
APPUNTI RETI DI CALCOLATORI

INSEGNAMENTO DI TENUTO DAL PROF. ANTONIO DELLA SELVA

BASATO SUGLI APPUNTI DI
STEFANO ZIZZI

*Università degli Studi di Urbino
Corso di Laurea in Informatica Applicata*



QUESTI APPUNTI COSTITUISCONO SOLTANTO UN AUSILIO PER LO STUDENTE,
NON SONO IN NESSUN MODO SOSTITUTIVI DEI TESTI CONSIGLIATI
NÉ TANTO MENO POSSONO SOPPERIRE COMPLETAMENTE ALLA MANCATA FREQUENZA ALLE LEZIONI.

Indice

1 Introduzione alle Reti di Calcolatori	7
1.1 Scopi delle Reti e Loro Classificazione	7
1.1.1 Scopi delle Reti di Calcolatori	7
1.1.2 Grids, Cloud e Virtualizzazione	10
1.2 Modelli di Riferimento OSI e TCP/IP	13
1.2.1 Software di rete	13
1.2.2 Il modello di Riferimento OSI	15
1.2.3 Il modello di Riferimento TCP/IP	18
1.2.4 Confronto tra i Modelli Presentati	20
1.3 Esempi di Reti e Standardizzazione delle Reti	20
1.3.1 Categorizzazione per Topologia	21
1.3.2 Categorizzazione per Tecnologia	22
1.3.3 Categorizzazione per Estensione	23
1.3.4 Standard di Rete e Organizzazioni	24
2 Livello Fisico	29
2.1 Basi Teoriche della Comunicazione e della Trasmissione Fisica dei Dati	29
2.1.1 Dati e Segnali	29
2.1.2 Modalità di Multiplazione	32
2.1.3 Criteri di Valutazione delle Reti	33
2.2 Mezzi di Trasmissione Guidati	36
2.2.1 Doppino o Cavo Ethernet	36
2.2.2 Cavo Coassiale	39
2.2.3 Fibra Ottica	39
2.2.4 Onde Convogliate	42
2.3 Mezzi di Trasmissione Wireless	42
2.3.1 Reti Satellitari	43
3 Livello Collegamento	45
3.1 Data Link Control	45
3.2 LAN Cablate: Ethernet	45
3.2.1 Sottolivello MAC	46
3.2.2 Lo Strato Fisico	49
3.3 LAN Wireless: IEEE 802.11 - Bluetooth	51
3.3.1 IEEE 802.11 - WiFi	52
3.3.2 Bluetooth	55
3.4 LAB - (Copertura Wireless - Potenza del Segnale)	58
4 Tecnologie di Accesso	59
4.1 Local loop e Vecchie Tecnologie	59
4.2 xDSL	62
4.3 Tecnologia d'Accesso in Fibra Ottica	63
4.4 Fixed Wireless Access and Satellite	65

5 Livello Rete	67
5.1 Problemi Architetturali dello Stato Rete	67
5.2 Protocollo IPv4 e Indirizzamento	67
5.2.1 Pacchetti IPv4	68
5.2.2 Indirizzamento IP	69
5.3 Protocollo IPv6 e Indirizzamento	71
5.3.1 Prefix Delegation e Subnetting	72
5.3.2 Altre Novità di IPv6	73
5.4 Algoritmi di Routing	74
5.4.1 Distance Vector	75
5.4.2 Link State	76
5.5 Congestione	78
5.5.1 Controllo Proattivo	78
5.5.2 Controllo Reattivo	79
5.5.3 Leacky Bucket/Token Bucket	80
5.5.4 Load Shedding	81
6 Internetworking	83
6.1 Protocolli di Controllo	83
6.1.1 Internet Control Message Protocol (ICMP)	83
6.1.2 ARP – Address Resolution Protocol	83
6.1.3 DHCP – Dynamic Host Configuration Protocol	84
6.2 NAT - Network Address Translation	85
7 Livello Trasporto	89
7.1 Descrizione dei Servizi di Trasporto	89
7.1.1 Primitive	89
7.1.2 Socket di Berkeley	91
7.2 Elementi dei Protocolli di Trasporto	91
7.2.1 Indirizzamento	92
7.2.2 Stabilire una Connessione	93
7.2.3 Rilascio della Connessione	94
7.2.4 Controllo degli Errori e Controllo di Flusso	95
7.2.5 Multiplexing	97
7.2.6 Ripristino Dopo un Crash	98
7.2.7 Protocolli a Finestra Scorrevole	99
7.3 Il Protocollo di Trasporto Internet Senza Connessione: UDP	100
7.3.1 Remote Procedure Call: RCP	102
7.3.2 Protocolli di Trasporto Real-Time	102
7.3.2.1 RTP	103
7.3.2.2 RTCP	104
7.4 Il Protocollo di Trasporto Internet Orientato alla Connessione: TCP	104
7.4.1 Il Modello di Servizi	105
7.4.2 Intestazione del Segmento TCP	107
7.4.3 Instaurazione di una Connessione	108
7.4.4 Rilascio di una Connessione	109
7.4.5 Modello di Gestione della Connessione	110
7.4.6 La Finestra Scorrevole	111
7.4.7 Controllo della Congestione	112
7.4.8 Gestione de Timer	115
7.4.9 WebRTC	116
7.5 LAB - Socket	116
7.6 LAB - Implementazione Sistemi Client-Server	116

8 Livello Applicazione	117
8.1 Posta Elettronica: MIME, SMTP, IMAP, POP3	117
8.1.1 Formato dei Messaggi	118
8.1.1.1 RFC 5322	118
8.1.1.2 MIME	119
8.1.2 Consegnna Finale	119
8.1.2.1 IMAP	120
8.1.2.2 Webmail	120
8.2 Protocolli di Trasporto File	120
8.2.1 FTP	120
8.2.2 TFTP	121
8.3 DNS - Domain Name System	121
8.3.1 Campi del Protocollo	122
8.3.2 Server	124
8.3.3 Problemi	125
8.4 World Wide Web	126
8.4.1 Il Lato Client	127
8.4.1.1 Minacce alla Sicurezza	128
8.4.2 Il Lato Server	129
8.4.3 Cookie	130
8.4.4 HTTP	131
8.4.5 Caching	132
8.5 Distribuzione dei Contenuti	132
8.5.1 Server Farm e Proxy Web	133
8.5.1.1 Server Farm	133
8.5.1.2 Proxy Web	134
8.6 Content Delivery Network	135
9 Sicurezza delle Reti	139
9.1 Controllo degli Accessi	139
9.2 Crittografia	140
9.3 Protocolli di Autenticazione	141
9.4 Certificati Digitali	142

Capitolo 1

Introduzione alle Reti di Calcolatori

1.1 Scopi delle Reti e Loro Classificazione

1.1.1 Scopi delle Reti di Calcolatori

La convergenza di computer e comunicazioni ha avuto un'influenza profonda sul modo in cui sono evoluti i dispositivi digitali così come li conosciamo oggi.

Il concetto di *centro di calcolo* come una stanza con un grande computer dove gli utenti portano il loro lavoro per l'elaborazione oggi è completamente superato. Il vecchio modello di un solo computer che soddisfa l'intera necessità di calcolo dell'organizzazione è stato sostituito da un altro, in cui il lavoro è svolto da un gran numero di nodi digitali distinti ma interconnessi. Questi sistemi sono chiamati **reti di calcolatori**. Due computer sono interconnessi se sono in grado di scambiare informazioni tra di loro. Quindi una rete di calcolatori è una connessione (fisica e logica) di due o più dispositivi per la ricezione e la trasmissione di informazioni e/o per la condivisione di risorse hardware e software.

L'obiettivo della **condivisione delle risorse** è quello di rendere disponibili a chiunque sulla rete tutti i programmi, le periferiche e soprattutto i dati, indipendentemente dalla posizione fisica dell'utente e della risorsa.

Spesso una rete può venire confusa con un **sistema distribuito**, un sistema quest'ultimo, atto all'espletamento di un determinato compito e basato su una rete di calcolatori. La garanzia della comunicazione e la coerenza delle informazioni scambiate in un sistema distribuito è compito del **middleware**. Si possono identificare come middleware i *DBMS*, i *Web Server* e tutti gli strumenti basati sul concetto di sviluppo e pubblicazione di applicazioni e contenuti.

L'interconnessione delle reti informatiche mondiali identifica **Internet**, mentre un sistema distribuito che si appoggia su questa infrastruttura è il **World Wide Web**.

Una rete di computer è anche un **sistema di telecomunicazione**, e come tale le sue performances sono influenzate da quattro caratteristiche:

1. **Consegna**: il sistema deve consegnare le informazioni al corretto destinatario, ed esso deve essere il solo a ricevere il dato a cui il mittente voleva inviarlo;
2. **Precisione**: il sistema deve consegnare i dati senza che essi vengano alterati (nel caso ciò avvenisse deve essere in grado di ristabilirne la correttezza);
3. **Tempestività**: il sistema deve garantire la consegna delle informazioni in maniera celere; dati che arrivassero in ritardo potrebbero essere inutili;
4. **Jitter**: essa è la variazione di errore con cui i dati giungono al destinatario.

Le componenti che identificano un sistema di telecomunicazione sono:

- **Messaggio**: le informazioni che devono essere inviate; esso può assumere varie forme (testo, video, etc.);
- **Mittente**: il dispositivo che spedisce il messaggio;
- **Destinatario**: il dispositivo a cui è indirizzato e che riceve il messaggio;
- **Mezzo di trasmissione**: il mezzo fisico attraverso il quale il messaggio viene spedito;

- **Protocollo:** l'insieme di regole che governano la comunicazione tra i nodi che si scambiano il messaggio.

Il flusso di informazioni tra i nodi, ossia tra i dispositivi che comunicano, può essere:

- **Simplex:** in tale tipo di comunicazione il passaggio di informazioni è possibile solo in una direzione (ossia un dispositivo sarà sempre mittente e l'altro sempre e solo destinatario); la televisione è un esempio di trasmissione in modalità simplex in quanto la TV trasmittente invia segnali al nostro apparecchio ma non riceve mai un ritorno da esso. I vantaggi di questo tipo di trasmissione sono che non ci sono problemi di traffico e la mittente può utilizzare tutto il mezzo trasmittivo, mentre lo svantaggio, come facilmente si intuisce, è il fatto che il nodo trasmittente non riceve né feedback né dati dall'altro nodo;
- **Half-Duplex:** con questo tipo di comunicazione un dispositivo può assumere sia il ruolo di mittente che quello di destinatario, ma non può farlo in maniera contemporanea; il classico esempio di tale comunicazione è quello dei walkie-talkie, in cui chi finisce di parlare "passa la palla" a chi fino ad allora ha ascoltato ma si accinge ora a rispondere;
- **Full-Duplex:** è la modalità in cui un dispositivo può fungere contemporaneamente sia da mittente che da destinatario. Il collegamento telefonico è un sistema che usa questo tipo di connessione. I vantaggi sono che si può comunicare contemporaneamente senza attesa che l'altro nodo finisca. Lo svantaggio è che non tutta l'ampiezza di banda viene utilizzata proprio per la possibile contemporaneità d'uso del canale.

Lo scopo primario di una rete di calcolatori è quello di facilitare la comunicazione attraverso lo scambio d'informazione. Immaginiamo una rete nella sua forma più semplice: ad esempio un'azienda che ha all'interno del proprio sistema informativo molti dati conservati in uno o più database e deve dare accesso a essi ai suoi dipendenti e collaboratori, che hanno bisogno di accedervi sia da locale che remotamente: la rete consente a un collaboratore di collegarsi come utente remoto o permette la connessione con un'altra rete per scambiare informazioni. Le reti quindi consentono di condividere file e risorse; la condivisione di risorse, soprattutto hardware, permette di abbattere i costi (ad esempio la condivisione di stampanti o sistemi di archiviazione).

Il principale e più conosciuto (ma non il solo) paradigma di comunicazione all'interno delle reti è quello chiamato **client-server**. Il processo client (badate bene, si parla di processo e non di macchina) è tipicamente dedicato a interagire con l'utente final; esso svolge un ruolo attivo, in quanto genera autonomamente richieste di servizi. Invece, il processo server è reattivo: esso svolge una computazione solo a seguito di una richiesta da parte di un qualunque client.

In questo modello i dati sono memorizzati in computer ad alte prestazioni sul quale gira il processo **server**. Al contrario gli impiegati devono poter accedere ai dati e hanno sulla loro scrivania dei terminali su cui gira il processo **client**.

Le macchine su cui girano client e server sono collegate da una **rete**. Questa configurazione è chiamata **modello client-server**.

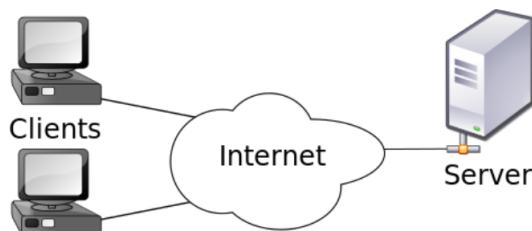


Figura 1.1: Modello client-server

La comunicazione è rappresentata da un processo client che manda un messaggio attraverso la rete al processo server e resta in attesa di un messaggio di risposta. Quando il processo server riceve la richiesta esegue il lavoro o recupera i dati desiderati e manda indietro una risposta.

- **Vantaggi**

1. La gestione dei dati informativi (ad esempio il backup) è facile ed economico in quanto non è necessario gestirlo sui nodi di rete che fungono da clients;

2. Le prestazioni sono generalmente migliori perché il tempo di risposta è fortemente influenzato dalla macchina che agisce come server in quanto esso è in linea di massima un computer più potente degli altri computer che partecipano all'architettura;
3. La sicurezza è più facile da gestire in quanto l'accesso non autorizzato viene negato dal computer che funge da server e tutti i dati passano attraverso esso;
4. La scalabilità non è un grosso problema in questa architettura, il limite è dato solo dalle risorse che il server ha a disposizione.

- **Svantaggi**

1. In caso di guasto del server, l'intera rete è inattiva;
2. Il costo di manutenzione del server è elevato ed è indispensabile perché il server è la componente principale di questa architettura; anche in caso di aggiornamento delle risorse per gestire un numero di richieste clients più grosso di quello preventivato o di abnorme crescita dei dati da conservare, i costi aumentano.

Il paradigma Client-Server non è l'unico a essere presente nelle reti. In alcune di esse non sono presenti ruoli "fissi" su chi sia il client e chi il server.

Queste reti prevedono un insieme di dispositivi equivalenti (**peer** per l'appunto) in grado sia di avviare che di completare la comunicazione. Benché il **P2P** sia tornato in auge con reti per la condivisione di file (il primo tra tutti fu Napster nel 2000), esso era l'architettura pensata inizialmente dalla rete **ARPANET**; l'idea originale era infatti quella di condividere cicli di CPU e quindi potenza computazionale all'interno della rete universitaria degli States (anni '60).

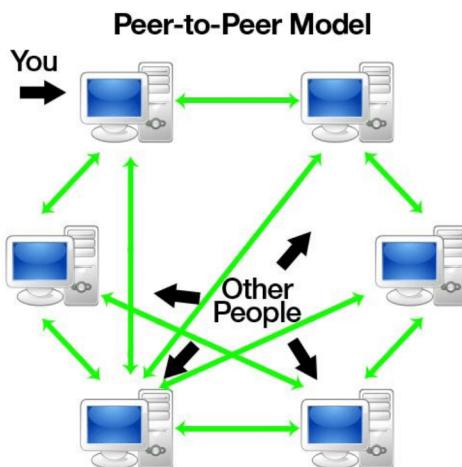


Figura 1.2: Modello peer to peer

L'idea del P2P è quella di non localizzare in un solo punto il carico di lavoro ma distribuirlo tra i partecipanti. Per quanto riguarda un'architettura di tipo P2P puro, non essendoci un server centrale in grado di far connettere tra di loro i diversi nodi, viene solitamente integrata una **Overlay network** virtuale (ovvero una "rete sovrapposta" virtuale), con la quale i nodi formano una sottorete rispetto alla rete fisica principale. L'overlay è utilizzata principalmente per indicizzare e mappare i nodi, in modo da conoscere la topologia della rete.

- **Vantaggi**

1. Meno costoso in quanto non esiste un server centrale che deve eseguire il backup;
2. In caso di guasto di un nodo, tutti gli altri computer della rete non sono interessati e continueranno a funzionare come prima del guasto;
3. L'installazione dell'architettura peer-to-peer è abbastanza semplice poiché ogni computer si gestisce da solo.

- **Svantaggi**

1. La scalabilità è un problema in un'architettura peer-to-peer in quanto risulta difficile collegare ogni computer a ogni altro nodo in una rete molto grande;
2. Ciascun computer deve gestirsi da solo sia in ambito sicurezza sia nella gestione dei dati.

Una rete di calcolatori è un potentissimo **mezzo di comunicazione**. Alcuni esempi di utilizzo sono i seguenti:

- Accesso a informazioni remote, presenti su dispositivi lontani da quello che ne sta richiedendo l'uso;
- Comunicazione tra persone, anch'esse lontane tra loro, ad esempio attraverso modalità di tipo asincrono (**email**) o sincrono (**IM**) o addirittura di tipo multimediale (**VoIP** o videoconferenze);
- Intrattenimento interattivo: giochi multiplayer, video On Demand, etc;
- Commercio elettronico: le reti di calcolatori hanno aperto nuovi tipi di mercato e nuove modalità di fare commercio;
- Formazione online: l'uso delle reti e di contenuti multimediali ha migliorato la qualità dell'apprendimento, attraverso l'accesso a risorse e servizi non disponibili in locale e la collaborazione di comunità create ad hoc;
- **Smart working** e telelavoro;
- **Grids e Cloud Computing**

1.1.2 Grids, Cloud e Virtualizzazione

Le reti sono in continua evoluzione, sia dal punto di vista fisico/tecnologico che da quello logico/commerciale. Negli ultimi anni questo ha portato allo sviluppo di nuovi paradigmi, come il Grid-computing, il cloud e la virtualizzazione.

I sistemi grid sono soprattutto una infrastruttura di calcolo distribuito; tali sistemi permettono la condivisione coordinata di risorse all'interno di un'organizzazione virtuale.

Il cloud è un'idea di marketing che prevede di considerare quelli che finora erano considerati beni (programmi, hardware, etc.) come servizi: essi vengono forniti attraverso internet (da qui il nome di cloud) da grossi provider.

La virtualizzazione, benché non sia un concetto tipicamente delle reti di calcolatori, è un presupposto imprescindibile per fornire servizi di tipo cloud.

Essa è la capacità di astrarre hardware e software su una sola macchina.

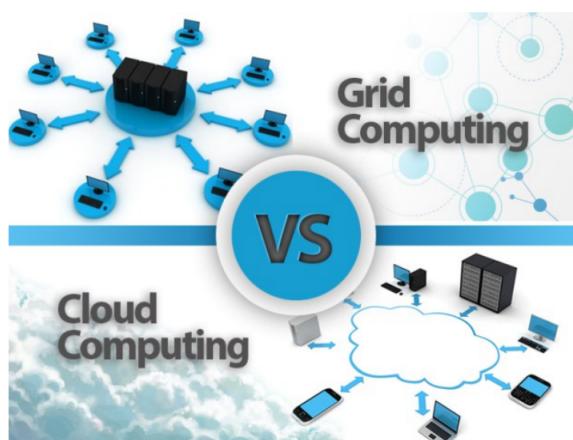


Figura 1.3: Grids e Cloud

Il concetto di Grid è nato dalla necessità di dover coordinare delle risorse condivise atte a risolvere in maniera dinamica problemi complessi con grosse quantità di dati. La condivisione non è limitata solo ai dati e al software ma anche, in generale, a tutto l'hardware, come cpu/gpu, necessario alla risoluzione del problema.

Essendo i problemi trattati dal grid-computing di tipo data-intensive, le applicazioni usate hanno bisogno di accedere a grandi quantità di dati geograficamente distribuiti, e il compito del grid è proprio quello di fare collaborare tali applicazioni nel modo migliore possibile attraverso protocolli standard e aperti, e interfacce in grado di condividere una qualità del servizio non banale.

Una rete di grid computing consiste principalmente in 3 nodi:

1. Un **nodo di controllo**, ossia un server che amministra l'intera rete e tiene conto delle risorse nel pool di rete;
2. Un **provider**, un computer che contribuisce con le proprie risorse al pool di risorse in rete;
3. Un **utente**, il nodo che utilizza le risorse.

Quando un computer effettua una richiesta di risorse al nodo di controllo, esso fornisce all'utente l'accesso alle risorse disponibili in rete; quando un nodo non è in uso, dovrebbe idealmente contribuire con le proprie risorse alla rete.

Quindi un computer può assumere sia il ruolo di utente che di provider, in base alle sue esigenze.

Per controllare la rete e le sue risorse viene utilizzato un protocollo/software di rete generalmente noto come **middleware**. Il vero responsabile della rete è il middleware, mentre i nodi di controllo sono semplicemente i suoi esecutori.

Il cloud computing, o semplicemente cloud, sta cambiando il modo dell'IT di offrire servizi e la maniera di un utente di usufruire delle risorse di elaborazione dal posto di lavoro, da casa e in movimento. Il cloud consente all'IT di rispondere alle opportunità di business con consegne on-demand che, a lungo termine, potrebbero essere economicamente efficienti e agili.

Un concetto chiave nel mondo cloud è quelli di servizio; nel contesto IT, tale termine viene usato frequentemente per descrivere una forma di consegna o disponibilità di una risorsa, quindi possiamo vedere un servizio come qualcosa di "consegnato su richiesta".

Per i professionisti IT, il cloud può significare: applicazioni, griglie di calcolo ad alta velocità, virtualizzazione, configurazione automatica e distribuzione di macchine, elaborazione on-demand e remota oppure qualsiasi combinazione di ciascuna di queste cose.

Un modo per descrivere il cloud computing è quello di basarsi sui modelli di distribuzione dei servizi, ne esistono tre: **Software-as-a-Service(SaaS)**, **Platform-as-a-Service(PaaS)** e **Infrastructure-as-a-Service(IaaS)**.

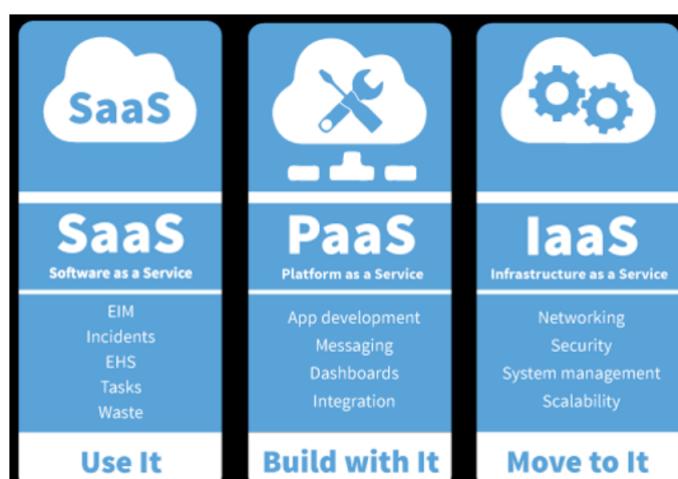


Figura 1.4: Servizi Cloud

SaaS è un modello in cui l'applicazione è disponibile su richiesta, è la forma più comune di cloud a oggi consegnata.

PaaS è una piattaforma disponibile su richiesta per lo sviluppo, il testing, la distribuzione e la manutenzione continua delle applicazioni, utile se non si vuole affrontare il costo iniziale dell'infrastruttura e il software sottostante.

IaaS è un ambiente IT che prevede un'infrastruttura completa per il sottoscrittore. Questa infrastruttura viene fornita, la maggior parte delle volte, con macchine virtuali in cui l'utente mantiene l'OS e le applicazioni installate; tutto quanto è al di sotto viene gestito dal fornitore del servizio.

La gestione dei servizi delle tre tipologie appena viste, dipende per l'appunto dal modello scelto:

- Con il SaaS l'azienda fruitrice del servizio non controlla l'infrastruttura che supporta il software che ha scelto; il provider si incarica dell'intera gestione a livello di rete, dei server, degli storage e dei sistemi operativi. L'azienda cliente può solo intervenire sulle funzionalità del software stabilendo criteri di gestione delle identità e delle prioritizzazioni degli accessi tramite un insieme di configurazione dedicate.
- Anche nel paradigma PaaS l'azienda cliente non si deve preoccupare di dover gestire o controllare l'infrastruttura cloud a livello di rete, server, sistemi operativi e storage, ma questa volta ha il pieno controllo sulle applicazioni implementate e le relative impostazioni di configurazione.
- Nel paradigma IaaS le aziende esternalizzano le risorse, gestite a livello di infrastruttura da un fornitore. Il cliente può gestire il proprio networking, lo storage e tutte le sue risorse di calcolo in modalità distribuita potendo visionare il tutto con una dashboard centralizzata senza doversi preoccupare dei dettagli di monitoraggio, di sicurezza e di aggiornamento legati alle macchine che abilitano questi servizi online.
- Logicamente la scelta del paradigma On Premise lascerebbe l'inconvenienza non solo della gestione, ma anche dell'acquisto, l'installazione e il tuning sia della parte software che di quella hardware dell'intera struttura.

Al momento dell'installazione esistono tre modelli di base. La **Public Cloud** è quella messa a disposizione attraverso internet per il pubblico o utenti mirati ed è proprietà di un'organizzazione che offre tale tipo di servizio. Questo modello è in primo luogo indicato per quelle aziende o utenti che lavorano con dati che non hanno problemi stringenti di privacy. Possiamo riassumere i suoi vantaggi come: flessibile, affidabile, altamente scalabile, costi bassi, non localizzati. Gli svantaggi si focalizzano sulla scarsa sicurezza e la bassa personalizzazione.

Il **Private Cloud** è disponibile invece solo per un'organizzazione ed è gestita da essa (o da terze parti) On Premise. Tale modello dovrebbe essere adottato da organizzazioni aziendali che hanno requisiti di una gestione dinamica dei dati e on demand, che trattano informazioni critiche o protette. Come si può facilmente capire i vantaggi di questo tipo di deployment sono un maggiore controllo sui propri dati e sull'accesso a essi e una protezione maggiore delle informazioni riservate e della privacy. Gli svantaggi vanno dal costo elevato (tutto on premise) alla poca scalabilità (le risorse sono tarate sulle specifiche aderenti all'azienda).

La **Community Cloud** è un ambiente cloud pubblico nel quale sono presenti però elementi tipici del cloud privato, in esso, diverse organizzazioni con background simili, condividono l'infrastruttura e le relative risorse. La selezione del giusto tipo di cloud hosting è essenziale in questo caso. Tale modello è consigliato per enti di ricerca e gruppi di aziende consociate. I vantaggi in questo caso sono una riduzione dei costi, una maggiore sicurezza e privacy dei dati rispetto al pubblico e una facilità nella condivisione dei dati e della collaborazione. Gli svantaggi prevedono una condivisione delle risorse da allocare e non è molto diffuso.

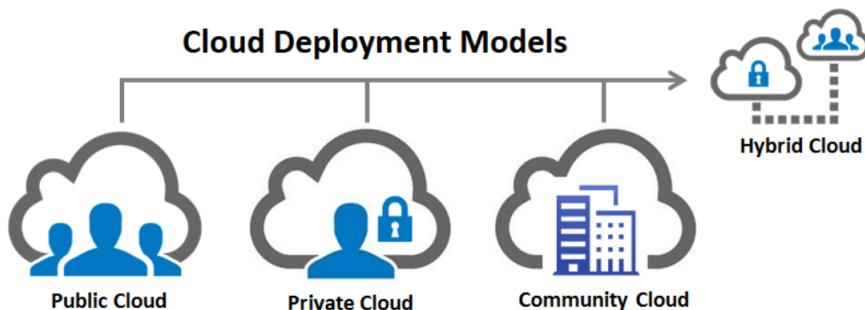


Figura 1.5: Modelli di deployment del Cloud

Il **Cloud Ibrido** è un altro modello che deriva da una combinazione di due o più modelli visti in precedenza integrati in un'unica architettura. Una tipica implementazione potrebbe vedere le attività non critiche, come carichi di sviluppo e testing, essere eseguite su cloud pubblico, mentre le attività critiche, come la gestione di dati sensibili, eseguite utilizzando un cloud privato. I vantaggi di entrambi i modelli di deployment, o anche di un modello di tipo community, sono possibili in un hosting cloud ibrido.

Questo modello di deployment non solo salvaguarda e controlla le risorse strategicamente importanti, ma lo fa nel modo più efficiente in termini di costi e risorse per ogni caso specifico. Inoltre, questo approccio facilita la portabilità dei dati e delle applicazioni.

Un sistema noto come macchina virtuale (VM) è un contenitore software strettamente isolato, con un proprio sistema operativo e le applicazioni che su esso girano. Ogni macchina virtuale autonoma è completamente indipendente. L'inserimento di più VM su un singolo computer consente l'esecuzione di più sistemi operativi e applicazioni su un solo sistema fisico, o "host". Un sottile strato di software disaccoppia le macchine virtuali dall'host e alloca dinamicamente le risorse di elaborazione a ciascuna macchina virtuale secondo necessità.

Alcuni concetti chiave della virtualizzazione sono:

- **Partizionamento**
 - Esegue più sistemi operativi su una macchina fisica
 - Divide le risorse di sistema tra macchine virtuali
- **Isolamento**
 - Fornisce isolamento dei guasti e della sicurezza a livello di hardware
 - Preserva le prestazioni con controlli avanzati delle risorse
- **Incapsulamento**
 - Salva l'intero stato di una macchina virtuale su file
 - Sposta e copia le macchine virtuali con la stessa facilità con cui si può spostare e copiare file
- **Indipendenza Hardware**
 - Effettua il provisioning o migra qualsiasi macchina virtuale su qualsiasi macchina fisica

Alcuni modelli di virtualizzazione possono essere il Desktop, un Server (inteso come macchina hardware di discrete risorse) e il Network.

L'implementazione della **virtualizzazione dei desktop** consente alle organizzazioni IT di rispondere in maniera più rapida alle mutevoli esigenze del luogo di lavoro e alle opportunità emergenti. I desktop e le applicazioni virtualizzati possono anche essere inviati e installati rapidamente e facilmente alle filiali, ai dipendenti in outsourcing e offshore e ai lavoratori mobili che utilizzano devices mobili.

La **virtualizzazione dei server** consente l'esecuzione di più sistemi operativi su un singolo server fisico: tali macchine virtuali risultano altamente efficienti in quanto hanno costi operativi ridotti, una distribuzione del carico di lavoro più rapido e deployment delle applicazioni facilitato.

Riproducendo completamente una rete fisica, la **virtualizzazione della rete** consente alle applicazioni di funzionare su una rete virtuale come se fossero in esecuzione su una rete fisica, ma con i vantaggi operativi della virtualizzazione; tale virtualizzazione presenta dispositivi e servizi di rete logici (porte logiche, switch, router, firewall, bilanciatori del carico, VPN e altro) ai carichi di lavoro connessi.

1.2 Modelli di Riferimento OSI e TCP/IP

Una rete, per funzionare, ha bisogno sia dell'hardware che del software: l'hardware permette di spedire un flusso di bit dal mittente al destinatario.

1.2.1 Software di rete

Per diminuire la complessità, le reti sono generalmente organizzate come pila di *strati* (layer) o *livelli*, costruiti uno sull'altro. Lo scopo di ogni strato è quello di offrire determinati servizi agli strati di livello superiore, nascondendo i dettagli sull'implementazione dei servizi (*information hiding*).

Lo strato *n* di un nodo è in comunicazione con lo stesso strato *n* di un altro nodo e la comunicazione è regolamentata da **protocolli**. Fondamentalmente un protocollo è un accordo tra le parti che comunicano, sul modo in cui deve procedere la comunicazione (formato, ordine in cui i messaggi vengono inviati e ricevuti, azioni da prendere, ...).

Tra ogni strato contiguo esiste inoltre un'**interfaccia**, che definisce le operazioni elementari e i servizi che lo strato inferiore rende disponibili a quello soprastante.

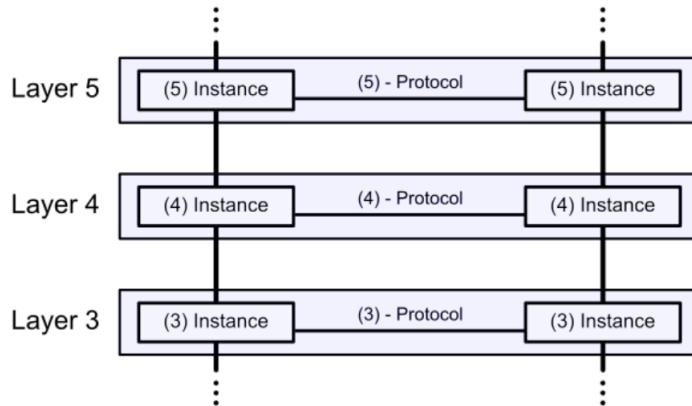


Figura 1.6: Organizzazione a strati

L'insieme di strati e di protocolli si chiama **architettura di rete** (non ne fanno parte le interfacce perché nascoste dal nodo). Un elenco di protocolli usati da uno specifico nodo si chiama **pila di protocolli**.

Gli strati possono offrire due tipi di servizio a quelli sovrastanti: orientati alla connessione (**connection oriented**) oppure senza connessione (**connectionless**). Un servizio **orientato alla connessione** assomiglia al sistema telefonico: il trasmettitore invia le informazioni (bit) da un'estremità (di un canale di comunicazione) e il ricevitore li prende dall'altra. Nella maggior parte dei casi il ricevitore e la subnet eseguono una **negoziazione** dei parametri da usare. Al contrario, un servizio **senza connessione** si comporta come la posta: ogni messaggio trasporta l'indirizzo completo del destinatario ed è instradato attraverso il sistema postale in modo indipendente dagli altri.

Il software permette lo scambio di informazioni da una qualsiasi applicazione su un qualsiasi nodo della rete, ad un'altra applicazione che gira su un altro nodo.

è utile pensare di affrontare il problema della trasmissione suddividendolo in sottoproblemi più semplici, ognuno dei quali individuabile in un dato livello: la mera trasmissione dei segnali informativi avviene nello strato più basso. In figura abbiamo un mittente e un destinatario e tra essi un mezzo trasmittivo. I livelli di destinatario e mittente che si trovano nella stessa posizione dello stack sono detti **peer**. I peer comunicano tra loro attraverso un protocollo. Un protocollo è un insieme organizzato e coordinato di procedure che caratterizzano operazioni tecniche di una certa complessità.

L'architettura e la costruzione di rete di base sono un buon punto di partenza per cercare di capire come funzionano i sistemi di comunicazione. Le architetture sono in genere basate su un modello che mostra come i protocolli e le funzioni si integrano. Storicamente, sono stati utilizzati molti modelli per questo scopo alcuni dei quali sono Systems Network Architecture (SNA-IBM), AppleTalk, Novell Netware (IPX/SPX) e il modello **Open System Interconnection (OSI)**. La maggior parte di questi ormai sono solo un ricordo a causa della popolarità del modello **TCP/IP**. TCP/IP rappresenta una suite di protocolli utilizzati su quasi tutti i moderni sistemi di comunicazione. Come già il nome suggerisce, questa è la vera architettura di Internet.

Nei modelli a strati, l'informazione che un mittente vuole inviare a un destinatario passa da un livello superiore a quello inferiore, dove avvengono delle operazioni che consentiranno a questi dati, alla fine, di poter essere trasmessi su un canale fisico. Tale meccanismo è un importante principio che sta alla base delle architetture di rete che utilizzano il modello a strati e viene chiamato **incapsulamento**.

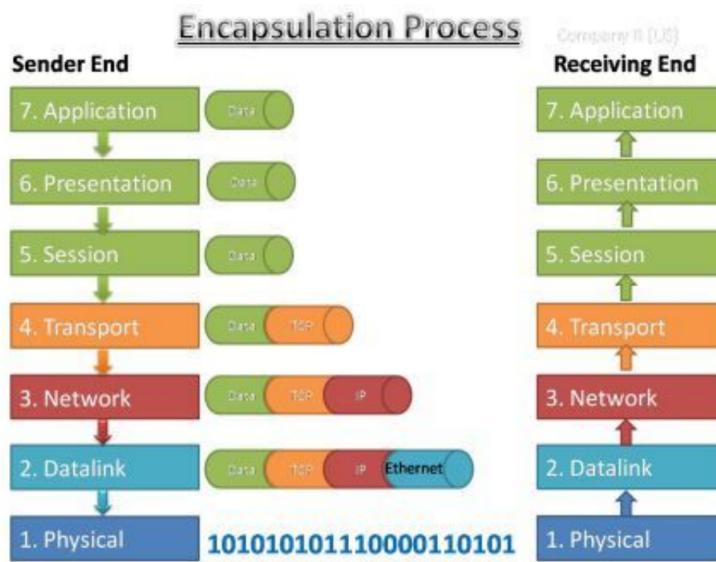


Figura 1.7: Incapsulamento

Questo processo si verifica nel momento in cui un nodo vuole inviare dati a un altro, e l'informazione (PDU) passa da un livello superiore a quello inferiore, e ogni livello aggiunge a quest'ultimo un insieme di dati chiamato **header** (e a volte anche una **tail**).

Al nodo ricevente il processo viene invertito, con le intestazioni eliminate a ogni livello man mano che l'informazione ricevuta procede verso il livello applicazione.

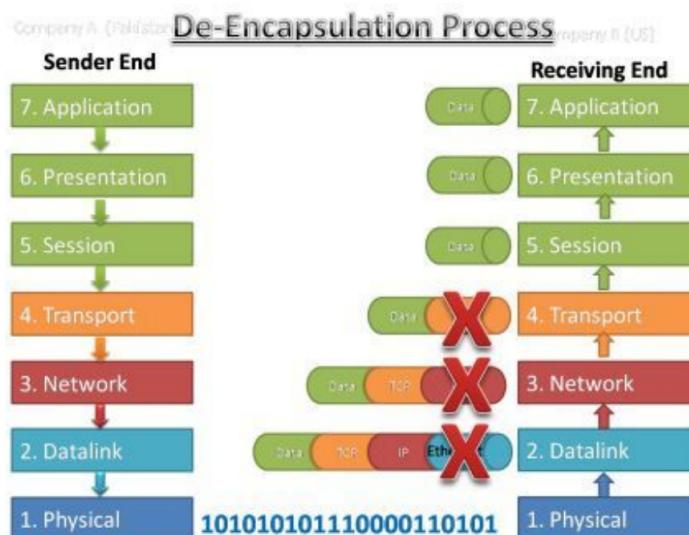


Figura 1.8: Decapsulamento

1.2.2 Il modello di Riferimento OSI

Il modello OSI (*Open System Interconnection*) è stato sviluppato da ISO - "International Organization of Standardization", nell'anno 1984. È un'architettura a 7 livelli, ciascuno dei quali con funzionalità specifiche da eseguire. Tutti questi 7 livelli lavorano in modo collaborativo per trasmettere i dati da un nodo all'altro della rete.

I principi che sono stati applicati per arrivare ai sette strati si possono brevemente riassumere come segue:

1. Si deve creare uno strato quando è richiesta un'astrazione diversa;
2. Ogni strato deve svolgere una funzione ben definita;

3. La funzione di ogni strato va scelta con uno sguardo rivolto alla definizione di protocolli internazionali;
4. I confini degli strati vanno scelti per minimizzare il flusso d'informazioni attraverso le interfacce;
5. Il numero di strati deve bastare a evitare la necessità di radunare funzioni distinte nello stesso strato, ma essere abbastanza piccolo da rendere l'architettura instabile.

Lo **strato fisico** si occupa della trasmissione di bit grezzi sul canale di comunicazione ed è il livello più basso del modello di riferimento OSI. È responsabile della ricezione del segnale e lo convertirà in 0 e 1 per passarli al livello Data Link, che li rimetterà insieme formando il frame.

Le funzioni dello strato fisico sono:

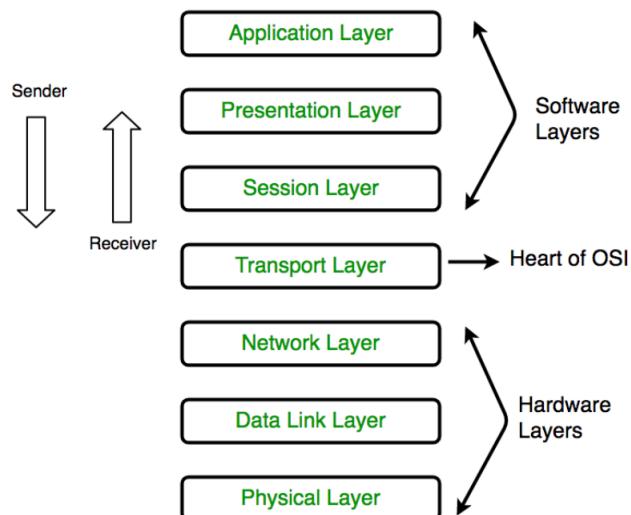


Figura 1.9: Modello ISO/OSI

1. **Sincronizzazione bit:** il livello fisico fornisce la sincronizzazione dei bit fornendo un clock. Questo clock controlla sia il mittente che il destinatario fornendo così la sincronizzazione a livello di bit;
2. **Controllo della velocità in bit:** il livello fisico definisce anche la velocità di trasmissione, ovvero il numero di bit inviati al secondo;
3. **Topologie fisiche:** il livello fisico specifica il modo in cui i diversi dispositivi/nodi sono disposti in una rete, ad esempio topologie a bus, a stella o mesh;
4. **Modalità di trasmissione:** il livello fisico definisce anche il modo in cui i dati fluiscono tra i due dispositivi collegati. Le varie modalità di trasmissione possibili sono: Simplex, half-duplex e full-duplex.

