rete (ad esempio perdita di pacchetti, ritardo), o gli strumenti o tecniche per ottenere una qualità di servizio desiderata. Si contrappone al termine best effort.

## 1.16 Unicasting, Multicasting e Broadcasting

**Unicasting, Multicasting e Broadcasting** sono delle metodologie per la trasmissione dei dati. La distinzione avviene in base al numero dei ricevitori:

- **Unicasting**: trasmissione di dati tra un trasmettitore e un ricevitore (1-1);
- **Multicasting**: trasmissione di dati tra un trasmettitore ed un sottoinsieme di tutte le macchine della rete (1-n);
- **Broadcasting**: trasmissione di dati tra un trasmettitore e tutte le macchine della rete (1-tutti).

## 1.17 RFC

Gli **RFC (Request For Comment)**, sono una serie di rapporti tecnici numerati cronologicamente che riportano informazioni o specifiche riguardanti nuove ricerche, innovazioni e metodologie di Internet. Sono redatti da un organismo internazionale chiamato **IETF (Internet Engineering Task Force)**, e sono tutti consultabili su www.ietf.org/rfc/.

# 2 Le Architetture di Rete

Esistono due importanti architetture di rete: il modello di riferimento **OSI** e il modello **TCP/IP**. Anche se i protocolli associati al modello OSI ormai sono in disuso, il modello in sé ha valore generale ed è ancora valido, e le caratteristiche discusse per ogni livello sono ancora molto importanti. Il modello TCP/IP ha caratteristiche opposte: il modello in sé è poco utilizzabile, ma i protocolli sono largamente impiegati; per questo motivo vanno esaminati entrambi in dettaglio. A volte s'impara più dai fallimenti che dai successi.

*OSI*

| | |
|---|---|
| **Livello 7** | Applicazione |
| **Livello 6** | Presentazione |
| **Livello 5** | Sessione |
| **Livello 4** | Trasporto |
| **Livello 3** | Rete |
| **Livello 2** | Data Link |
| **Livello 1** | Fisico |

*TCP/IP*

| |
|---|
| Applicazione |
| (non presente) |
| (non presente) |
| Trasporto |
| Internet |
| Host-to-network |

## 2.1 Il modello di riferimento ISO/OSI

L'ISO (International Standards Organization) è un organo consulente dell'ONU, che promuove lo sviluppo di standardizzazioni nel mondo. È il creatore dell'**OSI** (***Open Systems Interconnections***), così chiamato perché riguarda la connessione di sistemi aperti, cioè sistemi che sono "aperti" verso la comunicazione con altri. È uno standard ***de jure*** ("per legge", ovvero formale, adottato da qualche organismo di standardizzazione autorizzato, in questo caso ISO).
È caratterizzato da **7 livelli**: ogni livello sfrutta i servizi dei livelli inferiori. La comunicazione tra livelli adiacenti avviene tramite **i NAP (Neutral Access Points)**, mentre la comunicazione tra entità di livelli diversi avviene tramite il **SAP (Service Access Point)**. Le operazioni specifiche di un livello sono realizzate tramite un insieme di protocolli. Di seguito una breve descrizione di ogni livello:

**Physical Layer:** si occupa della trasmissione di bit grezzi sul canale di comunicazione. È l'insieme di regole che specificano le connessioni elettriche e fisiche tra i dispositivi. Definisce la specifica dei cavi e del tipo di segnale elettrico associato ai vari pin.

**Data Link Layer:** definisce l'accesso al mezzo specificato nel Physical Layer, il formato dei dati ed è responsabile dell'invio affidabile delle informazioni, ovvero gestisce tra le altre cose la frammentazione dei dati e le procedure di controllo di possibili errori del livello fisico. Appartengono a questo livello i protocolli Data Link (DLCP, BSC, HDLC...) e sono inoltre presenti anche i sottolivelli LLC e MAC.

**Network Layer:** si occupa della connessione tra due nodi della rete (detti nodo sorgente e nodo destinatario). Gestisce il routing e lo scambio di informazione tra i nodi. I servizi associati a questo livello sono legati al movimento dei dati nella rete, inclusi l'indirizzamento, il routing e le procedure di controllo dei flussi (vengono definite le raccomandazioni X.25 e X.75). A questo livello appartiene il protocollo IP.

**Transport Layer:** è il garante del trasferimento delle informazioni. Analizza il traffico fra i nodi controllando gli errori, la sequenza e i fattori di affidabilità dello scambio. A questo livello appartengono i protocolli TCP e UDP. È il primo livello **end-to-end**.

**Session Layer:** regolarizza l'inizio e la fine dei flussi dei dati fra i nodi. Si occupa dell'organizzazione del dialogo tra i programmi applicativi.

**Presentation Layer:** si occupa di effettuare trasformazione sui dati compatibilmente con il dispositivo di ricezione. Esempi di trasformazione riguardano crittografia e compressione.

**Application Layer:** è l'ultimo livello, comprende tutti i programmi applicativi che consentono l'uso della rete: fa da interfaccia tra rete e utente. Esempi di funzioni svolte da questo livello possono essere: terminale virtuale, trasferimento di file, posta elettronica, condivisione di risorse e accesso a database.

## 2.2 Il modello di riferimento TCP/IP

Il **TCP/IP** è così chiamato per ricordare i suoi protocolli più importanti: il Transmission Control Protocol (TCP) e l'Internet Protocol (IP). È uno standard ***de facto*** ("dalla realtà", ovvero stabilito senza piani formali). Di seguito una breve descrizione di ogni livello:

**Host-to-network Layer:** "il grande vuoto", nel modello di riferimento TCP/IP non viene specificato cosa accade in questo territorio, si limita a segnalare che l'host deve collegarsi alla rete usando qualche protocollo che gli permetta di spedire pacchetti IP. Questo protocollo però non è definito e varia da host a host.

**Internet Layer:** il suo scopo è quello di consentire agli host di mandare pacchetti in qualsiasi rete, e farli viaggiare in modo indipendente l'uno dall'altro fino alla destinazione (che magari è su una rete diversa). È importante notare che viene definito il protocollo **IP**.

**Transport Layer:** è progettato per consentire la comunicazione tra entità pari degli host sorgente e destinazione, come nel livello trasporto OSI. Sono definiti due protocolli di trasporto end-to-end: **TCP** e **UDP**.

**Application Layer:** nel modello TCP/IP non ci sono sessione e presentazione, poiché si è notato che nella maggior parte dei casi sono inutili. Come nel modello OSI, il livello Applicazione contiene un gran numero di programmi che si interfacciano con l'utente.

# 3 Internet Protocol (IP)

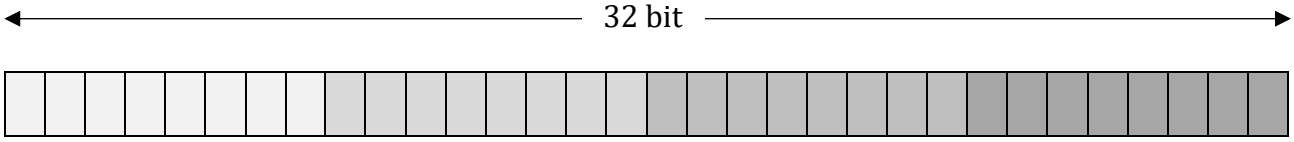
L'IP è la colla che tiene unito Internet. È il protocollo **connectionless** e **best effort** del livello 3 (Rete) e permette di interconnettere reti eterogenee: per questo è implementato sopra ai protocolli di livello Data Link, svolgendo la funzione di routing (ovvero scegliere il percorso che dovranno seguire i dati, spiegato in seguito).

## 3.1 Datagram IP

Il pacchetto utilizzato dall'IP è detto **datagram IP** (per analogia con il servizio telegrafico, *telegram service*) ed è costituita da una parte di intestazione (detta **header**) e da una parte di testo (detta **data**). La versione attualmente utilizzata è la v4, anche se al momento è in corso una transizione da IPv4 a IPv6, che ha avuto inizio anni fa e il cui completamento richiederà ancora parecchio tempo (o addirittura mai). Di seguito viene analizzato l'header dell'IPv4.

### 3.1.1 IPv4

L'intestazione ha una parte fissa di 20 byte e una parte opzionale di lunghezza variabile ed è trasmessa in ordine *big endian* (da sinistra a destra). Sulle macchine *little endian* è necessario eseguire una conversione software sia in ricezione che in trasmissione.




| Version | IHL | Type of Service | | | Total Length | | | |
|---|---|---|---|---|---|---|---|---|
| Identification | | | | | | D F | M F | Fragment Offset |
| Time to Live | | Protocol | | | Header Checksum | | | |
| Source address | | | | | | | | |
| Destination address | | | | | | | | |
| Opzioni (0 o più word) | | | | | | | | |

- **Version**: campo di 4 bit che indica la versione IP del datagram (attualmente IPv4);
- **IHL (Internet Header Length)**: campo di 4 bit che indica la lunghezza dell'header espressa in parole da 32 bit. Necessario poiché, come anticipato, la lunghezza dell'intestazione non è costante;
- **Type of Service**: campo di 8 bit che indica come deve essere gestito il datagram;
- **Total Length**: campo lungo 16 bit che identifica la lunghezza totale del datagram;
- **Identification, flag, offset**: controllano frammentazione e riassemblaggio del datagram;
- **TTL (Time To Live)**: indica la durata in secondi concessa al datagram per restare in rete, evitando che un datagram giri a vuoto per sempre, evento che potrebbe accadere in caso di danneggiamento delle tabelle di routing;
- **Protocol**: è un codice che identifica il protocollo utilizzato nel campo dati (intuitivamente, può essere associato all'estensione nel nominativo di un file);
- **Header Checksum**: garantisce il controllo dell'integrità dell'header;
- **Source address, Destination address:** rispettivamente indirizzo sorgente e indirizzo di destinazione, indicano il numero di rete e il numero di host, espressi tramite indirizzi IP;
- **Options (opzioni)**: via di fuga per dare alle versioni successive del protocollo la possibilità d'includere informazioni non presenti nel progetto originale.

**Indirizzo IPv4**

Un indirizzo IP (in entrambe le sue versioni) permette di identificare univocamente **la connessione di un host alla rete**. L'indirizzo IPv4 è espresso con **32 bit**, è **diviso in 2 parti** (rete ed host) ed è utilizzato nei campi *Source address* e *Destination address* dei pacchetti IP. È importante notare che un indirizzo IP non si riferisce veramente a un host, ma a una **scheda di rete**, perciò quando un host ha due schede di rete, deve avere due indirizzi IP. Comunque, la maggior parte degli host ha un'unica scheda di rete e perciò ha un solo indirizzo IP.

Un indirizzo IP si esprime in **notazione decimale a punti**, dove ognuno dei 4 byte è rappresentato con un numero che varia tra 0 e 255, del tipo:

***x.y.z.w***

Un indirizzo IP può essere:

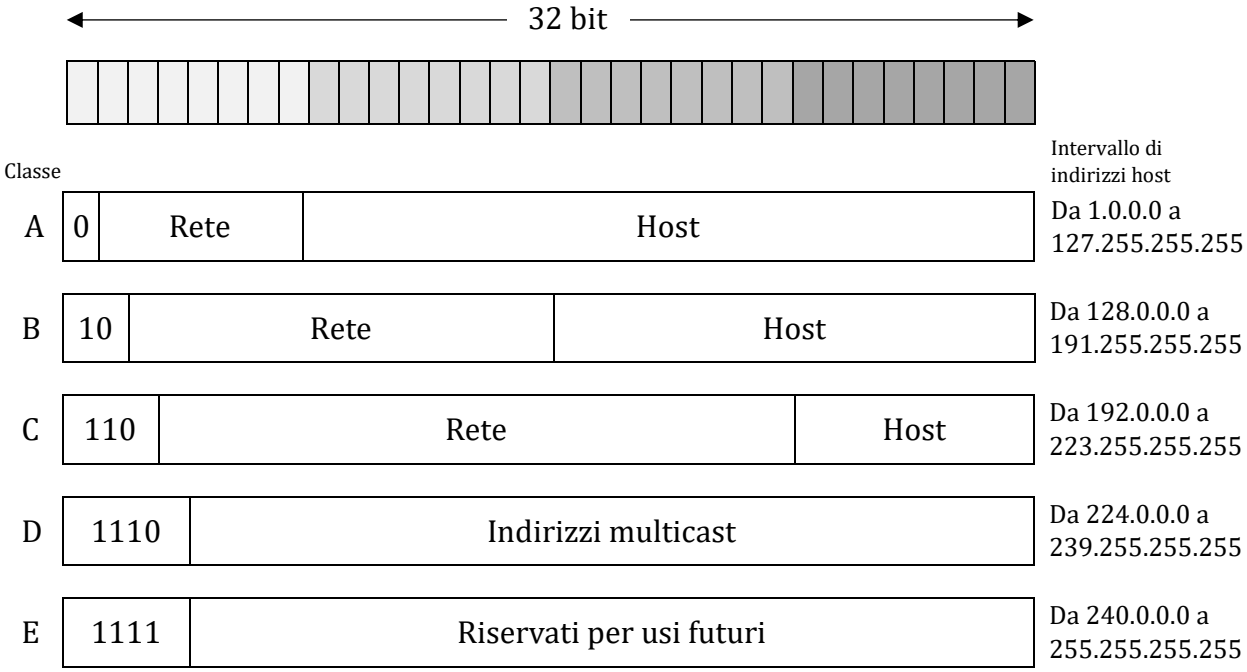
- **Pubblico**: rilasciato dall'ICANN, indica un indirizzamento di rete destinato a reti **globali**;
- **Privato**: indica un indirizzamento di rete destinato a reti **locali**;
- **Statico**: l'IP dell'host **non varia** nel tempo;
- **Dinamico**: l'IP dell'host **varia** nel tempo.

**Come si ottiene un indirizzo IP?**

Un indirizzo IP può essere assegnato automaticamente utilizzando il servizio DHCP (Dynamic Host Configuration Protocol) oppure in modo manuale.

**Classificazione di Indirizzi IP**

Per molti decenni, gli indirizzi IP sono stati divisi in cinque categorie: A, B, C, D, E. È bene notare che oggi il sistema non è più utilizzato poiché permetteva parecchi sprechi di indirizzi IP, sempre più preziosi con l'aumentare della richiesta. Come approfondimento, il nuovo sistema attualmente in uso è il **CIDR (Classless Inter-Domain Routing)**.