Il **livello di collegamento** (Data Link) è responsabile della consegna del messaggio da nodo a nodo. Il lavoro che esso deve svolgere, potrebbe riassumersi nella funzione di assicurarsi che il trasferimento delle informazioni avvenga senza errori. Quando un pacchetto viene passato al livello Data Link da quello superiore, questa ha la responsabilità di trasmetterlo al nodo destinatario utilizzando il suo indirizzo MAC.

Il Data Link Layer è diviso in due sottolivelli:

- Controllo del collegamento logico (LLC)
- Controllo dell'accesso al mezzo (MAC)

I servizi forniti dal livello di collegamento dati sono:

1. **Framing:** è un servizio che permette al mittente di raggruppare e trasmettere una serie di bit in modo che il destinatario sappia come leggerli;
2. **Indirizzamento fisico:** dopo aver creato i frame, il livello di collegamento dati aggiunge gli indirizzi fisici (indirizzo MAC) del mittente e/o del destinatario nell'intestazione di ciascun frame;

3. **Controllo degli errori:** il livello di collegamento dati fornisce il meccanismo di controllo degli errori in cui rileva e ritrasmette i frame danneggiati o persi;
4. **Controllo del flusso:** la velocità dei dati deve essere costante su entrambi i lati, altrimenti i dati potrebbero essere danneggiati, quindi il controllo del flusso coordina la quantità di dati che possono essere inviati prima ricevere il riconoscimento;
5. **Controllo degli accessi:** quando un singolo canale di comunicazione è condiviso da più dispositivi, il sottolivello MAC del livello di collegamento dati aiuta a determinare quale dispositivo ha il controllo sul canale in un determinato momento.

Lo **strato network** controlla la trasmissione delle informazioni da un nodo presente in una rete a un altro nodo situato in una rete differente: questo servizio prende il nome di **intradamento**, ovvero la scelta del percorso più consone che il pacchetto deve attraversare per arrivare al nodo target. Gli indirizzi IP del mittente e del destinatario sono inseriti nell'intestazione dal livello di rete.

1. **Routing:** i protocolli del livello di rete determinano quale percorso, dalla sorgente al destinatario, il pacchetto dati deve seguire. Questo servizio del livello di rete è noto come routing o intradamento;
2. **Indirizzamento logico:** al fine di identificare in modo univoco ciascun dispositivo sulla rete, questo livello definisce uno schema di indirizzamento: l'indirizzo IP. Tale indirizzo distingue ogni dispositivo in modo univoco.

Il **livello trasporto** fornisce servizi al livello di applicazione e usufruisce di quelli del livello rete. Nel livello trasporto, l'insieme dei dati è chiamato **segmento**. È responsabile della consegna end-to-end del messaggio completo.

Il livello di trasporto fornisce anche la 'ricevuta' dell'avvenuta trasmissione dei dati e si occupa della ritrasmissione se viene rilevato un errore.

Lato Nodo Mittente

Il livello di trasporto del nodo mittente riceve i dati formattati dai livelli superiori, esegue la segmentazione e provvede al controllo del flusso e degli errori per garantire la corretta trasmissione dei dati. Nell'header del segmento si aggiunge anche il numero di porta di origine e di destinazione; per ultimo inoltra i dati segmentati al livello di rete.

Attenzione: il mittente deve conoscere il numero di porta associato all'applicazione del destinatario.

Lato Nodo Ricevente

All'arrivo del segmento, il livello trasporto legge il numero di porta dalla sua intestazione e inoltra i dati che ha ricevuto alla rispettiva applicazione. Si occupa anche della gestione della sequenza e quindi dell'assemblaggio dei dati segmentati.

Le funzioni che questo livello esegue sono:

- **Segmentazione e riassemblaggio:** questo livello accetta il messaggio dal livello superiore (sessione) e lo suddivide in unità più piccole. A ogni segmento prodotto è associato un'intestazione. Il livello di trasporto nella stazione di destinazione riassembra il messaggio seguendo le indicazioni contenute nell'header.
- **Indirizzamento:** per consegnare il messaggio al processo corretto, l'intestazione del livello di trasporto include quello che viene chiamato indirizzo del punto di servizio o indirizzo di porta. Pertanto, specificando questo indirizzo, il livello di trasporto si assicura che il messaggio venga consegnato al processo corretto.
- **Controllo di flusso:** meccanismi per gestire la trasmissione delle informazioni, simili a quella che si trova nel livello collegamento ma di tipo end to end.
- **Multiplexing:** i protocolli di livello trasporto per eseguire il multiplexing e demultiplexing includono due campi nell'intestazione del segmento: quelli del numero di porta di origine e di porta di destinazione.
 - **Multiplazione** – è chiamato multiplexing la raccolta di dati da più processi lato mittente, l'inviluppo di tali dati nel segmento e l'invio di tale unico blocco al destinatario.
 - **Demultiplazione** – il demultiplexing è la consegna dei dati ricevuti nel segmento ai processi corretti del livello applicazione

Inoltre fornisce i seguenti servizi:

- Servizio orientato alla connessione: è un processo in tre fasi che include:
 - Instanziazione della connessione.
 - Trasferimento dati.
 - Terminazione/disconnessione

In questo tipo di trasmissione, il dispositivo ricevente invia una conferma di ricezione alla fonte dopo che un pacchetto o un gruppo di pacchetti è stato ricevuto. Questo tipo di trasmissione è affidabile e sicuro.

- Servizio senza connessione: è un processo a una fase e include il trasferimento dei dati. In questo tipo di trasmissione, il destinatario non conferma la ricezione di un pacchetto. Questo approccio consente una comunicazione molto più rapida tra i dispositivi ma è meno affidabile rispetto al servizio orientato alla connessione.

Il **livello sessione** è responsabile della creazione della connessione, del mantenimento delle sessioni, dell'autenticazione e garantisce anche la sicurezza. Le funzioni del livello di sessione sono:

- Gestione della sessione: il livello consente di iniziare, gestire e terminare la sessione tra due processi.
- Sincronizzazione: questo livello consente a un processo di aggiungere checkpoint in un flusso dati, in maniera tale da suddividere la sessione in parti temporali più piccole che, in caso di errori o malfunzionamenti, permettano di ripristinare la sessione esistente dall' ultimo checkpoint, senza ricominciarla da capo.
- Controllo del dialogo: il livello di sessione consente a due nodi di avviare la comunicazione tra loro in modalità half o full-duplex.

Il **livello presentazione** si occupa di manipolare i dati provenienti dal livello applicazione per rappresentarli nel giusto formato richiesto per una corretta trasmissione in rete che può avvenire anche tra nodi con sistemi operativi diversi. Le funzioni del livello di presentazione sono:

- **Traduzione:** ad esempio, da ASCII a EBCDIC, GIF etc.
- **Crittografia/Decrittografia:** questo è il livello che si occupa di eventuale cifratura dei dati.
- **Compressione:** riduce il numero di bit che devono essere trasmessi sulla rete.

Il livello più alto dell stack del modello OSI, è il livello applicazione; in esso vengono implementate le applicazioni di rete. Questo livello funge anche da finestra per l'accesso dei servizi applicativi alla rete e per la visualizzazione delle informazioni ricevute. Alcune di queste applicazioni sono l' email, il dns, i software di IM, etc.

1.2.3 Il modello di Riferimento TCP/IP

Il modello OSI è un modello di riferimento molto chiaro ma complesso e non è pedissequamente implementato in Internet a causa della sua tardiva realizzazione. Il modello in uso rimane quello **TCP/IP**, ovvero quello che gli sviluppatori hanno 'formato' man mano che codificavano protocolli per permettere la comunicazione tra dispositivi in Internet (non per nulla è anche noto come suite di protocolli Internet)

Il Defense Advanced Research Projects Office (DARPA), il dipartimento investigativo del Dipartimento della Difesa degli Stati Uniti, ha realizzato i TCP/IP negli anni '70 per il suo utilizzo in ARPANET, una WAN che ha preceduto il web. Questo modello è stato inizialmente progettato per il framework Unix ed è stato integrato poi in tutti gli altri sistemi.

TCP/IP sta per Transmission Control Protocol/Internet Protocol ed è un insieme di convenzioni e metodi organizzato in quattro livelli (contro i sette del modello OSI). Le caratteristiche principali su cui è basato il protocollo TCP/IP:

- Architettura flessibile.
- Facilità nell'aggiungere nuovi nodi alla rete.

- Fornire un servizio orientato alla connessione.
- Garantire affidabilità e riordino dei dati arrivati fuori sequenza.

I quattro livelli sono:

- **Livello Interfaccia Network**
- **Livello Internet**
- **Livello Trasporto**
- **Livello Applicazione**

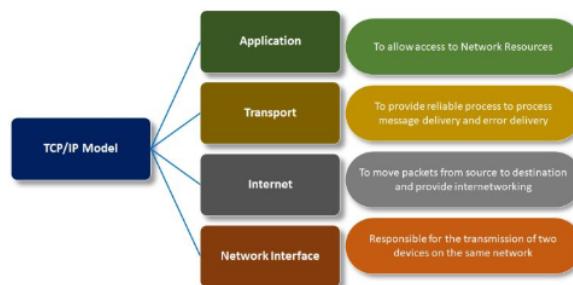


Figura 1.10: Modello TCP/IP

Il **livello interfaccia Network** corrisponde alla combinazione dei primi due livelli del modello ISO/OSI (Data Link e Fisico). Si occupa dell'indirizzamento hardware e gestisce i protocolli che consentono la trasmissione fisica dei dati. Una nota su ARP (che incontreremo più avanti): è un protocollo di livello Internet, ma è incapsulato dai protocolli di livello Interfaccia alla Rete.

Il **livello Internet** è il duale del livello Rete di OSI fornisce quindi le funzioni proprie anche di quel livello. Qui sono definiti i protocolli responsabili della trasmissione logica dei dati sull'intera rete. I principali di essi sono:

- **IP** – sta per Internet Protocol ed è responsabile della consegna dei pacchetti dall'host di origine all'host di destinazione osservando gli indirizzi IP nelle intestazioni dei pacchetti. IP ha 2 versioni: IPv4 e IPv6. IPv4 è quello attualmente utilizzato dalla maggior parte dei siti Web. Ma IPv6 sta crescendo poiché il numero di indirizzi IPv4 è limitato rispetto al numero di utenti.
- **ICMP** – sta per Internet Control Message Protocol. È incapsulato all'interno di datagrammi IP ed è responsabile di fornire agli host informazioni sui problemi di rete.
- **ARP** – sta per Address Resolution Protocol. Il suo compito è trovare l'indirizzo hardware di un host da un indirizzo IP noto.

Il **livello trasporto**, omonimo del quarto livello del modello ISO/OSI, è responsabile della comunicazione 'end to end' e della consegna senza errori delle informazioni. tra le funzioni che esso espleta ritroviamo quindi:

- La segmentazione del flusso dati proveniente dal livello applicazione.
- Il recapito del messaggio processo corretto sulla macchina di destinazione.
- Inoltre, assicura che l'intero messaggio arrivi senza errori, venga riassemblato nell'ordine corretto e gestisca il flusso dell'invio.

I più importanti protocolli che incontreremo in questo livello sono:

- **TCP** – è il protocollo più noto, usato per fornire una comunicazione affidabile e priva di errori tra i sistemi finali. è un protocollo molto efficace ma ha un costo importante in termini di performance temporali dovuto alle funzionalità che esso fornisce.

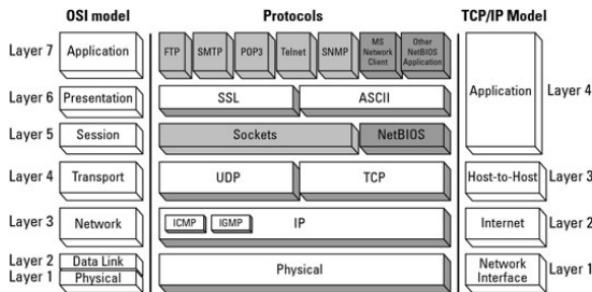


Figura 1.11: Modelli a confronto

- **UDP** – è un protocollo più agile e snello che non fornisce tali funzionalità. È il protocollo di riferimento se l'applicazione eseguita non richiede un trasporto affidabile ma veloce. A differenza del precedente protocollo, UDP non è orientato alla connessione.

Il **livello applicazione** nel modello TCP/IP riassume in esso le funzioni dei tre livelli più alti del modello OSI: Applicazione, Presentazione e Sessione. Si occupa della comunicazione da applicazione ad applicazione e controlla le specifiche dell'interfaccia utente. Alcuni dei protocolli presenti in questo livello sono: HTTP/S, FTP, TFTP, SSH, SMTP, SNMP, NTP, DNS etc.

1.2.4 Confronto tra i Modelli Presentati

Il modello TCP/IP e quello OSI sono i due modelli di rete più famosi e utilizzati nell'ambito delle odierni reti. Ci sono alcune somiglianze tra loro, ma anche differenze; partiamo dalle somiglianze.

1. La **struttura**: entrambi i modelli sono fatti a livelli, con uno stack di protocolli, ossia ogni livello incorpora in sè dei protocolli necessari a comunicare con i rispettivi livelli degli altri nodi.
2. Il **framework**: è un fondamento sia per TCP/IP che OSI; permette di creare e implementare in maniera chiara sia protocolli standard che dispositivi.
3. Entrambi i modelli prevedono che un produttore possa costruire dispositivi e componenti di rete che possano coesistere e funzionare con dispositivi e componenti realizzati da altri.

Le principali differenze invece invece potremmo elencarle così:

1. Da quanto finora incontrato, una delle principali differenze che salta in mente è che OSI è un modello concettuale quasi mai utilizzato nella pratica, mentre spesso incontriamo applicazioni che si rifanno al modello TCP/IP per stabilire una connessione e comunicare attraverso la rete.
2. TCP/IP è un modello nato dal paradigma client-server. OSI è un modello concettuale.
3. TCP/IP aiuta a stabilire una connessione tra diversi tipi di computer, mentre OSI aiuta a standardizzare router, switch, schede madri e altro hardware.
4. TCP/IP è un modello a quattro livelli contro i sette di OSI, in quanto ingloba i tre più alti del modello OSI in uno (Applicazione vs Sessione, Presentazione e Applicazione) e due livelli nella parte bassa dello stack (Network Access vs Fisico e Collegamento).
5. TCP/IP è uno standard orientato al protocollo, mentre OSI è un modello generico basato sulle funzionalità di ogni livello.
6. TCP/IP segue l'approccio orizzontale (ossia tra livelli pari di nodi diversi). D'altra parte, il modello OSI supporta l'approccio verticale.
7. TCP/IP segue un approccio dall'alto verso il basso, mentre il modello OSI segue un approccio dal basso verso l'alto.

1.3 Esempi di Reti e Standardizzazione delle Reti

Esistono tanti diversi modi per catalogare le reti: per tecnologia, per estensione, per mezzo di trasmissioni etc..

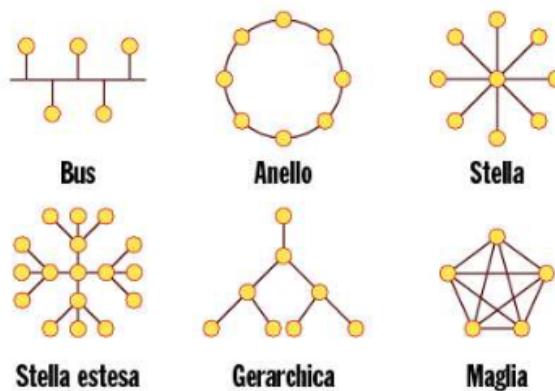


Figura 1.12: Topologia di rete

1.3.1 Categorizzazione per Topologia

La rappresentazione geometrica di come sono connessi i computer può essere un buon inizio per descrivere una rete, tale rappresentazione prende il nome di topologia. Nella topologia a **bus** c'è un cavo principale e tutti i dispositivi sono collegati a questo cavo tramite linee di derivazione. poiché tutti i dati vengono trasmessi tramite il cavo principale, esiste un limite di linee di discesa e alla distanza che può avere il cavo principale.

- **Vantaggi**

1. Installazione semplice, ogni cavo deve essere collegato con un cavo backbone.
2. Meno cavi necessari rispetto a topologie tipo Mesh e stella

- **Svantaggi**

1. Difficoltà nel rilevamento dei guasti.
2. Non è scalabile in quanto esiste un limite al numero di nodi che è possibile connettere con il cavo backbone

Nella topologia ad **anello** ogni dispositivo è connesso con i due dispositivi su entrambi i lati. Ci sono due collegamenti punto-punto dedicati che ciascun nodo ha con i dispositivi a lui adiacenti. Questa struttura forma un anello, quindi è nota come topologia ad anello. Se un dispositivo desidera inviare dati a un altro dispositivo, invia i dati in una direzione, ogni dispositivo nella topologia ad anello ha un ripetitore, se i dati ricevuti sono destinati a un altro dispositivo, il ripetitore inoltra questi dati fino a quando il dispositivo previsto non li riceve.

- **Vantaggi**

1. Installazione semplice.
2. La gestione è più semplice in quanto per aggiungere o rimuovere un dispositivo dalla topologia è necessario modificare solo due collegamenti.

- **Svantaggi**

1. Un errore di collegamento può far fallire l'intera rete in quanto viene a crearsi un buco sul collegamento principale e il segnale non sarà più in grado di essere inoltrato a causa di un errore.
2. Problemi di traffico dati, poiché tutti i dati circolano in un anello.

La topologia a **stella** prevede che ogni dispositivo della rete sia connesso a un dispositivo centrale chiamato hub (vedremo che questo dispositivo è stato sostituito da switch evoluti nelle moderni reti). Essa non consente la comunicazione diretta tra i dispositivi, un dispositivo deve comunicare con gli altri tramite il nodo centrale, ossia deve prima inviare i dati all'hub/switch e poi questo provvederà a trasmetterli al dispositivo designato.

- **Vantaggi**

1. Poco costoso perché ogni dispositivo necessita solo di una porta I/O e deve essere collegato al nodo centrale con un solo collegamento.
2. Di facile installazione
3. Non molti cavi necessari perché ogni dispositivo deve essere collegato solo al nodo centrale.
4. Se un collegamento fallisce il resto della rete continua a funzionare bene.
5. Facile rilevamento dei guasti perché il collegamento può essere facilmente identificato.

- **Svantaggi**

1. Se il nodo centrale si guasta, tutto si interrompe, nessuno dei dispositivi può funzionare senza hub/switch.
2. Il nodo centrale richiede più risorse e una manutenzione regolare in quanto da esso dipende tutta la rete.

Nella topologia mesh ogni dispositivo è connesso a ogni altro nodo della rete tramite un collegamento punto-punto dedicato. Quando diciamo 'collegamento dedicato' intendiamo dire che esso trasporta solo i dati per i due dispositivi che fanno da terminali. Se abbiamo n dispositivi nella rete, ognuno di essi dovrà essere connesso con i restanti $(n - 1)$ nodi. Il numero di collegamenti in una topologia mesh di n dispositivi sarà quindi

$$\frac{n(n - 1)}{2}$$

- **Vantaggi**

1. Nessun problema di traffico dati in quanto esiste un collegamento dedicato tra ciascuna coppia di dispositivi, il che significa che il collegamento è disponibile solo per quei due dispositivi.
2. La topologia mesh è affidabile e robusta poiché il guasto di un collegamento non influisce sugli altri collegamenti e sulla comunicazione tra altri dispositivi sulla rete.
3. La topologia mesh può in un certo senso essere considerata sicura perché esiste un collegamento punto a punto, quindi l'accesso non autorizzato non è possibile.
4. Permette un facile troubleshooting.

- **Svantaggi**

1. A causa della grossa quantità di cavi necessaria a collegare ciascun nodo il collegamento è molto complesso.
2. poiché ogni dispositivo deve essere collegato ad altri dispositivi, il numero di porte I/O richieste per ogni dispositivo è crescente al crescere dei nodi.
3. Questa topologia ha problemi di scalabilità in quanto un dispositivo ha un limite massimo di schede I/O di cui può disporre.

1.3.2 Categorizzazione per Tecnologia

Ci sono due tipi di tecnologie trasmissive impiegate a largo spettro:

1. Collegamenti broadcast (*broadcast links*);
2. Collegamenti punto-punto (*point-to-point links*).

Le **reti broadcast** hanno un solo canale di comunicazione che è condiviso fra tutte le macchine della rete. Brevi messaggi, chiamati in alcuni contesti **pacchetti**, sono inviati da ciascuna macchina e ricevuti da tutte le altre. Un campo indirizzo nel pacchetto individua il destinatario, l'indirizzo viene verificato e il pacchetto ricevuto.

Le **punto-punto**, al contrario, consistono di molte connessioni tra singole coppie di macchine, chiamate *unicasting*. Per andare dalla sorgente alla destinazione, in questo tipo di rete un pacchetto deve visitare una o più macchine intermedie

1.3.3 Categorizzazione per Estensione

Vi è poi una classificazione per estensione geografica:

- **PAN**: sono le reti che si estendono per un paio di metri nell' intorno dell' utilizzatore;
- **LAN**: reti che si estendono per un raggio di qualche centinaio di metri (di solito quelle casalinghe e di piccoli uffici);
- **MAN**: reti più estese, che di solito coprono il territorio cittadino (ad esempio sono reti di questo tipo le televisioni via cavo americane);
- **WAN**: sono le reti più estese, che raggiungono diversi Km e coprono macro-regioni o intere nazioni;

Le reti PAN sono reti che vengono stabilite tra dispositivi nel raggio di pochi metri. Si parte da reti personali abbastanza semplici come può essere quella di un computer portatile collegato a una stampante e a uno smartphone ad altre più complesse, i cui nodi sono rappresentati da sensori che, ad esempio, monitorano funzioni vitali di una persona (battito cardiaco, pressione sanguigna etc.) Tali reti usano tecnologie e protocolli differenti, tra i primi abbiamo:

- **Onde elettromagnetiche**
- **IRdA**
- **USB**
- **FireWire**

Per quanto riguarda invece i protocolli possiamo avere: **WiFi**, **Bluetooth**, **ZigBee**, **USB**, **FireWire**, **NFC**.

Le reti LAN, sono reti locali costituite da almeno due dispositivi come minimo e possono estendersi anche fino a coprire diversi edifici, ad esempio un campus universitario. Di solito le reti LAN sono progettate per condividere risorse e quindi abbattere i costi, ad esempio stampanti e dispositivi di archiviazione ma anche un'unica connessione a internet piuttosto che un programma che viene ospitato su un'unica macchina chiamata server. In una rete LAN presente negli uffici, le topologie più comuni sono quella a bus e quella ad anello; in entrambi questi casi vedremo che ci sarà bisogno di un meccanismo che regoli la trasmissione dei dati, in quanto, in caso di tentativo di invio di più host in maniera contemporanea, potrebbero verificarsi problemi.

Le reti MAN sono storicamente note per essere nate per fornire servizi di tv via cavo, soprattutto negli States. Queste reti, che sfruttano come mezzo fisico cavi in fibra ottica, sono state usate successivamente anche come collegamenti per accedere ad internet. La topologia maggiormente usata per queste reti è quella ad anello (Anelli metropolitani) ed ha funzionalità di backbone. In Europa si hanno tecnologie MAN anche di tipo wireless con il WiMax e il 5G: tale tecnologia fornisce un'alternativa alle connessioni DSL, permettendo una connettività sia wireless che mobile attraverso zone metropolitane o di una vasta area.

Una WAN è una rete con linee di comunicazione a grandi distanza. Essa può essere sia molto semplice con un singolo collegamento di tipo punto-punto, o molto complessa quando costituisce la dorsale di una interrete. In quest' ultimo caso la rete da nodi rappresentati da router che si incaricano di inoltrare i pacchetti inviati dagli hosts di una LAN. Man mano che i pacchetti arrivano ai router, essi vengono inoltrati su canali che permettono loro di raggiungere i destinatari. Tali reti vengono dette a commutazione (o anche packet-switched o store and forward).

La commutazione di pacchetto non è l' unico modo di trasferire le informazioni attraverso la rete. È possibile anche impostare un percorso unico attraverso il quale fluiranno tutti i pacchetti appartenenti a quel flusso. In questo caso abbiamo una rete a commutazione di circuito. In entrambe le tipologie di rete, i percorsi che i pacchetti seguiranno sono scelti in accordo a determinati algoritmi detti "algoritmi di routing".

Le reti di dispositivi digitali possono essere suddivise anche in base al mezzo di trasmissione. Una

tipologia di reti sempre più in auge negli ultimi anni sono quelle senza fili (wireless) e le reti mobili. Le due cose non sempre coincidono. Normalmente le reti senza fili funzionano attraverso onde elettromagnetiche. Ci sono diversi standard e architetture che una rete wireless può usare. Le reti wireless prevedono un gateway che le connette a reti cablate. Le reti cellulari (o mobili) sono anch'esse reti senza fili e come tali rispecchiano i principi base di tutte le reti wireless.

Le reti wireless sono classificate per:

- tipologia di interconnessione, ad esempio a corto raggio, P2P etc
- standard, adottati nelle diverse tipologie LAN, MAN, WAN; i più usati derivano dallo standard IEEE 802 ma non sono i soli, soprattutto se ad esser prese in considerazione sono le reti mobili.

Proprio a dimostrare quanto sopra detto, vediamo un paio di esempi:

- una tipica connessione bluetooth di tipo punto-multipunto (vedremo che esistono anche altri tipi di connessione per questo standard);
- una connessione LAN WiFi a infrastruttura, dove diversi clients si connettono a un AP (Access Point).
- tipologia di interconnessione, ad esempio a corto raggio, P2P etc
- standard, adottati nelle diverse tipologie LAN, MAN, WAN; i più usati derivano dallo standard IEEE 802 ma non sono i soli, soprattutto se ad esser prese in considerazione sono le reti mobili.

Alcune categorie di Reti casalinghe sono: Rete di computer: Reti di intrattenimento: videogiochi, tv etc Reti di comunicazione: telefoni, cellulari etc Reti di elettrodomestici; frigo, aria condizionata etc In realtà, tutte queste reti, possono convergere, in quella che viene chiamata domotica. La domotica è una rete di dispositivi all'interno della casa che comunicano tra loro.

Wireless: sono reti che trasferiscono i dati attraverso mezzi fisici non guidati, come ad esempio onde elettromagnetiche; Home Network: le reti che distribuiscono i segnali dentro casa; possono essere una commistione di tecnologie: cablate e wireless; Internetwork: sono le reti (di solito composte da pochi dispositivi) che interconnettono diverse reti tra loro (ad esempio una WAN con una LAN).

1.3.4 Standard di Rete e Organizzazioni

è impossibile studiare le reti di dispositivi digitali e le relative tecnologie senza dare uno sguardo rapido a tutta una serie di standard correlati all'argomento e alle organizzazioni che si occupano di tali standard. Gli standard sono delle norme che unificano e facilitano l'interoperabilità delle tecnologie di rete e sono estremamente importanti. Tutte le reti che conosciamo e ogni dispositivo hardware o protocollo è governato da almeno uno standard (ma di solito sono più di uno).

In questa breve sezione vengono presentati alcuni standard di rete e organizzazioni che si occupano di fornire tali standard. Inizieremo col capire perché gli standard sono importanti, evidenziando le differenze tra standard proprietari e quelli aperti; ci sarà poi una panoramica degli standard di rete e delle più importanti organizzazioni internazionali di standard e gruppi industriali che ricoprono un ruolo importante nel networking.

Inizialmente, agli albori dell'informatica, non si capiva l'importanza di avere uno standard, e la maggior parte delle aziende che decideva di immettere sul mercato un dispositivo o un protocollo pensava fosse più importante tenere per sé le informazioni sul come funzionasse in modo da mantenere il controllo sul mercato: condividere con la concorrenza le informazioni sulle 'scoperte' aziendali non era considerata una mossa furba. è così che sono nati gli standard proprietari. Quello che non va con gli standard proprietari è che altre aziende sono escluse dal processo di sviluppo delle tecnologie in questione e quindi hanno pochi incentivi a collaborare con il proprietario dello standard; al contrario, esse avranno una forte motivazione a sviluppare uno standard proprietario concorrente, anche se non migliora quello esistente.

A tutte le tecnologie di rete sono associati degli standard: questi sono solitamente documenti altamente tecnici. Ogni tecnologia di rete può avere più di uno standard e i motivi sono uno o più dei seguenti:

- Lo standard originale è stato rivisto o aggiornato;
- La tecnologia è sufficientemente complessa da dover essere descritta in più di un documento;
- La tecnologia si basa su documenti utilizzati in tecnologie correlate;
- Le organizzazioni coinvolte nello sviluppo della tecnologia sono molteplici.

Oggi la maggior parte degli standard di rete sono standard 'aperti', amministrati da un'organizzazione che si occupa di standard o da un gruppo industriale piuttosto ampio. In effetti, le poche tecnologie in cui non esiste uno standard aperto e universalmente accettato hanno perso terreno rispetto a quelle con standard aperti, in particolare nelle aree delle LAN wireless e delle reti domestiche.

Per facilitare lo sviluppo di standard aperti sono necessarie organizzazioni che coordinino la creazione e la pubblicazione di questi documenti. In genere, si tratta di organizzazioni senza scopo di lucro che assumono specificamente una posizione neutrale riguardo alle tecnologie e lavorano per il miglioramento dell'industria nel suo insieme. Di seguito una lista di organizzazioni che hanno questo ruolo:

- **International Organization for Standardization (ISO)**: probabilmente la più grande organizzazione di normazione al mondo, l'ISO è una federazione di organizzazioni di normazione di dozzine di nazioni. Nel mondo delle reti, l'ISO è famoso soprattutto per il suo modello di riferimento OSI che abbiamo incontrato prima.
- **American National Standards Institute (ANSI)**: ANSI è la principale organizzazione responsabile del coordinamento e della pubblicazione di standard informatici ed elettronici negli Stati Uniti. esso supervisiona e accredita le organizzazioni che effettivamente creano gli standard, qualificandole come organizzazioni per lo sviluppo di standard (SDO). L'ANSI pubblica anche i documenti degli standard creati dalle SDO e funge da rappresentante degli Stati Uniti presso l'ISO
- **European Telecommunications Standards Institute (ETSI)**: un'organizzazione con membri provenienti da dozzine di paesi appartenenti soprattutto all' Europa, ma anche dall'esterno, che si dedica allo sviluppo di standard di telecomunicazione per il mercato europeo (ma non solo). ETSI è noto, tra le altre cose, per la regolamentazione dell'uso della larghezza di banda radio in Europa e per lo sviluppo di standard come HiperLAN.
- **Institute of Electrical and Electronics Engineers (IEEE)**: è un'organizzazione professionale ben nota per coloro che operano nel campo elettrico o elettronico, inclusi computer e reti. Il maggiore successo di questa organizzazione nel settore delle reti è il progetto IEEE 802, che comprende diverse tecnologie di rete divenute ormai standard diffusissimi, tra cui Ethernet.
- **EIA/TIA**: La TIA (elecommunications Industry Association) è il settore che si occupa di telecomunicazioni dell'EIA (Electronic Industries Alliance) ed è responsabile dello sviluppo degli standard in questo campo (si pensi agli standard che riguardano il cablaggio delle reti).
- **International Telecommunication Union - Telecommunication Standardization Sector (ITU-T)**: è un altro grande organismo internazionale che sviluppa standard per l'industria delle telecomunicazioni.
- **Internet Engineering Task Force (IETF)**: l'IETF si concentra su questioni relative allo sviluppo delle attuali tecnologie Internet e TCP/IP. è diviso in una serie di gruppi di lavoro (WG), ciascuno dei quali è responsabile dello sviluppo di standard e tecnologie in un'area particolare, come il routing o la sicurezza. Ogni area è indipendente e gestita da un direttore di area (AD), che fa parte dell'IESG

Il successo di Internet come rete globale dipende dallo sviluppo di protocolli e tecnologie standard universalmente accettate. Le organizzazioni di standard nel mondo Internet, come l'IETF, sono quindi di fondamentale importanza: esse gestiscono il processo di sviluppo degli standard, per garantire che tutti siano d'accordo su come creare hardware e software che siano in grado di cooperare in tutto il mondo.

Sebbene, per quanto detto, sembri ovvia la necessità di standardizzare i protocolli nelle reti, ci sono un altro paio di aspetti che non sono secondari e, forse, non altrettanto ben compresi:

- **Standardizzazione dei parametri:** la maggior parte dei protocolli si basa sull'uso di parametri che sono importanti per il loro corretto funzionamento; per questo, così come è essenziale che i dispositivi siano d'accordo su quali protocolli utilizzare, essi devono anche accordarsi sui parametri da utilizzare per quei protocolli, affinché la comunicazione abbia successo.
- **Allocazione delle risorse globali e unicità dell'identificatore:** in Internet le risorse da allocare sono numerose e l'unicità nell'assegnazione è essenziale: si pensi ad esempio a un indirizzo IP, che deve essere univoco.

In Internet le autorità di registrazione responsabili del coordinamento e delle risorse assegnate a livello globale (come ad esempio gli indirizzi IP) sono due organizzazioni centralizzate. La prima a nascere e a occuparsi di questo tipo di problemi è stata la Internet Assigned Numbers Authority (**IANA**), che inizialmente era responsabile dell'assegnazione dell'indirizzo IP, della gestione dei nomi di dominio DNS e del registro dei protocolli. Oggi l'Internet Corporation for Assigned Names and Numbers (**ICANN**) ha la responsabilità generale di queste attività; la IANA opera sotto l'egida dell'ICANN ed è ancora responsabile dell'assegnazione degli indirizzi IP e del coordinamento dei parametri.

Rimanendo sul concetto di gestione delle risorse IP, originariamente gli indirizzi venivano assegnati alle organizzazioni direttamente da IANA in blocchi di indirizzi gestiti per classi: indirizzi di classe A, classe B e classe C. Oggi viene invece utilizzato un sistema di indirizzamento gerarchico e senza classi chiamato Classless Inter-Domain Routing (**CIDR**). L'assegnazione degli indirizzi in CIDR prevede l'allocazione gerarchica di blocchi di indirizzi di grandi dimensioni che vengono assegnati a grandi organizzazioni, che poi li dividono per assegnarli a gruppi più piccoli.

IANA, in quanto organizzazione responsabile di tutti gli indirizzi IP, assegna i blocchi di indirizzi più grandi a dei registri regionali (RIR, regional Internet registries) responsabili di ulteriori allocazioni: ogni RIR può assegnare blocchi di indirizzi direttamente agli Internet provider (ISP) o delegarli ulteriormente a registri Internet nazionali (NIR) o più piccoli locali (LIR). Ciascun RIR gestisce gli indirizzi IP e altre risorse Internet (come i numeri degli AS, Autonomous Systems) per una determinata regione. Di seguito la lista dei RIR:

- **AfriNIC** (African Network Information Centre) - Africa
- **APNIC** (Asia Pacific Network Information Centre) - Asia e Oceano Pacifico
- **ARIN** (American Registry for Internet Numbers) - Nord America
- **LACNIC** (Latin America and Caribbean Network Information Centre) – America Latina e Caraibi
- **RIPE NCC** (Rèseaux IP Europèens Network Coordination Centre) - Europa, Medio Oriente e Asia Centrale



Figura 1.13: Regional Internet Registries

La standardizzazione è stata ottenuta principalmente con la costruzione del consenso attraverso la discussione su nuove tecnologie e protocolli: chiunque avesse una proposta per un nuovo protocollo o tecnologia, o un'idea per cambiarne uno esistente, potrebbe creare un documento descrivendo tecnicamente quanto pensato e mettendolo a disposizione della comunità per eventuali commenti o miglioramenti. Poiché l'obiettivo è sollecitare commenti sulla proposta, questi documenti sono stati chiamati richieste di commenti (RFC, Requests for Comments). Benché sia meglio una descrizione tecnica, non tutti gli

RFC, comunque, descrivono standard formalizzati: molti sono solo documenti descrittivi, chiarimenti o informazioni varie. L'accesso aperto e gratuito alle RFC ha notevolmente contribuito al successo di Internet.

Un documento, per diventare uno standard, passa attraverso quattro fasi:

1. **Lo sviluppo di base:** la specifica non ha alcuno status formale ed è solo un'idea; non è detto che debba proseguire le fasi di standardizzazione.
2. **Proposed Standard:** la specifica è stabile, sufficientemente utilizzata e non vengono riscontrati problemi.
3. **Draft Standard:** la specifica è stata Proposed Standard per almeno sei mesi ed esistono almeno due implementazioni indipendenti che abbiano interagito per provarne le funzionalità;
4. **Internet Standard:** la specifica ha superato un' ulteriore soglia di quattro mesi durante i quali è stata sufficientemente utilizzata da un numero considerevole di utenti interessati.

Altre categorie RFC sono:

- **Best Current Practice:** un documento che fornisce informazioni sulle linee guida o raccomandazioni dell'IETF ma che non è uno standard.
- **Informativo:** un documento che fornisce informazioni generali o commenti.
- **Sperimentale:** una proposta per uno standard sperimentale che non è segnalato negli standard; ad esempio le modifiche proposte ai protocolli esistenti che non sono accettate come standard formali passano allo stato 'sperimentale'.

Nel mondo delle reti (e in generale dell' informatica) gli standard vengono spesso divisi in due categorie: **de jure** e **de facto**. Gli standard de jure sono quelli omologati da organizzazioni ufficiali, che procedono a una ratifica tramite rigorose procedure, mentre gli standard de facto, o market-driven, sono quelli ampiamente adottati dalle aziende di un certo comparto che decidono di farne uso. Uno standard de facto può ovviamente diventare de jure se approvato da un'organizzazione ufficiale che si occupa di standard, ad esempio secondo i processi prima descritti, che ne testeranno e garantiranno ripetibilità, qualità e sicurezza.

Capitolo 2

Livello Fisico

2.1 Basi Teoriche della Comunicazione e della Trasmissione Fisica dei Dati

2.1.1 Dati e Segnali

La maggior parte dei dati che registriamo dai fenomeni naturali è analogica. Ciò significa che esso ha una forma continua. Classico esempio è il sismografo che in caso di terremoto registrerà una linea continua indicante come la severità dell' evento. Spesso i dati analogici devono essere convertiti in un segnale, anch' esso analogico, per essere poi elaborati da un sistema.

Al contrario, i **dati digitali** si riferiscono a insiemi di valori discreti. Il codice Morse, ad esempio, si basa su due valori discreti (un trattino e un punto). Diverse combinazioni di questi due valori vengono utilizzate per rappresentare lettere, numeri e punteggiatura. Pertanto, un messaggio codificato utilizzando il codice Morse risulterà in un insieme di dati digitali. Un segnale è un'onda elettromagnetica o elettrica che trasporta dati da un sistema a un altro attraverso una canale o una rete. Così come i dati, anche i segnali sono di due tipi principali: segnali analogici e digitali. Discuteremo brevemente le caratteristiche, i vantaggi e gli svantaggi corrispondenti di ciascuno e le applicazioni tipiche dei segnali analogici rispetto a quelli digitali. La rappresentazione di un segnale analogico risulta in una linea continua così come rappresentato in figura 2.2.

Un **segnale analogico** è variabile nel tempo e generalmente legato a un intervallo (ad es. da +24V a 0V), ma esiste un numero infinito di valori all'interno di tale intervallo continuo. Un segnale analogico utilizza una determinata proprietà del mezzo per trasmettere le informazioni del segnale, come l'elettricità che si muove attraverso un filo. In un segnale elettrico possiamo variare la tensione, la corrente o la frequenza del segnale per rappresentare l'informazione. I segnali analogici sono spesso risposte calcolate a cambiamenti di luce, suono, temperatura, posizione, pressione o altri fenomeni fisici.