A) Classe utilizzata per le WAN e le MAN;
B) Classe utilizzata per le MAN e le grosse LAN;
C) Classe utilizzata per le LAN;
D) Classe utilizzata per gli indirizzi multicast;
E) Classe non commerciale (riservata).

**Indirizzi IP speciali**

Esistono alcuni indirizzi IP che hanno un significato speciale. Uno schema di essi, è visibile nella tabella sottostante.

| | | |
|---|---|---|
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | | Questo host |
| 00 . . . 0 0 | Host | Un host su questa rete |
| 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 | | Trasmissione broadcast sulla rete locale |
| Rete | 1 1 1 1 . . . 1 1 1 1 | Trasmissione broadcast su una rete distante |
| 127 | (Qualsiasi cosa) | Loopback |

L'indirizzo IP **0.0.0.0** è utilizzato dagli host al momento del boot. Gli indirizzi IP di tipo C che hanno lo **0 come numero di rete** si riferiscono alla rete corrente. Grazie a questi indirizzi, i computer possono utilizzare la rete locale senza conoscerne il numero. L'indirizzo composto da **tutti 1** permette la trasmissione broadcast sulla rete locale, in genere una LAN. Gli indirizzi con **un numero di rete opportuno e tutti 1** nel campo host, permettono ai computer l'invio di pacchetti broadcast a reti LAN distanti collegate a Internet (però molti amministratori di rete disattivano questa funzionalità). Infine, tutti gli indirizzi espressi nella forma **127.xx.yy.zz** sono riservati per le prove di loopback. I pacchetti diretti all'indirizzo di loopback non sono immessi nel cavo, ma elaborati localmente e trattati come pacchetti in arrivo.

### 3.1.2 Subnetting

Può rivelarsi estremamente utile (come in casi in cui una singola rete acquisisce dimensioni parecchio elevate) dividere internamente una rete in più parti, facendo in modo che il mondo esterno veda ancora una singola rete. Questa pratica è chiamata **subnetting** e le parti della rete sono chiamate **subnet (sottoreti)**.

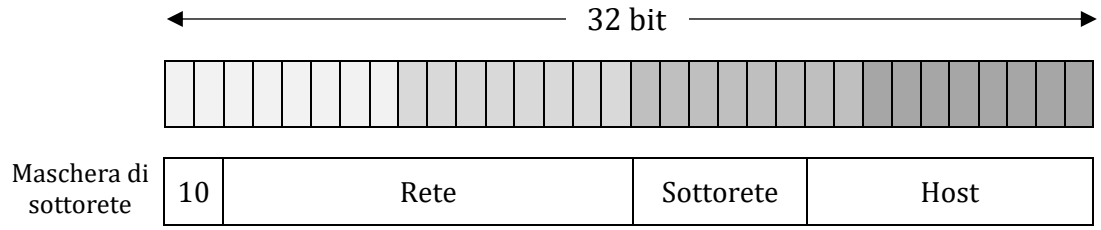
Affinché un router implementi le sottoreti e sia in grado di riconoscere il numero di host raggiungibili all'interno di ognuna, si definisce la **maschera di sottorete (subnet mask)**, un numero che, preso un indirizzo IP, indica il punto di demarcazione tra i bit di rete e i bit di host.

Come l'indirizzo IP, la subnet mask è un **numero a 32 bit** che si può indicare in notazione decimale a punti oppure come un unico numero decimale, indicante il numero di bit della parte di indirizzo della rete.

Normalmente indirizzo IP e subnet mask sono accoppiati nella forma **a.b.c.d/m**, dove *m* è la subnet mask in forma decimale, che è anche possibile ottenere in notazione decimale a punti: basta infatti porre i primi m bit significativi di un numero a 32 bit pari a 1 ed i rimanenti pari a 0, per poi raggruppare in ottetti e convertire in decimale.

Effettuando l'AND logico bit a bit tra la subnet mask e l'indirizzo IP, si ottiene l'indirizzo IP della sottorete.

Si noti che effettuare subnetting, vuol dire conseguentemente **aumentare** i bit della parte **rete** dell'indirizzo e **diminuire** i bit della parte **host**.




*Esempio di una rete di classe B divisa in 64 sottoreti.*
*La subnet mask è esprimibile come 255.255.252.0 o come /22.*

### 3.1.3 Supernetting

Analogamente, esiste il concetto di **supernetting**, che consiste nell'accorpare più reti fisiche primarie diverse in un'unica rete. Fare supernetting può risultare utile, ad esempio, per semplificare le informazioni di routing da trasmettere ai router.

Al contrario del subnetting, effettuare supernetting vuol dire conseguentemente **aumentare** i bit della parte **host** dell'indirizzo e **diminuire** i bit della parte **rete**.

### 3.1.4 Piano di indirizzamento IP

Il piano di indirizzamento IP è un documento che l'amministratore di rete deve scrivere e tenere aggiornato per descrivere l'utilizzo del proprio spazio di indirizzamento IP.

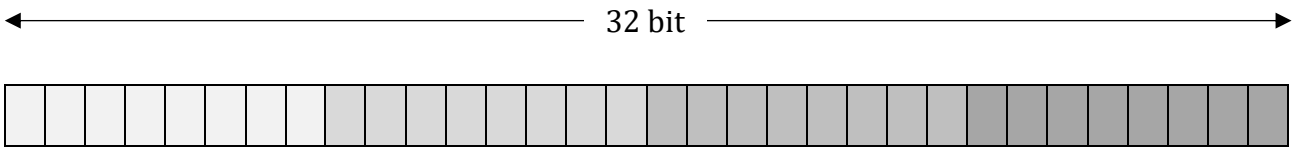
### 3.1.5 IPv6

L'IPv6 è la nuova versione di IP che non dovrebbe mai esaurire i suoi indirizzi IP, risolvere una varietà di problemi in IPv4 ed essere più flessibile ed efficiente, permettendo quindi una reale connettività globale, eliminando reti o host nascosti.

Nello specifico, sono 4 i cambiamenti portati da questo nuovo standard:

- **Indirizzi da 16 byte** (128 bit), il che si traduce in una scorta praticamente illimitata di indirizzi IP;
- **Header semplificato** (7 campi contro 13 di IPv4), che permette di elaborare i pacchetti più velocemente;
- **Migliore supporto per le opzioni**, necessario poiché campi prima necessari ora sono opzionali;
- **Incremento della sicurezza**, migliorando autenticazione e riservatezza.

Di seguito è descritta l'intestazione dell'IPv6.




| Version | Traffic class | Flow label | |
|---|---|---|---|
| Payload length | | Next header | Hop limit |
| Source address (16 bytes) | | | |
| Destination address (16 bytes) | | | |

- **Version**: campo di 4 bit che indica la versione IP del datagram (che sarà sempre 6);
- **Traffic class**: utilizzato per distinguere i pacchetti con diversi requisiti di distribuzione in tempo reale (simile a *Type of Service* di IPv4);
- **Flow label**: sebbene il suo utilizzo non sia ancora chiaro, consentirebbe a una sorgente e a una destinazione di impostare una pseudo-connessione con particolari proprietà e requisiti;
- **Payload length (lunghezza del carico utile)**: numero di byte del campo data;

- **Next header**: è il segreto con cui è stato possibile semplificare l'header dell'IPv4. Permette di indicare header estesi (opzionali) che seguono l'intestazione corrente. Se però l'intestazione corrente è l'ultima intestazione IP, il campo next header indica il gestore del protocollo di trasporto *TCP* o *UDP* al quale va passato il pacchetto (come il campo Protocol dell'IPv4);

- **Source address e Destination address**: come in IPv4, indicano l'indirizzo IP sorgente e l'indirizzo IP di destinazione.

Si noti che i campi riguardanti la frammentazione in IPv6 non sono più presenti, poiché IPv6 la gestisce in maniera diversa, così come il campo checksum è stato eliminato, perché l'elaborazione di questa informazione riduceva enormemente le prestazioni, che con le reti sempre più affidabili odierne, risulta quasi totalmente superfluo.

**Header estesi IPv6**

Come accennato prima, IPv6 implementa il concetto di header. Alcuni dei campi mancanti di IPv4 sono ancora saltuariamente necessari, perciò IPv6 ha introdotto il concetto di intestazione estesa (opzionale), chiamata anche extension header. Queste intestazioni possono essere fornite per offrire informazioni aggiuntive, codificate in modo efficiente. Al momento sono definiti sei tipi di intestazioni estese:

- **Hop-by-hop**: utilizzata per le informazioni che tutti i router lungo il percorso devono esaminare.
- **Destination options**: utilizzata dai campi che devono essere interpretati dall'host di destinazione;
- **Routing**: elenca uno o più router che devono essere visitati lungo la strada;
- **Fragment**: si occupa delle informazioni di frammentazione proprio come fa IPv4;
- **Authentication**: fornisce un meccanismo che garantisce al ricevente di un pacchetto l'autenticità del mittente;
- **Encapsulating Security Payload**: permette di proteggere con la crittografia il contenuto del pacchetto in modo da renderlo leggibile solo ai destinatari desiderati.

**Indirizzi IPv6**

Gli indirizzi IPv6 ovviamente svolgono ancora la stessa funzione degli indirizzi IPv4. Ciò nonostante, avendo ora a disposizione 128 bit per essere rappresentati, la loro definizione va rivista. Per scrivere gli indirizzi a 16 byte è stata escogitata una nuova notazione. Gli indirizzi sono scritti come otto gruppi di quattro cifre esadecimali separate da due punti:

**a:b:c:d:e:f:g:h**

Sono ammesse alcune scritture facilitate:

- È possibile omettere gli zero iniziali di un gruppo, ad esempio 0123 -> 123;
- Uno o più gruppi contenenti 16 bit a zero possono essere sostituiti da due punti, ma ciò è concesso solo una volta ad indirizzo. Ad esempio: 8000:0000:0000:0000:0123:4567:89AB:CDEF -> 8000::123:4567:89AB:CDEF;

Inoltre, gli indirizzi IPv4 possono essere scritti in notazione decimale a punti dopo una coppia di due punti, ad esempio ::192.31.20.46.

Per quanto riguarda la scrittura di un indirizzo IPv6 in un URL, esso deve essere scritto fra parentesi quadre. Ad esempio: http://[2001:1:4F3A::206:AE14]:8888/index.html

**Tipi di indirizzi IPv6**

Gli indirizzi IPv6 possono essere di vario tipo: **Unicast**, **Multicast**, **Anycast**.

**Unicast**

Uno a uno: è un indirizzo IP che identifica una sola interfaccia di rete e quindi un unico destinatario. L'indirizzo IPv6 unicast è diviso in 2 parti: i primi 64 bit identificano il prefisso di rete e vengono detti **Subnet Prefix**, i restanti 64 identificano l'host e vengono detti quindi **Host Identifier**. Gli indirizzi di tipo unicast possono essere:

- **Unspecified**: assenza di indirizzo, può essere usato nella richiesta iniziale DHCP per ottenere un indirizzo (formato: 0:0:0:0:0:0:0:0 oppure ::);
- **Loopback**: identifica l'host stesso, equivalente del 127.0.0.0 in IPv4 (formato: 0:0:0:0:0:0:0:1 oppure ::1); Per controllare se lo stack IPv6 funziona, è possibile eseguire infatti il comando `ping6 ::1`;
- **Link Local**: è uno *scoped address*, una novità di IPv6, ovvero un indirizzo di rete che è valido solo per la comunicazione all'interno di un segmento di rete (link). Fornisce ad ogni nodo un indirizzo IPv6 per iniziare le comunicazioni (formato: FE80:0:0:0:<interface identifier>);
- **Site Local**: è anch'esso uno *scoped address*, in questo caso l'ambito dell'indirizzo è un ***site*** (una rete di link). Non è configurato di default e può essere utilizzato per l'indirizzamento privato (format: FEC0:0:0:<subnet id>:<interface id>).

**Multicast**

Uno a molti: è un indirizzo IP che identifica un insieme di interfacce di rete e quindi un insieme di destinatari. Si noti che non esiste il broadcast in IPv6, ed al suo posto si usa il multicast. Il suo formato è FF<flags><scope>::<multicast group>, dove flag può essere 0 (permanente) o 1 (temporaneo).

**Anycast**

Uno al più vicino: un indirizzo IPv6 anycast è un indirizzo che può corrispondere a un insieme di interfacce di rete. Serve per le funzioni di discovery e non sono distinguibili dagli indirizzi unicast.

## 3.2 Configurazione IP

Un host IP può essere configurato automaticamente o staticamente. Della configurazione automatica se ne occupa il protocollo **DHCP (Dynamic Host Configuration protocol)** che, oltre a semplificare la fase di configurazione, permette una gestione facilitata della rete: di fatti così facendo è possibile cambiare gli indirizzi IP e/o i parametri di configurazione senza intervenire sui client, il che torna molto utile nel caso di grandi organizzazioni. Il suo funzionamento viene analizzato nei capitoli successivi.
Per la configurazione statica, occorre specificare le seguenti entità:

- **Indirizzo IP**;
- **Subnet Mask**;
- **Default gateway**;
- Eventualmente **Indirizzo IP del nameserver** (vedi *DNS*).

## 3.3 Internet multicasting

La normale comunicazione IP avviene tra un trasmittente e un ricevente, ma per alcune applicazioni è utile che un processo possa comunicare contemporaneamente con un gran numero di ricevitori. Si pensi banalmente alla gestione di video-conferenze digitali.

IP supporta la trasmissione **multicast**, in particolare IPv4 lo implementa con gli indirizzi di classe D. Ogni indirizzo di classe D identifica un gruppo di host e 28 bit sono utilizzati per identificare i gruppi. Il tentativo che viene fatto per trasmettere i dati ai membri del gruppo è di tipo **best-effort**.

In un gruppo, gli host possono cambiare dinamicamente, ovvero possono aggiungersi o uscire dal gruppo senza limitazioni, ed è possibile definire una chiave di accesso.

Sono supportati due tipi di indirizzi di gruppo: **permanenti** e **temporanei**.

Un gruppo **permanente** è sempre presente, non deve essere impostato e possiede un indirizzo di gruppo permanente, mentre un gruppo **temporaneo** deve essere creato prima di poter essere utilizzato, mediante entità specializzate chiamate **multicast agents**. Oltre a provvedere alla creazione di nuovi gruppi, i multicast agents sono responsabili dell'invio in Internet dei datagram multicast, nel caso un host sia in una rete diversa rispetto a quella degli altri host.

### 3.3.1 IGMP

Circa una volta al minuto, ogni router multicast genera una trasmissione hardware multicast (sul livello data link) diretta agli host sulla sua LAN (indirizzo 224.0.0.1) che chiede alle macchine di indicare i gruppi di appartenenza dei loro processi. Ogni host risponde comunicando tutti gli indirizzi di classe D che gli interessano.

Questi pacchetti di interrogazione e risposta utilizzano un protocollo chiamato **IGMP (Internet Group Management Protocol)**, descritto nel documento RFC 1112. Come intuibile, ha solo due tipi di pacchetti: interrogazione e risposta, ognuno con un formato semplice prefissato, che contiene alcune informazioni di controllo nella prima word del campo carico utile e un indirizzo di classe D nella seconda word.

Come piccola curiosità, il routing multicast è realizzato mediante le strutture *spanning tree*.

# 4 Protocolli di controllo Internet

Oltre a IP, utilizzato per il trasferimento dei dati, Internet ha diversi protocolli di controllo utilizzati nel livello network, tra cui **ICMP**, **ARP** e **RARP**.

## 4.1 ICMP

Il funzionamento di Internet è monitorato attentamente dai router. Quando avviene qualcosa di imprevisto, l'evento è comunicato da **ICMP (Internet Control Message Protocol)**. Svolge inoltre una seconda funzione, che è quella di verificare lo stato della rete. Di seguito i principali tipi di messaggi ICMP.

| Tipo di messaggio | Descrizione |
|---|---|
| DESTINATION UNREACHABLE | Il pacchetto potrebbe non essere inoltrato. |
| TIME ESCEEDED | Il campo TTL ha raggiunto il valore 0. |
| PARAMETER PROBLEM | Campo dell'intestazione non valido. |
| SOURCE QUENCH | Pacchetto di interruzione: in passato era utilizzato per rallentare gli host che trasmettevano troppi pacchetti.<br>Ora è usato raramente, perché in caso di congestione questi pacchetti tendono ad alimentare il fuoco, ed ora il controllo della congestione è affidato al livello trasporto. |
| REDIRECT | Il router si è accorto che un pacchetto sembra essere stato instradato male. |
| ECHO | Chiede a una macchina se è "viva". |
| ECHO REPLY | Sì, sono "vivo". |
| TIMESTAMP REQUEST | Simile alla richiesta di echo, ma con un contrassegno temporale. |
| TIMESTAMP REPLY | Simile alla risposta di echo reply, ma con un contrassegno temporale. |

## 4.2 ARP

Un host Internet può comunicare con un altro host solo se conosce il suo indirizzo fisico di rete (MAC), che è anche fondamentale per il funzionamento del gateway al fine di definire il percorso dei pacchetti. Attenzione però: i programmi applicativi in genere conoscono solo il nome dell'host o il suo indirizzo IP.

**ARP (Address Resolution Protocol)** fornisce il modo con cui gli indirizzi IP vengono associati agli indirizzi del livello data link (per esempio agli indirizzi Ethernet), risolvendo la corrispondenza Indirizzo IP-Indirizzo fisico.

L'host che ha bisogno dell'indirizzo fisico di un altro host della rete, invia un pacchetto **broadcast** di richiesta dell'indirizzo al destinatario (includendo il proprio indirizzo fisico di rete), il quale risponde includendo a sua volta l'indirizzo fisico di rete.

In ciascuna macchina è predisposta una cache, che memorizza gli indirizzi richiesti via ARP. Questo risulta utile per velocizzare le consultazioni successive. In particolare, ad ogni richiesta ARP tutti gli host possono aggiornare l'indirizzo fisico del richiedente nella propria cache.

## 4.3 RARP

Come detto, ARP risolve il problema della scoperta dell'indirizzo Ethernet che corrisponde a un dato indirizzo IP. In qualche caso è utile però risolvere il problema inverso: dato un indirizzo Ethernet, qual è il corrispondente indirizzo IP? Un caso in cui è molto utile risolvere questo problema, è l'avvio delle workstation diskless, che hanno bisogno di richiedere da un file server remoto un'immagine binaria del sistema operativo. In questo caso, come fanno a scoprire il proprio indirizzo IP?

Ecco quindi che interviene **RARP (Reverse Address Resolution Protocol)**. RARP effettua una richiesta **broadcast** contenente il proprio indirizzo Ethernet, in cui domanda se qualcuno conosce il suo indirizzo IP. I server che ricevono la richiesta, cercano l'indirizzo Ethernet nei propri file di configurazione e trasmettono una risposta contenente l'indirizzo IP corrispondente.

Se i riceventi non sanno rispondere, propagano la richiesta ai server secondari.

# 5 Routing

Il **routing** riguarda il livello 3 del modello di riferimento ISO-OSI ed è l'azione d'**instradare i pacchetti dal computer sorgente al computer di destinazione**. Nella maggior parte delle sottoreti i pacchetti compiono **salti multipli** per raggiungere la meta; l'unica eccezione degna di nota riguarda le reti broadcast, ma anche in questo caso se sorgente e destinazione non si trovano sulla stessa rete, il routing è un problema. Consiste in due attività principali:

- Determinare il percorso migliore;
- Trasportare pacchetti sulla rete.

Per riportare eventuali **malfunzionamenti** nel routing, viene usato **ICMP**. In particolare, il messaggio **redirection** indica la necessità di reinstradare i pacchetti in modo migliore, cioè segnala che un router è stato attraversato inutilmente.

## 5.1 Algoritmi di routing

L'**algoritmo di routing** è quella parte del software del livello network, che si preoccupa di scegliere lungo quale linea di uscita vanno instradati i pacchetti in arrivo.
Ad un algoritmo di routing devono appartenere le seguenti **proprietà**:

- **Semplicità**: consumo di meno risorse possibile;
- **Robustezza e stabilità:** comportamento corretto in condizioni inusuali (guasti hardware/software) e mai viste prima, reagendo in modo opportuno a congestioni;
- **Imparzialità**: la strada scelta non deve intenzionalmente avvantaggiare alcun nodo;
- **Ottimalità**: riduzione al minimo possibile del ritardo medio dei pacchetti, massimizzazione delle capacità di carico totale della rete (anche se molto spesso questi due obiettivi sono in conflitto ed occorre scendere a compromessi).

A seconda della tipologia di algoritmo scelto, il routing si distingue in **tre classi principali**:

- **Minimale**: le decisioni sono tutte definite al momento della configurazione dell'interfaccia di rete;
- **Statico**: utilizza algoritmi **non adattivi**, ovvero che non basano le loro decisioni su misure o stime del traffico e della topologia corrente. Il percorso utilizzato per collegare due nodi è calcolato in anticipo, in modalità offline, ed è scaricato nei router all'avvio della rete;
- **Dinamico**: utilizza algoritmi **adattivi**, che cambiano le loro decisioni secondo le modifiche apportate alla topologia e di solito anche al traffico.

Uno stato importante per gli algoritmi di routing è lo stato di **convergenza**: diremo che un set di router sono in uno stato di convergenza, se le informazioni topologiche riguardo la internetwork dove operano è la medesima. Analizzare quanto velocemente un algoritmo di routing raggiunge la convergenza è molto utilizzato come metro di paragone.

### 5.1.1 Routing table

Una **routing table** (o **tabella di routing**) è un database memorizzato in un router o in un host che elenca i vari **next hop** (**prossimo salto**, il prossimo nodo da attraversare) a cui può andare un pacchetto. Quindi nella tabella di routing, ogni riga corrisponde ad una possibile strada, i cui elementi essenziali sono:

- **ID di rete**: l'indirizzo dell'host o sottorete di destinazione;
- **Next hop**: indirizzo dell'eventuale prossimo gateway da attraversare;
- **Costo/metrica**: il costo o metrica del percorso. Questa unità di misura viene detta **metric** e non è univocamente definibile: può assumere infatti diversi significati:
    - **Path Length (o hop-count)**: numero di nodi da attraversare;
    - **Reliability**: l'affidabilità di una rete misurata ad intervalli costanti;
    - **Delay:** il tempo necessario ad ogni pacchetto per raggiungere la destinazione;
    - **Bandwidth**: l'ampiezza di banda;
    - **Load**: traffico misurato ad intervalli regolari;
    - **Communication Cost**: numero intero arbitrario che indica quanto un percorso sia conveniente: valori più bassi indicano un percorso migliore.

Le routing table possono ricevere informazioni da due sorgenti:

- Dal **file di configurazione** creato dall'amministratore di rete, salvato sul disco del dispositivo e interpretato in fase di inizializzazione dell'hardware;
- Da **protocolli dinamici di aggiornamento** (protocolli di routing).

Alla ricezione di un pacchetto, ogni nodo della rete controlla la classe dell'indirizzo IP o sottorete di destinazione:

- se l'indirizzo **è locale**, allora applica la subnet mask alla destinazione e invia direttamente all'host destinatario;
- se invece **non è locale**, allora cerca nella routing table la rete di destinazione e instrada il datagram verso il gateway corrispondente.

In ambiente **Unix**, il comando per visualizzare la tabella di routing è: **`netstat -r`**. Opzionalmente, è possibile usare la specifica -n per ottenere gli indirizzi di destinazione in forma numerica. Per ogni risultato può comparire un campo flag, i cui significati sono i seguenti:

- **U**: indica che l'interfaccia di rete è attiva (infatti U sta per UP);
- **G**: indica un'uscita verso un'altra rete tramite gateway;
- **H**: indica che la destinazione è l'indirizzo completo di un host;
- **D**: indica una route aggiunta da un ICMP redirect.

### 5.1.2 Routing in una internetwork

Il routing attraverso una internetwork è simile al routing eseguito in una singola sottorete, ma con alcune complicazioni in più. Il routing viene infatti effettuato a due livelli:

- dentro ogni rete viene utilizzato un protocollo di gateway interno (**IGP**: *Interior Gateway Protocol*);
- tra le reti è adottato un protocollo di gateway esterno (**EGP**: *Exterior Gateway Protocol*).

Esiste inoltre un'altra famiglia di protocolli di routing, che non rientra né in quella IGP né in quella EGP, ed è la **CIDR (Classless Inter-Domain Routing)**.

Poiché ciascuna rete è indipendente, ognuna può usare algoritmi differenti e a forte di questa loro indipendenza, un gruppo di router e reti sotto il controllo di una singola autorità amministrativa ben definita, viene chiamato **Autonomous System (AS)**. Ogni AS ha un numero identificativo univoco, detto **ASN (Autonomous System Number)**, utile ai fini del routing. Gli ASN sono assegnati da *ICANN* ai *Regional Internet Registries* (*RIRs*).

Gli AS possono essere suddivisi in:

- **multihomed autonomous system**: è un AS che mantiene connessioni con più di un AS;
- **stub autonomous system**: si riferisce ad un AS connesso solamente con un altro AS;
- **transit autonomous system**: è un AS che fornisce attraverso di sé connessioni con altre reti.

## 5.2 Classificazione di algoritmi adattivi

Le moderne reti di computer generalmente utilizzano algoritmi di routing dinamici al posto di quelli statici, poiché gli algoritmi statici non tengono conto del carico istantaneo della rete. I due algoritmi dinamici più popolari sono il routing basato sul vettore delle distanze e il routing basato sullo stato dei collegamenti.

### 5.2.1 Algoritmi di tipo distance-vector (vettore-distanza)

Gli algoritmi di routing basati sul **vettore delle distanze** (*distance vector routing*) operano facendo in modo che ogni router conservi una tabella (ossia un vettore), che definisce la migliore distanza conosciuta per ogni destinazione e la linea che conduce a tale destinazione. Queste tabelle sono aggiornate scambiando informazioni con i router vicini, e gli aggiornamenti sono in forma di coppie del tipo **(rete di destinazione, distanza)**, dove la metrica usata per la distanza potrebbe essere il numero di salti, il ritardo espresso in millisecondi, il numero totale di pacchetti accodati lungo il percorso o qualcosa di simile.

### 5.2.2 Algoritmi di tipo link-state (stato dei collegamenti)

Il routing basato sul vettore delle distanze è stato utilizzato in ARPANET fino al 1979, quando fu sostituito dal routing basato sullo **stato dei collegamenti**, ancora largamente usato in diverse varianti. L'idea alla base di questo algoritmo è semplice e può essere riassunta in cinque punti. Ogni router deve:

1. scoprire i propri vicini e i relativi indirizzi di rete;
2. misurare il ritardo (delay) o il costo di ogni vicino;
3. costruire un pacchetto detto *link-state* che contiene tutte le informazioni raccolte;
4. inviare questo pacchetto a tutti gli altri router;
5. elaborare il percorso più breve verso tutti gli altri router.

In pratica, si misurano sperimentalmente la topologia completa e tutti i ritardi per poi distribuirli a ogni router; successivamente viene eseguito l'algoritmo di Dijkstra per trovare il percorso più breve associato a ogni altro router.

Questo tipo di routing permette di effettuare calcolo distribuito (ogni router lavora), evita i problemi che si hanno quando un router trasmette informazioni errate e può gestire reti composte da un gran numero di nodi.

## 5.3 OSPF

Sviluppato nel 1988 da un sottogruppo di IEFT (Internet Engineering Task Force), **OSPF (Open Shortest Path First)** è oggi supportato dalla maggior parte dei produttori di router ed è diventato il principale protocollo di **routing per gateway interni**.

Si tratta di un protocollo aperto (**Open**, senza vincoli di brevetto) che soddisfa alcuni requisiti:

- supporta **diverse metriche di distanza** (come la distanza fisica, il ritardo…);
- è un algoritmo **dinamico**, in grado di modificarsi rapidamente e automaticamente in base alla topologia;
- è stato realizzato come supporto al **routing basato sul tipo di servizio** (ottenibile dal campo *Type of Service* di IP), ma a causa del suo scarso utilizzo (in sua assenza gli utenti avevano già trovato altre soluzioni), si è deciso di eliminarlo in quanto superfluo;
- esegue il **bilanciamento del carico**, ossia suddividere il carico su più linee: la maggior parte dei protocolli precedenti inviava tutti i pacchetti lungo il percorso migliore ed il percorso successivo al migliore non era mai utilizzato;
- supporta i **sistemi gerarchici (aree)**, in quanto prima nessun router poteva (né doveva) conoscere l'intera topologia di Internet, a causa della sua dimensione elevata e crescente;
- supporta l'**autenticazione dei messaggi**, migliorando la sicurezza ed impedendo agli utenti di imbrogliare i router inviando loro false informazioni di routing.