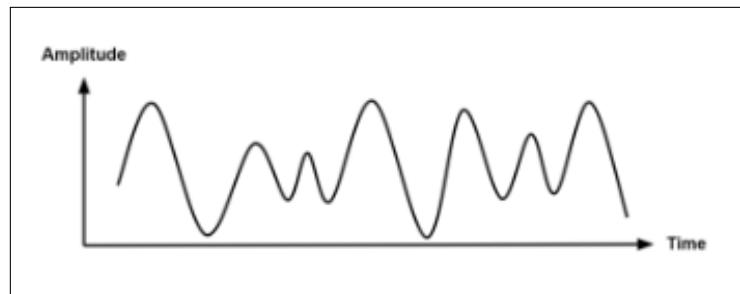


Figura 2.1: Segnale Analogico

Un **segnale digitale** è un segnale che rappresenta i dati come una sequenza di valori discreti: esso può assumere, in un dato momento, un valore solo da un insieme finito di possibili valori. I segnali digitali sono utilizzati in tutta l'elettronica digitale, comprese le apparecchiature informatiche e i dispositivi di

trasmissione dati. Quando vengono tracciati su un grafico tensione/tempo hanno l' andamento come quello mostrato in figura 2.1.

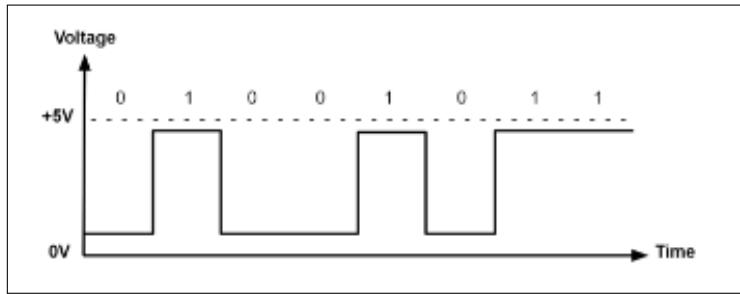


Figura 2.2: Segnale Digitale

Come per la maggior parte degli argomenti di ingegneria, ci sono vantaggi e svantaggi sia per i segnali analogici che per quelli digitali. L'applicazione specifica, i requisiti di prestazione, il mezzo di trasmissione e l'ambiente operativo possono essere determinanti sulla scelta di quale dei due usare (o una combinazione di entrambi).

Segnali Analogici: Pro e Contro

I vantaggi dei segnali analogici sono di seguito riportati:

- sono più facili da elaborare dunque, più adatti per la trasmissione audio e video;
- hanno una densità molto più elevata e possono presentare informazioni più raffinate;
- utilizzano una larghezza di banda inferiore rispetto ai segnali digitali;
- forniscono una rappresentazione più accurata dei cambiamenti nei fenomeni fisici, come suono, luce, temperatura, posizione o pressione;
- i sistemi di comunicazione analogici sono meno sensibili in termini di tolleranza elettrica.

Svantaggi di questi segnali sono:

- la trasmissione di dati a lunghe distanze può causare disturbi del segnale indesiderati;
- sono soggetti a perdita di generazione¹;
- sono soggetti a rumore e distorsione, al contrario dei segnali digitali che hanno un'immunità molto più elevata;
- sono generalmente segnali di qualità inferiore rispetto ai segnali digitali.

Segnali Digitali: Pro e Contro

I vantaggi dell'utilizzo dei segnali digitali nei sistemi di comunicazione, includono:

- possono trasmettere informazioni con meno rumore, distorsione e interferenza;
- l'elaborazione del segnale digitale è più flessibile perché le operazioni DSP possono essere modificate utilizzando sistemi programmabili;
- l'elaborazione del segnale digitale è più sicura perché le informazioni digitali possono essere facilmente crittografate e compresse;
- i sistemi digitali sono più accurati e la probabilità che si verifichino errori può essere ridotta impiegando il rilevamento degli errori e i codici di correzione;
- i segnali digitali possono essere facilmente memorizzati su qualsiasi supporto magnetico o supporto ottico utilizzando chip a semiconduttore;

¹diminuzione o alla degradazione della qualità del segnale nel corso del tempo

- i segnali digitali possono essere trasmessi su lunghe distanze.

Gli svantaggi sono:

- per la comunicazione digitale è necessaria, per la stessa quantità di informazione, una larghezza di banda maggiore rispetto alla trasmissione analogica;
- l' elaborazione del segnale ad alta velocità necessita di più risorse hardware. Ciò si traduce in una maggiore dissipazione di potenza rispetto all'elaborazione del segnale analogico, che include componenti passivi che consumano meno energia;
- i sistemi e l'elaborazione digitali sono in genere più complessi.

Nelle comunicazioni digitali normalmente si usano segnali analogici periodici e segnali digitali non periodici, questo perché i segnali analogici periodici possono essere classificati in *semplice* o *composito*. Un segnale analogico periodico semplice è un'onda sinusoidale, che non può essere scomposta in segnali più semplici. Un segnale analogico periodico composito è invece composta da più onde sinusoidali tra loro sovrapposte.

Un segnale semplice, una sinusoide, è identificata da tre parametri:

- ampiezza;
- frequenza;
- fase.

L' **ampiezza** rispecchia il valore assoluto del segnale ed è proporzionale all' energia da esso trasportata. Graficamente parlando l'ampiezza è il picco della sinusoide.

La **frequenza** è la velocità con cui il segnale cambia rispetto al tempo. Segnali ad alte frequenze implicano cambiamenti veloci, quelli a bassa frequenza cambiamenti lenti. Questo porta a dire che un segnale con valore costante ha frequenza pari a 0, mentre un segnale che cambia istantaneamente ha frequenza infinita. La frequenza si misura in Hz (Hertz). La frequenza è, per definizione, una quantità intrinsecamente positiva, posto che rappresenta essenzialmente il numero di ripetizioni (cicli) per unità di tempi di una data forma d'onda. Definiamo **periodo** il tempo, misurato in secondi, necessario affinché il segnale termini il suo ciclo. Tale definizione ci fa capire che il periodo è l' inverso della frequenza.

Si può introdurre, ora, la distanza che passa tra due massimi (o minimi) consecutivi di un segnale sinusoidale (considerando la sua periodicità spaziale): essa viene chiamata **lunghezza d'onda**, solitamente indicata con la lettera greca λ e misurata in metri.

La **fase** descrive la posizione dell' onda rispetto al tempo 0, o più in generale possiamo dire che è la frazione di periodo trascorsa rispetto a un tempo fissato a un certo istante.

Detto ciò possiamo ricordare che un' onda sinusoidale semplice è descritta dalla formula:

$$f(t) = A \sin(\omega t + \phi)$$

dove, per l'appunto, A è l'ampiezza, ω la frequenza e ϕ la fase.

È possibile analizzare un segnale sia nel dominio del tempo, che in quello della frequenza, traendone informazioni diverse: rappresentandolo nel dominio del tempo, noteremo le oscillazioni, i momenti in cui l'ampiezza è maggiore o minore; viceversa, nel dominio della frequenza il segnale viene per l'appunto scomposto nelle sue varie frequenze.

Nelle comunicazioni un segnale sinusoidale con singola frequenza non è utile in quanto nel dominio della frequenza esso rappresenta una costante; molto interessanti invece sono i segnali complessi in quanto essi possono essere scomposti in segnali più semplici il che significa, nel dominio della frequenza, molta più informazione.

Sotto condizioni di regolarità che sono verificate per la maggior parte dei segnali di interesse pratico, è possibile mostrare che una funzione $f(t)$, periodica di periodo T , può essere ottenuta combinando

linearmente più funzioni: questo permette di sviluppare il segnale nella **serie di Fourier**. Con la serie di Fourier, un segnale periodico viene decomposto in un insieme infinito di frequenze multiple rispetto alla frequenza fondamentale ω_1 ovvero $\omega_n = n\omega_1$: tali frequenze sono dette armoniche. La trasformazione del segnale in serie di Fourier è importante per il fatto che essa permette di scomporre un segnale generico in una somma infinita di sinusoidi con frequenze, ampiezze e fasi diverse; tale scomposizione (che avviene nel nodo mittente del segnale) permette una successiva rapida ricostruzione (lato destinatario) tramite la formula inversa (**antitrasformata**).

La **larghezza di banda**, la cui unità di misura è l' Hertz (Hz), è il range di frequenze usate per trasmettere un segnale senza che esso sia sostanzialmente disturbato. Tale larghezza è data dalla differenza tra la frequenza più alta e quella più bassa nello spettro del canale trasmittivo. La larghezza di banda di un mezzo dipende dalle sue caratteristiche fisiche come il materiale, lo spessore, la lunghezza etc.

I segnali in **banda base** sono i segnali che hanno la frequenza più bassa pari a 0 Hz, quindi possiamo dire che la larghezza di banda di un segnale in banda base corrisponde con la sua frequenza massima. I segnali a **banda passante** sono invece quelli in uscita da un filtro passabanda; quest'ultimo è un filtro elettronico che permette il passaggio solo di un determinato range di frequenze.

Nei sistemi di comunicazione, i segnali analogici viaggiano attraverso mezzi trasmittivi che tendono a deteriorarne la qualità, il che significa che il segnale inviato all'inizio della trasmissione non è lo stesso del segnale che giunge al destinatario. Nella realtà il mezzo trasmittivo oppone una resistenza al segnale che lo attraversa, causandone la modifica: consideriamo di seguito le cause di tale processo.

Attenuazione il segnale perde energia. La resistenza del mezzo fa sì che la forza del segnale diminuisca con l'aumentare della distanza: l'attraversare il mezzo infatti provoca la perdita di energia. Tale processo è noto come attenuazione del segnale. Dei dispositivi chiamati amplificatori vengono utilizzati per riportare il segnale attenuato all'energia originale e compensare questa perdita. L'attenuazione è misurata in decibel (dB); essa misura il rapporto di potenza tra due segnali o di un segnale in due punti diversi

2.1.2 Modalità di Multiplazione

La multiplazione è la tecnica che consente a più utenti di trasferire informazioni attraverso lo stesso canale di comunicazione: in questo contesto, per canale canale si intende una linea di trasmissione come ad es. un doppino intrecciato, un cavo coassiale ecc. Il canale offre una larghezza di banda specificata, che è disponibile per un tempo t , dove t può tendere all'infinito. Possiamo dire che, con riferimento al canale vi sono **2 "gradi di libertà"**:

- banda o frequenza
- tempo

Sono possibili vari metodi di multiplexing in termini di larghezza di banda del canale e tempo, e di segnale, in particolare considerando la frequenza, la fase o il tempo.

Possiamo riassumere i concetti base della multiplazione nei seguenti punti:

- è una tecnica mediante la quale diversi flussi di trasmissione, analogici e digitali, possono essere elaborati simultaneamente su un collegamento condiviso. Il multiplexing divide il mezzo ad alta capacità in un supporto logico a bassa capacità che viene poi condiviso da diversi flussi;
- tutti i mezzi hanno capacità di sfruttare il multiplexing: si possono usare canali via etere (radio-frequenza) oppure supporti fisici guidati come doppini e fibra ottica;
- quando più di un mittente tenta di inviare su un singolo mezzo, un dispositivo chiamato Multiplexer divide il canale fisico in diversi sottocanali e ne assegna uno a ciascun utente. Dall'altra parte del canale un Demultiplexer riceve i dati dall'unico canale, identifica ciascun flusso e invia le informazioni ai diversi destinatari;
- la trasmissione simultanea di due o più segnali può essere eseguita usando più canali di trasmissione o impostando una coppia di ricevitori trasmettitori per ciascun canale, ma questo è un approccio costoso che per l'appunto viene evitata usando il multiplexing.

2.1.3 Criteri di Valutazione delle Reti

Le prestazioni di una rete riguardano la misura della qualità del servizio (**QoS**) e potrebbero anche essere redatte sulla qualità percepita dall'utente (**QoE**).

Esistono diversi modi per misurare le prestazioni di una rete, a seconda della natura e dell'architettura della rete stessa. Le caratteristiche che misurano le prestazioni di una rete secondo la QoS sono:

- larghezza di banda;
- throughput;
- ritardo (Latenza);
- prodotto larghezza di banda - Ritardo;
- jitter.

Andiamo a vedere queste voci una per una:

Larghezza di Banda La larghezza di banda determina la velocità con cui le informazioni richieste sono trasferite. Sebbene abbiano visto che sono diversi fattori da considerare rispetto alle prestazioni di una rete, la larghezza di banda è spesso l'elemento limitante.

Possiamo definirla come la misura di dati o informazioni che possono essere trasmesse in una determinata misura di tempo. Il termine può essere utilizzato in due contesti differenti con due distinti valori di stima. Nel caso dei dispositivi digitali, la larghezza di banda viene misurata in bit al secondo (bps) o byte al secondo. Nel caso di dispositivi analogici, la larghezza di banda viene misurata in cicli al secondo, o Hertz (Hz).

La larghezza di banda non è il solo parametro a influire sulla percezione di velocità di rete di un individuo. Le persone spesso confondono la larghezza di banda con la velocità di Internet perché gli ISP tendono ad affermare di avere una "connessione a TOT Mbps" nelle loro campagne pubblicitarie. La velocità effettiva di Internet è la quantità di dati che si ricevono ogni secondo (e dipende anche dalla latenza). Quindi un concetto che deve essere chiaro è che '**Larghezza di banda**' significa capacità e '**Velocità**' significa quantità di bit trasferiti per l' unità di tempo. Più larghezza di banda non significa automaticamente più velocità, ma questa dipenderà da come si deciderà di 'impegnare' il collegamento. Quando si devono valutare i collegamenti di tipo WAN, prendiamo in considerazione principalmente la larghezza di banda, ma quando consideriamo la LAN, allora valutiamo principalmente la velocità: questo perché nel primo caso siamo soprattutto vincolati al costo della larghezza di banda che arriva sulla WAN e nel secondo caso (su LAN) i vincoli sono sul mezzo trasmissivo, sull'hardware e sulla velocità di trasferimento dati dell'interfaccia.

Larghezza di banda in Hertz: è l'intervallo di frequenze contenuto in un segnale composito, ossia l'intervallo di frequenze che un canale lascia passare.

Larghezza di banda in bps: si riferisce al numero di bit al secondo che un canale o un collegamento può trasmettere.

Esiste una relazione diretta tra larghezza di banda in Hertz e in bps: ad esempio se un ciclo di segnale trasporta 1 bit di informazione, la frequenza del sistema (in Hertz) è uguale alla sua velocità (in bit al secondo). Quindi una giusta idea sarebbe quella di trasportare più bit di informazioni per singolo ciclo. Pertanto è possibile aumentare la velocità di un sistema senza modificarne la frequenza.

Throughput Il throughput è la quantità di dati trasmessi per unità di tempo. Dipende dalla larghezza di banda disponibile, dal rapporto segnale-rumore e da limitazioni hardware. Il throughput nominale di una rete (ossia il valore massimo che esso può raggiungere) può essere quindi superiore al throughput effettivo raggiunto durante una sessione di comunicazione. I termini 'throughput' e 'bandwidth' vengono spesso usati come sinonimi, ma in realtà essi indicano due concetti diversi. La larghezza di banda è la misurazione della trasmissione potenziale di un collegamento, mentre il throughput è una misurazione effettiva della velocità con cui inviamo i dati.

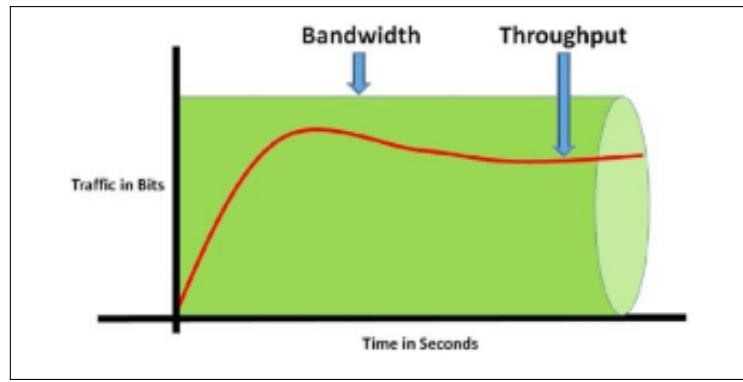


Figura 2.3: Larghezza di Banda e Throughput

Quanto detto può essere desunto dall' immagine sopra riportata dove assimiliamo la larghezza di banda alla portata massima di una conduttrice (colorata in verde), mentre il throughput è l'attuale flusso che da esso esce (linea rossa).

Il throughput non potrà mai superare la larghezza di banda.

Ritardo In una rete, durante il processo di comunicazione, il ritardo (detto anche latenza) è definito come il tempo totale impiegato da un pacchetto per arrivare a destinazione; la misurazione di questo parametro parte dal momento in cui il primo bit del messaggio viene inviato dalla sorgente e termina quando l'ultimo bit viene consegnato alla destinazione. Le connessioni di rete, a seconda che i ritardi siano piccoli o importanti, sono chiamate 'Reti a bassa latenza' o 'ad alta latenza'.

L'elevata latenza impedisce di sfruttare in toto il canale di comunicazione e riduce la larghezza di banda della rete. L'effetto della latenza sulla larghezza di banda può essere temporaneo o indefinito a seconda della fonte che causa i ritardi. Un altro modo di dire per identificare la latenza è 'frequenza di ping' ed è misurata in millisecondi (ms).

Essa può essere misurata in molti modi: 'round trip', 'one way', etc; inoltre ogni componente presente nel tragitto del pacchetto che va dalla sorgente al destinatario può avere la sua influenza sul ritardo totale, che può essere così calcolato:

$$R = t_P + t_T + t_A + r_E$$

dove:

- R è il ritardo;
- t_P è il tempo di propagazione;
- t_T è il tempo di trasmissione;
- t_A è il tempo di accodamento;
- r_E è il ritardo di elaborazione.

Descriviamoli meglio:

- **Tempo di propagazione:** è il tempo necessario all' unità dell' informazione (il bit) per viaggiare dalla sorgente alla destinazione. Il tempo di propagazione può essere calcolato come il rapporto tra la lunghezza del collegamento (distanza) e la velocità di propagazione sul mezzo di comunicazione:

$$t_P = \frac{d}{v_P}$$

dove:

- t_P è il tempo di propagazione;
- d è la distanza;

– v_P è la velocità di propagazione.

- **Tempo di trasmissione:** esso è il tempo necessario per inviare il segnale lungo la linea di trasmissione, ossia il costo in termini temporali per la propagazione di un segnale EM/ottico da un lato all'altro del mezzo trasmissivo. Il tempo di trasmissione di un messaggio dipende dalla dimensione del messaggio e dalla larghezza di banda del canale:

$$t_T = \frac{G}{L}$$

dove:

- t_T è il tempo di trasmissione;
- G è la grandezza del messaggio;
- L è la larghezza di banda.

- **Tempo di accodamento:** i pacchetti che transitano in una rete complessa, devono attraversare i router, e il tempo necessario affinché passino dalla porta di ingresso alla porta di uscita è chiamato tempo di accodamento; accodamento perché molto spesso non è possibile inoltrare un pacchetto immediatamente e quindi esso rimane in una coda in attesa di uscita. Normalmente questo tempo non è un fattore fisso ma cambia con il traffico in rete e con le politiche di QoS che i router usano.
- **Ritardo di elaborazione:** ossia il tempo che impiega il router a capire dove inviare il pacchetto; non appena il nodo scopre qual'è il destinatario del pacchetto e la via che esso deve seguire, provvederà a metterlo nella coda della porta giusta. Questi costi sono prevalentemente basati sulla complessità del protocollo e dalla potenza hardware del nodo.

Ci sono molte situazioni in cui è più importante sapere quanto tempo ci vuole per inviare un messaggio da un capo all'altro di una rete e viceversa, piuttosto che conoscere il ritardo 'one-way'. Questo tempo di andata e ritorno lo indichiamo come *round-trip time* (RTT) della rete.

Per quanto riguarda la differenza tra latenza e tempo di risposta, il primo parametro (minore del secondo) indica il tempo che un pacchetto impiega a percorrere il tragitto dal mittente al destinatario, mentre il tempo di risposta aggiunge alla latenza il tempo che un nodo della rete impiega per elaborare il pacchetto e rimetterlo sul canale: **Il tempo di risposta sarà quindi sempre maggiore della latenza.**

Prodotto Larghezza di Banda - Ritardo Nelle metriche finora introdotte abbiamo visto sia cosa è la larghezza di banda che il ritardo; e anche utile parlare del prodotto di queste due metriche, vediamo perché.

Intuitivamente, se pensiamo a un canale di comunicazione tra una coppia di processi come a un tubo cavo (vedi Figura 1.14), dove il ritardo corrisponde alla lunghezza del tubo e la larghezza di banda è il suo diametro, allora il prodotto

$$L \times R$$

rappresenta il volume del tubo, ossia il numero massimo di bit che potrebbero essere in transito attraverso il canale in un dato istante.

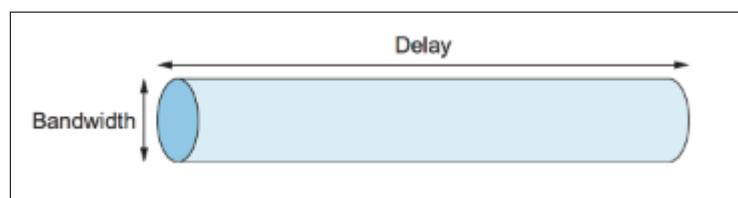


Figura 2.4: Prodotto Larghezza di banda e Ritardo

Il prodotto 'larghezza di banda \times ritardo' è importante soprattutto quando si costruiscono reti ad alte prestazioni perché corrisponde alla quantità di bit che il mittente deve trasmettere prima che il primo bit di questo arrivi al destinatario.

Se il mittente poi si aspetta che il destinatario segnali in qualche modo che i bit sono arrivati, ci vuole altro tempo, pari alla latenza di canale, affinché questo segnale ritorni al mittente, il che implica che il mittente invii una quantità di dati pari a $RTT \times L$ prima di sapere che il destinatario abbia ricevuto senza problemi il primo bit.

Jitter Il jitter è un altro indice di prestazioni, anch'esso legato al ritardo. In termini tecnici, il jitter è una 'varianza del ritardo del pacchetto'. Il jitter è considerato un problema quando il suo valore è alto, ossia diversi pacchetti di dati segnano ritardi differenti in una rete in cui l'applicazione del destinatario è tempo sensibile (questo è vero soprattutto per processi che lavorano con dati audio o video). Il jitter viene misurato in millisecondi (ms).

Le principali cause di jitter sono: interferenze elettromagnetiche (EMI) e diafonia tra i segnali. Il jitter può portare allo sfarfallio dello schermo, all'introduzione di 'clic' o altri effetti indesiderati nei segnali audio, crea inoltre congestione e perdita di dati trasmessi tra i dispositivi di rete.

L'introduzione di buffer può ridurre gli effetti del jitter, sia in una rete, su un router o switch, sia su un computer. Il nodo destinatario che riceve i pacchetti di rete, di solito li riceve dal buffer e non direttamente dall'ingresso al sistema; ogni pacchetto viene quindi espulso dal buffer a una velocità regolare. Un altro approccio per ridurre il jitter in caso di percorsi multipli consiste nell'instradare in maniera selettiva i pacchetti lungo i percorsi più stabili o scegliere sempre il percorso che può garantire il tempo di consegna desiderato.

- l'**affidabilità**, ossia la capacità della rete di trasmettere le informazioni senza errori;
- la **sicurezza**, dei dati o dei processi con i quali essi vengono trasmessi.

2.2 Mezzi di Trasmissione Guidati

I cavi LAN, cioè quelli che collegano i nodi di una LAN, sono i mezzi che permettono la trasmissione dei dati all'interno della suddetta rete locale. Esistono diversi tipi di cavi che possono essere impiegati in tale servizio, anche se spesso, ci si riferisce ai più diffusi di essi: i cavi in rame o Ethernet.

2.2.1 Doppino o Cavo Ethernet

È stato inventato da A. Graham Bell, e si chiama doppino in quanto è costituito da una coppia di conduttori (rame), ciascuno ricoperto da una guaina di isolante, tra loro intrecciati. Uno dei conduttori è utilizzato per trasportare il segnale e l'altro come riferimento di massa. Il ricevitore utilizza la differenza di segnali tra questi due conduttori. La **diafonia** in due conduttori paralleli è elevata (interferenza elettromagnetica che si genera tra due cavi vicini quando in uno di essi passa corrente non costante il segnale, generando uno scambio di energia da una linea all'altra), ma questa può venire notevolmente ridotta se i cavi vengono tra loro intrecciati (questo è il motivo della caratteristica torsione dei due cavi).

Nella prima torsione, un conduttore è vicino alla sorgente di rumore e l'altro è lontano dalla sorgente ma nella successiva torsione avviene il contrario e il rumore risultante è molto minore e quindi viene mantenuta una certa qualità del segnale che arriva al ricevitore. La qualità del segnale nei cavi a doppino intrecciato dipende molto dal numero di torsioni per unità di lunghezza del cavo.

Un modo semplice per identificare la velocità massima di un cavo è l'identificazione della categoria a cui esso appartiene. L'appartenenza a una categoria piuttosto che a un'altra è basata principalmente sulla larghezza di banda e il massimo data-rate (ma le varianti riguardano anche la schermatura e il numero di torsioni per unità di lunghezza).

- **Cat. 1:** Per un certo periodo, questo doppino intrecciato non schermato (UTP) era la forma più comune di cablaggio per i sistemi di telefonia vocale nelle case e negli uffici. Consisteva in due fili di rame isolati attorcigliati l'uno con l'altro ed era progettato per le comunicazioni vocali analogiche.
- **Cat. 2:** Il cablaggio di categoria 2 era usato per comunicazioni di tipo voce e dati ed è stato utilizzato principalmente durante gli anni '80 per le reti IBM Token Ring. Supporta una velocità di trasmissione dati fino a 4 Mbps.

- **Cat. 3:** Introdotto all'inizio degli anni '90, il cablaggio di categoria 3 è stato il primo con quattro doppini intrecciati e il primo a supportare sia reti Ethernet 10BaseT e che le comunicazioni vocali digitali. Si trova ancora negli edifici più vecchi, ma la sua velocità arriva a un massimo di 10 Mbps.
- **Cat. 4:** Come Cat. 3, il cavo di categoria 4 si trova in genere negli edifici più vecchi dove l'elevato costo della sostituzione ha favorito la loro permanenza. Aveva una velocità di trasmissione dati di 16 Mbps ed era utilizzato principalmente per le reti IBM Token Ring.
- **Cat. 5:** Introdotto nel 1995, il cavo di categoria 5 ha una velocità di trasmissione dati fino a 100 Mbps. Viene utilizzato per reti standard 10BaseT e 100BaseT (Fast Ethernet) e può distribuire segnali dati, video e telefonici a distanze fino a 100 metri (nominali). **Cat5** non è una designazione ufficiale, ma viene utilizzata dai produttori per descrivere un cavo Cat5 avanzato, in grado di raggiungere velocità di poco superiori ai 100 Mbps. La sua maggiore velocità di trasmissione dati si ottiene aumentando il numero di torsioni, rendendolo quindi più resistente alla diafonia. Cat5e è consigliato per le nuove installazioni di rete sub-Gigabit.
- **Cat. 6:** Rispetto a Cat5e, il cavo **Cat6** offre una maggiore larghezza di banda e velocità di trasferimento dati fino a 1 Gbps su 100 m. Tuttavia, a distanze inferiori fino a 35 m, Cat6 è in grado di raggiungere velocità di 10 Gbps grazie alla schermatura migliorata e alla maggiore larghezza di banda. Cat6 include un separatore fisico chiamato "spline" tra le quattro coppie per ridurre la diafonia e la schermatura a foglio per ridurre le interferenze elettromagnetiche. Il cablaggio Cat6 è retrocompatibile con lo standard Cat5/5e. Introdotto nel 2009, Cat6a è un cavo di categoria 6 "aumentato" con una larghezza di banda fino a 500 MHz.
- **Cat. 7:** La specifica Cat7 è uno standard proprietario sviluppato da un consorzio di aziende e non è approvato da IEEE o TIA/EIA. Sebbene sostanzialmente simili alle caratteristiche prestazionali di Cat6a, i cavi Cat7 presentano connettori GG45 proprietari e una schermatura robusta. Cat7a (Category 7 Augmented) è un ulteriore perfezionamento di Cat7, in grado di raggiungere velocità di 40 Gigabit oltre i 50 metri e 100 Gbps fino a 15 metri. La natura proprietaria degli standard Cat7 e Cat7a e la mancanza di supporto da parte di IEEE ed EIA ha portato a un'installazione basata su Cat7/Cat7a poco diffusa.
- **Cat. 8:** Con una larghezza di banda fino a 2 GHz (2000 MHz) su 30 metri e una velocità di trasmissione dati fino a 40 Gbs, il cavo Cat8 è ideale per le comunicazioni switch-to-switch in una rete da 25 GBase T o 40 GBase T. I suoi conduttori sono avvolti in un foglio per eliminare virtualmente la diafonia e consentire velocità di trasmissione dati più elevate. Il risultato è un cavo di calibro più importante (in peso e larghezza) che è piuttosto rigido e può essere difficile da installare in spazi ristretti. Utilizza ancora connettori RJ45 ed è retrocompatibile con gli standard precedenti.

I cavi Ethernet costituiti da quattro doppini intrecciati sono terminati utilizzando un connettore **RJ45** a 8 pin. I cavi telefonici più vecchi basati su due doppini in genere utilizzano connettori **RJ11**.

I termini 8p8c (Eight Position, Eight Contact) e RJ45 sono spesso usati in modo intercambiabile, ma 8p8c in realtà si riferisce a una categoria più ampia di connettori di cui RJ45 è fa parte. Il moderno connettore Ethernet RJ45 è l'esempio più comune di connettore 8p8c.

Le coppie di cavi twisted pair sono codificate per mezzo di un colore diverso come definito dal codice colore a 25 coppie sviluppato da AT&T Corporation, e la loro posizione nel cablaggio dipende dalla specifica tecnologia e/o standard.

Esistono due standard di cablaggio utilizzati dai connettori RJ45, denominati T568A e T568B. La differenza è l'assegnazione dei pin per le coppie verde e arancione. I pinout T568A sono i più comunemente usati, ma funzioneranno a patto che entrambe le estremità del cavo siano cablate in modo simile. Il governo degli Stati Uniti richiede l'uso dello standard T568A per le reti installate nell'ambito di un contratto federale (soprattutto per retrocompatibilità), anche se alcuni studi riferiscono di un migliore comportamento rispetto alla diafonia dello standard T568B (ridotto ormai a zero con le nuove disposizioni delle CAT superiori al 5).

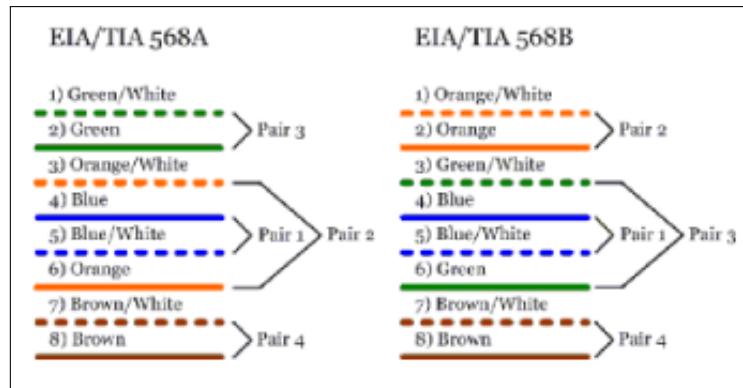


Figura 2.5: Intestazione del Doppino Secondo EIA568

Cavo UTP Il cavo a doppino intrecciato più comunemente utilizzato nelle cablature è spesso quello non schermato (UTP, Unshielded Twisted Pair). I fili conduttori sono tra loro intrecciati a coppie (di solito 4, vecchissimi cavi ormai in disuso ne avevano 3), senza alcun altro isolamento o schermatura. Riducono le interferenze esterne dovute alla presenza dell'isolamento.

Vantaggi:

- UTP è flessibile, economico e facile da installare;
- sono generalmente utilizzati per la trasmissione a breve distanza di voce e dati.

Svantaggi:

- questi cavi hanno una larghezza di banda limitata;
- sono efficienti solo per distanze fino a circa 80 metri (100 nominali) e devono essere sezionati in tratti minori di 100 metri.

Cavo STP Questi tipi di cavi hanno un isolamento extra, ossia un rivestimento protettivo sui conduttori sotto forma di una sottile maglia di rame intrecciata. Questa copertura fornisce anche una maggiore rigidezza alla struttura complessiva del cavo. Riduce il rumore e l'interferenza del segnale nel cavo.

Vantaggi:

- sono generalmente utilizzati per la comunicazione e la trasmissione a lunga distanza in ambienti rumorosi;
- la schermatura protettiva impedisce l'infiltrazione di disturbi elettromagnetici esterni nel cavo;
- hanno una larghezza di banda maggiore rispetto a UTP.

Svantaggi:

- i cavi STP sono costosi e più difficili da installare;
- la manutenzione è costosa più di quelle per UTP.

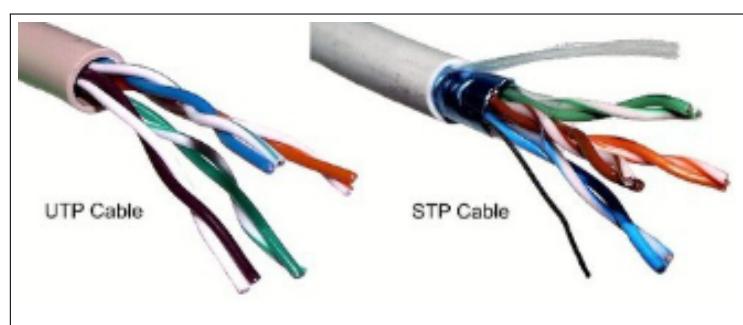


Figura 2.6: Doppino UTP ed STP

2.2.2 Cavo Coassiale

Esso porta segnali su frequenze più alte rispetto a quanto fatto dal doppino. Il cavo coassiale è fatto da una guaina di plastica esterna che contiene al proprio interno due strati di materiale isolante tra loro divisi da uno strato di conduttore metallico; al centro vi è un'anima metallica (di solito rame). Rispetto al doppino il segnale che passa in un cavo coassiale degrada più velocemente all'aumentare della distanza.

Le classificazioni RG (Radio Government) definiscono una funzione specializzata per ogni tipologia di cavo. I seguenti sono alcuni di quelli più comuni:

- **RG-8** – Viene utilizzato in thick Ethernet;
- **RG-9** - Viene utilizzato anche in thick Ethernet;
- **RG-11** – Ancora viene utilizzato in thick Ethernet;
- **RG-58** – Viene utilizzato in thin Ethernet.

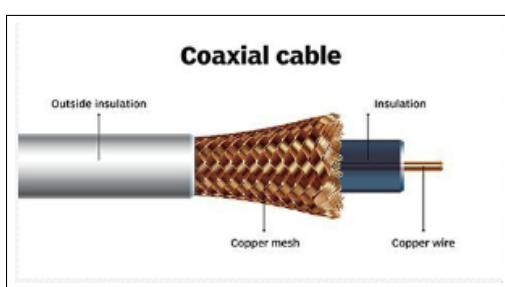


Figura 2.7: Cavo Coassiale

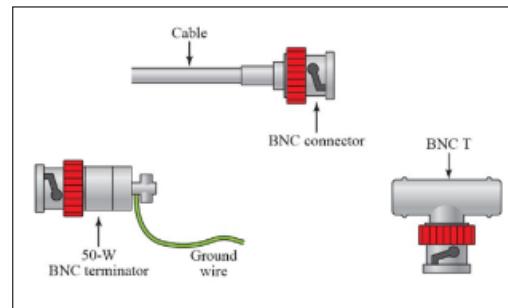


Figura 2.8: Tipi di Connatori Coassiali

Per collegare i dispositivi con il cavo coassiale, sono necessari appositi con nettori. Il più comune di questi è chiamato connettore a botte a causa della sua forma. In particolare quello più comune utilizzato oggi è il connettore a baionetta Neill-Concelman (BNC) che si inserisce e si blocca nella posizione di aggancio con mezzo giro. Altri tipi di connettori cilindrici prevedono che si avvitino tra loro o si spingano senza bloccaggio, che è meno sicuro.

2.2.3 Fibra Ottica

Un sistema di trasmissione ottico è formato principalmente di 3 parti: sorgente luminosa, mezzo di trasmissione e rilevatore di luce. Per convenzione, un impulso di luce indica il valore 1 e l'assenza di luce indica il valore 0. Il mezzo trasmittivo ovviamente è la fibra composta da un nucleo (core) di vetro di pochi micron avvolto in una guaina di vetro (**cladding**) con indice di rifrazione più basso e infine la solita rivestitura con guaina in plastica.

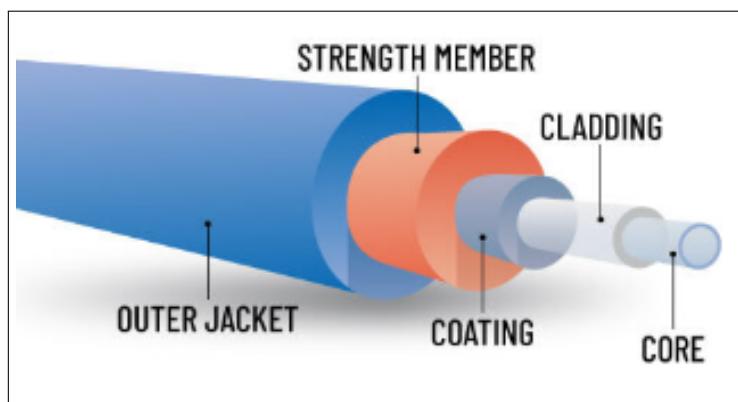


Figura 2.9: Cavo Fibra Ottica

La fibra si basa su un principio molto semplice, ovvero che la luce che la attraversa viene riflessa al suo interno fino ad arrivare all'altra estremità del cavo. Questo avviene perché la luce immessa nel core incontra il cladding con indice di rifrazione minore e il raggio luminoso viene così riflesso (se possiede un'inclinazione corretta). La velocità di tale raggio è circa quella della luce, infatti il limite di banda della fibra non è dovuto alla velocità di trasmissione ma di decodifica del segnale luminoso in impulso elettrico. Una fibra può contenere più raggi che si riflettono in essa l'importante è che il loro angolo di riflessione sia diverso. Questo tipo di fibre è detto **multimodale**. Le fibre che invece permettono la trasmissione di luce in linea retta sono le **monomodali** che non sono altro che guide d'onda ma possono raggiungere i 50Gbps per 100Km senza attenuazione.

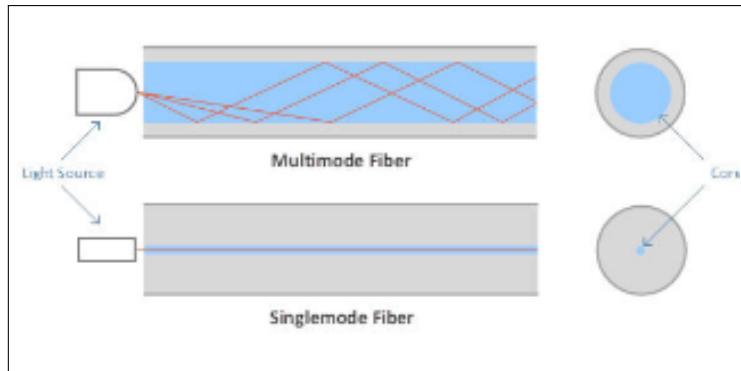


Figura 2.10: Fibra Multi e Mono Modale

L'attenuazione della luce attraverso il vetro dipende dalla lunghezza d'onda della luce. L'attenuazione, espressa in decibel, si ricava dalla seguente formula:

$$\text{attenuazione} = 10 \log_{10} \frac{\text{energiatrasmessa}}{\text{energiaricevuta}}$$

Per la comunicazione ottica si usano tre bande di lunghezza d'onda, centrate rispettivamente a 0,85, 1,30 e 1,55 micron. Gli impulsi luminosi trasmessi attraverso la fibra si espandono in lunghezza durante la propagazione; il fenomeno si chiama **dispersione cromatica**, che dipende dalla lunghezza d'onda. Per generare il segnale normalmente si impiegano due tipi di sorgenti luminose: i **LED** (*Light Emitting Diode*) e i **semiconduttori laser**. All'estremità finale della fibra si trova un fotodiodo che genera un impulso elettrico ogni volta che è colpito dalla luce. Il tipico tempo di risposta di un fotodiodo è un nsec, perciò la trasmissione dati è limitata a 1 Gpbs. Un problema delle fibre è il collegamento tra 2 di esse, che può avvenire in 3 modi:

1. le fibre vengono inserite in apposite prese grazie a dei connettori con perdita di circa 10-20% della luce;
2. le fibre vengono attaccate meccanicamente, messe di fronte una all'altra e poi viene avvolto in una macchina particolare per poi essere pinzate, con perdita comunque di circa il 10% della luce;
3. le fibre vengono fuse tra loro. Una soluzione quasi ottimale, anche se difficile, genera una piccola attenuazione di segnale.

A differenza del cavo di categoria in rame che utilizza l'onnipresente connettore RJ45 indipendentemente dal tipo di cavo, la fibra ottica (di vetro e plastica) può essere terminato utilizzando una varietà di tipi di connettore. La scelta del connettore è determinata dall'attrezzatura e dai requisiti dell'applicazione.

La fibra monomodale richiede un ricetrasmettitore pulito e allineato con precisione che inietti la luce nel suo piccolo nucleo con una precisione inferiore al micron. Al contrario, la fibra multimodale è un po' più tollerante.

- **Connettore a ghiera (FC)** L'FC è stato il primo connettore per fibra ottica a utilizzare una ghiera in ceramica. Questi connettori posizionano e bloccano con precisione il nucleo della fibra rispetto al trasmettitore e al ricevitore. I connettori FC sono stati ampiamente sostituiti dai connettori SC e LC più economici e più facili da installare, ma sono ancora preferiti in ambienti con vibrazioni elevate grazie alla loro pinza a vite.

- **Punta Dritta (ST)** ST era un tempo il connettore in fibra ottica più comune sia per fibra monomodale che multimodale. È dotato di un connettore twist lock a baionetta ed è economico e facile da installare. È ancora utilizzato in applicazioni industriali e militari, ma altrove è stato ampiamente sostituito da fattori di forma più piccoli.
- **Connettore subscriber (SC)** I connettori SC hanno un affidabile meccanismo di bloccaggio a scatto che si blocca con un semplice movimento push-pull. Sono un'opzione economica e durevole valutata per 1.000 cicli di accoppiamento. Questo connettore viene utilizzato nelle configurazioni simplex e duplex (illustrate). I connettori SC sono stati per lo più sostituiti da connettori LC nelle reti aziendali.
- **Connettore Lucent (LC)** Il connettore LC è stato progettato per rispondere alle lamentele che i connettori ST e SC erano troppo ingombranti e si staccavano facilmente. I connettori LC hanno un ingombro di circa il 50% inferiore rispetto al connettore SC. Grazie a queste dimensioni ridotte e alla funzione di blocco sicuro, è ampiamente utilizzato nei data center e nei centri di commutazione delle telecom.



Figura 2.11: Fibra Ottica: Connettori più Comuni

Le LAN basate sulle fibre ottiche solitamente sono ad anello con congiunzione a T per ogni PC o a stella passiva, questo per evitare gli ostacoli. La configurazione ad anello ha il difetto che se una congiunzione si guasta salta tutta la rete. Altre topologie di configurazione possono essere quelle a **stella passiva**

Tra gli innumerevoli vantaggi della fibra ottica rispetto alle connessioni in rame citiamo:

- maggiore ampiezza di banda;
- non viene influenzata dalle sorgenti elettriche;
- non viene influenzata dai campi magnetici e interruzioni della linea elettrica;
- sono molto più leggere;
- non perdono la luce;
- l'intercettazione dei dati è molto difficile, quindi c'è una maggior sicurezza.

Di contro è una tecnologia ancora relativamente nuova e poco sviluppata e presenta grossi costi di manutenzione.

Medium	Attenuation	Electromagnetic Interface	Security	Cost
Unshielded Twisted Pair	High	High	Low	Low
Shielded Twisted Pair	High	Moderate	Low	Moderate
Coaxial Cable	Moderate	Moderate	Low	Moderate
Fibre Optic Cable	Low	Low	High	High
Radio Waves	Low to High	High	Low	Moderate
Microwave Transmission	Can be higher or lower or moderate	High	Moderate	High
Satellite Communication	Can be higher or lower or moderate	High	Moderate	Very High

Figura 2.12: Riassunto Caratteristiche dei Mezzi Trasmissivi Guidati

Tabella Riassuntiva

2.2.4 Onde Convogliate

Il mezzo trasmissivo di questa tecnologia sfrutta i cavi conduttori in rame di una rete di alimentazione elettrica, evitando in questo modo un ulteriore cabaggio per la linea dedicata allo scambio di informazioni.

La tecnica che viene sfruttata è quella della sovrapposizione di un segnale ad alta frequenza e opportunamente modulato alla corrente elettrica già circolante (a bassa frequenza) in una determinata rete di alimentazione; tramite un sistema dedicato vengono poi filtrate e separate le frequenze utilizzate per la trasmissione delle informazioni che sono così rese disponibili all' utilizzatore finale

Questa tecnologia viene usata in molti campi nel mondo reale, dalla telelettura dei nuovi contatori di fornitura di corrente elettrica allo scambio di informazioni tra treni in marcia o semplicemente come rete di trasmissione dati in una LAN posta in edifici dalla difficile ricablatura.

2.3 Mezzi di Trasmissione Wireless

I mezzi wireless trasmettono le informazioni attraverso segnali elettromagnetici. Essi offrono le migliori opzioni di mobilità tra tutti i mezzi; proprio per questo motivo, il numero di dispositivi wireless è in continuo aumento. L'uso sempre maggiore di tale mezzo, spinge anche l'aumento delle opzioni sulla larghezza di banda usata da queste reti, permettendo performances sempre migliori.

I segnali trasmessi attraverso i mezzi non guidati possono essere raggruppati in tre macro-tipologie di segnali:

- **Onde Radio:** esse sono utilizzate per tecnologie come televisione e radio. Le onde radio vengono trasmesse facilmente attraverso l'aria, possono essere riflesse per far cambiare loro direzione e non provocano danni se assorbite dal corpo umano. Inoltre la trasmissione delle frequenze da molto basse a medie, segue la curvatura della Terra, mentre quelle alte e altissime viene riflessa dallo strato di Ionosfera che circonda il pianeta.
- **Microonde:** sono le frequenze che vanno da 1 GHz a 300 GHz. sono usate per comunicazioni come telefonia mobile, satelliti e altri standard wireless che usiamo tutti i giorni. Le microonde hanno frequenze che vengono facilmente assorbite dalle molecole di corpi 'molli'. L'energia interna delle molecole aumenta quando assorbono le microonde e ne provocano il riscaldamento. Le microonde passano facilmente anche attraverso l'atmosfera per questo vengono usate per la comunicazione tra le stazioni sulla Terra e i satelliti in orbita. Per alcune tecnologie sopra i 3 GHz è importante che la Stazione Base e il client siano 'a vista' (LoS, Line of Sight)
- **Infrarossi:** essi sono usati per comunicazioni di prossimità (pensate al telecomando TV) e sono assorbite da qualsiasi ostacolo.

Le onde elettromagnetiche formano uno spettro continuo di onde (chiamato per l' appunto **Spettro Elettromagnetico**).

La lunghezza d'onda è calcolata per mezzo della formula: $\lambda = c/f$, dove c è la velocità della luce nel vuoto, cioè circa $3.8 \times 10^8 m/s$. Questo implica che:

- Onde con una lunghezza d'onda molto corta hanno alta frequenza e alta energia;
- Onde con una lunghezza d'onda molto lunga hanno bassa frequenza e bassa energia.

Le frequenze di lavoro relative ai vari servizi radio vengono scelte in intervalli di frequenza (o bande) che sono definite in sede internazionale e nazionale, in modo da evitare il più possibile interferenze o sovrapposizioni.

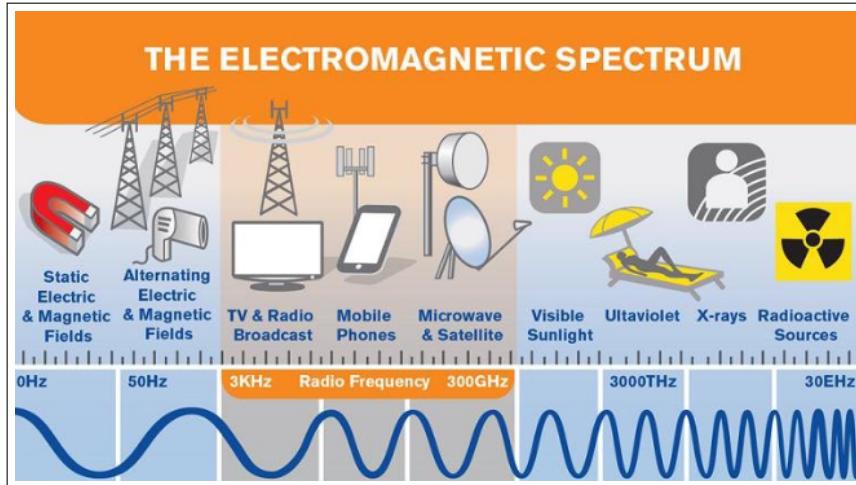


Figura 2.13: Spettro Elettromagnetico - Evidenziata Zona Usata nelle Comunicazioni

Lo Spettro elettromagnetico è una risorsa per gli stati, per questo vengono vendute delle licenze temporali per l'uso delle frequenze di telecomunicazione, tranne quelle che fanno parte della banda ISM (Industrial, Scientific and Medical); banda ISM è il nome assegnato dall'Unione Internazionale delle Telecomunicazioni (ITU) a una banda di frequenze riservate alle trasmissioni radio per uso industriale, scientifico e medico che non sono soggette a licenze, pertanto il loro uso è libero.

Freq. iniz,	Freq. fin,	Largh. banda
902 MHz	928 MHz	26 MHz
2,4 GHz	2,487 GHz	100 MHz
5,735 GHz	5,875 GHz	150 MHz
24,00 GHz	24,250 GHz	250 MHz

Altri temi di interesse che il wireless presenta sono:

- **Area di copertura:** le tecnologie di comunicazione dati wireless funzionano bene in ambienti aperti. Tuttavia, alcuni materiali da costruzione utilizzati negli edifici e nelle strutture e il terreno locale ne limitano la copertura effettiva, e così risulta sempre una sfida capire la qualità del segnale nelle zone coperte.
- **Interferenza:** Il wireless è suscettibile alle interferenze e può essere disturbato da dispositivi comuni che occupano le stesse frequenze, come telefoni cordless domestici, alcuni tipi di luci fluorescenti, fornì a microonde e altri standard wireless.
- **Sicurezza:** viaggiando nell'aria, le informazioni trasmesse in wireless possono essere accedute da dispositivi e utenti non autorizzati. Di conseguenza, la sicurezza della rete è un componente importante dell'amministrazione della rete wireless.

2.3.1 Reti Satellitari

Le reti satellitari hanno nodi (satelliti, stazioni terrestri e terminali) che forniscono collegamenti fra punti diversi della terra. Le reti satellitari, come quelle di telefonia mobili, suddividono la superficie terrestre in celle. In base alla posizione dell'orbita, i satelliti possono essere suddivisi in tre categorie:

- **Orbita Geostazionaria (GEO);**
- **Orbita Media (MEO);**
- **Orbita Bassa (LEO).**

La motivazione tra le diverse orbite è dovuta all' esistenza delle due fasce di Van Allen: strati dell' atmosfera contenenti particelle cariche che metterebbero fuori uso l' elettronica.

Le comunicazioni satellitari usano le microonde ad alta frequenza, e ogni satellite usa due canali, uno per l' uplink e l'altro per il downlink (quindi su bande a frequenza diversa).

Satelliti GEO Abbiamo detto che nella comunicazione ad alte frequenze le antenne del ricevente e del mittente devono 'vedersi'. Per questo motivo i satelliti geostazionari orbitano alla stessa velocità di rotazione della Terra, in modo che le antenne abbiano sempre una posizione 'fissa' tra loro. Ancora per lo stesso motivo servono almeno 3 satelliti, posizionati alla distanza di poco meno di 36000 Km dalla superficie, per coprire l' intero globo.

Satelliti MEO Sono posizionati nella zona che si trova fra le due fasce di Van Allen e vengono utilizzati soprattutto per il sistema GPS. Usano la triangolazione dei dati forniti dai satelliti che coprono quella zona in un determinato momento.

Satelliti LEO Essi orbitano in una zona che va tra i 500 e i 2000 Km e sono molto numerosi: questa trama di nodi viene detta costellazione. Ogni satellite funziona come uno switch: i satelliti vicini tra loro comunicano con dei collegamenti intersatellitari (CIS). Oltre alla comunicazione tra satelliti e terminali, si può avere anche un link con gateway posti sulla terra. Di questo tipi di reti fanno parte Iridium, Globalstar e Starlink SpaceX.

Fibra vs Satelliti Inutile dire che sono due tecnologie diverse tra loro e spesso complementari, proprio come la fibra e il Fixed Wireless Access (FWA). Sicuramente le reti satellitari riescono a raggiungere zone della Terra ove le reti in fibra ancora non sono arrivate o avrebbero difficoltà ad arrivare a causa dell' orografia del terreno. Inoltre la tecnologia satellitare è stata progettata con il broadcasting in mente, cosa che se usassimo la fibra comporterebbe molto consumo di banda. Proprio la larghezza di banda è invece una carta vincente per la fibra quando si parla di link punto-punto.

Capitolo 3

Livello Collegamento

3.1 Data Link Control

Questo è il livello in cui due dispositivi, detti genericamente **nodi**, comunicano tra loro attraverso un **collegamento** diretto; le informazioni vengono trasmesse in **frames**, ossia insiemi di informazioni che hanno una propria struttura che dipende dal protocollo adottato dalla rete a questo livello. I servizi offerti dal livello collegamento sono i seguenti:

- **Framing**: questo è il servizio che si occupa di incapsulare le informazioni che arrivano dai livelli superiori, inserendo dei dati in testa (header) e in coda (tail) che servono per gli altri servizi del livello. Il formato del frame dipende dal protocollo adottato.
- **Accesso al canale**: in alcuni casi i nodi hanno effettivamente un collegamento dedicato, in altri hanno invece un canale condiviso (pensate ai telefoni cellulari o a una rete LAN wireless).
- **Indirizzamento**: il servizio di indirizzamento è essenziale in quanto, su un collegamento condiviso, è necessario capire quale nodo comunica con un altro nodo. L'identificativo è di tipo hardware.
- **Controllo di flusso**: nodi presenti sullo stesso link possono avere velocità di trasmissione diverse o capacità di buffer differenti.
- **Gestione degli errori**: durante la trasmissione delle informazioni possono verificarsi degli eventi che causano errori. Compito del livello Data link è accorgersi e gestire tali errori.

3.2 LAN Cablate: Ethernet

Nel mercato delle reti cablate, sono molte le tecnologie che sono state usate nel tempo (Token Ring, Token Bus, ATM etc) ma sicuramente quella di maggior diffusione è stata Ethernet.

Le reti Ethernet sono state sviluppate da un progetto di ricerca del 1973-1976 alla Xerox di Palo Alto, da un'idea di Robert Metcalfe (che successivamente lascerà la Xerox e fonderà la 3Com per commercializzare la propria tecnologia). Nei primi anni '80 la IEEE ha iniziato un progetto (802) che permettesse l'interconnessione di hardware di produttori diversi, intervenendo sui primi due livelli dello stack ISO/OSI. Nella figura 3.1 è mostrata la relazione tra lo standard IEEE 802 e il modello ISO/OSI.

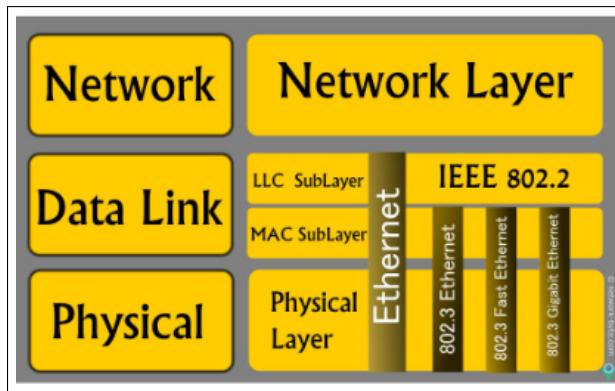


Figura 3.1: Relazione tra IEEE 802 e ISO/OSI

Come si nota dalla figura, il livello Data Link è stato suddiviso in due sottostrati:

- **Logical Link Control (LLC):** in questo strato vengono implementate le funzioni di controllo degli errori, controllo di flusso e parte delle funzioni di framing; c' è un solo sottolivello LLC che è indipendente dal tipo di rete fisica utilizzata.
- **Medium Access Control (MAC):** lo strato MAC definisce il metodo di accesso al mezzo fisico, e pertanto è legato al particolare hardware usato e ne esistono diversi, ognuno specializzato per il particolare tipo di rete che si vuole utilizzare.

Il progetto IEEE 802 ha prodotto una serie di standard che prendono lo stesso nome (802) ma a esso associano un sottogruppo che ne indica le particolarità tipiche della tecnologia usata; così vediamo che le specifiche generali sono esposte nello standard **802.1**; il sottolivello LLC nello standard **802.2**; il metodo di accesso CSMA/CD nello standard **802.3**; il token bus in **802.4** e così via.

Dalla loro nascita, le reti Ethernet si sono evolute in diverse generazioni che prendono il nome dalla velocità massima raggiunta, così dai 10 Mbps della Ethernet legacy arriviamo alle velocità 100GbE e oltre. In questo corso ci soffermeremo soprattutto sulle reti Ethernet standard.

3.2.1 Sottolivello MAC

Il MAC come già detto si occupa dell' accesso al mezzo e della formazione del frame. Un frame Ethernet contiene 7 campi:

- **Preambolo:** è un campo da 7 bytes composti da bit 0 e 1 alternati: esso permette la sincronizzazione (formalmente esso non fa parte del frame di livello DL ed è abbastanza lungo da permettere la sincronizzazione anche nel caso in cui il destinatario non si accorga subito della richiesta di collegamento).
- **Delimitatore:** questo campo di un byte delimita il frame (ne segnala l' inizio e la fine) ed è l' ultima possibilità di sincronizzazione (anche qui abbiamo sequenza di 1 e 0 alternati ma che finiscono con 11, ossia: 10101011).
- **Indirizzo Destinatario:** 6 bytes che definiscono l' indirizzo fisico della destinazione (che potrebbe essere anche più di un nodo).
- **Indirizzo Mittente:** 6 bytes che definiscono l' indirizzo fisico del nodo mittente.
- **Lunghezza o tipo:** 2 bytes che vengono usati per segnalare il protocollo di livello superiore trasportato nel campo dati (questo nella Ethernet legacy), oppure il numero di bytes del campo successivo (standard IEEE 802); entrambe le modalità sono abituali nell' Ethernet odierno.
- **Dati:** Il campo contiene i dati provenienti dal livello superiore e può contenere un numero di bytes minimo (46) fino ad un massimo di 1500.
- **CRC:** 4 bytes che contengono un codice a ridondanza ciclica per il controllo degli errori.

A causa della tecnologia di accesso al mezzo usata (CSMA/CD), Ethernet standard ha bisogno di avere una lunghezza minima di 512 bit (64 Bytes), con campi obbligatori che sono quelli degli indirizzi di destinazione e mittente, la lunghezza e il CRC (lasciando 46 bytes disponibili per i dati). È prevista inoltre anche una grandezza massima del frame (a causa della grandezza dei buffers) che corrisponde a 1518 Bytes.

Abbiamo parlato di indirizzi specificati all'interno del frame in quanto ogni nodo di rete possiede un identificativo univoco di 6 bytes, all'interno del NIC (Network Interface Card), che lo individua all'interno della rete: tali indirizzi vengono di solito specificati con la notazione esadecimale con il carattere di due punti per separare i bytes.

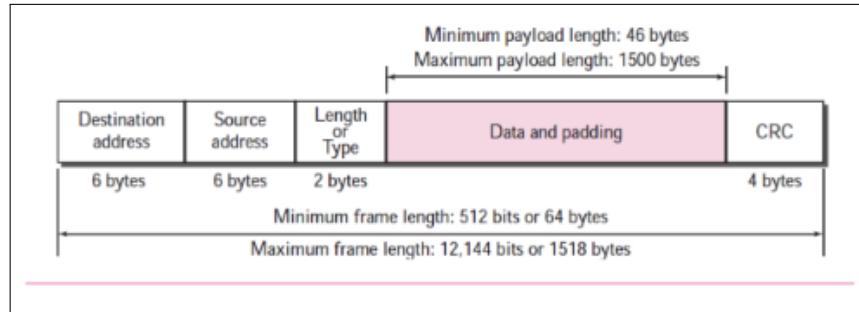


Figura 3.2: Frame Ethernet

I 6 bytes sono suddivisi in maniera che le prime 6 cifre esadecimali (i primi 3 bytes) identifichino il produttore della scheda di rete e sono detti **OUI (Organizationally Unique Identifier)**. Una prima differenziazione degli indirizzi MAC è quella che li divide in "globally unique" e "locally administered" e si ottiene settando il secondo bit meno significativo del byte più significativo dell'indirizzo (cioè del primo byte a sinistra, come si evince dalla fig. 3.3.): se il bit è impostato a 0 identifica un indirizzo globally unique (detto anche UAA, Universally Administered Address), se impostato a 1 invece, identifica un indirizzo locally administered. Gli indirizzi UAA, sono i più comuni e sono quelli assegnati dal produttore della scheda rispettando la regola che i primi 3 bytes corrispondano al proprio identificativo (del produttore) e i restanti 3 bytes permettano l'univoca identificazione del nodo. Quelli "locally administered" sono degli indirizzi che vanno a "coprire" il reale MAC Address della scheda di rete, per motivi necessari all'amministratore di rete, sostituendolo con uno valido solo in locale.

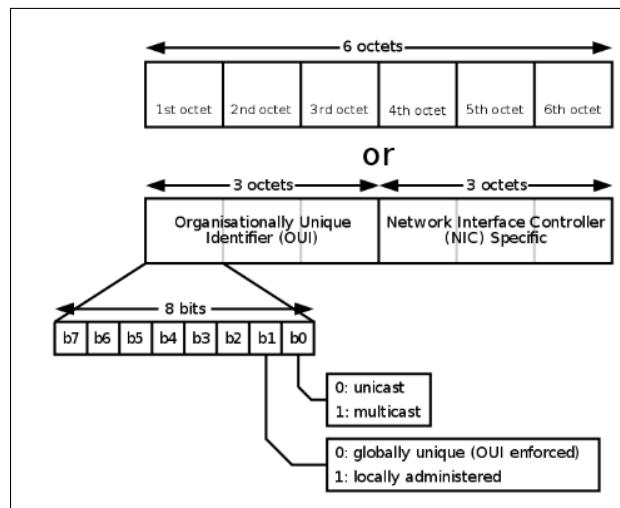


Figura 3.3: Indirizzo MAC

La seconda suddivisione è quella fatta in base all'ultimo bit significativo, sempre del primo byte, che a seconda del valore settato a 0 o a 1 individua l'indirizzo come singolo (ossia univoco di un singolo nodo) o di gruppo; logicamente questa suddivisione riguarda gli indirizzi del destinatario (il mittente è sempre univoco!). L'indirizzo di broadcast è un caso speciale di indirizzo di gruppo e corrisponde a una

sequenza con tutti 1 nei bit dell' indirizzo.

Vediamo ora come Ethernet standard prevede l'accesso al mezzo: **CSMA/CD 1-insistente**. Tra i vari standard progettati per l'accesso al mezzo condiviso ci si è accorti che per minimizzare le probabilità di collisione, è necessario che la stazione che vuole trasmettere controlli lo stato del mezzo prima di inviare. Il metodo CSMA (Carrier Sense Multiple Access) riduce le probabilità di collisione ma non le elimina del tutto a causa del ritardo di propagazione. Se definiamo **tempo di vulnerabilità** il periodo di tempo in cui è possibile che avvenga una collisione, allora, il valore che tale parametro assume per il protocollo CSMA è proprio pari al tempo di propagazione (T_p , ossia il tempo necessario al segnale per propagarsi da un capo all'altro del mezzo). Abbiamo detto che in CSMA la stazione che deve trasmettere controlla lo stato del mezzo prima di farlo e se lo trova occupato continua a controllare fino a che esso non si libera e a quel punto spedisce immediatamente il frame (questa decisione di trasmettere immediatamente è quella che caratterizza il metodo 1-insistente). Pertanto l' intervallo di tempo da utilizzare con il metodo CSMA/CD è dato da 2 volte il tempo di propagazione massimo (andata e ritorno) più il tempo necessario a spedire un segnale di jamming. Tali problematiche, aggiunte ad altre di carattere fisico, limitano la lunghezza massima della rete Ethernet standard a 2500 m (con velocità fino a 10 Mbps). Ethernet ha poi ratificato i maggiori cambiamenti che si sono avuti nel tempo, anche per competere con reti più veloci (fibra, ad esempio) nello standard IEEE 802.3u (**Fast Ethernet**, ossia Ethernet a 100 Mbps) centrando obiettivi come:

1. Aumentare la velocità di trasmissione a 100 Mbps.
2. Mantenere la compatibilità con Ethernet standard.
3. Mantenere lo stesso schema di indirizzamento a 48 bit.
4. Mantenere lo stesso formato e le stesse dimensioni (stessi limiti massimi e minimi) dei frame.

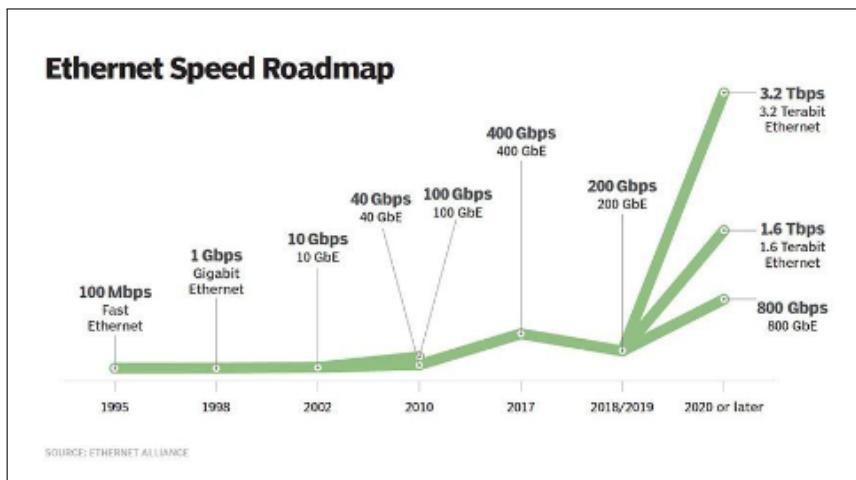


Figura 3.4: Ethernet Roadmap

A partire da questo standard è stato deciso di eliminare la topologia a bus e mantenere solo quella a stella che, come vedremo meglio nel paragrafo che parla della descrizione del livello fisico, prevede un accesso half-duplex o full-duplex, a seconda che i nodi siano connessi attraverso un hub oppure uno switch. Nel primo approccio il metodo per l' accesso al mezzo rimane CSMA/CD, per il secondo invece non c'è bisogno di nessun metodo di accesso (anche se qualsiasi implementazione di Ethernet mantiene tale metodo di accesso per retrocompatibilità).

La necessità di inviare dati a velocità sempre maggiori ha portato allo standard odierno che è Gigabit Ethernet che, come il nome stesso dice, raggiunge velocità misurabili in Gbps. Fermo restando che le considerazioni sugli obiettivi da raggiungere con questo standard sono gli stessi di Fast Ethernet (tranne il fatto che le velocità da raggiungere sono, logicamente, nell' ordine dei Giga), nel progetto di raggiungere questo alto bit rate di trasmissione, la funzionalità di usare connessioni half-duplex è rimasta solo per questioni di compatibilità, ma di fatto non è possibile usarla alle alte velocità.

Nell' approccio full-duplex vengono utilizzati switches direttamente connessi ai nodi o ad altri switches, che presentano buffers su ogni porta, ove conservare i pacchetti in transito: in questo modo non esistono possibilità di collisione e di fatto CSMA/CD non viene mai usato.

Nelle reti veloci che seguono l'approccio full-duplex, quindi, non ci sono collisioni e la lunghezza massima del cavo è determinata dall'attenuazione del segnale nel cavo. Nell' approccio half-duplex, scarsamente usato ma possibile, è possibile avere degli hub sulla rete (che fungono in realtà come estensione del cavo) che però creano possibili collisioni; per questo bisogna trovare delle tecniche alternative (ad esempio frame a raffica) per permettere l' uso corretto di tali reti veloci (essendo più veloci le collisioni vengono rilevate in un tempo più breve, facendo accorciare la lunghezza massima di essa: nelle reti Giga sarebbe di circa 25 metri).

3.2.2 Lo Strato Fisico

Il nome della particolare tecnologia Ethernet usata segue la codifica di sotto riportata:

- Il numero all'inizio indica la velocità massima raggiungibile;
- La parola "base" sta per baseband e indica che i dati sono codificati tramite la codifica Manchester;
- L'ultima sigla specifica la caratteristica saliente di quella particolare versione (ad esempio 2 e 5 indicano che la lunghezza massima è di 200 e 500 metri rispettivamente, mentre la T o la F indicano che il mezzo usato è di tipo Twisted Pair o Fibra).

In Ethernet standard lo strato fisico viene implementato in vari modi, i più comuni dei quali sono:

- 10Base5
- 10Base2
- 10BaseT
- 10BaseF

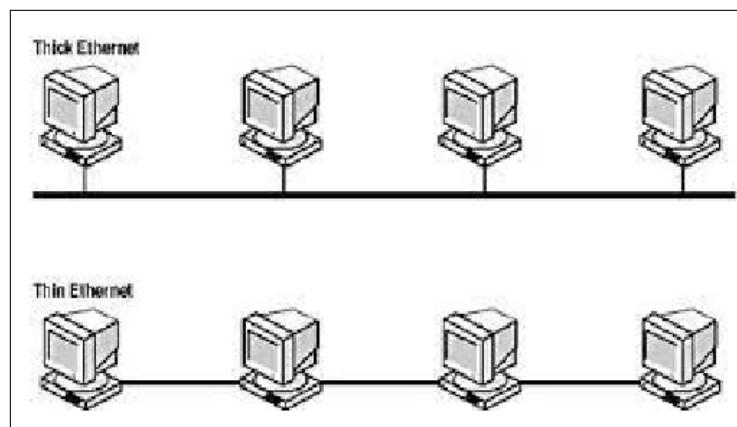


Figura 3.5: Cablaggio Thick e Thin Ethernet

I primi due metodi vengono anche detti Thick e Thin Ethernet (ossia a cavo grosso e cavo sottile) e usavano cavo coassiale con connettori particolari che congiungevano i nodi al cavo di trasmissione. In queste reti ciascun nodo può ricevere o spedire, ma non può fare entrambe le cose contemporaneamente, quindi la rete funziona in modalità half-duplex.

A partire da 10BaseT, usando il doppino (in cui i canali per trasmettere e ricevere sono sdoppiati) come mezzo trasmissivo, la topologia diventa a stella, usando come nodo centrale un hub: ed è proprio nell' hub che possono avvenire le collisioni in questa configurazione. Topologia a stella viene usata anche in 10BaseF, dove si usa una coppia di fibre per trasmettere e ricevere i dati.

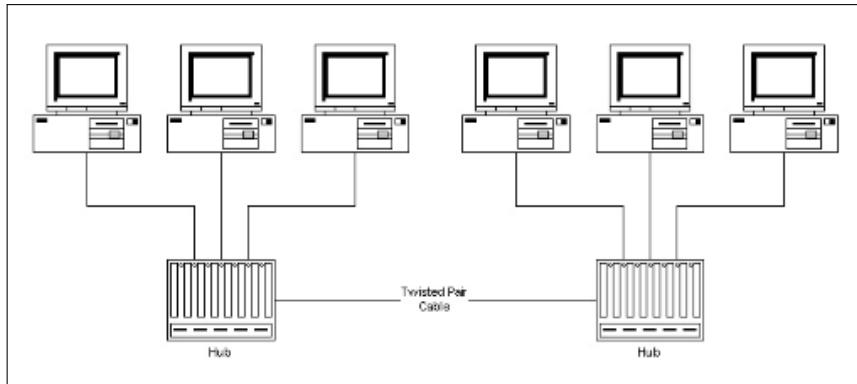


Figura 3.6: Cablaggio 10BaseT Ethernet

Prima di evolversi a velocità più alte, Ethernet standard ha subito diversi cambiamenti che hanno aperto la strada all'evoluzione delle reti come le conosciamo ora, primo tra tutti l'uso di **Bridge**: ciò ha portato all'aumento della larghezza di banda e alla creazione di domini di collisione. Un bridge divide la rete in una o più parti ognuna delle quali è a tutti gli effetti una rete indipendente, per quanto riguarda la larghezza di banda. È chiaro che dividendo una rete grande in più sottoreti, ognuna di essa, a meno che non debba comunicare con un nodo presente in una delle altre, deve controllare solo le comunicazioni afferenti alla propria sottorete, con minori nodi presenti e quindi meno possibilità di collisione.

Estremizzando il concetto di bridge per ogni nodo presente sulla rete si è arrivati al concetto di **Switch**, ossia un dispositivo che ha un solo nodo su ogni singola porta. Gli switch (di livello due) hanno anche hardware specializzato (come i buffer su ogni porta) nella gestione veloce dei frames. L'introduzione degli switch è stato uno stadio importante nell'evoluzione verso Ethernet veloce.

Con questo tipo di configurazione ogni nodo è collegato all'altro tramite una coppia di collegamenti (un cavo riceve e l'altro trasmette), consentendo un tipo di comunicazione di tipo full-duplex e quindi, di fatto, aumentando la velocità da 10 a 20 Mbps. Sempre per lo stesso motivo (i due cavi separati per Tx e Rx) non c'è più possibilità di collisioni e quindi non si necessita di CSMA/CD e il lavoro sottostrato MAC risulta alleggerito.

Nella versione Fast Ethernet, i cavi usati sono Twisted Pair (100BaseT) o fibra (100BaseS) e la topologia è di tipo a stella o punto-punto. Per quanto riguarda la codifica, essa non è più Manchester (richiederebbe un canale con larghezza di banda troppo elevata rispetto a quella possibile su cavi di cat5e, con i quali è nata Fast Ethernet) ma usa MLT-3 (che però non è autosincronizzante) e 4B/5B (per fornire sincronizzazione). Più o meno succede nelle reti 100BaseS anche se al posto di MLT-3 viene usata NRZ-I (anche essa non autosincronizzante).

Per quanto riguarda lo strato fisico delle reti Gigabit, esso è più complicato di quelle precedenti e non scenderemo nei particolari. Esse possono utilizzare topologie punto-punto, a stella oppure gerarchica mettendo in cascata tra loro più switch. Le loro implementazioni possono essere a due cavi in fibra (1000BaseLX - 1000Base SX) oppure a 4 cavi in rame (1000BaseTX). Anche le codifiche cambiano, riproponendo il problema di dualità tra sincronizzazione e rappresentazione efficiente.

3.3 LAN Wireless: IEEE 802.11 - Bluetooth

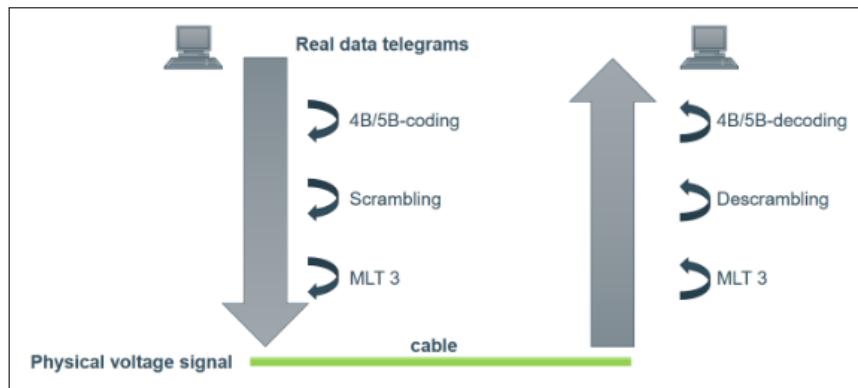


Figura 3.7: 00BaseTX Encoding-Decoding

Le reti senza fili differiscono per alcuni aspetti (ad esempio una gestione dell' energia più attenta) da quelle cablate, pur condividendo alcuni aspetti (framing, controllo degli errori, etc.). La potenza di trasmissione costituisce un problema rilevante per le connessioni wireless: non è possibile trasmettere onde radio a potenza arbitrariamente elevata a causa di leggi che regolano problemi di interferenza, e inoltre dispositivi mobili o sensori hanno problemi di durata della batteria non potendo accedere direttamente a sorgenti di energia. Inoltre il controllo dell' accesso al mezzo è un problema cruciale delle connessioni wireless:

- è difficile direzionare una connessione wireless verso un unico mittente
- è impossibile non avere interferenze da altri dispositivi in caso di ricezione, anche se questi usano standard differenti.
- è impossibile impedire ad altri dispositivi che si trovano nell'area di copertura di ascoltare in maniera "fraudolenta" trasmissioni non a essi diretti.

Name	Code	Standard	Data	Distance	Cable
Ethernet	10BASE-T	802.3i-1990	10 Mbits	100 m	Copper
Fast Ethernet	100BASE-T	802.3u-1995	100 Mbits	100 m	Copper
Fast Ethernet	100BASE-SX	802.3u-1995	100 Mbits	2000 m	Fibre
Giga Ethernet	1000BASE-T	802.3ab-1999	1000 Mbits	100 m	Copper
Giga Ethernet	1000BASE-LX	802.3z-1998	1000 Mbits	5 km	Fibre
10 Gigabit Ethernet	10GBASE-T	802.3an-2006	10 Gbits	100 m	Copper
10 Gigabit Ethernet	10GBASE-LR	802.3ae-2002	10 Gbits	10 km	Fibre
100 Gigabit Ethernet	100GBASE-LR4	802.3ba-2010	100 Gbits	10 km	Fibre
Terabit Ethernet	400GBASE-LR8	802.3bs-2017	400 Gbits	10 km	Fibre

Figura 3.8: Riepilogo Reti LAN Ethernet

Come si vede, alle problematiche tecniche si aggiungono leggi internazionali e nazionali per gestire al meglio la risorsa dello spettro elettromagnetico. In Italia tale gestione è demandata al PNRF (Piano Nazionale per la Ri-partizione delle Frequenze) che costituisce un vero e proprio piano regolatore dell'utilizzo dello spettro radioelettrico. Lo scopo del Piano è di stabilire, in ambito nazionale e per il tempo di pace:

- L'attribuzione ai diversi servizi delle bande di frequenze oggetto del piano.
- Di indicare per ciascun servizio, nell'ambito delle singole bande, l'autorità governativa preposta alla gestione delle frequenze, nonché le principali utilizzazioni civili.
- Di verificare l'efficiente utilizzazione dello spettro, al fine di liberare risorse per il settore televisivo e di gestire al meglio gli eventuali contenziosi con i Paesi frontalieri.

Il decreto ministeriale ora vigente ha rappresentato la conclusione di un iter che è durato anni ed ha consentito di recepire nella normativa nazionale alcune decisioni della WRC (World Radio Conference - Conferenza Mondiale delle Radiocomunicazioni), tra cui quelle riguardanti la radiodiffusione televisiva in tecnica digitale, nonché una serie di decisioni della Commissione Europea in merito alla gestione dello spettro radioelettrico. Abbiamo visto nella sezione precedente differenti modi di classificare le reti senza fili, e qui ci occuperemo dello standard WiFi e del Bluetooth.

3.3.1 IEEE 802.11 - WiFi

Wi-Fi è l'acronimo di Wireless Fidelity ed è sviluppato dalla IEEE (Institute of Electrical and Electronics Engineers) che stabilisce gli standard per il sistema Wi-Fi; in questo caso lo standard è l' 802 e il gruppo di studio generico 11 (da qui 802.11), suddiviso in vari sottogruppi identificati da lettere. Negli standard di rete Wi-Fi spesso si valutano i due parametri principali (ma no sono i soli) che sono:

- • Velocità: ossia la velocità di trasferimento dati della rete misurata in Mbps.
- • Frequenza: su quale radiofrequenza insiste la rete. Le bande di frequenza classiche per il Wi-Fi sono 2,4 GHz e 5 GHz, ma da poco tempo è previsto anche un canale sulle frequenze 6 GHz.

La banda da 2,4 GHz è una banda wireless comune e utilizzata anche da altri apparecchi come dispositivi Bluetooth, telefoni wireless, fotocamere, ecc. A causa della frequenza utilizzata da così tanti dispositivi, il segnale diventa rumoroso e la velocità rallenta. Quindi sono entrati in scena i canali sulla frequenza a 5 GHz, che fino a qualche tempo fa non erano comunemente usati: anch'essi oggi sono molto utilizzati da un numero crescente di dispositivi ma l'interferenza del segnale è ancora a livelli accettabili. Il 2,4 GHz trasmette i dati a una velocità inferiore rispetto a 5 GHz ma ha una portata maggiore rispetto a quest'ultimo. Il 5 GHz trasmette i dati a una velocità maggiore, ma ha una portata più corta in quanto ha una frequenza più alta. Gli standard Wi-Fi che si sono evoluti dal 1997 al 2021 sono diversi, e ne elenchiamo brevemente i principali:

- **IEEE 802.11** – È stato sviluppato nel 1997 e la sua velocità era di circa 2 Mbps.
- **IEEE 802.11a** – Questo standard è stato sviluppato nel 1999 e funziona nella banda ISM a 5 GHz. La velocità massima di 802.11a è di 54 Mbps. Questo standard è stato realizzato per evitare interferenze con altri dispositivi che utilizzano la banda a 2,4 GHz.
- **IEEE 802.11b** – Anch'esso creato anche con 802.11a nel 1999. La differenza è che utilizza una banda di frequenza di 2,4 GHz e la velocità è di 11 Mbps.
- **IEEE 802.11g** – Creato nel 2003, esso fondamentalmente, ha combinato le proprietà di 802.11a (la velocità di trasmissione) e 802.11b (il raggio di copertura, lavorando alla frequenza più bassa).
- **IEEE 802.11n** – Introdotto nel 2009, 802.11n funziona su entrambe le frequenze (2,4 GHz e 5 GHz), e gestisce singolarmente ognuna di esse. Può raggiungere una velocità nominale di trasferimento di circa 600 Mbps.
- **IEEE 802.11ac** – Questo standard è stato sviluppato nel 2013 e funziona solo su banda a 5 GHz. La velocità massima dichiarata è di 1,3 Gbps.
- **IEEE 802.11ax** – È la versione più usata di Wi-Fi tra quelle recenti, essendo stata rilasciata nel 2019. Funziona sia a 2,4 GHz che a 5 GHz per una migliore copertura e una migliore velocità: massima dichiarata di 10 Gbps con un miglioramento di circa il 30-40% rispetto a 802.11ac.