OSPF opera rappresentando la **rete reale come un grafo** e poi elabora il **percorso più breve** da ogni router a ogni altro router, in base al costo degli archi (distanza, ritardo e così via).

Siccome molti AS di Internet sono a loro volta grandi e difficili da gestire, OSPF permette di dividere tali AS in più **aree**, dove ogni area è una rete o un insieme di reti contigue. Ogni router conserva il **topological database** della sua area, ovvero una lista dei router adiacenti, indicando quelli designati a svolgere il ruolo di "rappresentanti dell'area", cioè **DR (*Designated Routers*)** e **BDR (*Backup Designated Routers)***. Ogni AS ha un'area dorsale chiamata **backbone area** o **area 0** e tutte le aree sono collegate alla dorsale, il che permette di passare da un'area a qualunque altra dello stesso AS attraverso la backbone.
I router della stessa area hanno lo stesso topological database, e la divisione in aree permette di ridurre il traffico di routing. L'OSPF backbone distribuisce le informazioni di routing tra le aree e ne può apprendere ulteriori mediante istruzioni di configurazione o tramite EGP o IGP.

La divisione in aree porta a 2 tipi diversi di routing (**intra-area** ed **inter-area**) e a 4 diversi tipi di router:

- **IR (Internal Router)**: router completamente interno ad un'area;
- **ABR (Area Border Router)**: collegano due o più aree, mantengono il topological database di ogni area che collegano;
- **BR (Backbone Router)**: router dell'area di backbone;
- **ASBR (AS Boundary Router)**: comunicano con i router che si trovano in altri AS.

OSPF è un protocollo **intra-AS (IGP)** (anche se può funzionare tra diversi AS) e **link-state**, poiché invia **link-state advertisements (LSA)** a tutti i router della stessa area per diffondere informazioni sulle interfacce attive, sulle metric e altre informazioni.

I **pacchetti di tipo LSA** di OSPF si dividono in **5 tipi**:

- **Hello**: usato per scoprire chi sono i vicini;
- **Database Description**: fornisce i propri criteri per la selezione del costo del link;
- **Link State Request**: richiesta di informazioni di stato ai router vicini;
- **Link State Update**: comunica i costi utilizzati nel database della topologia;
- **Link State Acknowledgement**: conferma un Link State Update.

## 5.4 RIP

**RIP (Routing Information Protocol)** è un popolare algoritmo di routing che utilizza UDP come protocollo di trasporto (porta 520) e si basa sull'algoritmo vettore-distanza utilizzando il numero di hop come metric.

RIP presenta un **limite di 15 hop**: reti più lontane di 15 hop (indicate con 16 hop) sono considerate irraggiungibili e considerate aventi metrica infinita. Nonostante ciò, non riesce ad evitare i routing loop. Questi sono alcuni dei motivi che hanno portato all'utilizzo ridotto di RIP, oltre al fatto che converge più lentamente rispetto ad altri algoritmi come OSPF e non supporta subnet variabili. RIP è più leggero di OSPF (richiede meno risorse) ed è disponibile per default sui sistemi Unix/Linux (daemon `**routed**`).

Il funzionamento di RIP è semplice: i router che lo implementano, inviano tutta la routing table o parte di essa ai router vicini ad intervalli di tempo regolari, dopodiché ogni router si tiene solo le informazioni relative alle best routes. Il RIP ha due forme a seconda dell'utente:

- Forma **passiva**: usata dagli host, riceve messaggi ma non ne invia;
- Forma **attiva**: usata dai router, riceve ed invia in broadcast messaggi.

Questi messaggi prendono il nome di **routing updates** e consistono in una porzione della routing table in cui si è trovato un percorso migliore. Gli aggiornamenti vengono inviati in due occasioni:

- Ad ogni tick del routing-update timer, cioè **a cadenza regolare** (di default 30 secondi);
- Quando **cambia la topologia** della rete dei router confinanti.

Un altro timer utilizzato è il **route timeout**: se una route non viene aggiornata nella tabella entro un tempo limite, viene segnata come non valida e successivamente rimossa allo scadere del **route-flush** timer.

Per evitare di annunciare false informazioni di routing, RIP implementa alcune tecniche:

- **Split horizon**: il router non propaga informazioni su una route al router che ha generato tale aggiornamento;
- **Hold down**: costringe un router a ignorare aggiornamenti inerenti a una rete per un certo tempo (60 sec. in genere), dopo aver ricevuto un messaggio di rete irraggiungibile;
- **Poison reverse**: quando un collegamento scompare, il router che effettuava l'annuncio continua ad annunciarlo ancora per un certo tempo, assegnandoli distanza infinita.

## 5.5 BGP

Dentro un singolo AS, il protocollo di routing raccomandato è OSPF (sebbene non sia l'unico utilizzato). Tra AS si utilizza un protocollo diverso, chiamato **BGP (Border Gateway Protocol)**, perché gli obiettivi sono differenti: un protocollo per gateway interni non deve far altro che spostare i pacchetti nel modo più efficiente possibile dalla sorgente alla destinazione; non deve preoccuparsi della **politica**, ovvero vincoli che si possono imporre. Alcuni esempi di vincoli:

- Nessun traffico di passaggio attraverso certi AS;
- Mai mettere l'Iraq su un percorso che inizia dal Pentagono;
- Passare per l'Albania solo se non esistono alternative per la destinazione.

In genere i criteri sono configurati manualmente in ogni router BGP (o inseriti usando qualche tipo di script), e non fanno parte del protocollo stesso.

Dato l'interesse di BGP verso il traffico di passaggio, le reti sono raggruppate in 3 categorie:

- **Stub networks (reti terminali)**: sono reti che non possono essere utilizzate per il traffico di passaggio perché non ci sono altre reti a cui si collega;
- **Multiconnected networks (reti multicollegate)**: reti utilizzate per il traffico di passaggio, se non rifiutano di farlo;
- **Transit networks (reti di transito)**: simili alle reti backbone nell'OSPF, sono reti pronte a gestire pacchetti di terze parti, di solito con alcune restrizioni o a pagamento.

BGP effettua **interdomain routing (EGP)** basandosi su un algoritmo **vettore-distanza** (un po' diverso da quello applicato ad altri protocolli) e utilizzando connessioni TCP (porta 179), che rende le comunicazioni affidabili e nasconde tutti i dettagli della rete attraversata.

Infatti, l'algoritmo **vettore-distanza** utilizzato in BGP è stato **modificato** in modo da tenere traccia anche del percorso utilizzato, anziché solo il costo di ogni destinazione. Inoltre, anziché comunicare periodicamente a ogni vicino il costo stimato associato a ogni possibile destinazione, ogni router BGP comunica l'esatto percorso in uso e basta. Ogni router BGP contiene un modulo che esamina i percorsi diretti a una destinazione e assegna loro un punteggio (distanza associata a quella destinazione) e nel caso un percorso violi un criterio di vincolo riceve punteggio ∞. Tuttavia la funzione che valuta il punteggio dei percorsi non fa parte del protocollo BGP e può essere scelta dagli amministratori del sistema.

I peer che eseguono il protocollo BGP eseguono 3 funzioni di base:

- **Autenticazione reciproca**, per stabilire la correttezza delle operazioni e vedere se è possibile comunicare;
- **Scambio di informazioni**, come le reti raggiungibili (e non) o dati del salto successivo;
- **Funzioni di verifica**, per controllare la correttezza dei peer e della connessione di rete.

Per definire queste 3 funzioni, BGP definisce un insieme di 5 messaggi:

- **Open**: primo messaggio inviato da ciascuna parte dopo l'attivazione della sessione TCP, serve ad aprire una sessione tra peer. Deve essere confermato da un messaggio keepalive prima che possa aver luogo la comunicazione;
- **Update**: aggiornamento del routing, consente al router di avere una mappa della rete ed eventualmente cancellare reti irraggiungibili;
- **Notification**: inviato in condizione di errore, chiude una sessione attiva e avvisa i router vicini del motivo della chiusura;
- **Keepalive**: messaggi di servizio inviati regolarmente per mantenere la connessione;
- **Refresh**: chiede il reinvio delle informazioni di routing per una rete da parte di un peer.

Il protocollo BGP può essere utilizzato in 3 differenti scenari:

- **Inter-Autonomous System Routing (eBGP)**: routing tra 2 o più router BGP di AS diversi;
- **Intra-Autonomous System Routing (iBGP)**: routing tra 2 o più router BGP di uno stesso AS;
- **Pass-through Autonomous System Routing**: routing tra 2 o più router BGP che scambiano traffico attraverso un AS che non esegue BGP.

# 6 Protocolli di trasporto Internet

Il livello trasporto non è uno strato qualsiasi: è il cuore dell'intera gerarchia di protocolli.

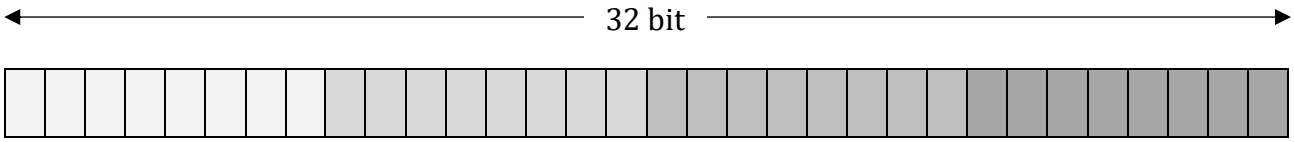
Il suo compito è fornire il trasporto dei dati, affidabile ed efficiente in termini di costi, dal computer di origine a quello di destinazione, indipendentemente dalla rete o dalle reti fisiche effettivamente utilizzate. Senza il livello trasporto, l'intero concetto di protocolli a strati avrebbe poco senso.

Internet possiede **due protocolli principali** nel livello trasporto: un protocollo senza connessione e uno orientato alla connessione, rispettivamente UDP e TCP.

## 6.1 UDP

**UDP (User Datagram Protocol**, ovvero protocollo per datagrammi utente) offre alle applicazioni un modo per inviare datagrammi IP incapsulati senza dover stabilire una connessione. Pur essendo di conseguenza poco affidabile, esso è impiegato in applicazioni *time-sensitive*, dove è richiesta velocità elevata: un esempio è lo streaming di video e audio.

UDP trasmette **segmenti** costituiti da un'intestazione di 8 byte seguita dal carico utile.




| Source port | Destination port |
|---|---|
| UDP length | UDP checksum |

- Le **due porte** servono per identificare i punti finali all'interno dei computer di origine e destinazione. Quando arriva un pacchetto UDP, il suo carico utile è consegnato al processo associato alla porta di destinazione. Senza le porte, il livello trasporto non saprebbe cosa farsene del pacchetto. È utile notare che l'inclusione del campo source port è utile nel caso si debba inviare una risposta all'origine;
- Il campo **UDP length** include l'intestazione di 8 byte e i dati;
- **UDP checksum** è utilizzato per verificare l'integrità dei dati. È facoltativo, anche se disattivarlo è insensato a meno che la qualità dei dati non sia importante (per esempio nel caso di voce digitalizzata).

Il protocollo non si occupa del controllo di flusso, del controllo degli errori o della ritrasmissione dopo la ricezione di un segmento errato. Questi compiti sono invece lasciati ai processi utente.

Un'altra funzionalità importante di UDP è il **multiplexing** di più processi utilizzando le porte, ovvero la trasmissione (praticamente simultanea) di due o più messaggi da parte di due o più processi attraverso la stessa linea.

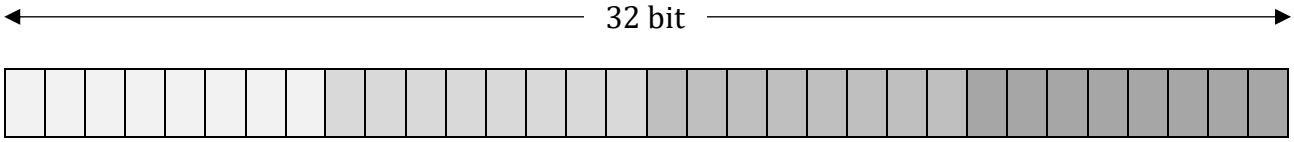
## 6.2 TCP

*Il TCP è un argomento abbastanza profondo e complicato e quindi, come anticipato, questo argomento è uno di quelli trattati in maniera semplificata.*

**TCP** (**Transmission Control Protocol**, ovvero protocollo di controllo della trasmissione) è un protocollo full-duplex con connessione. Affidabile e di applicabilità generale, è progettato per adattarsi dinamicamente alle peculiarità delle diverse reti che compongono Internet, contribuendo ad isolare le applicazioni dai dettagli di networking.

In una connessione TCP, ogni byte ha un proprio numero di sequenza a 32 bit. Questa è una funzionalità vitale di TCP.

Ogni computer che supporta TCP dispone di un'**entità di trasporto TCP**, che può essere una procedura di libreria, un processo utente o una parte del kernel, ma in tutti i casi gestisce i flussi TCP e le interfacce verso lo strato IP. Le entità TCP di invio e ricezione frazionano i dati in blocchi, detti segmenti. Un **segmento TCP** consiste di un'intestazione fissa di 20 byte (più una parte facoltativa) seguita da 0 o più byte di dati, ed è il software TCP a decidere la dimensione dei segmenti. Tuttavia, ogni rete ha una **MTU** (**Maximum Transfer Unit**, ovvero unità di trasferimento massima) e ogni segmento deve essere contenuto nella MTU, generalmente 1500 byte.




| Source port | | | | | | | | Destination port |
|---|---|---|---|---|---|---|---|---|
| Sequence number | | | | | | | | |
| Acknowledgement number | | | | | | | | |
| TCP header length | | URG | ACK | PSH | RST | SYN | FIN | Window size |
| Checksum | | | | | | | | Urgent pointer |
| Options (0 o più parole di 32 bit) | | | | | | | | |
| Dati (facoltativo) | | | | | | | | |

- **Source port** e **Destination Port**: estremi locali della connessione. Una porta e l'indirizzo IP del suo host formano un end point univoco e identificano quindi la connessione. Questa coppia viene chiamata **socket** e consente il multiplexing, facendo sì che una certa porta possa essere condivisa da più host;
- **Sequence number** e **acknowledgement number**: il primo specifica la posizione del primo byte contenuto nel campo dati all'interno dello stream di byte che il trasmettitore del segmento invia; il secondo specifica la posizione del prossimo byte aspettato all'interno del segmento inviato dal ricevitore del presente segmento;

- **TCP header length**: numero di parole di 32 bit contenute nell'header (necessario perché options ha lunghezza variabile);
- **6 flags**:
    - **URG**: 1 se si usa il campo urgent pointer;
    - **ACK**: 1 se l'ack number è valido (cioè se si trasporta un ack);
    - **PSH**: 1 se si vuole che i dati vengano consegnati all'applicazione all'arrivo e non archiviarli in un buffer per poi trasmettere un buffer completo (come potrebbe fare per migliorare l'efficienza);
    - **RST**: 1 se si richiede un reset della connessione (per problemi);
    - **SYN**: usato per stabilire le connessioni:
        - **SYN=1, ACK=0** -> richiesta connessione;
        - **SYN=1, ACK=1** -> accettata connessione;
        - **SYN=0, ACK=1** -> ricevuta la conferma di accettata connessione.
    - **FIN**: usato per rilasciare una connessione: non ci sono altri dati da trasmettere.
- **Window size**: indica quanti byte possono essere inviati a partire da quello che ha ricevuto acknowledgement. Utilizzato per la sliding window, spiegata in seguito;
- **Checksum**: simile a quello di UDP, verifica che sia l'header che i dati del segmento siano arrivati a destinazione senza errori;
- **Urgent pointer**: puntatore ai dati urgenti;
- **Options**: contiene caratteristiche extra non disponibili nella normale intestazione.

Le connessioni in TCP vengono stabilite mediante l'**handshake a tre vie**. In TCP, client e server tengono traccia di ciò che hanno inviato utilizzando un numero di sequenza, detto **ISN (Initial Sequence Number)**, un campo di 32 bit inizializzato casualmente che viene inviato nei campi acknowledgement number e sequence number.
Per stabilire una connessione, sono necessari 3 passaggi:

1) L'host client invia al server un pacchetto preliminare (**SYN**), **chiedendo di sincronizzarsi** con il proprio ISN;
2) L'host server riceve il pacchetto SYN, **riconoscendo** l'ISN del client e predisponendo il buffer che accoglierà i segmenti che gli verranno inviati. Il server si conserva quindi l'ISN del client aumentato di 1, poiché il prossimo pacchetto che gli manderà il client avrà questo ISN. Quindi il server invia al client un altro pacchetto (**SYNACK**), **chiedendo di sincronizzarsi** con il proprio ISN;
3) Ricevuto il SYNACK, anche il client **riconosce** l'ISN del server e alloca un buffer per la ricezione dei segmenti, conservando l'ISN del server aumentato di 1 (come ha fatto il server). Invia quindi un ultimo pacchetto (**ACK**), che sta ad indicare che la connessione è ormai instaurata con successo.

I nomi dei pacchetti spediti nei passaggi dell'handshake sono dati per ricordarsi delle flag poste a 1. Si noti che l'inizializzazione casuale dell'ISN è necessaria per diversi motivi, tra cui la sicurezza: un utente terzo potrebbe infatti effettuare *spoofing* di una delle due parti (ovvero spacciarsi per una delle due parti utilizzando il suo stesso indirizzo IP) e stabilire una connessione con l'altra semplicemente predicendo l'ISN.

Come anticipato durante la spiegazione dell'intestazione, il protocollo di base utilizzato dalle entità TCP è il protocollo **sliding window**, utilizzato per il controllo di flusso. Grazie al protocollo sliding window, il mittente può inviare più di un segmento anche se non ha ancora ricevuto il riscontro per il primo frame inviato. Comunque, è garantito che nessun segmento andrà perso e che i frame arriveranno nell'ordine corretto. Il protocollo ha questo nome perché utilizza 2 finestre (ovvero 2 intervalli) che vengono modificati durante l'algoritmo, avanzando sui numeri di segmenti spediti e ricevuti:

- Una finestra è dedicata al **mittente**: contiene i numeri dei **segmenti già spediti** ma per cui non si è ancora ricevuta una conferma (**no ACK**);
- l'altra finestra è per il **ricevente**: contiene i numeri dei **segmenti che può ricevere senza doverli subito confermare**.

Quando un mittente invia un segmento, viene fatto partire un timer. Una volta arrivato a destinazione, l'entità TCP ricevente invia un segmento contrassegnato da un numero di acknowledgement uguale al numero di sequenza successivo che prevede di ricevere. Se il timer del mittente scade prima della ricezione dell'acknowledgement, il mittente ritrasmette il segmento.

In sunto, le caratteristiche importanti da ricordare che TCP offre sono le seguenti:

- **Trasferimento bufferizzato**: il trasferimento viene ottimizzato creando pacchetti di dimensione il più possibile simile, accorpando o suddividendo in segmenti;
- **Orientamento dello stream**: i dati vengono passati al livello applicazione in ordine;
- **Stream non strutturato**: il servizio di stream del TCP non rispetta eventuali strutture presenti in dati strutturati: la comprensione della forma sta all'applicazione;
- **Connessione full-duplex**: trasferimento simultaneo in entrambe le direzioni;
- **Connessione connection oriented**: solo quando mittente e destinatario hanno verificato la sussistenza delle condizioni necessarie, ha inizio il trasferimento;
- **Riscontro positivo con trasmissione (acknowledgement)**: alla ricezione di un pacchetto, il destinatario risponde con un ACK, ossia una conferma di ricezione. Eventualmente il mittente ritrasmette i pacchetti persi;
- **Sliding Window**: permette una trasmissione ottimizzata, seppure sia comunque più lenta rispetto ad UDP.

# 7 Servizi di Rete

I livelli sotto lo strato applicazione forniscono il trasporto affidabile, ma non svolgono alcun lavoro per gli utenti. Di seguito vengono presentate alcune applicazioni concrete delle reti.

## 7.1 Telnet

**Telnet** è un protocollo di rete che ha come obiettivo il fornire un supporto per le comunicazioni sufficientemente **generalizzato**, **bidirezionale** ed **orientato ai byte**. È un servizio che permette di accedere in remoto ad una macchina, tramite emulazione di terminale attraverso la rete, basandosi sul protocollo **TCP** e sul paradigma client/server. Di solito a tale servizio è assegnata la **porta 23**, ed è possibile utilizzare un programma Telnet per stabilire una connessione interattiva ad un qualche altro servizio del server. Si prenda come esempio il collegamento alla porta 25, sulla quale si trova un server SMTP (utilizzato per la posta ed analizzato in seguito).

Telnet si basa su **3 aspetti**:

- **NVT (Network Virtual Terminal)**: è un terminale virtuale con caratteristiche generali. Ogni server o client traduce i propri controlli nativi in quelli del NVT;
- **Opzioni negoziate**: tale negoziazione avviene tra client e server per aumentare le funzionalità della sessione Telnet da aprire;
- **Viste simmetriche**: degli strumenti per evitare che la negoziazione delle opzioni generi cicli di opzioni senza fine (errata interpretazione).

Il problema principale di Telnet riguarda la sicurezza: è di fatti un protocollo non criptato, quindi password, nome utente e tutte le altre informazioni sono inviate in chiaro. A questa problematica pone rimedio **SSH** (analizzato in seguito).

## 7.2 Comandi r

I **comandi r** sono progettati per sistemi BSD Unix e forniscono delle facilitazioni specifiche per Unix. Di seguito i comandi più usati:

- **rlogin (remote login)**: permette di amministrare una serie di macchine autenticandosi una sola volta (e quindi non richiedendo altre volte la password). È una soluzione molto comoda che facilita le configurazioni di ambienti distribuiti, ma porta ovvie problematiche di sicurezza;
- **rsh (remote shell)**: consente l'esecuzione remota di un singolo comando.

Anche in questi comandi non vi è alcuna forma di crittografia e pertanto sono stati anch'essi rimpiazzati da **SSH**.

## 7.3 FTP

**FTP** (**File Transfer Protocol**) è un protocollo basato su TCP ed è impiegato per il trasferimento di file in rete. Di solito a tale servizio è assegnata la **porta 21**.

A differenza di altri protocolli, FTP utilizza 2 connessioni separate per gestire comandi e dati:

- **PI (Protocol Interpreter):** si occupa di trasmettere comandi fra il client e il server. Tramite esso inoltre si dà inizio al processo FTP;
- **DTP (Data Transfer Process)**: si occupa del trasferimento vero e proprio dei dati tra un client e un server. Durante l'esecuzione, client e server si scambiano i ruoli continuamente.

È bene notare che ad oggi, gran parte delle richieste FTP non avviene da linea di comando, ma da browser, ad esempio specificando un URL.

Normalmente, FTP richiede l'autenticazione del client tramite nome utente e password, tuttavia sono state pensate delle **sessioni anonime**, che permettono l'utilizzo del protocollo in sola lettura, il che evita l'upload sul server di dati non autorizzati. Per usare una sessione anonima, basta usare *anonymous* come username e come password qualsiasi stringa, come ad esempio l'indirizzo e-mail o semplicemente la parola "guest".

Nelle versioni iniziali, FTP prevedeva il trasferimento in chiaro delle password. In seguito, con RFC 2228, sono stati introdotti nuovi comandi per aumentare la sicurezza:

- **AUTH**: meccanismo utilizzato per l'autenticazione da utilizzare;
- **PROT**: il livello di sicurezza che verrà usato. Ce ne sono diversi:
    - **Clear**: trasmissione in chiaro;
    - **Safety**: richiesta la verifica sull'integrità dei dati, si usa il comando **MIC**;
    - **Confidential**: trasmissione cifrata, si usa il comando **CONF**;
    - **Private**: trasmissione cifrata e richiesta la verifica sull'integrità dei dati, si usa il comando **ENC**.

Nonostante ciò, anche FTP è stato inglobato in SSH per aumentare ancor di più la sicurezza.

Un'alternativa più leggera all'FTP standard è **TFTP (Trivial FTP)**, che sfrutta UDP al posto di TCP: pensata per l'ambito LAN, è spesso utilizzata dalle macchine diskless per ottenere l'immagine del sistema operativo.

## 7.4 SSH

**SSH (Secure Shell)** è un protocollo per comunicazioni di rete sicure progettato per essere relativamente semplice ed economico da implementare, utilizzando la crittografia asimmetrica.

La versione iniziale (SSH1) era focalizzata sulla fornitura di una struttura di accesso remoto sicura per sostituire Telnet e altri schemi di accesso remoto che non fornivano sicurezza. Nella nuova versione (SSH2), vengono corretti una serie di difetti di sicurezza dello schema originale e migliorate performance e flessibilità, che comprende anche l'implementazione di **SFTP (Secure File Transfer Protocol)**.

Le applicazioni client e server SSH sono ad oggi ampiamente disponibili per la maggior parte dei sistemi operativi ed è diventato il metodo di scelta principale per l'accesso remoto ad un server, rimpiazzando Telnet, i comandi R ed FTP.

SSH è organizzato in tre protocolli che in genere vengono eseguiti su TCP:

| User Authentication Protocol | Connection Protocol |
|---|---|
| Transport Layer Protocol | |
| TCP | |
| IP | |

- **Transport Layer Protocol**: definisce l'autenticazione del server, lo scambio delle chiavi, la cifratura, la compressione (opzionale) e il controllo d'integrità dei pacchetti;
- **User Authentication Protocol**: definisce l'autenticazione dell'utente, che può avvenire in 3 metodi: utilizzando la public key, la password o l'hostbased;
- **Connection Protocol**: gestisce funzionalità quali l'instaurazione di terminali interattivi, esecuzione di comandi remoti e l'inoltro di connessioni e di applicazioni grafiche X11, utilizzando canali multipli di comunicazione passanti per lo stesso tunnel criptato del Transport Layer.

I livelli interni al protocollo SSH coprono gli ultimi 3 livelli della pila OSI.

Sono resi disponibili diversi applicativi che utilizzano SSH. Tra questi, **OpenSSH** è un insieme di programmi open source, che rendono disponibili sessioni crittografate di comunicazione in una rete. Tra i programmi più utilizzati, sono presenti:

- ***ssh***, che sostituisce *rlogin* e *Telnet*;
- ***scp***, che sostituisce *rcp*;
- ***sftp***, che sostituisce ftp;
- ***ssh-add, ssh-agent,*** e ***ssh-keygen,*** una serie di programmi per la generazione e la gestione delle chiavi;
- i nomi dei processi demoni rispettivamente per server SSH e server SFTP sono ***sshd*** e ***sftp-server***.

## 7.5 DNS

*Il DNS richiederebbe una spiegazione sul funzionamento dei file di configurazione, ma per una migliore comprensione e focalizzazione è stata tralasciata.*

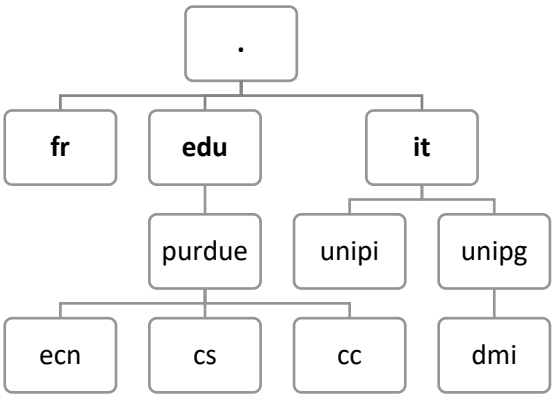
Anche se i programmi possono teoricamente fare riferimento a host tramite i loro indirizzi IP, per le persone sono difficili da ricordare. Inoltre, come detto, gli indirizzi IP possono cambiare. Per questo motivo è stato inventato il **DNS (Domain Name System)**, un applicativo che effettua la **risoluzione** di nomi di host in indirizzi IP, ovvero la traduzione di un nome sottoforma di stringa in indirizzo IP. È anche possibile effettuare una **risoluzione inversa**.

Le caratteristiche più importanti del DNS sono 2:

- uno **schema di denominazione gerarchico basato su dominio**;
- un **sistema di database distribuito** per l'implementazione del suddetto schema.