Di recente la Wi-Fi Alliance ha annunciato il nuovo schema di denominazione per gli standard Wi-Fi: invece di usare nomi che potevano risultare complessi come "802.11b" ora si è deciso di aggiungere un numero alla parola Wi-Fi che ne individua la generazione, tipo "Wi-Fi 1". Ciò aiuterà i consumatori a essere facilmente comprensibili poiché 802.11 è difficile da comprendere.

Vecchio Standard	Nuovo Standard
IEEE 802.11b	Wi-Fi 1
IEEE 802.11a	Wi-Fi 2
IEEE 802.11g	Wi-Fi 3
IEEE 802.11n	Wi-Fi 4
IEEE 802.11ac	Wi-Fi 5
IEEE 802.11ax	Wi-Fi 6

Wi-Fi può lavorare in due modalità: Ad-hoc o a infrastruttura, e quest'ultima è senz'altro la più comune e usata. Di seguito elenchiamo i componenti base delle reti Wi-Fi:

- **Stazioni (STA)** - Le stazioni comprendono tutti i dispositivi e le apparecchiature connesse alla LAN wireless. Una stazione può essere di due tipi:
 - **Wireless Access Pointz (AP)** - I punti di accesso (AP) sono generalmente router wireless che formano le stazioni base o danno l'accesso a un'altra rete.
 - **Cliente** - I clienti sono workstation, computer, laptop, stampanti, smartphone, ecc.
- Ogni stazione dispone di un controller di interfaccia di rete wireless.
- **Basic Service Set (BSS)** - Un set di servizi di base è un gruppo di stazioni che comunicano tra loro utilizzando lo stesso canale fisico. Una BSS può essere di due categorie a seconda della modalità di funzionamento:
 - **BSS a Infrastruttura** - Qui, i dispositivi comunicano con altri dispositivi tramite punti di accesso.
 - **BSS indipendente** - Qui, i dispositivi comunicano in modalità peer-to-peer, ossia direttamente tra loro, senza passare per un AP. Tale metodo viene anche chiamato ad hoc.
- **Extended Service Set (ESS)** - È un insieme di tutti i BSS connessi.
- **Sistema di distribuzione (DS)** - Collega i punti di accesso in ESS.

Alcune funzioni base del livello MAC del Wi-Fi gestiscono i servizi come il controllo dell'accesso al mezzo, il supporto per il roaming, l'autenticazione e il risparmio energetico. IEEE 802.11 definisce due sottolivelli MAC:

- **Funzione di coordinamento distribuito (DCF)**: DCF utilizza CSMA/CA come metodo di accesso poiché la LAN wireless non può implementare CSMA/CD. Offre solo un servizio asincrono.
- **Point Coordination Function (PCF)** - PCF è implementato sopra DCF e gestisce delle finestre temporali che si ripetono periodicamente. Utilizza un metodo di accesso al polling centralizzato e senza contese: viene fissato un Point Coordinator, di solito l'Access Point, che concede a una stazione il diritto di trasmettere.

Il frame 802.11, oltre ai campi di indirizzamento, il campo dati e quello del controllo errori, contiene 9 sottocampi di controllo su cui ci soffermeremo. La figura seguente mostra la struttura di base di tale frame:

Il Frame Control (FC) - È un campo lungo 2 byte che definisce il tipo di frame e alcune informazioni di controllo di seguito indicate:

1. **Versione**: è un campo lungo 2 bit che indica la versione corrente del protocollo che per ora è fissata a 0.
2. **Tipo**: è un campo lungo 2 bit che determina la funzione del frame, ad esempio gestione (00), controllo (01) o dati (10). Il valore 11 è riservato.
3. **Sottotipo**: è un campo lungo 4 bit che indica il sottotipo del frame come 0000 per la richiesta di associazione, 1000 per il beacon.
4. **Verso DS**: è un campo lungo 1 bit che quando messo a 1 indica che il frame di destinazione è per DS (sistema di distribuzione).

5. **Da DS:** è un campo lungo 1 bit che quando impostato indica il frame proveniente da DS.
6. **Più frammenti:** è un flag di 1 bit che quando impostato a 1 significa che il frame è seguito da altri frammenti.
7. **Ritrasmetti:** è un campo lungo 1 bit, se il frame corrente è una ritrasmissione di un frame precedente, esso è impostato su 1.
8. **Power Mgmt:** è un campo lungo 1 bit che indica la modalità di una stazione dopo la trasmissione riuscita di un frame. Impostato a 1 il campo indica che la stazione entra in modalità di risparmio energetico. Se il campo è impostato a 0, la stazione rimane attiva.
9. **Più dati:** è un flag di 1 bit utilizzato per indicare al destinatario che un mittente ha più dati da inviare rispetto al frame corrente. Può essere utilizzato da un punto di accesso per indicare a una stazione in modalità di risparmio energetico che più pacchetti sono bufferizzati, oppure può essere utilizzato da una stazione per indicare a un punto di accesso dopo essere stato interrogato che è necessario un polling più frequente in quanto la stazione ha più dati pronti per la trasmissione.
10. **WEP:** è un campo lungo 1 bit che indica che è applicato il meccanismo di sicurezza standard di 802.11.
11. **Ordine:** flag di 1 bit che se impostato a 1 indica che i frame ricevuti devono essere elaborati in ordine rigoroso.

I campi del frame principale sono:

- **Durata/ID:** È un campo lungo 4 byte che contiene il valore indicante il periodo di tempo in cui il supporto è occupato (in μs).
- **Indirizzo da 1 a 4:** si tratta di campi lunghi 6 byte che contengono indirizzi MAC standard IEEE 802 (48 bit ciascuno). Il significato di ciascun indirizzo dipende dai bit DS nel campo di controllo del frame.
- **SC (Sequence control):** È un campo lungo 16 bit che consiste in 2 sottocampi, ovvero Numero di sequenza (12 bit) e Numero di frammento (4 bit). Poiché i frame del meccanismo di riconoscimento possono essere duplicati, viene utilizzato un numero di sequenza per filtrare i frame duplicati.
- **Dati:** È un campo di lunghezza variabile che contiene informazioni specifiche provenienti dai livelli superiori.
- **CRC:** è un campo lungo 4 byte che contiene una sequenza di rilevamento degli errori CRC a 32 bit per garantire che il frame sia privo di errori

Per accedere al mezzo condiviso, i nodi della rete WiFi possono usare il protocollo **CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance)** che, a differenza di CSMA/CD (Carrier Sense Multiple Access/Collision Detection) che si occupa delle collisioni dopo che si sono verificate (nelle reti cablate), previene le collisioni, quindi interviene prima che esse si verifichino. L'algoritmo CSMA/CA può essere descritto con i seguenti passi:

- Quando un frame è pronto, la stazione trasmettente verifica se il canale è libero od occupato.
- Se il canale è occupato, la stazione attende finché il canale non diventa libero.
- Se il canale è inattivo, la stazione attende un intervallo di tempo Interframe gap (IFG) e quindi invia il frame.
- Dopo aver inviato il frame, imposta un timer.
- La stazione quindi attende la conferma dal ricevitore. Se riceve la conferma prima della scadenza del timer, segnala una trasmissione riuscita.
- In caso contrario, attende un periodo di backoff e riavvia l'algoritmo.

MACA è stato proposto successivamente all' implementazione di CSMA/CA nelle reti WiFi, nel 1990 presso i laboratori Bell per risolvere il problema dei nodi nascosti ed esposti proprio di CSMA/CA + RTS/CTS. Si è sbarazzato della parte CS (ossia dell'ascolto del canale) e ha invece introdotto i **NAV (Network Allocation Vector)** come soluzione per evitare le collisioni. IEEE 802.11 ha deciso di implementare MACA-with-carrier-sense come facoltativo in DCF in 802.11-1997.

I passaggi nella trasmissione MACA sono:

- La stazione trasmittente attende un tempo pari allo spazio inter-frame distribuito (DIFS) ed emette una richiesta di invio (RTS) se il canale è libero.
- Dopo aver inviato RTS, viene inizializzato un NAV (RTS), in modo che nessun'altra stazione tenti di trasmettere.
- La stazione ricevente attende un breve inter-frame space (SIFS) ed emette un clear to send (CTS).
- Con il CTS viene inizializzato un NAV (CTS).
- Il mittente attende un SIFS e trasmette il proprio frame di dati.
- Alla ricezione del frame di dati, il ricevitore attende un SIFS e invia un frame di riconoscimento (ACK).
- Entrambi i valori NAV diminuiscono a 0 durante questo periodo di tempo.
- Le stazioni attendono un SIFS e un periodo di backoff prima di contendere il canale.

3.3.2 Bluetooth

Bluetooth è uno standard per la comunicazione voce e dati wireless a corto raggio. È una tecnologia WPAN (Wireless Personal Area Network) e viene utilizzata per lo scambio di dati su distanze brevi (di solito entro un raggio di 10 metri). Questa tecnologia è stata inventata dall' azienda Ericson nel 1994. Funziona nella banda radio senza licenza, ossia quella industriale, scientifica e medica (ISM) che, ricordiamo, riguarda le frequenze (nel range dei 2 GHz) che vanno da 2,4 GHz a 2,485 GHz. Il numero massimo di dispositivi che possono essere collegati contemporaneamente è 7 e fornisce velocità dati che possono arrivare fino a 50 Mbps nell' ultima versione. La tecnica di diffusione che utilizza è FHSS (Frequency-hopping spread spectrum). Una rete Bluetooth è chiamata piconet e una raccolta di piconet interconnesse è chiamata scatternet.

Bluetooth Version	Data rate	High Data Rate Traffic	Release Year
1.2	1 Mbit/s	721 Kbit/s	2003
2.0 +EDR	3 Mbit/s	>80 Kbit/s	2007
3.0 HS	24 Mbit/s	802.11 link	2009
4.0	24 Mbit/s	802.11 link	2013

Figura 3.9: Bluetooth Data Rates fino alla versione 4

Lo standard bluetooth definisce due tipologie di rete:

- **PicoNet:** la rete elementare dello standard. Essa contiene un nodo primario chiamato nodo master e sette nodi secondari attivi, chiamati nodi slave. Quindi, possiamo dire che ci sono un totale di 8 nodi attivi che sono presenti a una distanza di 10 metri. La comunicazione tra i nodi primari e secondari può essere uno a uno o uno a molti. La comunicazione possibile è solo tra master e slave (segnalazioni di tipo slave-slave non sono contemplate). Essa prevede anche fino a un massimo di 255 nodi parcheggiati (parked), questi sono nodi secondari e non possono partecipare alla comunicazione a meno che non vengano convertiti nello stato attivo, mettendo in stato di park uno che attivo già lo è.
- **ScatterNet:** viene realizzata utilizzando varie piconet. Uno slave presente in una piconet può fungere da master (o primario) in un'altra piconet. Questo tipo di nodo può ricevere un messaggio da un master in una piconet e consegnare il messaggio al suo slave nell'altra piconet dove agisce come master. Questo tipo di nodo viene definito nodo bridge. Una stazione non può rivestire il ruolo di master in due piconet della stessa scatternet.

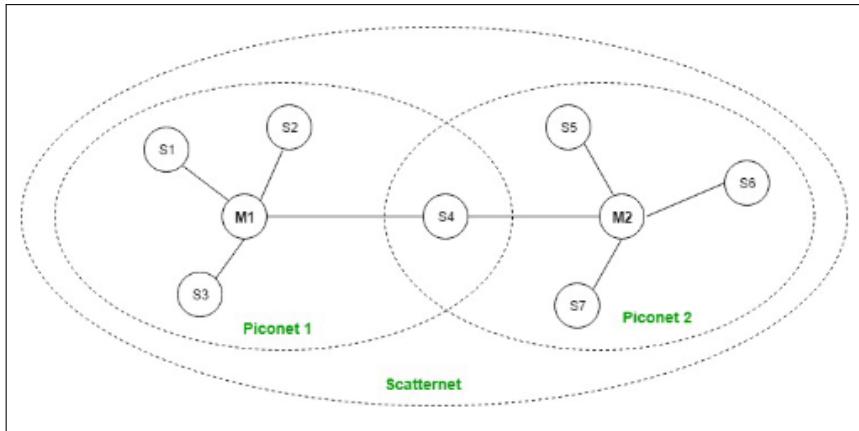


Figura 3.10: Piconet e Scatternet Bluetooth

L’interferenza a causa di altri standard che utilizzano le stesse frequenze è una delle maggiori sfide per qualsiasi tecnologia wireless per poter fornire una comunicazione affidabile: questi standard condividono il mezzo di trasmissione ed è possibile che un pacchetto dati possa essere danneggiato o perso se entra in collisione con un altro pacchetto trasmesso nello stesso momento e sulla stessa banda di frequenza. Una delle tecniche utilizzate da Bluetooth per superare le interferenze e trovare una trasmissione che eviti le collisioni è **FHSS**, ossia una trasmissione a spettro diffuso e salto di frequenza; in particolar modo viene usata un algoritmo di tale metodo di comunicazione chiamato **salto di frequenza adattivo (AFH)**. La tecnica consiste nel dividere la banda di frequenza in canali più piccoli (ad es. 40 canali nel caso di Bluetooth Low Energy) e passare rapidamente da un canale all’altro durante la trasmissione dei pacchetti. Per ridurre ulteriormente la possibilità di interferenze, il Bluetooth adatta la sua sequenza di hopping (cioè di salti tra i canali); in questo modo i canali rumorosi e occupati vengono tracciati dinamicamente ed evitati durante l’invio di pacchetti.

In precedenza abbiamo incontrato le collisioni e spiegato che una collisione si verifica quando due o più dispositivi trasmettono dati sullo stesso mezzo trasmittivo o canale radio, in periodi di tempo sovrapposti o, proprio in ambiente senza fili, quando diverse tecnologie radio si sovrappongono sulle stesse frequenze dello spettro radio.

La tecnologia Bluetooth mitiga il rischio di collisioni attraverso l’uso di tecniche a spettro esteso e in particolar modo con AFH. Ogni volta che si richiede un collegamento, una coppia di dispositivi ha l’opportunità di scambiare pacchetti a intervalli di tempo precisi, ma oltre a questo, all’inizio di ogni richiesta di connessione, si verifica un salto di frequenza, su un canale radio selezionato deterministicamente dall’insieme dei canali disponibili, utilizzando un particolare algoritmo di selezione del canale. Ogni dispositivo impegnato nella comunicazione passerà quindi al canale selezionato, durante tutto il periodo di connessione e a causa di una serie di eventi, cambiando frequentemente, e riducendo così, significativamente, le probabilità che si verifichino collisioni.

Il dispositivo master mantiene una mappa dei canali, classificando ognuno di essi come buono per la comunicazione o come inutilizzabile. La mappa dei canali viene condivisa con i dispositivi slave utilizzando una procedura a livello di collegamento, in modo che ciascuno di essi disponga delle stesse informazioni sui canali da utilizzare e quali evitare.

Bluetooth supporta due tipi di collegamento: un tipo, **chiamato SCO (synchronous connection oriented)**, utilizzato principalmente per la voce, e l’altro ACL (asynchronous connectionless) utilizzato principalmente per i dati a pacchetto.

Le coppie master-slave della stessa Piconet possono utilizzare diversi tipi di collegamento e il tipo di collegamento può cambiare arbitrariamente durante una sessione. Ciascun tipo di collegamento (di tipo full-duplex) supporta fino a sedici diversi tipi di pacchetto. Quattro di questi sono pacchetti di controllo e sono comuni sia per i link SCO che ACL.

Il collegamento SCO è simmetrico e in genere viene usato per il traffico vocale, dove non sono ammessi

valori alti di ritardo. I pacchetti SCO usano degli slot riservati per la comunicazione. Una volta stabilita la connessione, sia le unità master che quelle slave possono inviare pacchetti SCO indifferentemente. Il tipo di collegamento SCO supporta connessioni punto-punto e può raggiungere un data rate di 64 kbps.

Il collegamento ACL è orientato ai pacchetti e supporta sia il traffico simmetrico che quello asimmetrico. L'unità master controlla la larghezza di banda del collegamento e decide quanta larghezza di banda deve essere assegnata a ciascun slave della Piconet. Gli slave devono essere interrogati prima di poter trasmettere i dati.

Il collegamento ACL supporta anche messaggi di tipo broadcast del master verso tutti i nodi slave nella Piconet. In tali comunicazioni possono essere utilizzati pacchetti multi-slot che raggiungono velocità dati massime di 721 kbps in una direzione e 57,6 kbps nell'altra direzione se non viene utilizzata alcuna tecnica di correzione degli errori.

In ACL si usa lo schema di ritrasmissione automatica (ARQ), così, quando arriva un pacchetto, viene eseguito un controllo su di esso: se viene rilevato un errore, l'unità ricevente lo indica nel pacchetto di ritorno, e così la ritrasmissione viene effettuata automaticamente solo per i pacchetti difettosi. Per collegamenti che trasmettono dati real time (tipo voce), la ritrasmissione non è fattibile, quindi deve essere utilizzata una tecnica per migliorare la protezione dagli errori. Andiamo ora ad analizzare lo stack protocollare del Bluetooth.

- **Livello radio (RF):** specifica i dettagli dell'interfaccia radio, inclusa la frequenza, l'uso del salto di frequenza e la potenza di trasmissione. Esegue la modulazione/demodulazione dei dati in segnali RF. Definisce le caratteristiche fisiche dei ricetrasmettitori Bluetooth. Definisce due tipi di collegamenti fisici: senza connessione e orientati alla connessione.
- **Baseband Link layer:** La baseband è il motore digitale di un sistema Bluetooth ed è l'equivalente del sottolivello MAC nelle LAN. Si occupa della creazione della connessione all'interno di una piconet, dell'indirizzamento, definisce il formato del pacchetto, la temporizzazione e il controllo dell'alimentazione.
- **Protocollo Link Manager:** esegue la gestione dei collegamenti già stabiliti, incluso i processi di autenticazione e crittografia. È responsabile della creazione dei collegamenti, della loro integrità e della loro terminazione in caso di comando o errore.
- **Protocollo Logical Link Control and Adaption (L2CAP):** anche noto come il cuore dello stack del protocollo Bluetooth. Consente la comunicazione tra i livelli superiori e inferiori dello stack del protocollo Bluetooth. Impacchetta i dati ricevuti dai livelli superiori nella forma prevista dai livelli inferiori. Esegue anche la segmentazione e il multiplexing.
- **Service Discovery Protocol (SDP):** consente di scoprire i servizi disponibili su un altro dispositivo abilitato Bluetooth.
- **Livello di comunicazione RF:** è un protocollo sostitutivo della comunicazione cablata. È l'abbreviazione di Radio Frontend Component. Fornisce un'interfaccia seriale con WAP, OBEX e PPP. Fornisce inoltre l'emulazione delle porte seriali tramite il controllo del collegamento logico e il protocollo di adattamento (L2CAP). Il protocollo si basa sullo standard ETSI TS 07.10.
- **OBEX:** è l'abbreviazione di Object Exchange. È un protocollo di comunicazione per lo scambio di oggetti tra 2 dispositivi.
- **Wireless Access Protocol (WAP):** viene utilizzato per l'accesso a Internet e può essere esteso ai dispositivi Bluetooth attraverso lo stack protocollare PPP/IP/TCP.
- **Telephony Control Protocol (TCS):** fornisce il servizio di telefonia. La funzione di base di questo livello è il controllo delle chiamate (configurazione e rilascio) e la gestione dei gruppi per il gateway che serve più dispositivi.
- **Livello applicazione:** consente all'utente di interagire con l'applicazione.

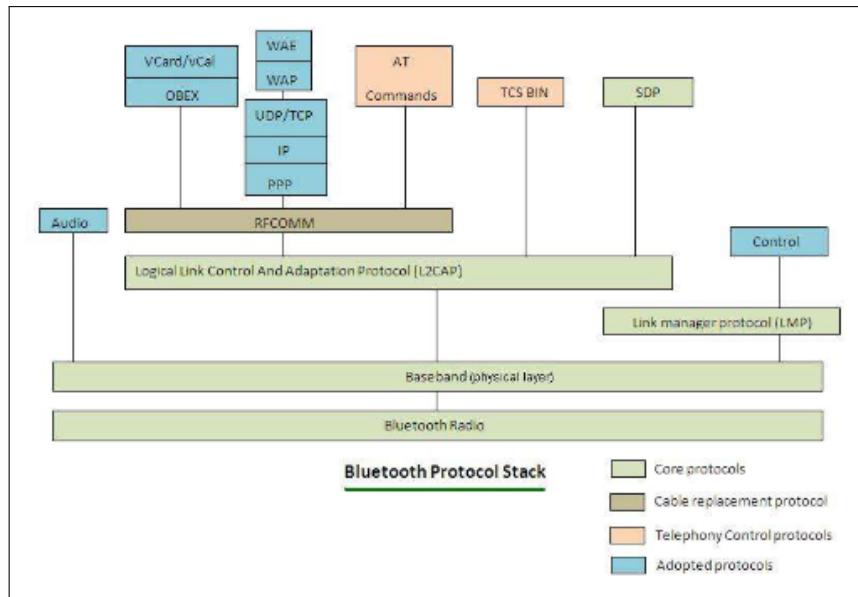


Figura 3.11: Bluetooth: stack protocollare

3.4 LAB - (Copertura Wireless - Potenza del Segnale)

Comandi di rete:

- ipconfig: Mostra la configurazione corrente della rete TCP/IP, come indirizzo IP, subnet mask, gateway predefinito e server DNS. Può anche essere usato per aggiornare i lease DHCP e rilasciare le configurazioni DHCP attuali.
- ping: Verifica la connettività a livello IP. Può essere utilizzato per controllare se un computer host può connettersi alla rete TCP/IP e alle risorse di rete. Può pingare vari indirizzi, inclusi l'indirizzo di loopback, l'indirizzo IP dell'host locale, l'indirizzo IP del gateway predefinito e l'indirizzo IP dell'host remoto.
- tracert: Determina il percorso da un host a un altro attraverso una rete. Mostra il percorso ottimale, non necessariamente quello effettivo, utilizzando il campo TTL IP e i messaggi di errore ICMP.
- pathping: Combina la funzionalità di ping e tracert. Fornisce latenza di rete e perdita di pacchetti per ogni router e collegamento nel percorso ed è utile per risolvere problemi di rete.
- netstat: Mostra le statistiche del protocollo e le connessioni TCP/IP attuali. Può essere usato con varie opzioni per visualizzare informazioni diverse, come tutte le connessioni, la tabella di route più le connessioni attive, le statistiche Ethernet, le statistiche per protocollo, e altro ancora. Utile per monitorare le connessioni di rete e identificare potenziali problemi.
- arp: Visualizza e modifica le voci della tabella ARP sul computer locale. ARP (Address Resolution Protocol) consente a un host di trovare l'indirizzo di controllo di accesso al supporto di un host sulla stessa rete fisica, dato l'indirizzo IP dell'host. Il comando arp è utilizzato per gestire e manipolare queste voci della tabella ARP.

Capitolo 4

Tecnologie di Accesso

4.1 Local loop e Vecchie Tecnologie

A livello globale, si prevede che il numero totale di utenti Internet crescerà da 3,9 miliardi nel 2018 a 5,3 miliardi entro il 2023 con un tasso di crescita del 6% in termini di popolazione, questo rappresenta il 51% della popolazione mondiale nel 2018 e il 66% della popolazione globale prevista entro il 2023. Sebbene la crescita del numero di utenti Internet sia una tendenza globale, si osservano variazioni regionali, con un tasso di crescita maggiore in Medio Oriente e Africa (circa il 10%).

A livello globale, i dispositivi e le connessioni stanno crescendo più rapidamente (ad un tasso del 10%) rispetto sia alla popolazione che agli utenti di Internet (rispettivamente del 1% e del 6%). Questa tendenza sta accelerando l'aumento del numero medio di dispositivi e connessioni per nucleo familiare e pro capite. Si pensi agli SmartPhone che ogni anno vengono introdotti e adottati sul mercato, in diversi fattori di forma, con nuove capacità aumentate e intelligenza artificiale.

Inoltre, un numero crescente di applicazioni M2M, come contatori intelligenti, videosorveglianza, monitoraggio sanitario, trasporti e tracciamento di pacchi o risorse, sta contribuendo in modo sostanziale alla crescita di dispositivi e connessioni. Entro il 2023, le connessioni M2M saranno la metà dei dispositivi e delle connessioni totali.

Le connessioni M2M saranno la categoria di dispositivi e connessioni in più rapida crescita, aumentando di quasi 2,5 volte durante il periodo di previsione, ossia fino a 14,7 miliardi di connessioni entro il 2023. Al secondo posto di tasso di crescita ci sono gli smartphone cresceranno al secondo posto più velocemente, aumentando di un fattore 1,4. Le SmartTV connesse (televisori smart, set-top box, adattatori multimediali digitali [DMA], e console di gioco) cresceranno del 6% circa, arrivando a 3,2 miliardi entro il 2023. I dispositivi connessi in decrescita (e rispetteranno tale tendenza) sono i PC, con un calo del 2,3

è importante tenere traccia del cambiamento della combinazione di dispositivi e tipologia di connessioni e della crescita da parte dei clienti nell'avere più dispositivi poiché influisce sui modelli di traffico. I dispositivi video, in particolare, possono avere un effetto moltiplicatore sul traffico. Un televisore HD connesso a Internet che fruisca di tre ore di contenuti un paio di volte al giorno genererebbe in media lo stesso traffico Internet di un'intera famiglia odierna.

L'effetto video dei dispositivi sul traffico ha un peso specifico' più pronunciato a causa dell'introduzione dello streaming video Ultra-High-Definition (UHD) o 4K. Questa tecnologia ha un tale effetto perché il bit rate richiesto per tale standard è circa 15-18 Mbps, più del doppio di quella richiesta dai video HD e nove volte superiore alla velocità in bit dei video a definizione standard (SD). Si stima che entro il 2023 i due terzi (66%) dei televisori installati saranno UHD, rispetto al 33% del 2018.

Ancora, una delle principali soluzioni che soddisfano le richieste della crescente domanda di connessione a Internet sfrutta da tempo le reti Wi-Fi. Con i progressi e le ratifiche di nuovi standard Wi-Fi, ambienti densi con molti dispositivi connessi contemporaneamente edge devices IoT come possono essere aeroporti, centri commerciali, sanità, Smart City, stadi ecc, stanno diventando oggetto sempre più di soluzioni wireless. A livello globale, ci saranno quasi 628 milioni di hotspot Wi-Fi pubblici entro il 2023,

rispetto ai 169 milioni di hotspot nel 2018, quattro volte di più.

Tutto questo si spiega anche con la **legge di Metcalfe**. Un business digitale di successo è componente essenziale nella nuova società. Internet stesso è diventato un facilitatore per gli effetti di rete, poiché diventa sempre meno costoso connettere gli utenti sulle piattaforme presenti sul Web, e quelle in grado di attirarli in massa diventano estremamente preziose nel tempo. Inoltre, gli effetti di rete facilitano la scala. Man mano che le aziende e le piattaforme digitali acquisiscono nuovi utenti, vengono create nuove applicazioni e servizi, a loro volta in grado di attirare nuovi utenti. Nuove applicazioni e nuovi utenti generano la richiesta di nuovi e più performanti dispositivi...e così via.

La rete telefonica pubblica commutata, o PSTN, è l'insieme mondiale di reti telefoniche pubbliche tra loro interconnesse e principalmente usata per veicolare dati voce. Tradizionalmente essa era una rete a commutazione di circuito. Queste reti forniscono l'infrastruttura e i servizi per le telecomunicazioni pubbliche e negli ultimi anni la situazione sta cambiando rapidamente con l'introduzione delle fibre ottiche e della tecnologia digitale.

Partiamo dalla struttura della classica rete telefonica per vedere come si è evoluta. Inizialmente, i telefoni venivano venduti in coppia e spettava al cliente collegare i cavi tra i due nodi. Si capì quasi subito che la situazione sarebbe diventata presto ingestibile, quindi Bell fondò la Bell Telephone Company che aprì il suo primo ufficio di commutazione a New Haven, nel Connecticut, nel 1878: per effettuare una chiamata, il cliente faceva squillare il telefono nell'ufficio della compagnia telefonica dove l'operatore collegava manualmente il chiamante al chiamato utilizzando un cavo jumper. Successivamente vennero collegati gli uffici di commutazione per rendere possibili le chiamate interurbane. Si sono quindi resi necessari uffici di commutazione di secondo livello; così la gerarchia è cresciuta fino ad arrivare a cinque livelli. Questo schema è rimasto sostanzialmente intatto per oltre 100 anni.

In sintesi, il sistema telefonico è costituito da tre componenti principali:

- **Local Loop (o ultimo miglio)**: formato ancora oggi da doppini intrecciati (rame), e quindi ancora analogici, e alcuni tratti in fibra ottica **FTTH(Fiber to the Home)**.
- **Trunks o Backbone (Dorsale)**: ormai quasi interamente in fibra ottica o microonde, trasmesse in tecnologia per lo più digitale.
- **Centrali di smistamento**: ormai trasformatesi in armadi e box di distribuzione.

Nel loro insieme le reti e le linee costituiscono la rete di accesso, e sono il mezzo di collegamento tra gli utenti e le centrali telefoniche e/o i nodi di fornitura a larga banda poste all'interno delle centrali tradizionali o distribuiti sul territorio.

Per capire lo sviluppo della rete iniziamo dalla parte interna al nodo cliente, che di solito è di proprietà dell'utente e collega i punti telefonici all'interno dell'edificio fino al Box dell'edificio dove arriva il cavo della Rete di Accesso messo a disposizione dal provider.

Il cavo telefonico della rete di Accesso Secondaria parte dal Box dell'edificio fino al Box Stradale, da cui, attraverso la rete di Accesso Primaria, arriverà fino al Giunto di Sfioccamento (ogni giunto gestisce fino ad un massimo di 800 doppini derivanti dai vari Box Stradali); da qui finalmente si arriverà alla centrale telefonica.

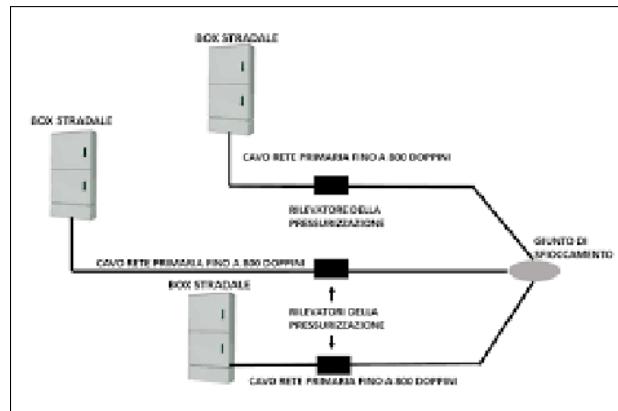


Figura 4.1: Parte della Rete Primaria in PSTN

La Metro Access Network è una rete su standard Ethernet. Viene comunemente utilizzata per connettere gli abbonati a una rete di servizi più ampia, come ad esempio Internet: gli accessi possono andare da 20 Mbps a 1 Gbps, a seconda della tecnologia usata:

- Basata su rame: è quella che ha servito tutta l'Italia fino a qualche anno fa e che sta man mano venendo sostituita dalla moderna rete in fibra ottica.
- Basata su fibra ottica: è la naturale evoluzione della rete precedente, a causa della sempre maggior richiesta di servizi ad alte prestazioni (streaming, gaming etc.). Questa tipologia di rete si divide in due principali architetture:
 - **TDM-PON:** l'implementazione di questa architettura si basa sulla divisione del periodo di trasmissione per ciascun utente su una singola lunghezza d'onda: TDM infatti sta per Time Division Multiplexing. Infatti, il multiplexing consiste nell'assegnare sequenzialmente (a turno per ciascuno di essi) l'intera banda a tutti gli utenti durante un fissato intervallo di tempo, denominato "Time Slots".

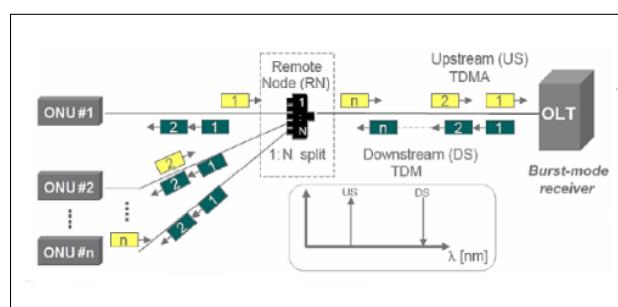


Figura 4.2: Il principio del multiplexing TDM-PON

- **WDM-PON:** a differenza del sistema precedente, si può pensare di fare multiplexing a divisione di lunghezza d'onda, in cui a ciascun abbonato viene assegnata una specifica lunghezza d'onda. Questo tipo di multiplexing ottico viene utilizzato nelle reti di accesso per aumentare la larghezza di banda, fino a una velocità dell'ordine di 10 Gbps.

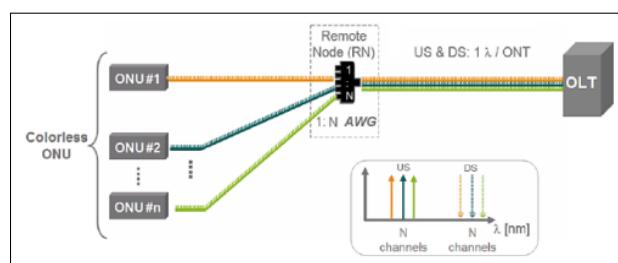


Figura 4.3: Il principio del multiplexing WDM-PON

4.2 xDSL

Il primo elemento che si interfaccia con la rete di accesso e si trova all' interno della casa del cliente viene generalmente chiamato **CPE (Customer Premises Equipment)**: per CPE si intende qualsiasi dispositivo di telecomunicazione presso l' utente, che viene utilizzato in sostituzione di un provider di servizi. Esempi comuni di apparecchiature usate presso le sedi del cliente includono ricevitori telefonici, TV via cavo e , per l'appunto, modem/router per l' accesso a Internet.

Un modem è un dispositivo elettronico , disponibile come case indipendente o come una scheda da inserire in un computer, che permette ai dati digitali di circolare (ricevere e inviare) su un canale analogico . Esegue la modulazione: codifica di dati digitali in segnale analogico, che generalmente è una frequenza portante modulata. L'operazione di demodulazione esegue l'operazione inversa e consente al ricevitore di ottenere l'informazione digitale. Nei modem vengono utilizzati diversi tipi di modulazione:

- modulazione di ampiezza (AM Amplitude Modulation).
- modulazione di frequenza (FSK, Frequency Shift Keying).
- modulazione di fase (PSK, Phase Shift Keying.).
- modulazione combinata di ampiezza e fase (QAM, Quadrature Amplitude Modulation).

Alcuni termini che riguardano la trasmissione sono bps, baud e banda passante. Baud e Bps (bit al secondo) non sono la stessa cosa, anche se spesso vengono confusi dai profani. Bit al secondo è la misura di quanti bit di dati vengono trasmessi ogni secondo su un canale di comunicazione. Il baud è invece, quando il segnale è bivalente (cioè usa solo due segnali), il numero di simboli trasmessi per unità di tempo: ad esempio un modem che trasmette in modalità 4-QAM ha una velocità di 1200 Baud, ma di 2400 bps (perché trasmette 2 bit per simbolo).

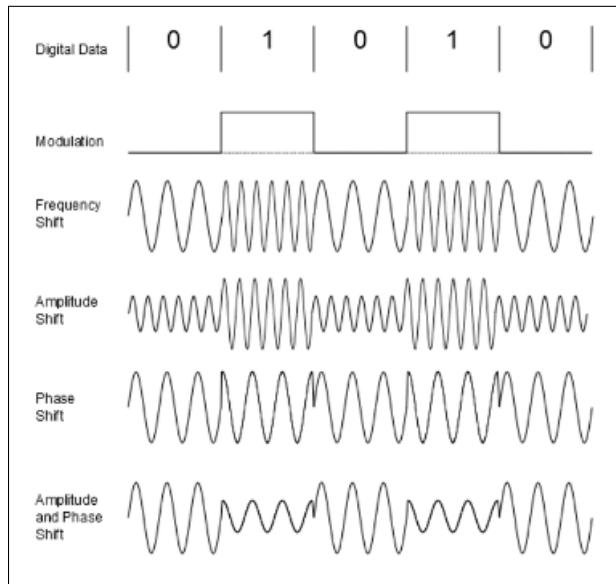


Figura 4.4: Modulazioni

I primi modem usavano la banda passante, cioè un range di frequenze in cui passava il segnale, detta **fonica** (da 300 Hz a 3400 Hz, la banda di frequenze udibile dall' orecchio umano) per trasmettere i dati. Tale banda permetteva un passaggio di 1200 baud. Poi con l'evoluzione della tecnica si è passati a 2400, 4800, 9600, fino a 56 Kbps con lo standard **V90**, anche se i baud trasmessi sono sempre 1200: questo è stato possibile aumentando le informazioni di ogni singolo baud che è passato da un bit fino a raggiungere i teorici 46,6 bit per baud.

Lo standard V dei modem è iniziato con la trasmissione in modulazione QAM (che su linea telefonica usa una variante che prende il nome di CAP). Ciascun tipo di modulazione QAM è caratterizzato da un diagramma sul piano complesso (detto **costellazione**), su cui sono rappresentati tutti i possibili stati

della portante (le legittime posizioni del vettore).

La modulazione **DMT**, usata nei modem xDSL, consiste nel suddividere la banda disponibile in un alto numero di sottocanali di uguale dimensione. Nella prima versione, **ADSL**, la banda di 1104KHz viene suddivisa in 256 sottocanali, ciascuno di ampiezza di banda pari a 4,3125 KHz. I primi 6 sottocanali vengono impiegati per la fonia. In realtà ogni sottocanale ha un'ampiezza effettiva di 4KHz. La differenza da 4,3125KHz viene considerata come banda di guardia per evitare che i sottocanali possano sovrapporsi, dopo l'azione dei filtri non ideali, con inevitabili interferenze che provocherebbero errori di comunicazione.

L'allocazione spettrale dei sottocanali avviene con la tecnica FDM (Frequency Division Modulation). Nella realizzazione standard si destinano 32 sottocanali per l'upload (dall'utente all'ISP) e 218 sottocanali per il download (dall'ISP all'utente). I 32 sottocanali per l'upload sono allocati a partire dalla frequenza 25,875KHz fino a 163,875KHz. Il modem è realizzato in modo da trasmettere, per ogni sottocanale, da 2 a 15 bit al periodo in funzione del rapporto S/N: più questo è basso minore è il numero di bps che il modulatore DMT genera. Questo è il compromesso che si deve accettare se non si vogliono elevati tassi di errore di trasmissione. Rimanendo nella categoria dei protocolli asimmetrici, nel tempo questi si sono evoluti tentando di ottimizzare le proprie prestazioni, come l'ADSL2 e l'ADSL2+, fino ad arrivare alla **Very-high-bit-rate Digital Subscriber Line (VDSL)** che comprende a sua volta l'evoluzione VDSL2. Vediamo a grandi linee quali sono state le migliorie messe in campo man mano:

- ADSL2 utilizza un miglior sistema di modulazione del segnale rispetto alla normale ADSL, questo permette l'utilizzo più intensivo delle bande di frequenze disponibili il che si traduce immediatamente in una maggiore "portanza" e quindi in un incremento delle velocità di trasmissione.
- ADSL2+: ha semplicemente raddoppiato la banda a disposizione per il download, spostando il limite superiore da 1,1 MHz previsto per l'ADSL e l'ADSL2, a 2,2 MHz.
- VDSL: introduce la fibra ottica nell'infrastruttura fisica e lavora a frequenze più elevate, fino a 12 MHz, pur basandosi sul comune doppino telefonico in rame: la fibra ottica arriva fino all'armadio di strada e l'ultimo miglio (il collegamento fra l'armadio e l'unità immobiliare) resta in rame.
- VDSL2: è una tecnologia sviluppata a partire dalla meno recente VDSL con lo scopo di essere pienamente compatibile con l'esistente ADSL2+ e offrire velocità sensibilmente maggiori. VDSL2 separa i due versi di trasmissione con la tecnica FDD (Frequency Division Duplexing) e utilizza lo spettro compreso tra 8-30 MHz, definendo diversi profili utilizzabili a seconda degli scenari:
 - i profili a 8 MHz e 12 MHz sono solitamente utilizzati per linee più lunghe.
 - quello a 17 MHz trova impiego per utilizzi da cabinet.
- Enhanced VDSL2: così è stata definita un'estensione dello spettro del VDSL2 fino a 35 MHz, con lo scopo di incrementare ulteriormente la velocità raggiungibile sulle reti in rame, preservando la compatibilità in scenari FTTC con il profilo VDSL2 a 17 MHz.

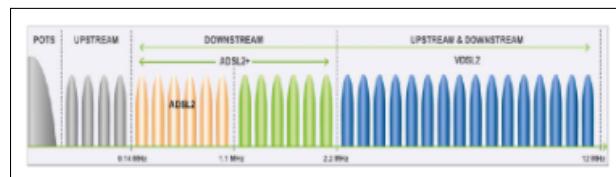


Figura 4.5: Confronto tecnologie xDSL

4.3 Tecnologia d'Accesso in Fibra Ottica

Una rete generica di accesso fisso **NGAN** (rete di nuova generazione), o FTTx (generalizzazione delle varie architetture di rete che utilizzano la fibra, come ad esempio FTTH, FTTB, FTTC, ecc.) è costituita da un insieme di apparati trasmissivi attivi e passivi che collegano il primo punto dell'operatore che eroga i servizi all'utente finale.

Spesso una rete in fibra viene indicata come OTN (Optical Transport Network) e con essa si persegue l'obiettivo di una Rete Tutta Ottica. Essa è composta da un insieme di elementi di rete ottici connessi da collegamenti in fibra atti a fornire funzionalità di trasporto, multiplazione, instradamento, gestione e supervisione di canali ottici che trasportano segnali. Per il trasporto di segnali cliente la rete definisce tre tipologie di contenitori **ODU (Optical Data Unit)**:

- la prima (ODU-1) adatta a trasportare segnali cliente con trasmissione fino a 2,5 Gbit/s;
- la seconda (ODU-2) per segnali da 2,5 a 10 Gbit/s;
- la terza (ODU-3) per segnali da 10 a 40 Gbit/s.

La gerarchia ottica consente, normalmente, il trasporto di quattro ODU-1 in un ODU-2 e di quattro ODU-2 in un ODU-3, ma sono possibili anche diverse altre combinazioni di canali.

Nel segmento di rete di accesso la larga diffusione e capillarità della classica rete di distribuzione in rame, sviluppata per il servizio telefonico, ha indotto fino a oggi tutti gli operatori a sfruttare al massimo tale patrimonio infrastrutturale disponibile.

Tra le possibili architetture che fanno uso della fibra, quelle che si basano su soluzioni punto-multipunto **PON (Passive Optical Network)**, presentano alcuni vantaggi che le rendono convenienti rispetto alle soluzioni punto-punto:

- Affidabilità elevata rispetto a soluzioni a stella attiva con apparati attivi "in strada".
- Manutenzione semplificata.
- Riduzione del numero di fibre in centrale e del costo per utente.
- Possibilità di evoluzione mediante la sola sostituzione degli apparati terminali

Una rete PON è una rete di accesso caratterizzata dall'assenza di apparati attivi al di fuori delle sedi ove sono collocate le **OLT (Optical Line Termination)** e le **ONT-ONU (Optical Network Termination-Optical Network Unit)**. È in genere basata su topologie di rete ad albero, realizzate mediante l'uso di ripartitori ottici di tipo passivo, con architettura tipo quella mostrata in figura.

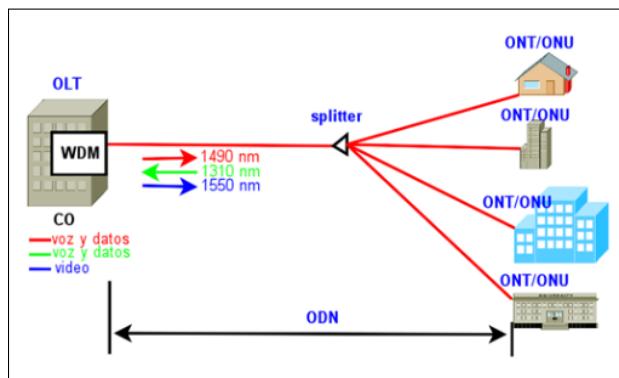


Figura 4.6: Architettura PON

Sul lato rete è presente la terminazione di linea ottica (**OLT**) che tipicamente si trova in un punto di raccolta, quale una centrale, e funziona da interfaccia condivisa tra tutti gli utenti connessi e la rete core. L'utente accede ai servizi offerti dalla rete tramite la terminazione di rete ottica (**ONT** o **ONU**).

Le OLT e le ONU sono connesse dalla rete di distribuzione ottica **ODN (Optical Distribution Network)** in configurazione punto-multipunto che può essere realizzata con uno o più livelli di diramazione e con splitter ottici disposti più o meno vicini alla OLT o alle sedi cliente, a seconda della disponibilità di fibra e delle strategie adottate dal gestore di rete; la rete di distribuzione ottica è totalmente passiva e la fibra ottica impiegata è tipicamente di tipo single mode.

Esaminando la figura (su riportata) della struttura generale della rete PON, tale modello può essere

applicato sia ad architetture di tipo FTTH (Fiber to the Home) (nelle quali si ha una ONT per ogni singolo cliente) sia ad architetture con un maggior grado di condivisione della terminazione ottica (ONU) quali FTTB (Fiber to the Building) o FTTC (Fiber to the Cabinet). È evidente che in questi due ultimi casi l'architettura di rete d'accesso potrà prevedere un parziale impiego di rete in rame, sfruttando così la capillarità di quest'ultima nel 86 tratto terminale e riducendo notevolmente la necessità di posa di nuova fibra. Per il drop su rame si possono utilizzare i sistemi trasmissivi ad alta velocità su rame della famiglia VDSL.

4.4 Fixed Wireless Access and Satellite

Nel concetto di **Fixed Wireless Access** rientrano una serie di tecnologie anche molto diverse e ogni operatore sceglie la strada che gli è più congeniale. Ad esempio possiamo trovare un'infrastruttura **FTTT (fiber to the tower)**, ovvero un collegamento in fibra ottica che arriva fino a una torre, da cui parte poi il segnale radio fino alle abitazioni. Ci sono poi i ponti radio in cui tutta l'infrastruttura si muove attraverso ripetitori e onde radio fino a una torre principale e da qui viene erogata alle abitazioni (che dovranno orientare un'apposita antenna in direzione della torre)

Tutti i sistemi FWA hanno comunque un modello comune: sono tipicamente costituiti da una stazione base collegata a una dorsale, e un numero di CPE, in dotazione agli abbonati, distribuite su un'ampia area. La stazione base (**BTS**) utilizza quindi le onde radio per comunicare con le unità degli abbonati, consentendo ai consumatori di connettersi alla rete fissa e accedere a servizi dati ad alta velocità. Le Base Stations sono fissate strategicamente a strutture fisse, di solito posizionate su altezze, come pali, edifici o torri.

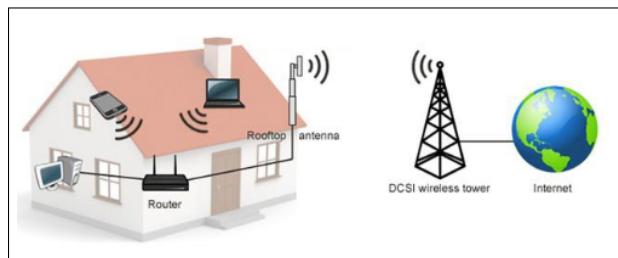


Figura 4.7: Architettura FWA

Ci sono diversi vantaggi e limitazioni che riguardano le reti FWA, essi includono:

- Benefici:
 - Colmare il divario digitale nelle aree meno servite.
 - FWA, lato cliente, non richiede una connessione alle infrastrutture via cavo.
 - L'installazione dell'infrastruttura è abbastanza semplice e non richiede particolari lavori, tipo scavi di trincee etc
- Svantaggi:
 - La distanza di trasmissione è limitata. Poiché le connessioni FWA si basano su tecniche a linea di vista, sono limitate alle aree vicine che riescono a scorgere la Base Station.
 - I fattori ambientali possono influire sulle prestazioni. Alcune connessioni FWA possono essere influenzate da ostacoli come alberi, edifici e path loss, che dipende dall'orografia del terreno.
 - Dipendenza dalla frequenza. Frequenze d'uso alte riducono la portata del segnale benché possano aumentare il throughput

La storia del FWA è iniziata a metà anni 2000 con la tecnologia HiperLAN2: essa aveva il vantaggio di essere una tecnologia “semplice” che consentisse di realizzare velocemente una rete radio senza bisogno di troppi servizi centralizzati e, per quei tempi, una buona efficienza spettrale (circa 1bit/Hz in condizioni reali). E' un protocollo semplice, non sincronizzato e senza alcuna gestione del canale radio. Ogni client che deve trasmettere qualcosa impiega immediatamente il canale radio, senza nessun sistema di prenotazione della risorsa trasmissiva. Questo sistema ha il beneficio che, se il canale è scarsamente

utilizzato, la latenza per trasmettere il dato è molto bassa (1-2ms), ma col crescere dell'occupazione del canale la latenza diventa imprevedibile, indeterministica e molto variabile.

Poi è stata la volta del **WiMax** che, nonostante la sua velocità di trasmissione dati, l'ampio raggio di copertura e l'affidabilità, non è mai riuscito veramente a decollare. WiMAX (Worldwide Interoperability for Microwave Access) lavora in una banda di frequenze compresa tra 3,4 - 3,6 GHz e ha due modalità operative:

- la *non-line-of-sight*, in cui la CPE collegata al computer comunica direttamente con la Base Station. La CPE può anche essere messa in casa (proprio grazie alla modalità non LoS e le basse frequenze) ma, se non posizionata in un breve raggio dalla BTS, lavora male e necessita montarla in modalità LoS.
- la modalità *line-of-sight*, viene utilizzata per le CPE distanti dalla BTS e per il trasferimento dei segnali in dorsale del tipo Torre-Torre.

Iniziando dal presente e nei prossimi anni assisteremo a un massiccio incremento delle offerte FWA grazie a una maggiore diffusione dello standard 5G. Il 5G sarà importante per le prestazioni che riesce a offrire, velocità di download/upload che possono eguagliare quelle di una rete in fibra FTTC.

Capitolo 5

Livello Rete

5.1 Problemi Architetturali dello Stato Rete

Lo strato di rete si occupa di interconnettere tra loro i diversi collegamenti punto-punto, in modo da poter mettere in comunicazione tra loro anche dispositivi geograficamente lontani. Le informazioni che viaggiano nei pacchetti sono trasmesse dal mittente alla destinazione attraversando tutti i nodi della rete che si trovano sul tragitto congiungente questi due, I pacchetti sono interamente ricevuti da ogni nodo prima di essere ritrasmessi: (**store and forward**).

La progettazione del livello di rete deve tener presente i servizi da fornire al livello superiore (livello di trasporto) rendendoli trasparenti il più possibile e tra loro omogenei. Ci sono due scuole di pensiero per realizzare quanto appena detto: coloro che sostengono che la rete debba fornire un servizio orientato alla connessione e chi invece dice il contrario. Quelli che sostengono la prima tesi, fanno riferimento all'esperienza delle reti telefoniche e vorrebbero che la rete fornisse un servizio affidabile, in cui la QoS è un fattore fondamentale: un rete senza connessioni non potrebbe fornire un servizio di qualità (soprattutto per applicazioni Real Time). La seconda corrente di pensiero fa invece capo a coloro che hanno sviluppato Internet nel tempo, per i quali l'unica preoccupazione del livello di rete dovrebbe essere quella di accettare e inoltrare i pacchetti, ognuno completo di indirizzo destinazione e mittente, perché potrebbe prendere vie differenti durante il tragitto (a causa degli algoritmi di routing), mentre operazioni quali l'ordinamento e il controllo di flusso non dovrebbero appartenere a questo livello in quanto già implementate a livello superiore.

Le **reti non orientate** alla connessione sono anche dette reti a **Datagram**, perché l'unità di informazione è proprio il datagramma che si muove tra i nodi della rete: come detto poco innanzi, esso incorpora sempre gli indirizzi del destinatario e del mittente, ognuno di 32 o 128 bit a seconda che si usi IPv4 o IPv6. In queste reti, inoltre, poiché ogni datagramma è inoltrato in maniera indipendente dagli altri, i nodi che si trovano sul percorso attraversato (router) non mantengono alcuna informazione sullo stato delle connessioni, rendendo al quanto difficile la Qualità del Servizio e un eventuale controllo della congestione. Di contro, in caso di malfunzionamento di un router, non ci sarebbero grossi problemi sulla connessione in atto, se non la perdita dei datagrammi presenti nel router al momento del malfunzionamento.

5.2 Protocollo IPv4 e Indirizzamento

Per trasferire i pacchetti sulla rete c'era bisogno di un protocollo che gestisse questa operazione individuando univocamente mittente e destinazione, così venne proposto **Internet Protocol**. IP è un protocollo connectionless e inaffidabile in quanto:

- tratta ogni pacchetto in maniera indipendente;
- non garantisce la consegna;
- non garantisce la correttezza dell'informazione trasportata;
- non richiede acknowledgment da alcun host, si tratti del destinatario o di nodi intermedi.

Ogni tipologia di rete ha una propria lunghezza massima di dati trasportabili da un frame, detta MTU (Maximum Transmission Unit), quindi può succedere che i dati provenienti dai livelli superiori debbano venire frammentati e trasmessi in pacchetti diversi.

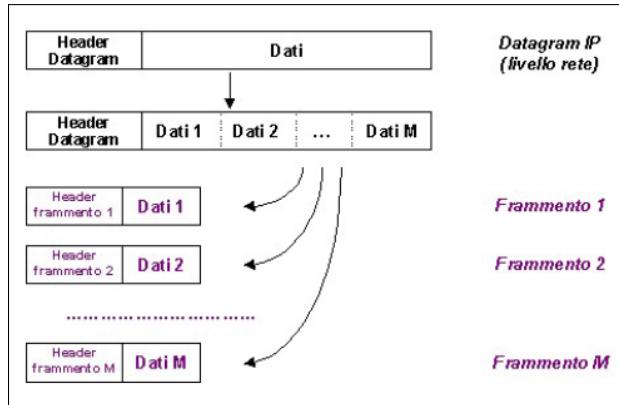


Figura 5.1: Frammentazione dei Pacchetti IPv4

Il fatto che IP sia un protocollo non orientato alla connessione e che un messaggio è suddiviso in un certo numero di pacchetti, implica che possa capitare che ciascuno di essi segua sulla rete un percorso diverso; in più, i pacchetti possono arrivare in un ordine diverso da quello in cui sono stati inviati, toccherà quindi a un protocollo di livello superiore rimetterli nell'ordine corretto.

5.2.1 Pacchetti IPv4

Il protocollo non solo definisce come i dati si spostano su Internet, ma anche come deve essere formata 'l'unità informativa' che effettivamente viene spostata su Internet: il pacchetto IP.

Un pacchetto IP è come un pacco una lettera con una busta che indica le informazioni sull'indirizzo, i dati contenuti al suo interno e attraverso quale altro protocollo di livello superiore essi devono essere interpretati. L'intestazione del pacchetto fornisce le informazioni necessarie per instradare il pacchetto verso la sua destinazione e in IPv4 può essere lunga fino a 24 bytes, con un minimo di 20 bytes: include, tra l'altro, l'indirizzo IP di origine, l'indirizzo IP di destinazione e le informazioni sulla dimensione dell'intero pacchetto.

L'altra parte fondamentale di un pacchetto IP è la componente dati, che può variare in termini di dimensioni. I dati all'interno di un pacchetto IP sono il contenuto che viene trasmesso. Il pacchetto IPv4 è allineato a 32 bit e la sua struttura, in particolar modo quella relativa al formato dell'header e delle sue componenti, può essere osservato in figura

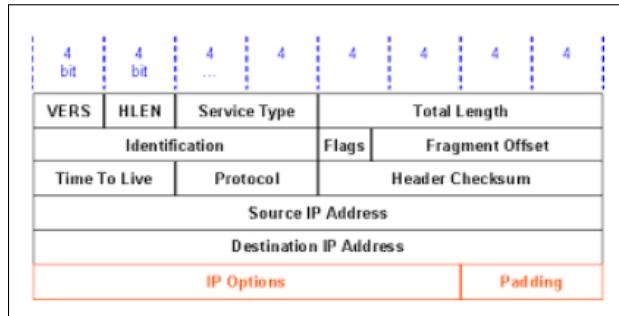


Figura 5.2: Intestazione IPv4

Vediamo ora i significati dei vari campi:

- **Version:** specifica la versione di IP utilizzata. La versione corrente del protocollo IP è di 4.
- **Length:** indica la lunghezza dell' intestazione del datagramma, misurata in parole di 32 bit.

- **Type of Service:** contiene cinque sottocampi che sono utilizzati per la QoS; essi specificano il tipo di precedenza, il ritardo, la velocità di trasmissione e l'affidabilità desiderata per quel pacchetto. Le impostazioni predefinite per questi cinque sottocampi sono la precedenza di routine, il ritardo normale, la normale velocità di trasmissione e la normale affidabilità.
- **Total Length:** specifica la lunghezza del datagramma comprensiva di intestazione e dati (misurata in byte).
- **Identification:** contiene un intero univoco che identifica il datagramma
- **Flags:** sono 3 bit utilizzati per il controllo del protocollo e della frammentazione dei datagrammi:
 - **Reserved:** sempre settato a 0.
 - **Don't Fragment:** di default settato a 0; se settato a 1 il pacchetto non viene frammentato, ma se sulla sua strada incontra un router/host che non prevede la non-frammentazione, allora viene scartato.
 - **More Fragment:** se settato a 0 indica che il pacchetto non è frammentato o è l'ultimo frammento del pacchetto originario; pertanto tutti gli altri suoi frammenti hanno il bit MF settato a 1.
- **Fragment Offset:** indica lo scostamento di questo frammento nel datagramma originale misurato in byte.
- **Time To Live:** è il tempo di vita (TTL) del pacchetto, in modo da evitare la sua persistenza indefinita sulla rete nel caso in cui non si riesca a recapitarlo al destinatario. Esso inizialmente misurava i "secondi di vita" del pacchetto, mentre ora rappresenta il numero di "salti" da nodo a nodo nella rete: ogni router che riceve il pacchetto prima di inoltrarlo ne decrementa il valore.
- **Protocol:** specifica il tipo di protocollo di livello superiore contenuto nel pacchetto; la lista dei codici di tali protocolli è mantenuta dalla IANA.
- **Header Checksum:** è un campo usato per il controllo degli errori dell'header.
- **Source/Destination IP Address:** indicano l'indirizzo IPv4 associato all'host del mittente/destinatario, formato da 32 bit suddiviso in 4 bytes.
- **Options:** informazioni facoltative e non molto usate, per usi più specifici del protocollo, come informazioni sui router o percorsi da seguire.

5.2.2 Indirizzamento IP

Siamo abituati a individuare un indirizzo IPv4 nella forma così come riportata di seguito:

208.67.220.220

Cioè composto da 4 numeri decimali separati da punti. Questa notazione testuale è pensata per renderne più semplice la lettura a noi umani. Internamente gli indirizzi IP vengono memorizzati in formato binario, cioè con una sequenza di 0 e 1.

La rappresentazione reale (come visto nell' intestazione del pacchetto IPv4 nel paragrafo precedente) viene fatta in binario ed è lunga 32 bit (divisi in 4 ottetti). Ad esempio l'indirizzo sopra corrisponde a:

11010000.01000011.11011100.11011100

Il numero totale di indirizzi IPv4 esistenti è 2^{32} , ossia 4294967296 (4 miliardi e 300 milioni circa), ma quelli realmente utilizzabili sono un po' di meno in quanto alcuni di essi sono riservati (ad esempio gli indirizzi 0.0.0.0, 127.0.0.1, 255.255.255.255, la classe 192.168.0.1/16).

Originariamente lo schema per individuare un indirizzo IPv4 era a classi (**classfull**); con questo schema l'indirizzo è ad autoidentificazione: per capire la classe a cui apparteneva bastava identificare i primi 4 bit dell'IP (i più significativi).

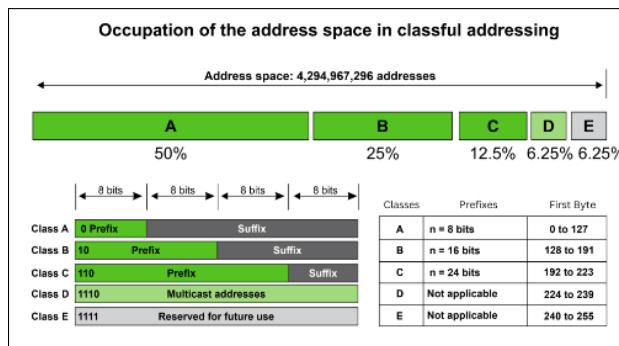


Figura 5.3: Indirizzamento Classfull

Quelle che si venivano a formare poi erano cinque classi nominate A,B,C,D,E e, in ciascuna classe, ogni indirizzo è stato diviso in due campi:

- l'indirizzo di rete (**netid**), che identifica la rete su cui si trova il computer e che costituisce numericamente una specie di prefisso;
- l'indirizzo del computer (**hostid**), che identifica il computer all'interno della rete e che costituisce numericamente una specie di suffisso.

L'indirizzamento a classi presenta diversi limiti dovuti soprattutto al numero di host gestibili dalle diverse classi. Se ad esempio si esauriscono gli indirizzi univoci resi disponibili da una classe occorre fare ricorso a un indirizzo di classe superiore, ma il cambiamento non è indolore.

Nella pratica, la crescita che negli anni ottanta ebbero le reti LAN ha portato all'esaurimento rapido degli indirizzi disponibili, quindi per risparmiare i prefissi di rete c'è stata necessità di escogitare altre tecniche come quella del mascheramento (**NAT**) per far sì che IPv4 continuasse ad adempiere al suo ruolo.

Un'altra tecnica entrata in uso e che ha permesso di mantenere in auge IPv4 anche con l'alto numero di dispositivi connessi che abbiamo ora è quella soprannominata classless, conosciuta anche con l'acronimo CIDR: consiste nel mantenere l'IP suddiviso in due parti, ma questa volta il numero di bit riservati alla rete e di quelli dedicati all'host viene fatta in maniera dinamica. La modifica introdotta da CIDR consiste essenzialmente nell'utilizzare maschere di sottorete di lunghezza arbitraria, al contrario dell'indirizzamento con classi che ne prevedeva solo tre. Un esempio che tutti abbiamo visto almeno una volta è la classica rete privata spesso utilizzata in casa, che in notazione CIDR possiamo descrivere con: 192.168.0.0/24

Il numero 24 si riferisce al fatto che i primi 24 bit (quindi i primi tre ottetti) identificano la rete mentre la parte rimanente (8 bit) l'host. Ovviamente il numero non è necessariamente un multiplo di 8, rendendo a volte non immediato capire dove finisce la rete.

La possibilità di dedurre la rete da un indirizzo IP è cruciale per il funzionamento del sistema di indirizzamento IP: il prefisso viene infatti utilizzato dai router per capire verso quale direzione un pacchetto deve essere inoltrato per raggiungere la destinazione (attraverso la maschera di sottorete e la procedura di ANDing con l'indirizzo IP esso suppone il gateway, che di solito viene posto al primo o all'ultimo indirizzo del range). Un'altra cosa da tenere presente è la differenza tra IP pubblici e privati:

- gli **indirizzi IP pubblici** vengono utilizzati quando si interagisce con Internet;
- gli **indirizzi IP privati** operano sulla rete locale (LAN).

Su una tipica rete con configurazione semplice, il router utilizza un indirizzo IP pubblico per identifierla in maniera univoca sulla rete Internet. All'interno di questa rete può funzionare una varietà di dispositivi diversi. Il router (o altro device che ne fa le veci) assegna a ciascun dispositivo su questa rete un indirizzo IP privato (anch'esso univoco) in modo che i dati possano essere inviati al dispositivo che li sta effettivamente richiedendo.

Un indirizzo IP pubblico è l'indirizzo IP esterno (rivolto cioè verso la rete pubblica) assegnato al router dal provider di servizi Internet; essi possono essere di tipo statico o dinamico:

- un **indirizzo IP statico** è un tipo di indirizzo IP che non cambia. Quando un IP statico viene assegnato a un dispositivo, tale indirizzo rimane tale nel tempo e questo consente ad alcuni servizi online di funzionare in modo più uniforme e funzionale;
- un **indirizzo IP dinamico** può cambiare se necessario, come ad esempio quando un dispositivo si connette a una nuova rete. La maggior parte delle reti domestiche utilizza indirizzi IP dinamici perché sono più semplici e più economici da assegnare per i Provider di servizi Internet; essi, inoltre, possono risultare più protetti e offrire maggiore anonimato.

Ritornando alla divisione tra indirizzi pubblici e privati, si può dire che tutti gli indirizzi IPv4 sono pubblici se non rientrano nella categoria degli IP speciali (che abbia visto prima) o nel range degli IP privati. Nell' intervallo degli IP privati non importa se lo stesso indirizzo è assegnato anche in altre reti private, poiché essi devono essere univoci solo all'interno della stessa rete locale.

I range di indirizzi IP privati sono i seguenti:

- Intervallo IP privato di classe A: 10.0.0.0 – 10.255.255.255
- Intervallo IP privato di classe B: 172.16.0.0 – 172.31.255.255
- Intervallo IP privato di classe C: 192.168.0.0 – 192.168.255.255

5.3 Protocollo IPv6 e Indirizzamento

Il protocollo IPv6 è stato definito negli anni 90, a causa dell' esaurimento degli indirizzi IPv4, e la sua diffusione è iniziata a metà degli anni 2000; trattandosi comunque di un **protocollo molto diverso da IPv4**, e con retrocompatibilità limitata, **la sua crescita è stata molto lenta**. Tra i vari miglioramenti apportati esso doveva prevedere soprattutto un numero maggiore di indirizzi. Se IPv4 prevede indirizzi lunghi 32 bit, **in IPv6 essi sono invece lunghi 128 bit, il quadruplo**, portando così ad avere uno spazio di indirizzamento estremamente vasto: si tratta infatti di 2^{128} indirizzi. La maggiore lunghezza ha portato anche a utilizzare il sistema esadecimale per la rappresentazione testuale, componendo l' indirizzo con un totale di 32 caratteri raggruppati in 8 "parole" o gruppi da 4 caratteri ciascuno, separate dal simbolo due punti. Un indirizzo IPv6 è effettivamente lungo e difficile da ricordare, ma c'è la possibilità di comprimerlo togliendo le parti poco utili. Possiamo ad esempio rimuovere tutti gli zeri all'inizio di ciascun gruppo (**leading zeros**) e comprimere le parti composte solo da zeri (non più di una volta nello stesso indirizzo). Come conseguenza, lo stesso indirizzo IPv6 può essere scritto in più modi diversi ma equivalenti.

L' indirizzo ip seguente:

2001:0db8:85a3:**0000:0000**:8a2e:0370:7334

può essere scritto ad esempio

2001:0db8:85a3::8a2e:0370:7334

Il fatto che ci siano così tanti bit a zero nell'indirizzo non compresso non è un caso, in quanto in IPv6 **la seconda parte di un indirizzo identifica infatti sempre l'host**, più precisamente chiamato **interface ID** perché lo stesso dispositivo può avere più interfacce di rete (es. WiFi ed Ethernet): esso è sempre lungo 64 bit e occupa perciò la seconda metà dell'indirizzo, dal bit 65 al bit 128 (una differenza rispetto a IPv4, dove la parte che separa rete e host è variabile); guardiamo il formato generico di tale indirizzo: Le tre parti principali della struttura dell' indirizzo le possiamo elencare come:

- I primi 3 bit rappresentano il range di indirizzi '**global unicast**' allocato da IANA (Internet Assigned Numbers Authority) in questo momento.
- Il **Subnet Prefix** (la "rete estesa"), che aggiunge al **Site Prefix** altri 16 bit fissi di sottorete (permettendo fino a 65 mila sottoreti per ogni rete, ossia per ogni organizzazione o "Site")
- L'**Interface ID** (l'Host), che occupa i rimanenti 64 bit, e che come abbiamo visto poc'anzi identifica la specifica macchina.

Dalla IANA è stato finora allocato 1/8 dello spazio di indirizzamento unicast IPv6, e precisamente quello che inizia per “001” (primo byte=001xxxxx, cioè gli indirizzi IPv6 = 2xxx::/3; quando finiranno si passerà a 3xxx::/3). In pratica questo significa che attualmente tutti gli indirizzi IPv6 pubblici iniziano con la cifra 2.

Esistono diversi tipi di indirizzi IPv6 consultabili a questo documento. Innanzitutto partiamo dal fatto che, come in IPv4, anche qui si usa la notazione CIDR, e una singola interfaccia viene indicata con il suffisso /128. Facciamo l’ esempio con un paio di indirizzi ‘particolari’: quello non specificato e quello di loopback. Il primo, che in IPv4 corrisponde allo 0.0.0.0 e cioè qualsiasi indirizzo, viene scritto come:

::/128 (Unspecified)

Mentre il secondo, il nostro ‘localhost’, ossia il server presente sulla macchina sulla quale stiamo lavorando e che in IPv4 corrisponde al 127.0.0.1, viene indicato come:

::1/128 (Loopback)

Altri indirizzi speciali in IPv6 sono:

- gli **indirizzi riservati**, pari ad 1/256 dell’intero spazio di indirizzamento (hanno il primo byte = 0); sono usati per scopi di ricerca dell’IETF e per ‘IPv4 address embedding’.
- gli **indirizzi locali** (i corrispondenti degli indirizzi privati in IPv4), che possono essere usati solo all’interno di ogni rete (o Site) e non vengono instradati verso l’esterno. Essi iniziano con i 9 bit: 1111 1110 1 (da FE8x::/9 a FEFx::/9) e sono anche detti ‘unregistered’ o ‘nonroutable’. Questo sottogruppo è diviso a sua volta in due categorie:
 - i **Link-local Addresses**, che vengono sempre bloccati dai Router, e sono quindi locali solo a un segmento di rete (switched LAN) o ad una subnet. Vengono usati per la “automatic address configuration”, usano le funzioni ND-Neighbor Discovery e per l’ARP. Hanno come decimo bit uno “0”, per cui cominciano con FE8x, FE9x, FEAx e FEBx
 - i **Site-local Addresses**, che possono essere instradati dai Router di una organizzazione solo all’interno della rete privata (Site), quindi tra le sue subnet, ma non verso Internet; iniziano con FECx, FEDx, FEEEx ed FEFx, avendo come decimo bit un “1”

In IPv6 l’idea è che ciascun dispositivo abbia assegnato almeno un indirizzo pubblico visibile su Internet. Per questo motivo un indirizzo di questo tipo è definito global unicast (GUA), indicazione che spesso si trova nella configurazione del proprio sistema operativo. Un esempio di indirizzo global è quello visto all’ inizio.

Il fatto che ogni dispositivo abbia assegnato un indirizzo IPv6 pubblico è una differenza fondamentale rispetto a IPv4, dove, a causa della penuria di indirizzi, molti dispositivi sono costretti a usare indirizzi privati e ricorrere all’ausilio del NAT; come visto sopra, però, in IPv6 l’uso degli indirizzi privati risulta utile per svolgere servizi che devono rimanere interni, come può avvenire ad esempio nelle reti aziendali.

5.3.1 Prefix Delegation e Subnetting

Per gli ISP che supportano IPv6, il metodo più utilizzato per la configurazione delle CPE (Customer Premises Equipment) consiste nell’uso del protocollo DHCPv6 con Prefix Delegation: un router dell’operatore delega al router del cliente la possibilità di gestire un prefisso IPv6, di creare delle sottoreti e di assegnare indirizzi pubblici a tutti i dispositivi in esse presenti. Secondo le raccomandazioni europee, nel nostro continente gli ISP dovrebbero delegare un prefisso statico /56 oppure /48 a ciascuna linea. Tali indirizzi prenderebbero la forma di

2001:db8:aaaa::/48

e

2001:db8:aaaa:1a00::/56

Abbiamo visto che l'interface ID di IPv6 occupa sempre la seconda metà dell'indirizzo, quindi il prefisso delegato può essere suddiviso in tante sottoreti (ad esempio /64). Prendendo in esame il caso di un prefisso /56 possiamo suddividere la rete locale in 256 sottoreti /64 ($2^{64-56} = 2^8$). Gli indirizzi assegnati ai dispositivi della rete vengono quindi scelti da una o più di queste sottoreti /64, permettendo di creare differenti sottoreti dedicate ad esempio a quella cablata, alla WiFi e alla rete ospite, permettendo scenari anche più complessi in ambienti aziendali; ognuna di esse (/64) permette di avere circa 18 miliardi di miliardi di indirizzi IPv6! Riferendoci ad esempio agli indirizzi sopra citati, le sottoreti /64 contenute nel prefisso sarebbero, in ordine:

2001:db8:aaaa:1a**00**::/64 2001:db8:aaaa:1a**01**::/64 . . . 2001:db8:aaaa:1a**fe**::/64 2001:db8:aaaa:1a**ff** ::/64

Tutti gli indirizzi IPv6 possono essere assegnati in maniera manuale o tramite protocollo DHCPv6; inoltre tutti i dispositivi che supportano IPv6 dispongono in automatico di un indirizzo link-local, attraverso il quale essi sono in grado di scoprire com'è fatta la rete in cui si trovano e di auto-configurarsi assegnandosi un indirizzo IPv6. Questo sistema si chiama **SLAAC** (Stateless Address Auto-Configuration) e non richiede di mantenere una lista di IP assegnati come farebbe un server DHCPv6. Gli indirizzi autoassegnati possono seguire due procedure:

- **assegnazione casuale**: come dice il nome stesso del processo l'indirizzo IPv6 viene generato in modo casuale;
- **sistema EUI-64**: l'interface ID viene generato a partire dall'indirizzo MAC della scheda di rete: questo è un indirizzo IPv6 facilmente riconoscibile perché contiene sempre al centro i caratteri ff:fe

Le versioni più recenti dei sistemi operativi implementano le **Privacy Extensions**, che consistono sostanzialmente nel cambiare periodicamente l'indirizzo auto-configurato per rendere più difficile il tracciamento e mantenere la privacy.

5.3.2 Altre Novità di IPv6

Nel rivedere il protocollo IP si è approfittato per fare anche ulteriori miglioramenti che non riguardano solo l'indirizzamento.

L'header dei pacchetti IPv6 è stato rivisto rimuovendo il campo checksum e prevedendo una lunghezza fissa per semplificare e velocizzare l'inoltro dei pacchetti sui router. L'header IPv6 è inoltre lungo il doppio rispetto a quello IPv4 (40 byte vs 20 byte) per via degli indirizzi più lunghi. L'header IPv6 è anche estensibile, cioè permette di definire funzionalità aggiuntive inserendo header 'a catena' come contenuto del pacchetto: essi però devono essere processati in modo sequenziale. Ogni Extensions Header è un intero multiplo di 8 ottetti per permettere l'allineamento naturale dei successivi. Un'altra novità è che IPv6 non include il supporto agli indirizzi di broadcast, ma prevede solo le tre macro-categorie di indirizzi: **unicast** (che abbiamo visto all'inizio), **anycast** e **multicast**; particolarmente fondamentali questi ultimi (ff00::/8).

Un indirizzo multicast viene infatti attribuito a un gruppo di interfacce di rete differenti (in linea di principio ognuna di esse appartenente ad altrettanti dispositivi distinti). I pacchetti inviati a un indirizzo multicast sono diretti a tutte le interfacce che fanno riferimento a questo indirizzo, ossia a quelle alle quali è stato attribuito.

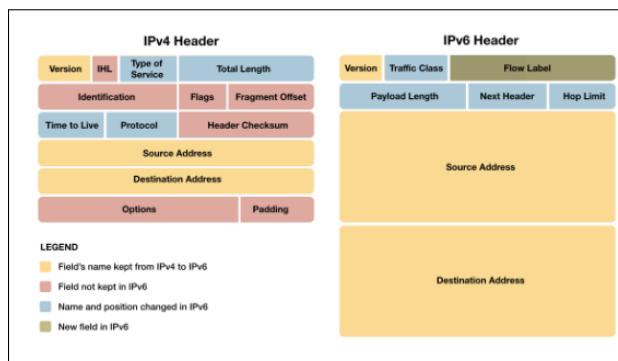


Figura 5.4: Confronto Header IPv4 e IPv6

Per quanto attiene agli indirizzi anycast, essi hanno le stesse caratteristiche di quelli unicast, ma vengono attribuiti a diverse interfacce di altrettanti nodi: lo scopo di un indirizzo anycast è quello di far sì che la rete consegna le informazioni al nodo del gruppo che risponde prima (ad esempio quello più vicino n base al protocollo di instradamento). Per la precisione, i pacchetti inviati a un indirizzo anycast dovrebbero raggiungere un'unica interfaccia di rete (la migliore in termini di costo).

Anche la sicurezza è stata curata in maniera migliore in IPv6 (ricordiamo infatti che in IPv4 essa è demandata a protocolli esterni che colmano il gap in questo campo): ci sono infatti due specifici extension header (abbiamo visto sopra di cosa si tratta) che forniscono servizi di questo tipo:

- **IP Authentication Header (AH)** ha il compito di fornire ai datagrammi IP integrità e autenticazione, ma non confidenzialità (questo per mantenere compatibilità con ambienti nei quali l'uso della crittografia è limitato). AH rende sicura la comunicazione tra due o più host, tra due o più gateway, tra host e gateway.
- **IP Encapsulation Security Payload (ESP)** a differenza del precedente questa modalità fornisce integrità, autenticazione e anche confidenzialità ai pacchetti. Come la precedente può essere usata per la comunicazione sicura tra due o più host, tra due o più gateway, tra host e gateway.

Sia AH che ESP possono essere utilizzati in due modalità: Transport Mode e Tunnel Mode; nel primo caso viene fornita protezione solo ai dati, mentre nel secondo caso viene fornita protezione all'intero datagramma IP.

Un concetto fondamentale sia per AH che per ESP è quello di Security Association (SA): una SA è una relazione, fra due entità in comunicazione, che identifica particolari condizioni di sicurezza.

5.4 Algoritmi di Routing

Nella sezione precedente abbiamo visto che il compito del router è quello di fornire due servizi principali:

1. Determinare il percorso ottimale
2. Trasportare le informazioni tra 2 reti diverse

Le reti sono spesso identificate con gli Autonomous System (AS), cioè in gruppi di network controllati e gestiti da fornitori di servizio Internet: i sistemi autonomi sono quindi di proprietà dei fornitori di servizi Internet (ISP). In altre parole, un AS è una rete individuale su Internet e ogni rete, o AS, ha un numero intero che la identifica (ASN), che viene assegnato dalla stessa autorità che rilascia gli indirizzi Internet. Tali numeri (ASN) sono quindi gli identificatori univoci rilasciati a un AS. L'identificatore consente a un AS di comunicare sia esternamente che internamente (in questo caso usa un ASN privato). I router che instradano i messaggi all'interno dello stesso AS sono spesso detti interior router e scambiano le informazioni tramite un protocollo, che è uguale su tutti i router, interno all' AS; mentre i router che instradano i messaggi tra AS diversi sono anche detti edge router o router di frontiera. In generale, ogni router mantiene in memoria una tabella di routing. Come minimo ogni entrata nella tabella deve contenere due elementi:

- Un **indirizzo di destinazione**: l' indirizzo IP del nodo di destinazione o della rete che lo ospita.
- Il **tipo d'indirizzo di destinazione**: può indicare che la destinazione è direttamente collegata al router oppure può indicare l'indirizzo di un altro router, anche questo collegato direttamente alla rete (next-hop router), ma che potrebbe portare a ulteriori reti.

Gli algoritmi di routing possono essere classificati in due grandi tipologie:

- **Statici**: in essi le tabelle di routing che vengono memorizzate sono predisposte da una persona (Network Administrator) e i valori di tali tabelle non cambiano per nessun motivo fino a quando il Network Administrator non li cambia.
- **Dinamici**: le tabelle vengono continuamente aggiornate e cambiate seguendo l' evoluzione della rete (caduta/inserimento di un nodo o di una network).

Per selezionare il percorso migliore, gli algoritmi scelgono secondo diversi criteri (**metriche**): per lunghezza del percorso, per banda passante, per affidabilità del link, per ritardo, per carico di rete. Tuttavia, due parametri universalmente accettati sono:

- **Hops**: numero di salti effettuati, cioè il numero di nodi attraversati lungo il cammino
- **Costo**: somma dei costi di tutte le linee attraversate (il costo di una linea è inversamente proporzionale alla sua velocità), e per costo si può intendere una qualsiasi delle metriche sopra elencate.