In breve, DNS viene utilizzato come segue: per associare un nome ad un indirizzo IP, un programma applicativo chiama una procedura di libreria chiamata **resolver (risolutore)**, passando il nome come parametro. Il risolutore invia quindi un pacchetto UDP a un server DNS locale, che quindi cerca il nome e restituisce l'indirizzo IP al risolutore, che a sua volta lo restituisce al chiamante.

### 7.5.1 Lo spazio dei nomi DNS




Il DNS è organizzato ad albero, la cui radice è indicata come "." e ogni nodo al di sotto di esso è detto **dominio**. Ogni dominio può essere partizionato ovviamente in **sottodomini**, che possono essere a loro volta divisi e così via. Le foglie rappresentano i domini che non hanno sottodomini, che possono sia rappresentare un singolo host che rappresentarne migliaia (come nel caso di una società).

Si noti che i nomi di dominio sono **case insensitive**.

Il primo livello dell'albero è l'insieme dei **domini di primo livello** (o **TLD, Top Level Domain**), assegnati da **ICANN**, e ne esistono di 3 tipi:

- **gTLD (generic TLD)**: domini di primo livello generici, usati da particolari classi di organizzazioni (ad esempio si usa **.com** per organizzazioni commerciali);
- **ccTLD (country-code TLD)**: domini di primo livello nazionali, usati da uno stato o una dipendenza territoriale (ad esempio **.it** per domini italiani);
- **Infrastrutturali**: l'unico è **.arpa**, usato nella risoluzione inversa dei nomi.

I **domini di secondo livello (o SLD, Second Level Domain)**, in genere, appartengono alle organizzazioni che li hanno registrati e comprendono il loro nome e il dominio di primo livello separati da un punto. Tutti i **sottodomini** che seguono da destra verso sinistra il SLD vengono creati per rendere la gestione del DNS modulare.

Si prenda come esempio **dmi.unipg.it**:

- il TLD è **.it**, che sta ad indicare la gerarchia italiana;
- il SLD è **.unipg.it**;
- il sottodominio è **.dmi**.

Al contrario di quanto avviene con gli indirizzi IP, si noti che la parte più importante di un nome è la prima partendo da destra (il TLD).

Ottenere un dominio di secondo livello è semplice: basta contattare un **registrar** (autorità di registrazione dei domini) per il dominio di primo livello corrispondente (ad esempio **.it**), per controllare se il nome desiderato è disponibile. Se non vi sono problemi, il richiedente paga una tariffa annuale e ottiene il nome.

Ad ogni dominio, che sia rappresentato da un singolo host o da un dominio di primo livello, può essere associato un insieme di **Resource Record (RR, record delle risorse)**.

Un **RR** è una tupla del database di un DNS Server che contiene la mappatura tra indirizzo IP (supportata anche la v6) e nome. Ve ne sono di diversi tipi, ma per sinteticità sono stati omessi.

### 7.5.2 Risoluzione

Come anticipato, la traduzione di un nome in un indirizzo IP è detta **risoluzione**. Essa può essere di due tipi:

- **Statica**: la mappatura tra indirizzo IP e nome viene stabilita permanentemente mediante una **host table**, memorizzata in un semplice file ASCII (**/etc/hosts**);
- **Dinamica**: la mappatura tra indirizzo IP e nome viene stabilita ad ogni avvio dell'host, effettuando richieste a server DNS. Permette ad un nome di essere sempre associato all'indirizzo IP di uno stesso host, anche se l'indirizzo cambia nel tempo.

### 7.5.3 BIND

**BIND (Berkeley Internet Name Domain)** è l'implementazione più comune del DNS su ambiente Unix, ed è composto da **una parte client (il resolver)** ed **una parte server (named)** che comunicano tramite la porta 53.

**Resolver** è una libreria di funzioni che permette di generare e inviare al server (tramite UDP) le richieste di informazioni sui nomi dei domini; **Named** è un processo demone in grado di servire le richieste del resolver e rispondere (tramite TCP).

**BIND** può essere configurato come:

- **Caching-only**: il server reindirizza ogni richiesta del resolver ad altri server e memorizza le risposte in una cache locale;
- **Primary**: il server è gestore di informazioni riguardanti specifici domini. Legge le informazioni da appositi file configurati dall'amministratore, detti ***zone file***;
- **Secondary**: il server scarica gli zone file dal primary server e li memorizza localmente in appositi file detti ***zone file transfer***.

Per configurare il **resolver**, il file relativo è '**/etc/resolv.conf**'.
Per configurare il **named**, i file relativi sono '**/etc/named.conf**', '**/etc/named.ca**', '**/named.local**', '**/etc/named.hosts**', '**/etc/named.rev**'.

## 7.6 NIS

**NIS (Network Information Service)** è un servizio (spesso utilizzato nel contesto di applicazioni parallele e distribuite) che permette di definire delle risorse di amministrazione comuni ad un insieme di host, permettendo un controllo centralizzato e la condivisione automatica di risorse. Così facendo, un utente può utilizzare host differenti mantenendo gli stessi username, password, cartella home e permessi.

Il NIS non utilizza i file amministrativi (come ad esempio /etc/passwd e /etc/group) così come sono, ma ne crea una copia: queste copie sono denominate **NIS map**. Le **NIS map** sono dei database in cui si memorizzano solo coppie di dati (chiave-valore) e vengono memorizzate nel server principale, che le rende disponibili ai client tramite il processo **ypserv**. I client possono aggiornare le loro informazioni ricevendo i database tramite il processo demone **ypbind**. Generalmente, le informazioni NIS sono memorizzate nella directory **/var/yp**.

Quando si attiva il NIS, si consiglia di non usare più i vecchi comandi amministrativi per cambiare password/permessi o quant'altro, perché il NIS non si accorge (autonomamente) dei cambiamenti apportati ai file amministrativi tradizionali. Bisogna invece utilizzare i comandi specifici del NIS.

## 7.7 NFS

**NFS (Network File System)** è un protocollo che permette la condivisione di directory e file su una rete. L'effetto finale che ottiene un utente è poter accedere ed utilizzare file memorizzati su sistemi remoti come se fossero locali.

Lato **client**, l'inserimento nel proprio filesystem di una directory situata in un host remoto viene detto **mounting,** realizzata col comando ***mount***. Il processo demone che gestisce l'I/O è **biod**.

Lato **server**, la condivisione di una directory locale ad host specifici è detta **sharing** e viene realizzata col comando ***export***, configurabile tramite il file **/etc/exports** dove è possibile elencare directory da esportare con relativi host che hanno l'accesso. Il processo demone che gestisce le richieste NFS è **nsfd**.

Gli sviluppatori hanno implementato NFS creando 3 software indipendenti, compreso **NFS**. Gli altri due sono **RPC** e **XDR**.

**RPC (Remote Procedure Call)** consente di codificare e decodificare le richieste tra i client ed i server, permettendo di accedere ai file system remoti con le stesse procedure che utilizzano per accedere ai file locali.

**XDR (eXternal Data Representation)** consente di scambiare dati tra macchine con architettura eterogenea fornendo una rappresentazione dei dati indipendente dalla macchina. Grazie ad esso, non bisogna preoccuparsi della conversione tra le diverse rappresentazioni dei dati a livello hardware.

## 7.8 SNMP

**SNMP (Simple Network Management Protocol)** è ad oggi il più potente e diffuso protocollo di **gestione di reti**, sistemi e applicazioni, introducendo una semplice **architettura** chiamata **INMF (Internet Network Management Framework)**.

SNMP definisce la modalità di scambio di informazioni tra apparecchiature di rete, consentendo inoltre agli amministratori di tenere sotto controllo le prestazioni della rete e accorgersi in tempo reale di malfunzionamenti della stessa.

Prevede **2 ruoli**:

- **Agent**: moduli software che risiedono sui dispositivi da gestire (come *host, stampanti, router, switch, hub…*);
- **Manager**: uno o più host che possono interrogare e inviare comandi agli agent, tramite UDP, sulle porte 161 e 162.

SNMP è il protocollo di gestione e definisce le modalità d'interazione tra il manager e gli agent.

Le informazioni trattate dagli agent sono dette **managed objects** (in italiano vengono detti semplicemente **oggetti**) e vengono raccolte in un database chiamato **MIB (Management Information Base)**. Un MIB ha una **struttura ad albero**:

- i **nodi** sono organizzazioni e sotto-organizzazioni rappresentate da una etichetta ed un numero. Il nodo più importante è **mib-2**, dove vengono raccolte le variabili usate da SNMP attualmente raggruppabili in 12 categorie (altre verranno aggiunte);
- le **foglie** rappresentano i managed objects. Ogni oggetto all'interno del MIB viene identificato mediante un **OID (Object ID)**, ovvero la sequenza di numeri dei nodi attraversati nel cammino dalla radice alla foglia, intervallati da punti.

Le strutture dati usate nel MIB sono definite sotto forma di regole nello **SMI (Structure of Management Information)**, specificate con lo standard ISO chiamato ASN.1 (Abstract Syntax Notation One), un linguaggio di definizione di oggetti. In particolare vengono definiti nome, sintassi e codifica.

SNMP conta finora 3 versioni. La **terza versione** del protocollo rimedia alle carenze in merito a **sicurezza e privacy** delle versioni precedenti, tramite un sistema di autenticazione e strumenti di controllo agli accessi.

Il software open source più popolare per la gestione delle reti utilizzando SNMP è **nagios**.

## 7.9 DHCP

**DHCP (Dynamic Host Configuration Protocol)** permette agli host di una rete locale di ricevere, ad ogni loro richiesta di accesso ad una rete, la configurazione IP necessaria per stabilire una connessione ed operare. Utilizza **UDP** sulla **porta** 67 (lato server) e 68 (lato client).

Come RARP e **BOOTP** (**Bootstrap Protocol**, predecessore di DHCP, ormai in disuso perché permetteva solo la configurazione manuale), anche DHCP utilizza l'architettura **client/server**: il **server** (un host, molto spesso un **router**) provvede all'assegnazione degli indirizzi IP ai **client** che li richiedono.

Poiché il server DHCP potrebbe non essere raggiunto dalle trasmissioni broadcast (i router non fanno passare i pacchetti broadcast), è necessario installare in ogni LAN un **DHCP relay agent (agente di inoltro DHCP)**, che si occupa di intercettare e rigirare in modalità ***unicast*** al server DHCP (che può trovarsi anche in una rete distante) le trasmissioni DHCP intercettate. È chiaro che il relay agent ha bisogno di un'unica informazione: l'indirizzo IP del server DHCP.

DHCP presenta 3 opzioni di configurazione:

- **Automatica**: i client ottengono un indirizzo IP permanente;
- **Dinamica**: ad ogni nuova connessione i client ottengono un indirizzo IP, il quale ha un tempo di validità (detto ***lease***) al cui termine ritorna tra gli indirizzi disponibili. Ciò permette di riutilizzare indirizzi non più in uso dai client;
- **Manuale**: i client ottengono un indirizzo IP assegnato dall'amministratore e il DHCP è utilizzato semplicemente per comunicare l'indirizzo scelto.

Il processo di assegnamento degli indirizzi si divide in 4 fasi:

- **Discovering**: il client invia in modalità broadcast un pacchetto **DHCP Discover**, per richiedere l'assegnamento di un indirizzo. Si noti che in questa fase il client è ancora sprovvisto di indirizzo IP e il pacchetto avrà come mittente 0.0.0.0;
- **Offering**: i server che ricevono la richiesta, rispondono (se hanno indirizzi liberi a disposizione) con **DHCP Offer**, in cui propongono una configurazione IP al client;
- **Requesting**: una volta ricevute le offerte dai server, il client le valuta e risponde in modalità broadcast con **DHCP Request**, in cui comunica quale server ha scelto;
- **Acknowledgment**: se l'assegnamento è avvenuto con successo, il server risponde al client con **DHCP ACK**, altrimenti con **DHCP NACK (Negative Acknowledgment)**.

Nei sistemi Unix, una possibile implementazione Open Source del DHCP è rappresentata da **dhcpd**, la cui configurazione viene eseguita mediante un file di testo chiamato **dhcpd.conf**.

## 7.10 NAT

Il problema dell'esaurimento degli indirizzi IP non è un problema teorico che potrebbe presentarsi in un lontano futuro: sta accadendo proprio qui e ora. La soluzione a lungo termine è rappresentata da IPv6, ma la transizione sta procedendo lentamente e si è resa necessaria una soluzione attuabile in tempi brevi.
La soluzione adottata si chiama **NAT (Network Address Translation)**. L'idea di base di NAT è assegnare ad ogni azienda un singolo indirizzo IP (o, al massimo, un piccolo numero di indirizzi) per il traffico di Internet. Dentro l'azienda, ogni host riceve un indirizzo IP privato e locale. Nessuno di questi indirizzi potrà apparire su Internet, infatti quando un pacchetto lascia l'azienda e va verso Internet, viene effettuata una **traduzione di indirizzo** (di solito dal router).

# 8 Posta Elettronica

La **posta elettronica** (o **e-mail**, dall'inglese **electronic mail**) è un servizio Internet che permette lo scambio di messaggi tra utenti.

I sistemi di posta elettronica sono normalmente composti da 2 software:

- **MUA (Mail User Agent, agente utente)**: consente alle persone di leggere e inviare la posta elettronica (es: gmail, outlook…).
  Costituisce un'interfaccia utente con le seguenti funzionalità:
  - **Composizione**: processo di creazione di messaggi e risposte, che comprende assistenza per l'indirizzario dei corrispondenti e i numerosi campi di intestazione associati a ogni messaggio. Si pensi ai casi in cui si sceglie di rispondere ad una mail e vengono predisposti già alcuni campi;
  - **Visualizzazione**: lettura dei messaggi in arrivo, che comprende fasi preliminari di conversione per gli allegati e strumenti per l'archiviazione dei messaggi;
  - **Collocazione**: operazione svolta dal destinatario con il messaggio dopo la ricezione. Il messaggio può essere cancellato, recuperato, riletto, inoltrato o elaborato in altri modi.
- **MTA (Mail Transfer Agent, agente di trasferimento)**: sposta i messaggi dall'origine alla destinazione (es: **sendmail**).
  Corrisponde al "postino" della posta reale, con le seguenti funzionalità:
  - **Trasferimento**: spostamento dei messaggi dal mittente al destinatario. Avviene tramite l'attivazione di una sessione con il server di destinazione o qualche macchina intermedia qualora il server sia congestionato o irraggiungibile e sia disponibile il servizio di ***mail relay***, un gestore di posta secondario, che in caso di necessità sia in grado di salvare temporaneamente i messaggi in transito, facendo le veci del server di destinazione. Siccome veniva sfruttata dagli spammer, che si fingevano server secondari, adesso si indicano espressamente gli host dai quali si accetta relaying;
  - **Reporting**: comunicazione al mittente di ciò che è avvenuto al messaggio, ad esempio se è stato consegnato, rifiutato, perso o si è presentato qualche errore.