Proprio nella scelta del costo, ricade una delle caratteristiche tipiche degli algoritmi di routing, che si compongono di:

- **Ottimizzazione**: cioè l'abilità dell'algoritmo a scegliere la strada migliore (minimizzazione o massimizzazione della metrica).
- **Semplicità**: l'algoritmo deve essere funzionale ed efficiente con basso utilizzo delle risorse hardware.
- **Rapidità di convergenza**: quando ad esempio avviene un cambiamento nella rete, i routers distribuiscono, nel più breve tempo possibile, i messaggi di aggiornamento di tale evento affinché non si verifichino malfunzionamenti.
- **Scalabilità**: i routers devono adattarsi velocemente e accuratamente a una varietà di circostanze quali per esempio la caduta di una network: in questo caso i router devono scegliere il miglior percorso per tutti quei pacchetti che usavano la network caduta.
- **Robustezza**: a fronte di guasti hardware, uso intensivo delle risorse hardware e traffico elevato, l'algoritmo deve continuare a lavorare.

5.4.1 Distance Vector

Il primo algoritmo di routing dinamico che andremo a individuare è quello nominato Vettore delle Distanze (Distance Vector), basato sull'algoritmo di Belmann-Ford. L'idea base di questo algoritmo è che ogni nodo della rete mantenga al proprio interno una tabella che ne indica la distanza da ogni altro nodo della rete (il vettore delle distanze, per l'appunto). Supponiamo di inizializzare la rete alimentando tutti i nodi contemporaneamente (**cold start**) al tempo t_0 : in questo stato ogni nodo è caratterizzato da una conoscenza locale, ossia che esso sa solamente il proprio indirizzo e ignora totalmente la topologia della rete e la distanza dagli altri. La tabella di routing sarà quindi minima, caratterizzata da una singola voce: quella del nodo stesso. Prendiamo ad esempio una rete composta da 3 nodi e valutiamo cosa contiene il nodo A al 'cold start'.

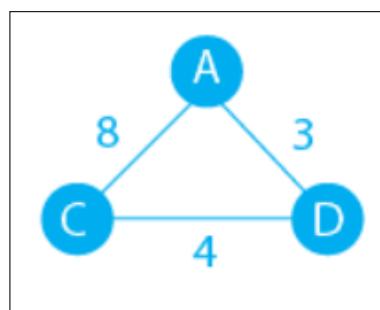


Figura 5.5: Rete Composta da Tre Nodi

Un vettore simile sarà presente sugli altri due nodi della rete, logicamente indicando rispettivamente C e D (al posto di A) in ognuno dei nodi. Successivamente, ognuno dei nodi invierà il proprio vettore delle distanze ai nodi a esso direttamente connessi; il vettore delle distanze presente nel nodo A sarà quindi il seguente

Ma cosa succede nel caso di un malfunzionamento su uno o più link? Praticamente viene fatto un merge dei vettori delle distanze tranne di quelli ricevuti dal link in errore, cosa può essere riassunta così:

- Il nodo scarta tutti i distance vector ricevuti da quel link
- Ricalcola la propria tabella mediante la fusione dei distance vector
- Distribuisce eventualmente il suo nuovo distance vector ai propri vicini

Il fenomeno della caduta di un link (o di un nodo) può innestare diversi problemi tipici di questo algoritmo come ad esempio:

- l' effetto rimbalzo (**Bouncing**) in cui, a causa di un possibile tempo di vulnerabilità tra l' indisponibilità del link e l' invio del distance vector del nodo che non sarà più disponibile, l'aggiornamento delle tabelle inizierà a far divergere l' algoritmo fermandosi solo al giungere del TTL dei pacchetti.
- La **divergenza**; si instaura un loop dovuto alla caduta di più link, ma, a differenza del precedente esempio, se i link caduti rendono effettivamente isolati i nodi non vi è alcuna possibilità per l'algoritmo di convergere ad uno stato stabile. Questo processo è anche conosciuto come conteggio all'infinito (**counting to infinity**) e può essere terminato solo attraverso una convenzione sulla rappresentazione di infinito con una distanza settata ad un valore maggiore del diametro della rete (ossia del percorso più lungo presente nella rete). Quando la distanza raggiunge questo valore, l' entry nella tabella di routing viene considerata infinitamente distante e, di conseguenza, irraggiungibile.

Da quanto detto, si capisce che l'effetto bouncing e il lungo tempo richiesto per il count to infinity sono effetti indesiderati a cui sono soggetti i protocolli di routing che fanno capo ai distance vectors: vediamo alcune tecniche (tra le molte studiate) per minimizzare questi effetti.

Lo **Split Horizon** è la principale di questi e si basa su una precauzione molto semplice: se un nodo (A) sta instradando pacchetti per una destinazione (Z) attraverso un nodo intermedio (B), non è logico che quest'ultimo tenti di raggiungere (Z) attraverso A, per cui non c'è ragione per cui (A) debba annunciare a (B) che (Z) ha una distanza più breve da A. Nella pratica ciò si realizza facendo in modo che i nodi inviano differenti versioni del vettore (per tenere conto del fatto che alcune destinazioni sono instradate attraverso i suddetti links in uscita) invece di diffondere lo stesso distance vector su tutte le porte del router.

La versione di **"Split horizon con poison reverse"** è più aggressiva: il vettore delle distanze dei nodi continua a includere tutte le destinazioni raggiungibili, ma essi imposteranno a infinito quelle distanze che sono instradate attraverso il link caduto; non si limitano cioè a escludere dal vettore le entries che sono per loro irraggiungibili, evitando di fatto che altri nodi possano erroneamente 'credere' di poter raggiungere quelle destinazioni per vie alternative.

5.4.2 Link State

La seconda tipologia di algoritmi di routing che vediamo è quella dello stato dei collegamenti (Link State). Questo tipo di algoritmi scambiano messaggi contenenti informazioni sullo stato dei collegamenti e consentono a ciascun router di apprendere l'intera topologia di rete: in tal modo ogni router riesce a calcolare la propria tabella di instradamento utilizzando l'algoritmo del percorso più breve.

Il principio di questa famiglia di algoritmi è semplice: non calcolano in maniera distribuita i percorsi migliori, bensì, ogni nodo mantiene una copia intera della mappa di rete, scambiando con ogni altro nodo della rete le informazioni sui suoi vicini, ed esegue un calcolo dei migliori percorsi per raggiungere gli altri. I concetti chiave per capire il Link State Routing sono tre:

- **Conoscenza dei nodi vicini:** al posto di inviare la propria tabella di routing, un router invia solo le informazioni sui suoi vicini. Ciascun router perciò invia a tutti i nodi della rete la sua identità e 'pesi' dei link che lo collegano direttamente ai suoi vicini.
- **Flooding:** ogni router invia le informazioni che detiene a tutti gli altri router della rete tranne a quelli a esso direttamente connessi; questo processo è conosciuto come flooding. Ogni router che riceve il pacchetto di informazione ne invia una copia ai propri vicini. A regime ogni router riceve una copia contenente le stesse informazioni.

- **Condivisione delle informazioni:** un router invia informazioni a ogni altro router solo quando vi è un cambiamento che modifica lo stato dei suoi link.

L'idea dei protocolli di routing 'link state' è di mantenere una copia sincronizzata della tabella sullo stato dei link in ogni nodi della rete. Abbiamo visto che per essere a regime è essenziale che tutte le copie presenti nei nodi siano identiche, altrimenti la coerenza degli instradamenti non potrebbe essere garantita. In realtà, durante alcune transizioni, potrebbero esserci dei nodi più aggiornati di altri; una situazione del genere potrebbe capitare, ad esempio, dopo un failure di un link.

L'algoritmo può essere suddiviso in quattro fasi:

1. **Inizializzazione:** ogni router della rete 'si informa' sui link che sono ad esso direttamente connessi. Le informazioni così apprese vengono inserite nella tabella del router
2. **Costruzione e distribuzione dei pacchetti:** Ogni router crea un pacchetto contenente la lista dei suoi vicini e i rispettivi costi di collegamento verso ciascuno di loro. Nell'intestazione del pacchetto, ogni router aggiunge la propria identità insieme a un numero di sequenza e un valore di durata (age): quest'ultimo viene utilizzato per garantire che i pacchetti non vaghino nella rete un periodo di tempo indefinito. Dopo il processo di costruzione, il pacchetto viene inviato nella rete tramite 'flooding', per raggiungere ogni nodo.
3. **Calcolo del percorso minimo:** ogni router usa le informazioni appena ricevute per creare un 'sink tree' e trovare il percorso minimo verso ogni nodo della rete; tipicamente viene applicato l'algoritmo di Dijkstra per trovare tali valori.
4. **Consolidamento delle route:** in quest'ultima fase i percorsi calcolati vanno a formare la tabella che poi verrà salvata all'interno del dispositivo: essa è formata da tutti i percorsi ottimi verso ogni nodo della rete.

I vantaggi (1 - 2) e gli svantaggi (3 - 4) di questa famiglia di algoritmi sono:

1. Hanno una convergenza abbastanza veloce in quanto ogni nodo ha bisogno di sapere solo lo stato dei link verso i nodi adiacenti, ed è molto importante nelle reti molto estese e con alto tasso di cambiamento di topologia.
2. Non soffre della problematica della divergenza, anche questo dovuto al fatto che le informazioni che si scambiano sono lo stato dei link circoscritti ai propri vicini.
3. È difficile da implementare correttamente in reti molto vaste in quanto non è scalabile come la famiglia del Distance Vector
4. Richiede un uso intensivo di risorse (memoria e tempo di CPU) per gestire le tabelle dell'intera rete e calcolare il percorso minimo a ogni cambiamento.

OSPF Uno dei più importanti algoritmi appartenente alla famiglia Link State è OSPF, acronimo che sta per Open Shortest Path First e ne esistono 3 versioni:

- OSPFv1: la prima versione, che ha gettato le basi per questo algoritmo molto usato, creato nel 1989, ma ormai quasi dl tutto sparito nella realtà.
- OSPFv2: in vigore dal 1998, usata nelle reti che hanno in uso IPv4.
- OSPFv3: l'ultima versione di OSPF, creata nel 2008, introduce l'uso di IPv6 pur rimanendo compatibile a IPv4.

I routers immagazzinano le informazioni riguardanti la rete in pacchetti di tipo Link State Advertisements (LSAs) e poi le organizzano in una struttura chiamata Link State Database (LSDB). I routers 'inondano' la rete con LSAs finché tutti i nodi della regione OSPF hanno la stessa mappa di rete. Semplificando quindi il processo di come lavora OSPF in 3 operazioni principali, possiamo scrivere:

- Il primo passo è quello di collegarsi con gli altri routers connessi sullo stesso segmento.
- Quindi ogni router scambia un Link State Advertisements (LSAs) con i suoi vicini.

- Infine, ogni router, in maniera indipendente, calcola il percorso migliore (attraverso l' algoritmo di Dijkstra) verso ogni altra destinazione. Fatto ciò inserisce i percorsi migliori nella tabella di routing.

Una rete OSPF, soprattutto se molto estesa, può essere suddivisa in sottodomini chiamati aree. Un'area è una raccolta logica di reti OSPF, router e collegamenti che hanno la stessa identificazione di area. I router all'interno di un'area devono gestire i dati riguardanti la topologia della sola area a cui appartiene; questo fa sì che le dimensioni del DB siano ridotte rispetto alla conoscenza dell' intera rete.

Le aree limitano l'ambito di distribuzione delle informazioni di route. Il database dello stato del collegamento (LSDB, Link State Database) dei router della stessa area deve essere sincronizzato e deve essere esattamente lo stesso, tuttavia è possibile inviare un riepilogo dei percorsi tra aree diverse attraverso router prestabili. Ogni rete OSPF suddivisa in aree diverse deve utilizzare le regole seguenti:

- Deve esistere un'area backbone, che collega un insieme di aree indipendenti in un singolo dominio.
- Ogni area non backbone deve essere connessa direttamente all'area backbone.
- L'area della backbone non può essere frammentata in parti non collegate tra loro in condizioni di errore (disconnessione di un router o di un collegamento).

5.5 Congestione

La congestione della rete è il fenomeno che si verifica quando i nodi sorgenti immettono nella rete più pacchetti di quanti essa riesca a consegnare a destinazione; oltre che per quantità di pacchetti presenti sulla rete, essa può avere molteplici cause come:

- diversi flussi attraversano la stessa linea e i pacchetti che arrivano a un router non riescono a essere processati, si forma quindi una coda nel router
- l'esaurimento dei buffer nei router innesca un meccanismo che porta allo scarto dei pacchetti successivi
- la lentezza della CPU del router (nell'analizzare i pacchetti, la tabella di routing, o per attività di logging degli eventi) può portare a rallentamenti nella risposta ed esaurimento della memoria

L'insorgere di congestione nella rete, può essere riscontrato tramite due parametri che sono il ritardo e il throughput, e in mancanza di contromisure il suo effetto peggiora nel tempo.

Il ritardo in una rete è minimo e, in condizioni normali, è dovuto solo al ritardo di propagazione e al tempo di processamento nei router; man mano che il traffico aumenta, anche il ritardo cresce a causa delle ultime due cause sopra esposte, innescando per di più la rispedizione dei pacchetti scartati.

Quando la rete lavora in condizione di carico normale, il throughput cresce proporzionalmente al traffico; se il traffico supera una percentuale piuttosto alta della capacità del canale (intorno all' 85-90%), allora il throughput comincia a diminuire (le cause sono le stesse elencate sopra).

Esistono diverse tecniche sia per prevenire che per mitigare la congestione quando essa si verifica, e vengono applicate sia a livello rete che a livello trasporto (ad esempio nel protocollo TCP). A livello rete esse vengono suddivise in due macrocategorie:

- **Proattive** (o a ciclo aperto)
- **Reattive** (o a ciclo chiuso)

5.5.1 Controllo Proattivo

Queste tecniche cercano di prevenire la congestione. Il controllo viene fatto o dalla sorgente o dalla destinazione.

Tecniche di Ritrasmissione È la politica in cui viene posto l'accento sulla ritrasmissione dei pacchetti. Se il mittente ritiene che un pacchetto inviato sia andato perso o comunque sia arrivato danneggiato, il pacchetto deve essere ritrasmesso. Particolare attenzione però deve essere posta nella ritrasmissione in quanto può aumentare la congestione sulla rete. Per prevenirla, i timer di ritrasmissione devono essere progettati con molta cura, ottimizzando al massimo l'efficienza (tali tecniche vengono molto usate anche a livello trasporto, TCP).

Gestione della Finestra Scorrivole Il meccanismo di finestra scorrevole usato dal mittente può a sua volta avere più o meno influenza sulla congestione: la ripetizione selettiva sicuramente invia un numero minore di pacchetti sul canale rispetto al Go Back N.

Politiche di Eliminazione dei Pacchetti Le politiche di eliminazione dei pacchetti all'interno dei router è senz'altro una tecnica che aiuta sia a prevenire che a moderare la congestione. A volte i router sono costretti a eliminare preventivamente i pacchetti per evitare la congestione (ad esempio è possibile eliminare alcuni pacchetti di una trasmissione audio che poi verranno ricostruiti per interpolazione dal destinatario).

Politiche di Accesso Anche meccanismi studiati per la QoS possono aiutare a gestire la congestione: è il caso delle politiche di accesso per reti a circuito virtuale (orientate alla connessione). In queste reti i router coinvolti nella comunicazione devono indicare la disponibilità o meno delle risorse richieste dal circuito virtuale (il canale di comunicazione) che mittente e destinatario vogliono instaurare. In caso di risorse scarse (possibile congestione) meglio che i router neghino l'accesso alla rete.

5.5.2 Controllo Reattivo

Questa tecnica viene messa in atto quando la congestione si è verificata e bisogna attuare un qualche meccanismo che sia volto a eliminarla o, quantomeno, ad attenuarla.

Segnalazione al Mittente L'azione più semplice che può essere intrapresa da un router che sta riscontrando congestione è quella di inviare un 'segnaletico' al mittente per far sì che esso rallenti la sua trasmissione. Eventuali altri router intermedi coinvolti nella comunicazione non saranno informati. I choke packets sono un meccanismo in cui il router invia direttamente il pacchetto di segnalazione al mittente: quando si accorge della congestione, per diminuire il carico, il router rimanderà indietro solo i pacchetti choke. Questa tecnica, se usata in modo non attento, può contribuire a far aumentare la congestione. Nella suite TCP/IP questo può essere fatto usando il protocollo ICMP e il meccanismo Source Quench, che notifica al mittente che i buffers del router sono sovraccarichi. Quando il messaggio viene ricevuto dal mittente esso riduce la sua finestra di invio verso la destinazione 'in difficoltà', limitando il traffico in uscita. L'abuso di questo meccanismo potrebbe portare a un DoS, e infatti, nella moderna internet non viene quasi mai usato.

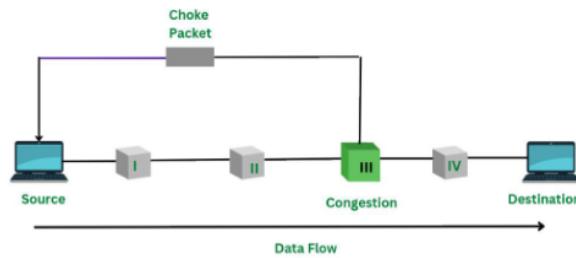


Figura 5.6: Segnalazione al mittente

Pressione all'Indietro Capita che la congestione non sia presente alla sorgente ma durante il tragitto. La pressione all'indietro è una tecnica in cui un nodo congestionato smette di ricevere pacchetti dal nodo che lo precede, ma ciò potrebbe spostare la congestione verso i nodi upstream (ossia nei nodi predecessori di quello che ha iniziato la backpressure). Il termine pressione all'indietro indica molto bene quello che succede, ossia la congestione si propaga, da nodo a nodo, nella direzione opposta al flusso di dati. Questa

tecnica può essere attuata solo in un circuito virtuale in cui ciascun nodo ha informazioni sul suo nodo che lo precede. Questa tecnica è conosciuta anche con il termine Choke packet hop-by-hop.

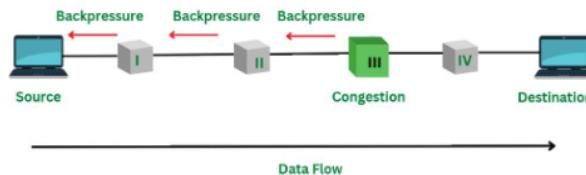


Figura 5.7: Pressione all'indietro

Segnalazione implicita Nella segnalazione implicita, non c'è comunicazione tra i nodi congestionati e la sorgente. La fonte suppone che ci sia una congestione in una rete in base ad alcuni eventi:

- quando il mittente invia diversi pacchetti e non viene ricevuta alcuna conferma prima della scadenza del timer;
- anche la ricezione di duplicati di conformi può essere indice di congestione nella rete.

Segnalazione esplicita Nella segnalazione esplicita, se un nodo riscontra una congestione invia esplicitamente un pacchetto alla sorgente o alla destinazione per informare della congestione. La differenza tra choke packet e segnalazione esplicita è che l'avviso di congestione fa parte dei pacchetti che trasportano i dati anziché crearne uno ad hoc come, per l'appunto, nel caso della tecnica choke packet. La segnalazione esplicita può avvenire sia in avanti che all'indietro:

- **Segnalazione diretta:** nella segnalazione diretta, l'avviso di congestione viene inviato nella direzione del flusso principale (quello in cui si riscontra il problema). La destinazione viene avvisata della congestione. Il destinatario in questo caso adotta politiche volte a prevenire ulteriori congestioni.
- **Segnalazione all'indietro:** nella segnalazione all'indietro, un avviso viene inviato nella direzione opposta al flusso dei dati. La fonte è avvisata della congestione e deve rallentare l'invio.

5.5.3 Leacky Bucket/Token Bucket

Due metodi per 'modellare' il traffico e ridurre la congestione sono l'algoritmo del **leaky bucket** e quello del **token bucket**. Il leaky bucket aiuta a controllare sia la quantità totale di traffico che la velocità con cui esso viene inviato alla rete, riuscendo anche a rilevare sia l'aumento graduale che i picchi repentinii di errori (e quindi di pacchetti scartati) nel buffer. L'algoritmo funziona in maniera simile al modo in cui un secchio bucato trattiene l'acqua: l'acqua rappresenta i pacchetti (che possono arrivare fino a una capacità massima) che entra dall'alto nel secchio (input) e fuoriesce dal buco sul fondo (output). Se il flusso dati in entrata fosse troppo alto e i pacchetti riempissero il secchio, quelli in eccesso verrebbero considerati "non conformi" e quindi scartati. I pacchetti ritornano ad essere ammessi nel secchio non appena esso ritorna ad avere spazio svuotandosi.

L'algoritmo del leaky bucket è ideale per livellare il traffico di tipo burst ('sprazzi' di traffico al massimo della banda). Proprio come un buco sul fondo di un secchio che fa fuoriuscire l'acqua a una data velocità costante, così l'algoritmo del leaky bucket fa lo stesso con il traffico di rete, accumulando i pacchetti all'interno del secchio e lasciandolo fuoriuscire a velocità controllata attraverso il buco sul fondo.

Il buco può essere visto come settaggio di un particolare data rate della rete: il contenitore che perde modello il traffico in entrata per garantire che sia conforme, in uscita, al data rate settato. Si può implementare l'algoritmo Leaky bucket con un buffer gestito in modalità FIFO: la coda trattiene i pacchetti e non consente loro di passare se superano la larghezza di banda settata sulla rete in uscita, tuttavia, se il traffico continua ad arrivare e il buffer è pieno, l'algoritmo scarterà i pacchetti appena ricevuti; questo fenomeno, noto come "tail drop", avviene indipendentemente dall'importanza del pacchetto o dal flusso a cui esso appartiene.

L'algoritmo del leaky bucket modella il traffico a raffica in traffico a velocità fissa calcolando la media della velocità dati ma non tiene conto del tempo di inattività di un host inattivo. Al contrario, l'algoritmo del token bucket consente all'host, durante l'inattività, di accumulare credito per il futuro sotto forma di token, superando così il limite dell'algoritmo del leaky bucket. Nel token bucket si possono inviare pacchetti anche alla massima velocità se ci sono abbastanza token a disposizione (che vengono aggiunti a un rate costante, fino a un massimo previsto dal bucket, e vengono tolti ogni volta che si spedisce). Ci sono dei contro anche per questo secondo algoritmo: se, a esempio, il bucket dei token è vuoto, i pacchetti dovranno attendere nuovi token, con conseguente aumento della latenza e potenziale perdita di pacchetti; inoltre è meno flessibile del leaky bucket quando si tratta di modellare il traffico di rete a causa della velocità fissa di generazione dei token che non può essere facilmente modificata per soddisfare i mutevoli requisiti della rete.

5.5.4 Load Shedding

Il **load shedding** o riduzione del carico è un'altra tecnica utilizzata per il controllo della congestione e interviene sui buffer del router. Sappiamo che i buffer sono utilizzati per conservare i pacchetti e poi instradarli verso la loro destinazione. La riduzione del carico è definita come un approccio che consiste nello scartare i pacchetti quando il buffer è pieno secondo una politica implementata a livello di collegamento. La scelta dei pacchetti da scartare è un compito importante e non può essere lasciato al caso, spesso ci viene in aiuto la conoscenza del traffico sulla rete:

- Priorità al più vecchio (**wine policy**): per il trasferimento di file, un vecchio pacchetto vale più di uno nuovo: questo perché eliminare un vecchio pacchetto potrebbe forzare la ritrasmissione di altri pacchetti, inoltre il file risulterebbe inservibile in caso di 'buchi' nelle informazioni. Per questo tipo di applicazioni il pacchetto 'più vecchio è, meglio è'.
- Priorità al più fresco (**milk policy**): per il traffico multimediale, un nuovo pacchetto è più importante di uno vecchio (pensate alla comunicazione telefonica digitale, VoIP). In questo caso, quindi 'più è nuovo il pacchetto, più esso è importante'.

Come abbiamo appena visto la scelta dei pacchetti da scartare nei router può essere fatta **in base alle applicazioni** che girano sulla rete ma anche **in base alla priorità**. Il mittente può contrassegnare la priorità del pacchetto ad esempio usando il campo ToS (Type of Service) all'interno dell'header IP. A seconda della priorità settata, il pacchetto può essere selezionato o scartato. La priorità può essere assegnata in base al costo, all'algoritmo o ai servizi a cui appartiene.

Random Early Detection Un altro metodo molto usato (a livello rete) per decidere quale pacchetto scartare in caso di congestione è il **Random Early Detection** (RED); in realtà esso ha un comportamento proattivo in quanto cerca di prevenire la congestione. I punti essenziali dell'algoritmo sono i seguenti:

- per rilevare le condizioni di congestione si utilizza la dimensione dei buffer di uscita;
- per prevenire la congestione si marcano i pacchetti in uscita. Rispetto ad altri metodi utilizzano questi strumenti RED introduce alcune migliorie che ne aumentano le prestazioni:
- Non si considera la dimensione istantanea della coda, ma una forma di media pesata. Questo consente di ignorare picchi episodici, e trattare solo picchi di traffico persistenti.
- In base all'hardware e ai flussi di dati vengono scelte due soglie, una minima e una massima.
- Quando la lunghezza media della coda supera una certa dimensione contrassegnata dalla soglia minima, alcuni pacchetti vengono marcati: i pacchetti marcati vengono scelti a caso, e in un numero che cresce con la lunghezza media della coda.
- Oltre una certa soglia, tutti i pacchetti sono marcati.
- La marcatura del pacchetto è un'operazione in un certo senso simbolica: a seconda della tecnologia del router tale marcatura può consistere:
 - nella rimozione del pacchetto
 - nell'impostazione di un bit o campo nel pacchetto

In ogni caso si intende che la marcatura avrà un effetto di feedback sul mittente del pacchetto

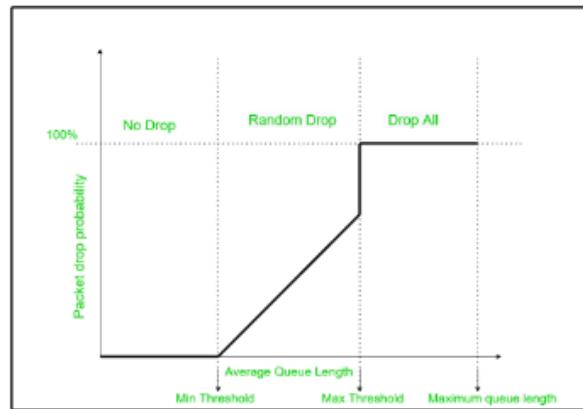


Figura 5.8: Random Early Detection

Capitolo 6

Internetworking

Fino a questo momento è stata implicitamente presupposta l'esistenza di una singola rete omogenea in cui ciascuna macchina utilizza lo stesso protocollo in ogni livello. Sfortunatamente questa premessa è esageratamente ottimistica; infatti esistono molte reti diverse, tra cui PAN, LAN, MAN e WAN. Abbiamo descritto Ethernet, le reti telefoniche mobili e fisse, 802.11, 802.16 e molte altre ancora. Inoltre esistono numerosi protocolli utilizzati comunemente in ogni livello. Nei paragrafi seguenti esamineremo con attenzione i problemi che sorgono quando si collegano tra loro due o più reti per formare una internetwork o, semplicemente, internet.

6.1 Protocolli di Controllo

Oltre a IP, utilizzato per il trasferimento dei dati, Internet ha diversi protocolli di controllo utilizzati nel livello di rete, tra cui ICMP, ARP, e DHCP. In questo paragrafo li esamineremo descrivendo le versioni che corrispondono a IPv4 in quanto sono quelle comunemente usate. ICMP e DHCP hanno versioni simili per IPv6; l'equivalente di ARP per IPv6 è chiamato NDP.

6.1.1 Internet Control Message Protocol (ICMP)

ICMP è un protocollo di rete utilizzato per inviare messaggi di errore e altre informazioni operative. Esistono due versioni principali:

- **ICMP per IPv4 (ICMPv4):** Utilizzato per inviare messaggi di errore (ad esempio, quando un pacchetto non può raggiungere la sua destinazione) e per funzioni di diagnostica come il comando "ping", che verifica la connettività tra due dispositivi.
- **ICMP per IPv6 (ICMPv6):** Include tutte le funzionalità di ICMPv4, oltre a supportare le operazioni di NDP e altre funzioni specifiche di IPv6.

6.1.2 ARP – Address Resolution Protocol

Anche se ogni macchina di Internet ha uno o più indirizzi IP, in realtà questi non possono essere utilizzati per inviare pacchetti, in quanto le schede di rete, come le schede Ethernet che operano a livello data link, non comprendono gli indirizzi Internet. Per esempio nel caso di Ethernet qualsiasi scheda di rete incorpora un indirizzo Ethernet a 48 bit. Ogni produttore richiede a IEEE un blocco di indirizzi per garantire che nessuna coppia di schede utilizzi lo stesso indirizzo (per evitare con itti se dovessero capitare sulla stessa LAN). Le schede di rete inviano e ricevono frame basati sugli indirizzi Ethernet a 48 bit e non sanno nulla degli indirizzi IP a 32 bit.

La soluzione migliore è: l'host 1 trasmette attraverso la Ethernet un pacchetto broadcast che chiede chi sia il proprietario dell'indirizzo IP 192.32.65.5, la trasmissione broadcast arriva a tutte le macchine della Ethernet CS e ognuna controlla il proprio indirizzo IP. Solo l'host 2 risponde inviando il proprio indirizzo Ethernet (E2). In questo modo l'host 1 scopre che l'indirizzo IP 192.32.65.5 è assegnato all'host che ha l'indirizzo Ethernet E2. Il protocollo utilizzato per fare questa domanda e ottenere una risposta si chiama **ARP** (*address resolution protocol*) e quasi tutte le macchine di Internet lo utilizzano. ARP è definito nell'RFC 826.

Utilizzare ARP al posto dei le di configurazione rende tutto più semplice. Il gestore del sistema non deve fare altro che assegnare a ogni macchina un indirizzo IP e stabilire le maschere di sottorete. ARP si occupa di tutto il resto.

Per permettere cambiamenti nelle associazioni, per esempio quando un host è configurato per utilizzare un nuovo indirizzo IP, mantenendo quello Ethernet, le informazioni nella cache ARP dovrebbero scadere in pochi minuti. Un modo più intelligente per tenere le informazioni nella cache e ottimizzare le prestazioni è che ogni computer trasmetta in broadcast la propria associazione durante l'accensione. Questa trasmissione broadcast è generalmente eseguita sotto forma di una ricerca ARP mirata al proprio indirizzo IP. L'operazione, che non dovrebbe ricevere alcuna risposta, ottiene l'effetto di inserire una voce nella cache ARP di tutti gli altri computer. Questa operazione è chiamata **gratuitous ARP** (ARP *gratuito*). Se (inaspettatamente) arriva una risposta, significa che due computer hanno ricevuto lo stesso indirizzo IP; l'errore deve essere risolto dall'amministratore di rete prima che le due macchine possano usare la rete.

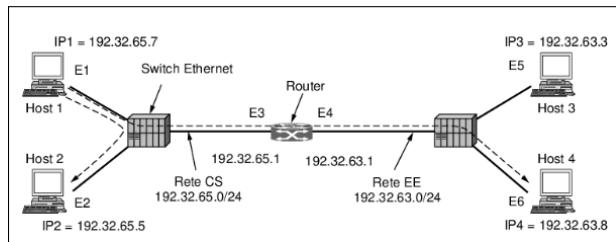


Figura 6.1: Due LAN con Ethernet commutata unite tramite un router.

Si osservi nuovamente la Figura 6.1 e si supponga che questa volta l'host 1 voglia inviare un pacchetto all'host 4 (192.32.63.8) sulla rete EE. L'host 1 vedrà che l'indirizzo IP di destinazione non si trova sulla rete CS. Sa che deve inviare tutto questo tra co fuori rete al router che prende anche il nome di default gateway e che, per convenzione, ha l'indirizzo più basso sulla rete (198.31.65.1). L'host 1 per inviare un frame al router deve comunque conoscere l'indirizzo Ethernet della sua scheda di rete sulla rete CS. Lo scopre inviando un broadcast ARP per 198.31.65.1 dal quale impara E3 e poi può inviare il frame. Lo stesso meccanismo viene usato per inviare un pacchetto da un router a un altro su un percorso.

È anche possibile mandare un pacchetto dall'host 1 all'host 4 senza che l'host 1 sappia che l'host 4 si trova su una rete diversa. La soluzione è di usare il router sulla rete CS per rispondere alle richieste ARP per l'host 4 e fornire il suo indirizzo Ethernet, E3. Non è possibile far rispondere l'host 4 direttamente, perché questo non vedrà la richiesta ARP in quanto i router non inoltrano i messaggi broadcast a livello Ethernet. Il router, poi, riceverà i frame indirizzati a 192.32.63.8 e li inoltrerà sulla rete EE. Questa soluzione prende il nome di **proxy ARP** ed è utilizzata in casi in cui un host voglia apparire su una rete anche se effettivamente risiede in un'altra. Una situazione comune, ad esempio, è quando una stazione mobile vuole che qualche altro nodo raccolga il proprio tra co quando non si trova nella sua LAN di appartenenza.

6.1.3 DHCP – Dynamic Host Configuration Protocol

ARP, come altri protocolli di Internet, assume che gli host siano configurati con alcune informazioni di base, come il proprio indirizzo IP. Come fanno gli host a ottenere questa informazione? Si potrebbe effettuare una configurazione manuale su ogni computer, ma sarebbe tedioso e suscettibile di errori. La soluzione è chiamata **DHCP** (*dynamic host configuration protocol*).

Con DHCP ogni rete deve avere un server DHCP responsabile della con gurazione. Quando un computer viene avviato ha un indirizzo Ethernet o altro di livello data link all'interno della scheda di rete, ma non un indirizzo IP. Analogamente ad ARP, il computer invia sulla rete una richiesta broadcast tramite un pacchetto DHCP DISCOVER per ottenere un indirizzo IP. Il pacchetto deve raggiungere il server DHCP. Se il server non è direttamente collegato alla stessa rete, il router deve essere configurato per ricevere il messaggio DHCP e inoltrarlo al server, ovunque esso sia.

Quando il server riceve il pacchetto alloca un indirizzo IP libero e lo invia all'host in un pacchetto DHCP OFFER (che potrebbe dover essere inoltrato nuovamente dal router). Per essere in grado di eseguire questa operazione anche quando gli host non hanno indirizzo IP, il server identifica un host tramite il suo indirizzo Ethernet (contenuto nel pacchetto DHCP DISCOVER).

Un problema che si presenta quando si assegnano automaticamente indirizzi IP estratti da una riserva comune riguarda la durata dell'allocazione. Se un host abbandona la rete e non restituisce il proprio indirizzo IP al server DHCP, quell'indirizzo andrà definitivamente perso, perciò dopo un po' di tempo molti indirizzi IP potrebbero non essere più disponibili. Per impedire che ciò accada, l'assegnazione dell'indirizzo IP può essere fissata per un periodo di tempo, mediante una tecnica chiamata leasing. Poco prima della scadenza del leasing, l'host deve chiedere al server DHCP il rinnovo dell'indirizzo. Se non riesce a fare la richiesta o la richiesta viene rifiutata, l'host non può più utilizzare l'indirizzo IP che gli era stato assegnato precedentemente.

DHCP è descritto negli RFC 2131 e 2132. È molto usato in Internet per configurare ogni sorta di parametro oltre a fornire indirizzi IP agli host. Oltre alle reti aziendali e domestiche DHCP è usato anche dagli ISP per configurare i parametri dei dispositivi di accesso a Internet, in modo che i clienti non debbano telefonare all'ISP per avere tale informazione. Esempi comuni sono la maschera di rete, l'indirizzo IP del default gateway, del DNS e del server per la sincronizzazione dell'orologio interno (time server). DHCP ha quasi del tutto sostituito protocolli precedenti (RARP e BOOTP) che avevano funzionalità più limitate.

6.2 NAT - Network Address Translation

Gli indirizzi IP sono scarsi; un ISP potrebbe avere un indirizzo /16 che consente di gestire 65.534 host, ma se il numero dei clienti supera questo valore, sorge un problema.

Questa scarsità ha portato alla creazione di tecniche per usare gli indirizzi IP cercando di fare economia. Un approccio è quello di assegnare gli indirizzi IP ai computer in modo dinamico al momento della connessione e recuperarli al termine della sessione per essere assegnati ad altri computer che si connettono. In questo modo un singolo indirizzo /16 permette di gestire fino a 65.534 utenti attivi.

La questione 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 che tutta Internet passi a IPv6, protocollo che adotta indirizzi a 128 bit. Questa transizione sta procedendo lentamente e ci vorranno anni prima che si completi. Di conseguenza alcune persone si sono resi conto che era necessario trovare una soluzione rapida attuabile in tempi brevi. La soluzione adottata si chiama **NAT** (*network address translation*), è descritta nell'RFC 3022 ed è esaminata brevemente in questo paragrafo.

L'idea di base di NAT è assegnare a ogni azienda o casa un singolo indirizzo IP (o, al massimo, un piccolo numero di indirizzi) per tutto di Internet. Dentro la rete del cliente, ogni computer riceve un indirizzo IP unico, utilizzato per instradare tutto interno alla rete locale. Tuttavia quando un pacchetto sta per lasciare la rete locale per dirigersi verso l'ISP viene eseguita una traduzione di indirizzo dall'unico indirizzo IP interno a quello pubblico condiviso. Questa traslazione usa tre intervalli di indirizzi IP dichiarati privati, che le aziende possono utilizzare internamente come preferiscono. L'unica regola è che nessun pacchetto contenente questi indirizzi possa apparire su Internet. I tre intervalli riservati sono:

10.0.0.0 – 10.255.255.255/8 (16.777.216 host)
172.16.0.0 – 172.31.255.255/12 (1.048.576 host)
192.168.0.0 – 192.168.255.255/16 (65.536 host)

Il primo intervallo permette di gestire 16.777.216 host (sono esclusi, come sempre, i valori 0 e 1) ed è la scelta comune in molte aziende, anche in quelle che non hanno bisogno di così tanti indirizzi.

Il funzionamento di NAT è mostrato nella Figura 5.55. Dentro i locali dell'azienda, ogni macchina ha un unico indirizzo espresso nella forma 10.x.y.z, ma quando un pacchetto lascia la rete locale, passa attraverso un **apparato NAT** che converte l'indirizzo IP interno (10.0.0.1 nella gura) nel vero indirizzo

IP assegnato all'azienda (198.60.42.12 in questo esempio). L'apparato NAT è spesso abbinato a un firewall, inserito all'interno di un singolo apparecchio che protegge la rete locale, controllando attentamente tutti i dati che entrano e che escono dalla LAN. Si può anche integrare l'apparato NAT nel router o nel modem ADSL aziendale.

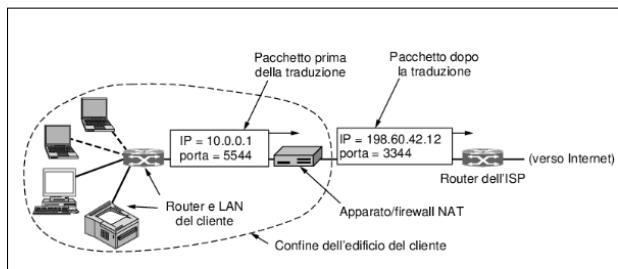


Figura 6.2: Collocazione e funzionamento di un apparato NAT.

Uno dei problemi principali del NAT riguarda la gestione delle risposte ai pacchetti. Quando i dati tornano, ad esempio da un server web, sono indirizzati all'IP pubblico (come 198.60.42.12). A questo punto, il NAT deve decidere a quale indirizzo di destinazione interno inviare questi dati. Idealmente, un campo nell'intestazione IP potrebbe essere utilizzato per tenere traccia dell'indirizzo sorgente originale, ma nell'intestazione IP rimane inutilizzato solo un singolo bit.

I progettisti di NAT hanno notato che la maggior parte dei pacchetti IP trasporta al suo interno TCP o UDP, entrambi con intestazioni che contengono una porta sorgente e una porta di destinazione. Le porte sono numeri interi a 16 bit che indicano dove inizia e finisce la connessione TCP. Quando un processo vuole stabilire una connessione TCP con un processo remoto, si lega a una porta TCP inutilizzata sulla sua macchina, chiamata porta sorgente, e fornisce anche una porta di destinazione. Questo permette di identificare i processi che utilizzano la connessione.

Per risolvere il problema dell'associazione, ogni volta che un pacchetto esce dalla rete interna e raggiunge il NAT, l'indirizzo sorgente interno (ad esempio 10.x.y.z) viene sostituito dall'indirizzo IP pubblico dell'azienda, e il campo della porta sorgente viene sostituito con un indice che punta alla tabella di traduzione del NAT. Questa tabella contiene l'indirizzo IP originale e la porta sorgente originale. I checksum delle intestazioni IP e TCP vengono ricalcolati e inseriti nel pacchetto. Questo meccanismo permette di mantenere l'associazione tra indirizzi interni ed esterni, nonostante due macchine diverse possano utilizzare la stessa porta sorgente.