  Nel caso di *sendmail*, la configurazione manuale (da scrivere nel file **/etc/mail/sendmail.cf**), è diventata nel tempo complessa, ed è stata semplificata mediante l'uso di uno pseudolinguaggio denominato **M4**, che consente di attivare delle funzioni mediante invocazione di macro. Alcuni file che riguardano *sendmail* sono:
  - **/etc/mail/access**: domini o host per i quali si controlla l'accesso;
  - **/etc/mail/relay-domains**: domini per i quali si accetta relaying;
  - **/etc/mail/sendmail.cw**: domini per i quali si fa virtual hosting;
  - **/etc/aliases**: alias per indirizzi di posta elettronica.

La maggior parte dei sistemi di posta consente di creare **mailbox (caselle di posta)** per archiviare la posta in arrivo di singoli utenti. Talvolta però, può risultare utile inviare un messaggio a più persone correlate tra loro. Da qui nasce l'idea delle **mailing list**, elenchi di posta elettronica, in cui quando viene inviato un messaggio, copie identiche vengono consegnate a tutti i membri dell'elenco. Tutti gli indirizzi di posta elettronica sono del tipo:

***utente@indirizzo-dns***

Dove *indirizzo-dns* è un dominio DNS (che può contenere sottodomini).

## 8.1 RFC822

**RFC822** definisce una serie di **campi di intestazione** per le e-mail. Ogni campo consiste in una singola riga di testo ASCII contenente il **nome del campo**, un carattere di **due punti** e, per la maggior parte dei campi, un **valore**.

Di seguito vengono elencati i campi di intestazione principali:

| **INTESTAZIONE** | **SIGNIFICATO** |
|---|---|
| Received: | I server attraversati, aggiunti da ogni **MTA** lungo il percorso |
| Message-ID: | Identificativo del messaggio |
| Date: | Data e ora di invio del messaggio |
| From: | La persona che ha creato il messaggio |
| Sender: | L'indirizzo di posta elettronica del mittente vero e proprio |
| Subject: | Breve riepilogo del messaggio da visualizzare su una riga |
| To: | Gli indirizzi di posta elettronica dei destinatari principali |
| Cc: | Gli indirizzi di posta elettronica dei destinatari secondari |
| Bcc: | Gli indirizzi di posta elettronica nascosti agli altri destinatari |

### 8.1.1 MIME

Per poter far fronte alle nuove esigenze di codifica di caratteri speciali (es: accenti, alfabeti non latini…) e contenuti multimediali (es: audio, immagini…), gli RFC1341 e 1521 hanno introdotto lo standard di codifica **MIME (Multipurpose Internet Mail Extensions)**. Esso aggiunge nuovi campi di intestazione all'RFC822, senza influire sul normale funzionamento del MTA.

Di seguito alcuni dei più importanti:

| **INTESTAZIONE** | **SIGNIFICATO** |
|---|---|
| MIME-Version: | Versione di MIME |
| Content-Description: | Contenuto del messaggio in forma leggibile |
| Content-Id: | Un identificatore univoco |
| Content-Transfer-Encoding: | Indica come il corpo del messaggio<br>è stato preparato per la trasmissione. |
| Content-Type: | Il tipo e il formato del contenuto |

Per il *Content-Transfer-Encoding* di seguito sono elencati i principali schemi di codifica:

- **ASCII**: la codifica più diffusa, rappresenta i caratteri in 7 bit e per ciascuna riga del messaggio non si può eccedere i 1000 caratteri;
- **EBCDIC**: come ASCII e con la sua stessa limitazione, ma rappresenta i caratteri in 8 bit;
- **Base64**: a volte chiamato armatura ASCII, è utilizzata spesso per rappresentare i file binari che violano il limite di 1000 caratteri per riga. Vengono fatti gruppi di 24 bit, a loro volta suddivisi in 4 unità di 6 bit, che vengono inviate come carattere ASCII legale. Le sequenze == e = indicano rispettivamente che l'ultimo gruppo contiene 8 o 16 bit;
- **Quoted-printable**: per i messaggi quasi interamente ASCII ma contenenti alcuni messaggi non ASCII, la codifica base64 è inefficiente e dal suo posto si utilizza questa. Si tratta di ASCII a 7 bit, con tutti i caratteri superiori a 127 codificati come segni uguale seguiti dal carattere espresso da due cifre esadecimali.

Il *Content-Type* è costituito da 7 tipi, ognuno con un certo numero di sottotipi, ed è nella forma:

**Content-Type: tipo/sottotipo**

I 7 tipi sono: Text, Image, Audio, Video, Application, Message, Multipart.

## 8.2 Il trasferimento dei messaggi: SMTP

**SMTP (Simple Mail Transfer Protocol)** è un protocollo adibito al trasporto di messaggi efficiente ed affidabile, in ambienti eterogenei che coinvolgono un client ed un server (dove il server può essere il destinatario finale o un server intermedio) tramite la realizzazione di un **IPCE (InterProcess Comunication Environment)**, ossia la realizzazione di un circuito virtuale tra server e client che consente la consegna del messaggio.

Come prima cosa, il client stabilisce una connessione **TCP** con la **porta 25** del server ed attende una risposta. Il server invia quindi una riga di testo che comunica la sua identità e la possibilità di inviare la posta, con conseguente attivazione del canale trasmissivo. Se ciò non avviene, il client rilascia la connessione e riprova in seguito.

Una volta attivato il canale, il primo comando obbligatorio che il client deve inviare è **HELO**, necessario per inizializzare una sessione SMTP ed identificarsi. Se l'argomento non è accettabile, il server ritorna un errore *501 Failure*.

In seguito, il client può inviare una sequenza di comandi, a passi bloccanti. Per l'invio di messaggi, si usano in sequenza:

- **MAIL FROM**: viene passato l'indirizzo del mittente;
- **RCPT TO**: viene passato l'indirizzo del destinatario. Può essere ripetuto per specificare più destinatari;
- **DATA**: vengono passati i dati del messaggio. Al termine dell'immissione del testo, deve seguire la sequenza ***<CR><LF>.<CR><LF>***, dove *<CR><LF>* è una serie di caratteri che viene interpretato come "vai a capo".

La transazione può essere abortita in qualsiasi momento, con il comando **RSET**. Ogni qualvolta il server interpreta correttamente un messaggio, risponde con **OK**. Altri comandi utilizzabili ovunque durante una sessione sono *NOOP, HELP, EXPN* e *VRFY*. L'ultimo comando di una sessione è **QUIT**.

## 8.3 POP3

**POP3** (**Post Office Protocol** version **3**), noto anche come **Popper**, è un protocollo client-server che permette l'accesso da remoto ad un server di posta elettronica. Una volta connesso al server, è possibile recuperare la posta (conservandola in locale) e/o cancellare i messaggi dal server SMTP. Si noti che l'invio della posta resta comunque affidato a SMTP.

Esso viene avviato quando l'utente apre il lettore della posta. Il lettore della posta chiama l'ISP (a meno che sia già disponibile una connessione) e stabilisce una connessione TCP con il MTA sulla porta 110. Una volta stabilita la connessione, POP3 attraversa sequenzialmente 3 stati:

- **Authorization**: login dell'utente (si noti che le credenziali vengono passate in chiaro), effettuato tramite i comandi **USER** e **PASS**;
- **Transaction**: ottenimento o cancellazione della posta dalla casella dell'ISP. I comandi utilizzati sono:
    - **STAT**: ritorna numero dei messaggi nella casella e dimensione in byte;
    - **LIST** *[msg]*: dato un msg (ID del messaggio), ritorna ID e dimensione. Se non si passa alcun ID, elenca tutti i messaggi (con ID e dimensione di ognuno);
    - **RETR** *msg*: ricezione del messaggio specificato;
    - **DELE** *msg*: cancellazione del messaggio specificato;
    - **NOOP**: *No Operation*, si usa per mantenere la connessione aperta;
    - **LAST**: ritorna l'ID dell'ultimo messaggio ricevuto;
    - **RSET**: resetta la connessione al suo stato iniziale, smarcando tutti i messaggi marcati per l'eliminazione.
- **Update**: provoca l'effettiva eliminazione della posta, terminando la sessione. Si effettua tramite il comando **QUIT**.

Il server risponde alle richieste con esito positivo (**+OK**) oppure con esito negativo (**-ERR**).

## 8.4 IMAP

Uno svantaggio di POP3 è che i messaggi devono essere scaricati sul client per poter essere letti. Si pensi al caso in cui si vuole accedere alla stessa casella di posta elettronica da macchine diverse (smartphone, portatile, computer fisso…): utilizzando POP3, si ottiene come risultato che la posta dell'utente viene rapidamente dispersa su più macchine.

Questo svantaggio ha portato alla nascita di un protocollo di consegna alternativo, **IMAP (Internet Message Access Protocol)**, un protocollo definito in RFC2060 che utilizza la porta 143. A differenza di POP3, che presume l'utente cancelli la casella di posta ogni volta, IMAP presume che tutti i messaggi rimangano sul server per un tempo indeterminato.

Le differenze e migliorie con POP3 non sono poche e di seguito se ne vedono alcune:

- **Decine di nuovi comandi e opzioni** (molta più flessibilità e personalizzazione);
- **Leggere in tutto o in parte i messaggi** (il che ad esempio è utile mentre si utilizza un modem lento, per leggere il testo di un messaggio con grandi allegati audio e video);
- **Effettuare ricerche indicizzate** (senza il bisogno di scaricare tutte le mail);
- **Sincronizzazione** tra i vari client (nel caso si acceda da più client simultaneamente);
- **Supporto per attributi dei messaggi** (per esempio per sapere se un messaggio è già stato letto o se ha avuto una risposta).

## 8.5 Posta elettronica privata

*L'argomento è stato trattato in maniera sintetica.*

Anche per la posta elettronica può essere necessario garantire la segretezza dei messaggi, ma nativamente non ci sono meccanismi che possono garantirla. È risaputo che crittografia e certificati digitali permettono di risolvere questo problema, e sono diversi i software che sono stati creati per implementarli nelle mail.

### 8.5.1 PGP

**PGP (Pretty Good Privacy)** è una famiglia di software multipiattaforma di crittografia utilizzati per autenticazione e privacy. Nell'ambito della posta elettronica, consente la crittografazione, la firma digitale e la compressione dei messaggi ed è attualmente utilizzato da *Enigmail* e *Mozilla Thunderbird*. Utilizza 3 algoritmi di crittografazione durante le sue operazioni: **RSA, IDEA** e **MD5**.

Un'alternativa Open Source è **GnuPG**, pensato come rimpiazzato di PGP. Utilizza un metodo per la condivisione efficiente delle chiavi pubbliche, usando una rete particolare: **Web of Trust**.

# 9 News

Molto tempo prima di Internet esisteva il sistema di news **USENET**, un sistema distribuito per il dialogo ed il confronto tra utenti. Consiste in circa 30.000 **newsgroup,** gruppi di discussione gerarchici in cui si inviano e leggono articoli correlati. Una newsgroup è indicata nella forma: **radice.gruppo.sottogruppo**, con un numero di sottogruppi indefinito. Un utente vi si può iscrivere, in modo da ricevere la lista dei messaggi del gruppo, e a differenza delle mail, le news vengono scaricate sul client solo nel momento in cui l'utente vi accede.

Quando un utente invia un messaggio ad un newsgroup, si dice che effettua un **post**. Nel caso in cui i messaggi vengono inviati a diversi newsgroup, si parla di **crosspost**. I messaggi nei gruppi sono però **moderati**, ovvero passano per un intermediario, detto **moderatore**, che determina la pubblicazione o il rifiuto di un articolo. Questa scelta viene anche basata sulla **netiquette**: delle regole informali che disciplinano il buon comportamento di un utente.

In alcuni casi, come la creazione di un particolare newsgroup, è possibile indire una **CfV (Call for Votes)**, dove gli utenti possono esprimere le loro preferenze. Generalmente, le CfV vengono effettuate sotto la radice **news**, mentre sotto la radice **alt** c'è più libertà, ed è in genere dove si può trovare di tutto, compreso materiale al limite della legalità.

Essendo un sistema distribuito, è necessario scambiare i messaggi tra i vari server dislocati. Il meccanismo per assolvere questo compito è detto **newsfeed**, dove il server che riceve il messaggio, lo inoltra ai server vicini, i quali a loro volta effettueranno le stesse operazioni fino a raggiungere tutti. I messaggi vengono archiviati nei server fino ad una data di scadenza prefissata e variabile da gruppo a gruppo, momento nel quale vengono fisicamente cancellati.

## 9.1 NNTP

**NNTP (Network News Transfer Protocol)** è il protocollo client/server che regola lo scambio dei messaggi tra i server nel meccanismo di newsfeed. Nello specifico, definisce distribuzione, interrogazione, accesso e invio di news. Utilizza TCP e dialoga sulla porta 119.
Quando c'è un nuovo messaggio, un server informa gli altri tramite il comando **IHAVE**.

# 10 World Wide Web

Il **World Wide Web** (abbreviato **WWW** o **Web**) è un'architettura che consente di accedere a documenti distribuiti su un vasto numero di macchine nell'Internet globale.

I documenti sono tecnicamente chiamati **pagine Web** (spesso dette solo **pagine** per brevità) e possono essere di 2 tipi: **statiche** (la pagina ha sempre lo stesso contenuto ad ogni visualizzazione) o **dinamiche** (viene generato il contenuto della pagina ad ogni richiesta, tramite un linguaggio di scripting come il PHP). Ogni pagina è localizzata da uno o più **URL (Uniform Resource Locator)**, una stringa composta di 3 parti:

- Il nome del protocollo (es: http, https);
- Il nome DNS della macchina su cui è sita la risorsa (es: www.unipi.it);
- La posizione locale sul server della risorsa richiesta (es: index.php).

Un esempio di URL è quindi: https://www.unipi.it/index.php. Si noti che l'URL è un tipo specifico di **URI (Uniform Resource Identifier)**. La differenza sta nel fatto che l'URI identifica una risorsa, mentre l'URL la localizza. Si noti che localizzare una risorsa vuol dire conseguentemente anche identificarla, ma non è sempre vero il contrario, per cui i due termini non possono essere usati come sinonimi: ad esempio, l'ISBN di un libro x è un URI (in quanto univoco), ma non localizza il libro, per cui non è anche un URL.

Ogni pagina può contenere collegamenti ad altre pagine situate ovunque nel mondo. L'insieme delle pagine collegate tra loro è detto **hypertext (ipertesto)**, ed il collegamento cliccabile che reindirizza ad altre pagine è detto **hyperlink** o **link (collegamento ipertestuale)**.

Il software per l'ottenimento, il rendering e la navigazione di pagine è detto **browser**.

## 10.1 HTTP

Il protocollo di trasferimento usato sul Web è **HTTP (HyperText Transfer Protocol)**, un protocollo **stateless** (ovvero non salva informazioni sulla sessione) che permette la realizzazione di sistemi informativi distribuiti, collaborativi ed ipermediali (ossia composti da *multimedialità* distribuita nella rete ed acceduta mediante *hyperlink*).

Il metodo utilizzato da un browser per contattare un server prevede di stabilire una connessione TCP di solito alla porta 80. Da HTTP/1.1 sono supportate le **connessioni persistenti**, che permettono di inviare più di una richiesta al server con la stessa connessione.

Una **HTTP request** (richiesta HTTP) è composta da:

- **Request line**: singola stringa composta da un metodo (**GET, POST, HEAD, PUT, DELETE, TRACE, OPTIONS** o **CONNECT**), un URL e la versione del protocollo.
  Un esempio è: GET url_del_file HTTP/1.1;
- **Header**: una o più stringhe contenenti ulteriori informazioni, seguiti da una riga vuota. Tra i più comuni vi sono:
  - **Host**: indica il nome del server a cui si riferisce l'URL;
  - **User-agent**: identifica il tipo di client: tipo di browser, versione, ecc.;
  - **Cookie**: utilizzati dalle applicazioni web per archiviare e recuperare informazioni a lungo termine sul lato client.
- **Body**: dati opzionali che si vogliono trasmettere.

Una **HTTP response** (risposta HTTP) è composta da:

- **Status line**: una riga contente uno status code ed un messaggio esplicativo. Gli status code sono dei numeri a 3 cifre che la cui prima cifra specifica una delle 5 categorie:
  - 1xx Informational;
  - 2xx Success;
  - 3xx Redirection;
  - 4xx Client Error;
  - 5xx Server Error.
- **Header**: una o più stringhe contenenti ulteriori informazioni, seguiti da una riga vuota. Tra i più comuni vi sono:
  - **Content-Type**: contiene il tipo MIME della pagina;
  - **Server**: contiene informazioni sul server;
  - **Set-Cookie**: contiene un cookie che il server vuole che il client salvi.
- **Body**: dati opzionali che si vogliono restituire.

Vi sono state varie versioni di HTTP. Di seguito vengono analizzati in breve i cambiamenti principali tra le versioni:

- **HTTP/0.9**: semplice protocollo per il trasferimento di dati grezzi sulla rete Internet, prima di esso il protocollo di riferimento per tali scopi era FTP;
- **HTTP/1.0**: pur consentendo il trasferimento di messaggi di tipo MIME, non era adatto a supportare la crescita esponenziale del Web;
- **HTTP/1.1**: versione consolidata del protocollo, utilizzata per ben 15 anni;
- **HTTP/2.0**: nuovo standard basato su un protocollo sviluppato da Google chiamato SPDY/2, che va a migliorare di molto le prestazioni mantenendo la retrocompatibilità per applicazioni già sviluppate;

## 10.2 Pagine Web dinamiche

Come anticipato, esistono due categorie di pagine web: statiche e dinamiche. Per richiedere una pagina statica il procedimento è banale: si effettua una HTTP request con il nome del file e si ottiene una HTTP response con il documento richiesto. Per le pagine dinamiche il procedimento è un po' più articolato poiché, una volta ricevuta la richiesta HTTP, il documento va generato su richiesta, con possibilità che ciò avvenga sia lato client che server.

Per gestire le pagine Web dinamiche **lato server**, è possibile utilizzare un sistema chiamato **CGI (Common Gateway Interface)**. È un'interfaccia standardizzata che consente ai server Web di comunicare con script e programmi di back-end che possono accettare l'input e generare pagine HTML in risposta. Un linguaggio usato per la scrittura di CGI è **Perl**. È anche possibile **incorporare piccoli script nelle pagine HTML** stesse (ad esempio con il **PHP**), e farli eseguire al momento della generazione.

Il processo, che inizia alla ricezione di una HTTP request e termina all'invio di una HTTP response è detto **round trip**, ed essendo HTTP un protocollo stateless, i linguaggi di scripting server-side mettono a disposizione apposite variabili dette **variabili di sessione**, per poter salvare dei dati tra una connessione e l'altra.

Per gestire le pagine Web dinamiche **lato client**, è necessario allo stesso modo incorporare degli script nelle pagine HTML. Il linguaggio di script più popolare per il lato client è **JavaScript**.

# 11 Sicurezza delle reti

Tra gli attributi dell'oggetto dell'analisi di sicurezza informatica, rientra la seguente triade:

- **Confidenzialità**: le informazioni possono essere **lette** solo dagli aventi diritto;
- **Integrità**: le informazioni possono essere **modificate** solo dagli aventi diritto;
- **Disponibilità**: le informazioni possono essere **accedute** al momento del bisogno.

Questa triade prende il nome di **CIA,** dalle iniziali inglesi dei 3 termini.
In seguito, sono state definite 2 nuove proprietà:

- **Accountability**: ogni utente e azione devono poter essere identificati;
- **Auditability**: l'efficacia dei meccanismi scelti deve poter essere verificata.

Nell'ambito della sicurezza informatica, è inoltre comune usare i seguenti termini:

- **Privacy**: il diritto che ha ogni individuo di determinare quando, quanto, come e a chi comunicare le informazioni che riguardano sé stesso;
- **Non ripudio**: fornire l'evidenza irrefutabile che una certa azione è stata compiuta da una determinata persona e non da altre o che un certo messaggio è stato spedito da un determinato mittente (firma digitale);
- **Autenticazione**: fornire la prova che l'utente è esattamente chi dice di essere (da non confondere con l'identificazione dell'accountability);
- **Tracciabilità**: capacità di tracciare le azioni compiute dagli utenti sulle risorse;
- **Forensics**: capacità di provare che determinati attacchi hanno avuto luogo.

Definiamo inoltre **vulnerabilità** un difetto di una componente di un sistema informatico, classificandola in 3 tipi principali:

- **Procedurali**: difetto nella modalità con cui si opera;
- **Organizzative**: difetto nelle persone che operano;
- Degli **strumenti informatici**: difetto nell'hardware/software utilizzato. Per questo tipo di vulnerabilità, esistono 3 sottotipi:
    - **Specifica**: difetto nella **progettazione** di un componente: ad esempio quando un componente è, inutilmente, più generale del necessario;
    - **Implementativa**: difetto nella **realizzazione** di un componente: ad esempio la mancanza di controlli sugli input;
    - **Strutturale**: difetto che nasce dalla **combinazione** di uno o più componenti del sistema.

Sfruttando una vulnerabilità, è possibile effettuare un **attacco informatico**. In generale, le varie fasi sono:

- Raccolta di informazioni;
- Individuazione delle vulnerabilità;
- Ricerca o costruzione di un programma (detto **exploit**) che sfrutti le vulnerabilità;
- Esecuzione dell'exploit;
- Installazione di strumenti per il controllo;
- Cancellazione delle tracce;
- Accesso ad un sottoinsieme di informazioni.

Alcune delle categorie dei possibili attacchi che possono andare a violare l'affidabilità di un server Web sono:

- **Estensibilità del server**: attacchi ai servizi offerti dal server. Ad esempio, potersi connettere ad un database via CGI;
- **Estensibilità del browser**: attacchi ai servizi fruibili da browser. Ad esempio, alcuni usi di ActiveX, Java, Javascript, VBSript possono compromettere la sicurezza;
- **Distruzione del servizio**: attacchi mirati a bloccare l'erogazione di un servizio. Un esempio sono gli attacchi DoS;
- **Supporto complicato**: attacchi mirati alle vulnerabilità dei protocolli di basso livello, alcune volte necessari per un funzionamento corretto del sistema.

Per poter descrivere lo stato di sicurezza di un sistema informatico si effettua la cosiddetta **analisi del rischio**. L'analisi del rischio consiste nell'identificazione dei beni da proteggere, valutandone le possibili minacce in termini di probabilità di occorrenza e relativo danno potenziale. In base alla stima del rischio si decide se, come e quali contromisure di sicurezza adottare, giustificando i costi per la messa in sicurezza.

L'analisi del rischio comprende:

- **Analisi delle vulnerabilità**: descrive le vulnerabilità di un sistema informatico;
- **Analisi degli attacchi**: descrive le tipologie di attacchi plausibili per le vulnerabilità trovate e, per ogni attacco, le informazioni e le risorse necessarie ad eseguirlo;
- **Analisi degli impatti**: per ogni attacco, si stabilisce la perdita dell'azienda;
- **Analisi delle minacce**: per ogni attacco, si stabilisce chi ha interesse nell'attacco e chi dispone delle risorse necessarie ad eseguirlo;
- **Individuazione del rischio accettabile ed introduzione delle contromisure**: si determinano le contromisure da attuare ed il rischio residuo, talvolta trasferibile a terzi (come le assicurazioni).

## 11.1 SSL

*Il funzionamento di SSL e TLS non sono stati trattati durante le lezioni, per cui si è evitato di spiegare i vari processi.*

La sicurezza nel Web è un argomento molto vasto e sicuramente mettere in sicurezza le connessioni è uno degli obiettivi più importanti oggigiorno. Per raggiungere questo obiettivo, è stato inventato un pacchetto per la sicurezza chiamato **SSL (Secure Socket Layer)**, un encryption system a doppia chiave pubblica e privata, usato nei server per garantire la privacy durante le trasmissioni su Internet.

SSL si pone in una sorta di strato intermedio della pila OSI, tra il livello Applicazione e il livello Trasporto, permettendo di cifrare le informazioni prima dell'invio ai client e evitando la lettura da parte di potenziali terzi in grado di intercettare il traffico tra server e client.

Ogni server deve avere una doppia chiave (pubblica e privata) e un certificato X.509 usato per firmare i dati, rilasciato da una *Certificate Authority (CA)*, degli enti considerati *trusted*.

Per effettuare una connessione con SSL, si deve:

1) Negoziare l'algoritmo di crittografia (come ad esempio DES o IDEA)
2) Il server si autentica al client (sempre) ed il client si autentica al server (opzionalmente);

Quindi è possibile inviare dati criptati con la chiave concordata.

Si noti che HTTP usato sopra SSL prende il nuovo nome di **HTTPS (Secure HTTP)**, anche se si tratta ancora del protocollo HTTP standard ed è normalmente disponibile su una porta diversa, la **443**.

Nel 1996, Netscape Corp. (creatrice di SSL), sottopose il proprio protocollo all'IETF per la standardizzazione. Il risultato fu **TLS (Transport Layer Security)**, anche noto con il nome di *SSL 3.1*. I cambiamenti operati all'SSL furono minimali e hanno reso TLS un po' più robusto, ma sono bastati a far sì che SSL versione 3 e TLS non riescano a comunicare tra loro. Non è ancora chiaro se TLS soppianterà SSL nella pratica.

## 11.2 Firewall

Un **firewall** è un sistema che ha lo scopo di controllare il traffico tra due o più reti sulla base di un insieme di regole. Si definiscono *trusted* la rete interna da proteggere (solitamente una LAN) e *untrusted* le reti esterne (generalmente una o più WAN). Può essere implementato sia lato hardware e software tramite un host con più schede di rete, costituendo il solo punto di collegamento tra le reti coinvolte, sia unicamente lato software sui vari host della rete interna da proteggere.

Ogni socket che viene aperto dall'interno verso l'esterno (e viceversa) viene verificato in base alle regole definite all'interno del software, che viene detto *packet filter*. Esistono due criteri generali per l'applicazione delle singole regole:

- *default-deny*: viene **permesso** solo quanto dichiarato esplicitamente, il resto è **bloccato**;
- *default-allow*: viene **vietato** solo quanto dichiarato esplicitamente, il resto è **permesso**.

I firewall normalmente utilizzano il criterio *default-deny*, in quanto garantisce una maggiore sicurezza e precisione nella definizione delle regole.

### 11.2.1 Packet Filter

*Il packet filter richiederebbe una spiegazione breve sul funzionamento del software iptables.*

Il packet filter è un tool che analizza l'intestazione dei vari pacchetti in entrata ed uscita e decide il da farsi in base a determinati criteri. Gli elementi utilizzati nel filtraggio sono:

- **Header IP**: che si ricorda contiene informazioni come indirizzo IP del mittente e del destinatario o il protocollo usato;
- **Header TCP/UDP**: che si ricorda contiene informazioni come le porte o le flag utilizzate.

Per ogni pacchetto è possibile effettuare una delle seguenti operazioni:

- **ACCEPT**: il pacchetto viene accettato e lasciato proseguire;
- **DROP**: il pacchetto viene scartato, senza mandare al mittente alcun messaggio;
- **REJECT**: come DROP, ma il mittente riceverà un pacchetto ICMP con un messaggio;
- **CHAIN** e **RETURN**: utilizzati per definire una concatenazione di regole.

L'insieme delle regole per il packet filtering è memorizzato in una tabella detta **filter**, che contiene a sua volta 3 diverse liste di regole, dette **chains**, distinte a seconda del percorso di routing del pacchetto:

- **INPUT**: riguarda i pacchetti che hanno come destinazione il firewall;
- **OUTPUT**: riguarda i pacchetti emessi dal firewall;
- **FORWARD**: riguarda i pacchetti che transitano attraverso il firewall.

Il packet filter più utilizzato (e preinstallato) su OS Linux è **iptables**.