Quando un pacchetto di risposta dall'ISP raggiunge il NAT, il campo della porta sorgente nell'intestazione TCP viene utilizzato come indice nella tabella di associazione. Da questa tabella, il NAT estrae l'indirizzo IP interno e la porta sorgente TCP originale, inserendo queste informazioni nel pacchetto e ricalcolando i checksum IP e TCP. Infine, il pacchetto viene inoltrato al router interno per la normale consegna basata sull'indirizzo 10.x.y.z. Obiezioni e Limiti del NAT

Molti nella comunità IP considerano il NAT problematico per diverse ragioni. In primo luogo, il NAT viola il modello gerarchico di IP, secondo cui ogni indirizzo IP identifica in modo univoco una singola macchina a livello globale. Con il NAT, migliaia di macchine possono condividere lo stesso indirizzo IP, come 10.0.0.1. In secondo luogo, il NAT rompe il modello di connettività end-to-end di Internet, impedendo a un host pubblico di iniziare una comunicazione con un host privato senza una connessione preesistente.

Il NAT trasforma inoltre Internet da una rete non orientata alla connessione a una rete orientata alla connessione, in quanto il NAT deve mantenere lo stato delle connessioni che attraversano. Questo rende la rete vulnerabile a malfunzionamenti del NAT, che possono interrompere tutte le connessioni TCP in corso.

Il NAT viola anche il principio della stratificazione dei protocolli, che afferma che ogni livello non deve fare ipotesi su ciò che il livello superiore ha inserito nel payload. Se in futuro il protocollo TCP venisse aggiornato con uno schema di intestazione diverso, il NAT non funzionerebbe più correttamente. Inoltre,

il NAT non supporta bene i protocolli non TCP/UDP e applicazioni che utilizzano più connessioni TCP o porte UDP preimpostate, come il protocollo **FTP** (*file transfer protocol*).

Infine, poiché il campo della porta sorgente TCP è di 16 bit, un indirizzo IP può essere associato a un massimo di 65.536 macchine, un numero leggermente inferiore poiché alcune porte sono riservate per usi speciali. Questi e altri problemi legati al NAT sono discussi nell'RFC 2993. Nonostante questi limiti, il NAT è ampiamente utilizzato nelle reti domestiche e nelle piccole aziende come soluzione pratica al problema della mancanza di indirizzi IP, spesso combinato con funzioni di firewall per bloccare i pacchetti in arrivo non richiesti. Anche con l'adozione diffusa di IPv6, è improbabile che il NAT scompaia completamente.

Capitolo 7

Livello Trasporto

Il livello di trasporto è il cuore della gerarchia del protocollo, si basa sul livello di rete per fornire il trasporto dei dati da un processo su una macchina sorgente a un processo su una macchina destinazione con un livello di affidabilità desiderato e indipendente dalle reti fisiche attualmente in uso.

7.1 Descrizione dei Servizi di Trasporto

L'obiettivo finale del livello di trasporto è fornire un servizio di trasmissione dati efficace, affidabile ed efficiente in termini di costi ai suoi utenti. Per raggiungere tale obiettivo, il livello di trasporto utilizza i servizi forniti a livello di rete. L'hardware e il software all'interno del livello di trasporto che svolge il lavoro è chiamato **entità di trasporto**. L'entità di trasporto può essere situata nel kernel del sistema operativo, in un processo utente separato, in librerie associate alle applicazioni di rete o nella scheda di rete. Esistono due tipi di servizi di trasporto, orientato o non alla connessione, in entrambi i casi le connessioni seguono tre fasi: impostazione, trasferimento dati e rilascio.

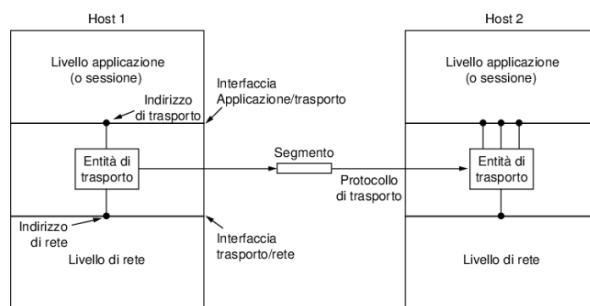


Figura 7.1: Livelli di rete, trasporto e applicazione

Se in una rete senza connessione vengono persi o alterati i pacchetti, l'entità di trasporto può rilevare il problema e compensarlo tramite ritrasmissioni, oppure se in una rete orientata alla connessione, un entità di trasporto è informata a metà di una lunga trasmissione che la sua connessione è stata interrotta, può impostare una nuova connessione di rete verso l'entità di trasporto remota. In sostanza permette al servizio di trasporto di essere più affidabile del sottostante servizio di rete.

A tutti gli effetti il livello di trasporto svolge la funzione di isolare i livelli superiori (**utenti del servizio di trasporto**), dalla struttura e dalle imperfezioni della rete, ovvero i livelli inferiori (**fornitori del servizio di trasporto**).

7.1.1 Primitive

Per consentire agli utenti di accedere al servizio di trasporto, il livello di trasporto deve fornire alcune funzionalità ai programmi applicativi, vale a dire un'interfaccia per il servizio di trasporto.

Per vedere come queste primitive possano essere utilizzate consideriamo un applicazione con un server e diversi client remoti. Per iniziare il server esegue una primitiva LISTEN, chiamando generalmente una procedura di libreria, che esegue una chiamata di sistema la quale blocca il server fino al sopraggiungere di un client. Quando un client desidera comunicare con il server, esegue una primitiva CONNECT. L'entità di trasporto esegue questa primitiva bloccando il chiamante e inviando un pacchetto al server. Incapsulato nel payload di questo pacchetto troviamo un messaggio del livello di trasporto per l'entità di trasporto del server.

Primitiva	Pacchetto inviato	Significato
LISTEN	(nessuno)	Si blocca fino a quando un processo cerca di connettersi
CONNECT	CONNECTION REQ.	Tenta di stabilire una connessione
SEND	DATA	Invia informazioni
RECEIVE	(nessuno)	Si blocca fino all'arrivo di un pacchetto DATA
DISCONNECT	DISCONNECTION REQ.	Questo lato desidera rilasciare la connessione

Figura 7.2: Le primitive per un semplice servizio di trasporto

In mancanza di un termine migliore, chiameremo segmento un messaggio spedito da un entità di trasporto all'altra. TCP, UDP e altri protocolli usano questo termine, mentre alcuni protocolli più vecchi usavano il termine **TPDU**(transport control data unit). Dunque i segmenti (livello trasporto) sono contenuti in pacchetti (livello rete), che a loro volta sono contenuti in frame.

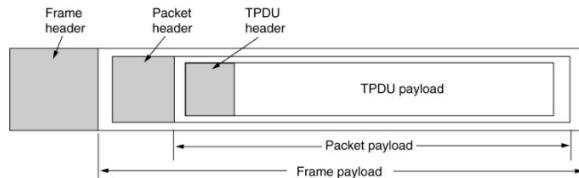


Figura 7.3: Nidificazione di segmenti, pacchetti e frame.

Tornando all'esempio client/server, la chiamata **CONNECT** del client provoca l'invio al server di un segmento **CONNECTION REQUEST**. Quando arriva, l'entità di trasporto controlla se il server è bloccato su una chiamata LISTEN (vale a dire se è interessato a gestire delle richieste). In tal caso sblocca il server e invia un segmento **CONNECTION ACCEPTED** al client. Quando questo segmento arriva, il client viene sbloccato e la connessione è stabilita. Ora i dati possono essere scambiati con le primitive **SEND** e **RECEIVE**. Nella forma più semplice, ogni lato può inviare una primitiva RECEIVE (bloccante) per attendere un SEND da parte della controparte. Quando il segmento arriva il destinatario viene sbloccato, può quindi elaborarlo e inviare una risposta. Finché entrambi i lati riescono a tenere traccia di a chi tocca inviare, lo schema funziona bene.

Quando una connessione non è più necessaria, deve essere rilasciata per liberare spazio nelle tabelle all'interno delle due entità di trasporto. La disconnessione presenta due varianti: asimmetrica e simmetrica. Nella variante asimmetrica, ogni utente del trasporto può emettere una primitiva **DISCONNECT** che provoca l'invio di un segmento DISCONNECT all'entità di trasporto remota. Al suo arrivo la connessione viene rilasciata. Nella variante simmetrica ogni direzione viene chiusa separatamente, indipendentemente dall'altra. Quando un lato usa la primitiva DISCONNECT, significa che non ha altri dati da inviare, ma è ancora in grado di accettare dati dal suo partner. In questo modello, una connessione viene rilasciata quando entrambi i lati hanno inviato un segmento DISCONNECT.

Un diagramma di stato per la costituzione e il rilascio della connessione tramite queste semplici primitive è mostrato nella Figura 7.4. Ogni transizione è innescata da un evento, cioè una primitiva eseguita dall'utente locale del trasporto o un pacchetto in ingresso.

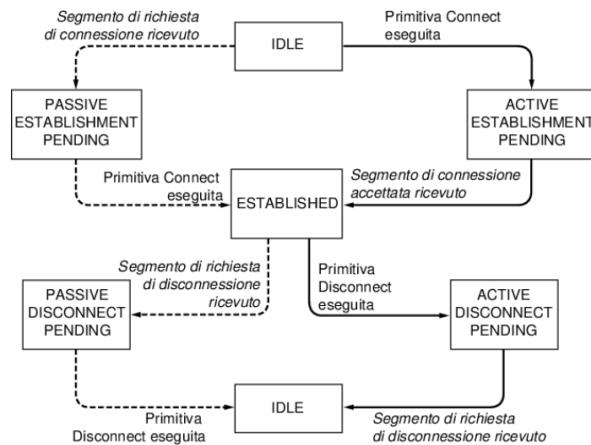


Figura 7.4: Schema di gestione della connessione

7.1.2 Socket di Berkeley

Esaminiamo ora brevemente un altro gruppo di primitive di trasporto, le primitive delle socket utilizzate per TCP. Queste primitive, elencate nella Figura 7.5, seguono all'incirca il modello del nostro primo esempio, ma offrono maggiori caratteristiche e flessibilità.

Primitiva	Significato
SOCKET	Crea un nuovo punto finale di comunicazione
BIND	Associa un indirizzo locale a una socket
LISTEN	Annuncia la capacità di accettare connessioni; fornisce la dimensione della coda
ACCEPT	Blocca il chiamante fino all'arrivo di un tentativo di connessione
CONNECT	Tenta in modo attivo di stabilire una connessione
SEND	Invia alcuni dati sulla connessione
RECEIVE	Riceve alcuni dati sulla connessione
CLOSE	Rilancia la connessione

Figura 7.5: Le primitive delle socket per TCP

Il funzionamento è essenzialmente questo:

- **Creazione e assegnazione di indirizzo:** La sequenza inizia con la creazione di una socket, seguita dall'assegnazione di un indirizzo utilizzando la primitiva **BIND**. Questo passaggio è cruciale per consentire ai client remoti di connettersi al server.
- **Ascolto delle connessioni in arrivo:** La chiamata **LISTEN** permette al server di accodare le connessioni in arrivo nel caso in cui più client tentino di connettersi contemporaneamente. Contrariamente all'esempio dato, questa fase di ascolto non è bloccante nelle socket.
- **Accettazione delle connessioni in ingresso:** Quando arriva una richiesta di connessione, il server utilizza la primitiva **ACCEPT** per creare una nuova socket e gestire la connessione. Questo permette al server di gestire più connessioni simultaneamente.

Dal lato del client:

- **Connessione al server:** Il client utilizza la primitiva **CONNECT** per avviare il processo di connessione al server. Una volta stabilita la connessione, entrambe le parti possono trasmettere e ricevere dati utilizzando le primitive **SEND** e **RECEIVE**.

7.2 Elementi dei Protocolli di Trasporto

Il servizio di trasporto è implementato da un protocollo di trasporto utilizzato tra le due entità di trasporto. Per certi aspetti i protocolli di trasporto ricordano i protocolli data link. Entrambi hanno a che

fare, tra le altre cose, con il controllo degli errori, l'ordinamento e il controllo di flusso.

Tuttavia tra i due esistono differenze significative, dovute alle diversità tra gli ambienti in cui operano i protocolli, come mostrato nella Figura 7.6. Nel livello data link due router comunicano direttamente tramite un canale fisico sia esso cablato o wireless, mentre nel livello di trasporto il canale fisico è sostituito dall'intera rete. Questa differenza porta con sé molte importanti implicazioni per i protocolli.

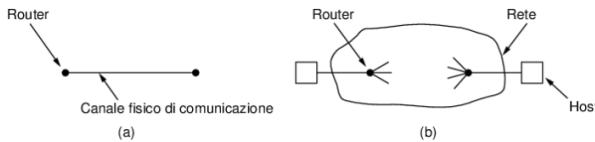


Figura 7.6: (a) Ambiente del livello data link. (b) Ambiente del livello di trasporto

Altre differenze importanti sono:

- **Indirizzamento esplicito nel livello di trasporto:** A differenza dei collegamenti punto a punto come cavi o fibre ottiche, dove ogni linea in uscita specifica univocamente il router destinatario, nel livello di trasporto è necessario un indirizzamento esplicito delle destinazioni.
- **Complessità nella costituzione della connessione:** Nel livello di trasporto, la costituzione della connessione iniziale è più complessa rispetto al livello data link, in cui l'altra estremità è sempre presente e non richiede molto lavoro.
- **Possibilità di memorizzazione nella rete:** Nel livello data link, i pacchetti non possono rimbalzare avanti e indietro nella rete o nascondersi per emergere successivamente. Tuttavia, nel livello di trasporto, esistono capacità di memorizzazione nella rete che possono causare ritardi, perdite o duplicazioni di pacchetti, richiedendo l'uso di protocolli speciali.
- **Buffering e controllo di flusso:** Entrambi i livelli richiedono buffering e controllo di flusso, ma nel livello di trasporto la presenza di un numero elevato e variabile di connessioni può richiedere un approccio diverso, con una distribuzione più efficiente delle risorse di buffer rispetto al livello data link.

7.2.1 Indirizzamento

Quando un processo applicativo (per esempio un utente) vuole creare una connessione verso un processo applicativo remoto, deve specificare a quale intende connettersi. Il trasporto non orientato alla connessione presenta lo stesso problema: a chi bisogna inviare ogni singolo messaggio? Il metodo normalmente utilizzato consiste nel definire indirizzi di trasporto su cui i processi possono restare in ascolto delle richieste di connessione. Su Internet questi endpoint (o punti terminali) sono chiamati **porte** (port). Useremo il termine generico **TSAP** (transport service access point) riferendoci a uno specifico endpoint nel livello di trasporto. Gli analoghi endpoint nel livello di rete (come gli indirizzi del livello di rete) sono banalmente chiamati **NSAP** (network service access point). Gli indirizzi IP sono esempi di NSAP.

La Figura 7.7 illustra la relazione tra NSAP, TSAP e una connessione di trasporto. I processi applicativi, client e server, possono collegarsi a un TSAP locale per stabilire una connessione con un TSAP remoto. Queste connessioni scorrono attraverso gli NSAP su ogni host. Il motivo per cui servono i TSAP è che in alcune reti ogni computer possiede un singolo NSAP, pertanto è necessario distinguere in qualche modo i diversi endpoint di trasporto che condividono lo stesso NSAP.

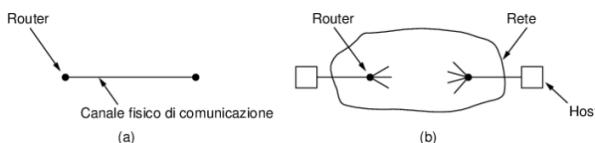


Figura 7.7: TSAP, NSAP e connessioni di trasporto

7.2.2 Stabilire una Connessione

L'instaurazione di una connessione può sembrare un processo semplice inizialmente, ma in realtà è soggetto a una serie di complicazioni dovute alle caratteristiche intrinseche delle reti di comunicazione. Innanzitutto, il fatto che la rete possa perdere, ritardare, corrompere e duplicare i pacchetti rende l'instaurazione di una connessione un'operazione sorprendentemente complessa. Sebbene possa sembrare sufficiente per un'entità di trasporto inviare un messaggio di richiesta di connessione e attendere una risposta di accettazione, il comportamento imprevedibile della rete può causare serie complicazioni.

Un possibile scenario di problematiche si presenta in una rete congestionata, dove gli **acknowledgements** (conferme di ricezione) non riescono quasi mai a tornare indietro in tempo. In questa situazione, ogni pacchetto potrebbe subire un timeout ed essere ritrasmesso più volte, generando ulteriore congestione e potenziali ritardi. Se la rete utilizza datagrammi e i pacchetti seguono percorsi diversi, alcuni pacchetti potrebbero restare bloccati in zone congestionate della rete, arrivando in ritardo o fuori sequenza rispetto alle aspettative del mittente.

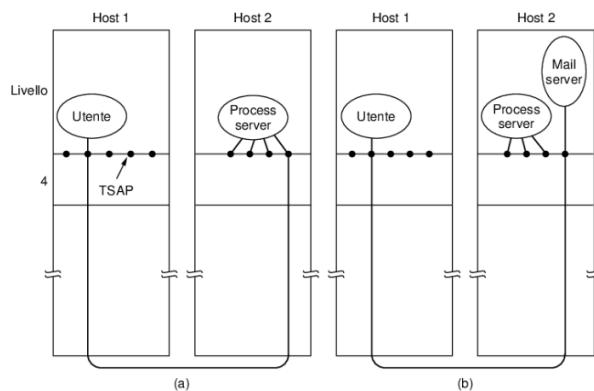


Figura 7.8: Modalità di connessione di un processo utente nell'host 1 con un mail server nell'host 2

Il peggior scenario immaginabile coinvolge una transazione finanziaria critica, come un trasferimento di denaro da parte di un utente a una banca. Se i pacchetti coinvolti nella transazione decidono di prendere percorsi alternativi nella rete, il mittente potrebbe subire timeout e ritrasmettere i pacchetti, convinto che siano persi. Tuttavia, i pacchetti potrebbero emergere improvvisamente in ritardo e fuori sequenza, causando confusione e potenzialmente generando transazioni duplicate e involontarie.

Per affrontare queste problematiche, è necessario adottare strategie che garantiscano l'integrità e l'efficienza delle connessioni di rete. Per semplificare il problema, è possibile limitare la vita dei pacchetti utilizzando tecniche come la progettazione di reti con restrizioni, l'inserimento di un contatore di salti in ogni pacchetto o l'applicazione di un timestamp. Limitare la vita dei pacchetti consente di evitare che pacchetti duplicati in ritardo possano causare confusioni o errori nelle connessioni di rete.

Tomlinson propose invece di equipaggiare ogni host con un orologio. Gli orologi dei differenti host non hanno bisogno di essere sincronizzati; si assume che ogni orologio sia implementato come un contatore binario che si incrementi a intervalli regolari. Inoltre il numero di bit nel contatore deve essere uguale o maggiore al numero di bit nel numero di sequenza. Ultimo e più importante si assume che l'orologio continui a misurare il tempo anche se l'host non sta funzionando. Quando viene instaurata una connessione, i k bit di ordine più basso dell'orologio sono usati come primo numero di sequenza. La regione proibita mostra i tempi per i quali i numeri di sequenza dei segmenti non sono ammissibili per essere utilizzati. Se un segmento è spedito con numero di sequenza all'interno di questa regione, potrebbe venire ritardato ed essere scambiato per un diverso pacchetto con lo stesso numero di sequenza emesso più tardi.

Il metodo basato sull'orologio risolve il problema di non riuscire a distinguere i segmenti duplicati da quelli nuovi. Tuttavia c'è un intoppo pratico nell'utilizzarlo per stabilire delle connessioni: poiché normalmente non ricordiamo i numeri di sequenza tra una connessione e l'altra, non abbiamo modo di sapere se un segmento CONNECTION REQUEST contenente un numero di sequenza iniziale sia un duplicato di una connessione recente. Questo problema non sussiste durante una connessione perché il protocollo

a finestra scorrevole ricorda il numero di sequenza corrente.

Per risolvere questo problema, Tomlinson ha introdotto **l'handshake a tre vie** (three-way handshake). Questo protocollo per l'impostazione di una connessione richiede la verificare reciproca da parte dei peer che l'attuale richiesta di connessione non sia un duplicato. La normale procedura per stabilire la connessione quando è l'host 1 a iniziare è mostrata nella Figura 6.11(a). L'host 1 sceglie un numero di sequenza, x , e invia un segmento CONNECTION REQUEST contenente x all'host 2, che risponde con un segmento ACK per confermare x e annunciare il suo numero di sequenza iniziale, y . Per finire l'host 1 conferma la scelta del numero di sequenza iniziale dell'host 2 con il primo segmento dati inviato.

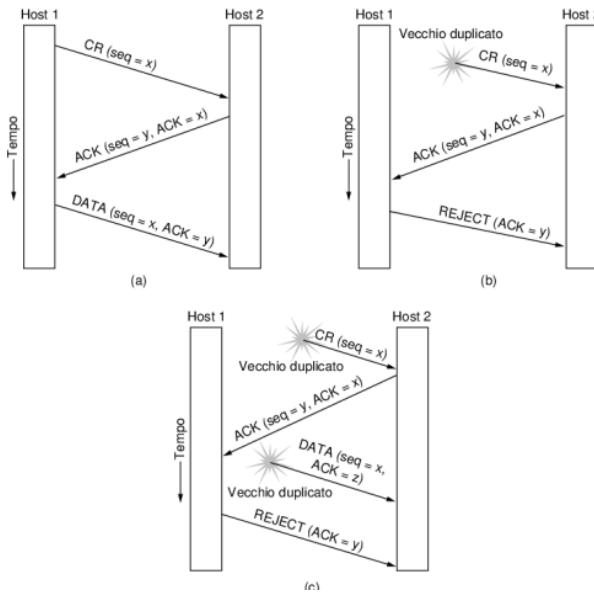


Figura 7.9: Tre scenari relativi all'istituzione di una connessione con l'handshake a tre vie. CR indica CONNECTION REQUEST. (a) Operazioni normali. (b) Una vecchia CONNECTION REQUEST duplicata appare dal nulla. (c) Con presenza di CONNECTION REQUEST e ACK duplicati.

7.2.3 Rilascio della Connessione

Rilasciare una connessione è più facile che stabilirla, però le trappole sono più di quante se ne possano immaginare. Come affermato in precedenza, esistono due modi per terminare una connessione: il rilascio asimmetrico e quello simmetrico. Il rilascio asimmetrico è il metodo utilizzato dal sistema telefonico: quando una delle due parti riattacca la connessione viene interrotta. Il rilascio simmetrico tratta la connessione come se fosse composta da due connessioni unidirezionali separate e richiede il rilascio separato di ognuna di esse.

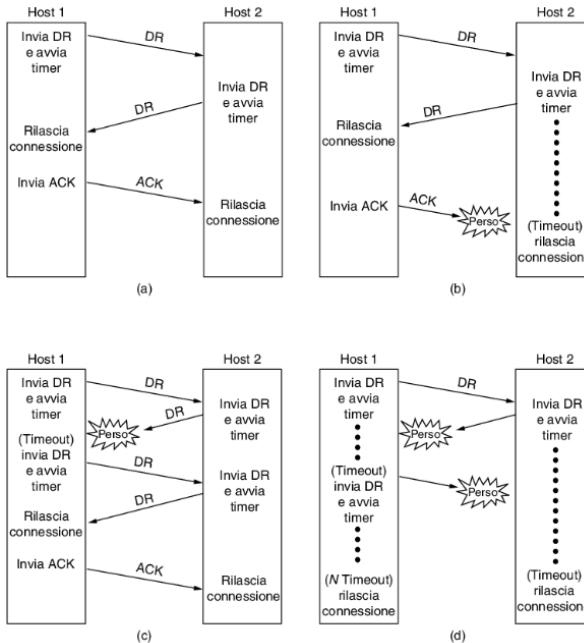


Figura 7.10: Quattro scenari per il rilascio di una connessione. (a) Caso normale di handshake a tre vie. (b) ACK segnale perduto. (c) Risposta perduta. (d) Risposta perduta e DR successivo perduto

Nella Figura 7.10 (a) è mostrato il caso normale in cui uno degli utenti invia un segmento DR (DISCONNECTION REQUEST) per iniziare il rilascio di una connessione. Quando arriva, il destinatario restituisce un segmento DR e avvia un timer (servirà nel caso il suo DR vada perso). All'arrivo di questo DR, il mittente originale invia un segmento ACK e rilascia la connessione. Per finire, quando arriva il segmento ACK, il ricevente rilascia la connessione. Rilasciare una connessione significa che l'entità di trasporto rimuove le informazioni sulla connessione dalla propria tabella delle connessioni aperte e lo segnala al proprietario della connessione (l'utente del livello di trasporto). Questa azione è diversa da un utente del livello di trasporto che fa uso di una primitiva DISCONNECT.

Se il segmento ACK finale viene perso, come mostrato nella Figura 7.10 (b), la situazione viene risolta dal timer, alla cui scadenza la connessione viene comunque rilasciata. Consideriamo ora il caso della perdita del secondo DR. L'utente che inizia la disconnessione non riceve la risposta prevista, si verifica un timeout e tutto ricomincia; la sequenza degli eventi è visibile nella Figura 7.10 (c), facendo l'ipotesi che la seconda volta nessun segmento venga perso e che tutti i segmenti siano consegnati correttamente e in tempo.

L'ultimo scenario, mostrato nella Figura 7.10 (d), equivale a quello della Figura 7.10 (c), tranne per il fatto che ora si presume che tutti i tentativi di ritrasmettere il DR falliscano a causa di segmenti persi. Dopo N tentativi il mittente si arrende e rilascia la connessione; nel frattempo si verifica un timeout anche nel ricevente che a sua volta esce. Questo protocollo è generalmente sufficiente, ma in teoria può fallire se vengono persi il DR iniziale e le N ritrasmissioni. Il mittente abbandonerà i tentativi e rilascerà la connessione, mentre l'altro lato non saprà nulla di tutti i tentativi di disconnessione e sarà pienamente attivo. Questa situazione genera una connessione aperta a metà.

Avremmo potuto evitare il problema non consentendo al mittente di abbandonare dopo N tentativi, ma obbligandolo a proseguire no al ricevimento di una risposta. Tuttavia, se si consente alla controparte di terminare per timeout, il mittente procederà all'inizio perché non riceverà mai risposta. Se non consentiamo il timeout del lato ricevente, il protocollo rimarrà in sospeso, come nella Figura (d).

7.2.4 Controllo degli Errori e Controllo di Flusso

Dato che i protocolli di trasporto utilizzano generalmente finestre grandi, esamineremo con più cura il problema dell'immagazzinamento dei dati in un buffer (buffering). Poiché un host può avere molte connessioni, ognuna delle quali è trattata separatamente, potrebbe essere necessaria una notevole quantità

di buffer per le finestre scorrevoli. I buffer sono necessari sia al mittente che al destinatario; certamente sono necessari al mittente per contenere tutti i segmenti che sono stati trasmessi, ma che non hanno ancora ricevuto acknowledgement; gli sono necessari perché questi segmenti potrebbero andare persi e dovrebbero essere ritrasmessi.

Se la maggior parte dei segmenti ha una dimensione simile, è naturale raggrupparli per dimensione identica e usare un buffer per ogni segmento, come mostrato nella Figura 7.11 (a). Tuttavia, se esiste una variazione notevole nella dimensione dei segmenti (da richieste brevi per pagine Web a pacchetti di grandi dimensioni per trasferimenti di file peer-to-peer), un insieme di buffer con dimensioni fisse genera dei problemi. Se la dimensione del singolo buffer è scelta in base al segmento con la massima dimensione possibile, si verificava uno spreco di spazio ogni volta che viene ricevuto un segmento breve. Se la dimensione del buffer è inferiore alla dimensione massima dei segmenti, saranno necessari più buffer per i segmenti lunghi, incrementando la complessità della gestione della memoria.

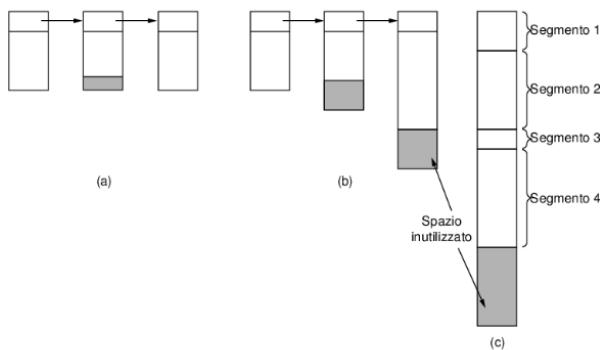


Figura 6.15 (a) Buffer concatenati a dimensione fissa. (b) Buffer concatenati a dimensione variabile. (c) Un grosso buffer circolare per ogni connessione.

Figura 7.11: (a) Buffer concatenati a dimensione fissa. (b) Buffer concatenati a dimensione variabile. (c) Un grosso buffer circolare per ogni connessione.

Un altro approccio al problema della dimensione del buffer è l'utilizzo di buffer con dimensioni variabili, come mostrato nella Figura 7.11 (b). Il vantaggio è un utilizzo migliore della memoria, al prezzo di una gestione più complicata. Una terza possibilità prevede di dedicare un singolo buffer circolare di grandi dimensioni a ogni connessione, come mostrato nella Figura 7.11 (c). Questo sistema è semplice ed elegante e non dipende dalla dimensione dei segmenti, ma fa un buon utilizzo della memoria solo quando tutte le connessioni sono pesantemente utilizzate.

A differenza dei protocolli a finestra scorrevole del Capitolo 3, un metodo generale per gestire l'allocazione dinamica dei buffer consiste nel separare il buffering dagli acknowledgement. La gestione dinamica dei buffer si traduce in definitiva nell'uso di una finestra di dimensione variabile. Inizialmente il mittente richiede un certo numero di buffer, basato sulla previsione delle proprie esigenze e il destinatario ne concede il numero che può permettersi. Ogni volta che il mittente trasmette un segmento deve decrementare la sua allocazione, fermandosi completamente quando raggiunge il valore zero. Il destinatario può quindi aggiungere acknowledgement e informazioni sulle allocazioni del buffer in coda al traffico che scorre nella direzione opposta. TCP usa questo approccio, trasportando le allocazioni di buffer in un campo dell'intestazione chiamato *Window size*.

A	Messaggio	B	Commenti
1	→ < request 8 buffers>		→ A desidera 8 buffer
2	← <ack = 15, buf = 4>		→ B conferma l'arrivo solo dei messaggi 0-3
3	→ <seq = 0, data = m0>		→ A dispone di 3 buffer
4	→ <seq = 1, data = m1>		→ Adesso A dispone di 2 buffer
5	→ <seq = 2, data = m2>		••• Messaggio perso, ma A pensa che ne sia rimasto 1
6	← <ack = 1, buf = 3>		→ B conferma l'arrivo di 0 e 1 e permette 2-4
7	→ <seq = 3, data = m3>		→ A dispone di 1 buffer
8	→ <seq = 4, data = m4>		→ A dispone di 0 buffer e deve fermarsi
9	→ <seq = 2, data = m2>		→ Timeout e ritrasmissione di A
10	← <ack = 4, buf = 0>		→ Tutti gli acknowledgement sono arrivati, ma A è ancora bloccato
11	← <ack = 4, buf = 1>		→ A ora può inviare 5
12	→ <ack = 4, buf = 2>		→ B ha trovato un nuovo buffer altrove
13	→ <seq = 5, data = m5>		→ A dispone di 1 buffer
14	→ <seq = 6, data = m6>		→ A è di nuovo bloccato
15	← <ack = 6, buf = 0>		→ A è ancora bloccato
16	••• <ack = 6, buf = 4>		→ Possible stallo

Figura 7.12: Allocazione dinamica dei buffer. Le frecce mostrano la direzione della trasmissione. I puntini di sospensione (...) indicano un segmento perso.

La Figura 7.12 mostra un esempio di come potrebbe funzionare la gestione dinamica delle finestre in una rete datagram con numeri di sequenza a 4 bit. Nell'esempio i dati viaggiano in segmenti dall'host A all'host B, mentre acknowledgement e informazioni sui buffer viaggiano in segmenti nella direzione opposta. Inizialmente A vuole otto buffer, ma gliene vengono garantiti solo quattro. Successivamente invia tre segmenti, di cui il terzo viene perso. Il segmento 6 fornisce l'acknowledgement a tutti i segmenti no a quello, con numero di sequenza 1 incluso, consentendo quindi ad A di rilasciare i buffer coinvolti; inoltre informa A che ha il permesso di inviare altri tre segmenti con numero di sequenza superiore a 1 (vale a dire i segmenti 2, 3 e 4). A sa di avere già inviato il numero 2, pertanto pensa che potrebbe inviare i segmenti 3 e 4 e procede a farlo; a questo punto A è bloccato e deve aspettare un'ulteriore allocazione dei buffer. Durante il blocco possono comunque avvenire ritrasmissioni indotte dal timeout (riga 9), dal momento che utilizzano buffer già allocati. Nella riga 10, B conferma la ricezione di tutti i segmenti no a 4 compreso, ma impedisce ad A di continuare. Tale situazione sarebbe impossibile con i protocolli a finestra fissa del Capitolo 3. Il segmento successivo da B ad A alloca un altro buffer e consente ad A di proseguire. Questo accadrà quando B ha spazio nel buffer, probabilmente perché l'utente del livello di trasporto ha accettato altri dati.

Quando lo spazio per i buffer non limita più il flusso massimo, nasce un altro collo di bottiglia: la capacità di trasporto della rete. Quello che serve è un meccanismo che limita le trasmissioni del mittente basato sulla capacità di trasmissione della rete anziché sulla capacità di buffering del destinatario. Belsnes (1975) ha proposto l'utilizzo di uno schema di controllo di flusso a finestra scorrevole in cui il mittente regola dinamicamente la dimensione della finestra per farla corrispondere alla capacità di trasporto della rete. Ciò significa che una finestra scorrevole dinamica può implementare sia il controllo di flusso sia il controllo di congestione. Dal momento che la capacità di rete disponibile per un dato flusso varia nel tempo, la dimensione della finestra dovrebbe essere regolata frequentemente per tenere traccia dei cambiamenti nella capacità di trasporto.

7.2.5 Multiplexing

Il multiplexing di più conversazioni su connessioni, circuiti virtuali e collegamenti fisici svolge un ruolo importante in diversi livelli dell'architettura di rete. Nel livello di trasporto l'esigenza del multiplexing può sorgere in molti modi. Per esempio se un host ha a disposizione un solo indirizzo di rete, tutte le connessioni di trasporto del computer devono utilizzare tale indirizzo. Quando arriva un segmento, è necessario stabilire in qualche modo a quale processo passarlo. Questa situazione, chiamata **multiplexing**, è mostrata nella Figura 6.17(a). In questa Figura quattro connessioni di trasporto distinte utilizzano la stessa connessione di rete (e cioè l'indirizzo IP) con l'host remoto.

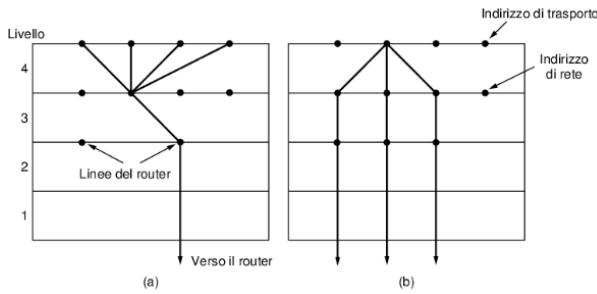


Figura 7.13: (a) Multiplexing. (b) Multiplexing inverso.

Se un utente ha bisogno di più ampiezza di banda o più affidabilità di quella fornita da uno dei percorsi di rete, un modo per risolvere la situazione è avere una connessione che distribuisca il traffico su diversi percorsi di rete facendo uso di round-robin, come indicato nella Figura 7.13 (b). Questo modo di operare è detto **multiplexing inverso** (*inverse multiplexing*)

7.2.6 Ripristino Dopo un Crash

Se host e router sono soggetti a malfunzionamenti e smettono di funzionare improvvisamente (gergalmente diciamo che “vanno in crash”) mentre sono in atto delle connessioni che durano molto (come download di grossi software o le multimediali), il ripristino può diventare un problema. Quando l’entità di trasporto è interamente all’interno degli host, il ripristino dopo un crash di rete o del router è banale. Le entità di trasporto prevedono sempre la perdita di segmenti e sanno come gestirla tramite ritrasmissioni. Un problema più complesso riguarda il ripristino dopo un crash dell’host. In particolare sarebbe auspicabile che i client fossero in grado di continuare a lavorare quando i server subiscono un crash e vengono rapidamente riavviati. Per illustrare la difficoltà, supponiamo che un host, il client, stia inviando un lungo file a un altro host, il server, utilizzando un semplice protocollo stop-and-wait. Il livello di trasporto sul server si limita a passare i segmenti in arrivo all’utente del livello di trasporto uno per uno. Durante la trasmissione il server subisce un crash; dopo il riavvio le sue tabelle vengono reinizializzate, pertanto non sa più precisamente a che punto era arrivato.

In un tentativo di recuperare il suo stato precedente, il server potrebbe inviare un segmento broadcast a tutti gli altri host, annunciando di aver subito un crash e richiedendo che tutti i suoi client lo informino dello stato delle connessioni aperte. Ogni client può trovarsi in uno dei due stati: S1, con un segmento in circolazione o S0, senza segmenti in circolazione. In base solo a queste informazioni di stato il client deve decidere se ritrasmettere il segmento più recente.

A prima vista il risultato appare ovvio: il client dovrebbe ritrasmettere se e solo se, quando viene informato del crash, ha in circolazione un segmento che non ha ancora ricevuto acknowledgement (stato S1). Tuttavia un’analisi più attenta evidenzia alcune di coltà di tale approccio. Si consideri, per esempio, la situazione in cui l’entità di trasporto del server invia prima l’acknowledgement e in seguito, dopo l’invio, passa i dati al processo applicativo. La scrittura di un segmento nel usso di output e l’invio di un acknowledgement sono due eventi distinti che non possono essere svolti contemporaneamente. Se si verificasse un crash dopo l’invio dell’acknowledgement ma prima della scrittura, il client riceverà l’acknowledgement e pertanto si troverà nello stato S0 quando giunge l’annuncio di ripristino dal crash. Il client non eseguirà di nuovo la trasmissione, pensando (erroneamente) che il segmento sia arrivato. Questa decisione del client genera un segmento mancante.

A questo punto i lettori potrebbero pensare: “Il problema può essere risolto facilmente. È sufficiente riprogrammare l’entità di trasporto per svolgere prima la scrittura e poi inviare l’acknowledgement”. In altre parole, si supponga che la scrittura sia stata eseguita, ma che il crash avvenga prima dell’invio dell’acknowledgement. Il client sarà nello stato S1 ed eseguirà di nuovo la trasmissione, generando un segmento duplicato non rilevato nel usso di output verso il processo applicativo del server.

Indipendentemente dall’implementazione di client e server, esistono sempre situazioni in cui il protocollo fallisce il corretto ripristino. Il server può essere implementato in due modi: prima l’acknowledgement

o prima la scrittura. Il client può essere programmato in quattro modi: per ritrasmettere sempre l'ultimo segmento, per non ritrasmettere mai l'ultimo segmento, per ritrasmetterlo solo nello stato S0 o per ritrasmetterlo solo nello stato S1. Si ottengono così otto combinazioni: come potremo vedere, per ogni combinazione esiste un insieme di eventi che provocano il fallimento del protocollo.

Per il server sono possibili tre eventi: invio di un acknowledgement (A), passaggio dei dati al processo di output (W) e crash (C). I tre eventi possono avvenire in sei ordini diversi: AC(W), AWC, C(AW), C(WA), WAC e WC(A); le parentesi sono usate per indicare che né A né W possono seguire C (una volta che l'host ha subito un crash non c'è più nulla da fare). La Figura 6.18 mostra le otto combinazioni della strategia per client e server e le sequenze di eventi valide per ognuno. Occorre notare che per ogni strategia esiste una sequenza di eventi che provoca il fallimento del protocollo. Per esempio, se il client ritrasmette sempre la ritrasmissioni, l'evento AWC genera un duplciato non rilevato, anche se gli altri due eventi funzionano correttamente.

		Strategia utilizzata dall'host che riceve					
		Prima ACK, poi scrittura			Prima scrittura, poi ACK		
Strategia utilizzata dall'host che invia	AC(W)	OK	DUP	OK	C(WA)	OK	DUP
	AWC	LOST	OK	LOST	W AC	LOST	OK
	C(AW)	OK	DUP	LOST	WC(A)	LOST	DUP
		LOST	OK	OK		OK	OK

OK = il protocollo funziona correttamente
 DUP = il protocollo genera un messaggio duplicato
 LOST = il protocollo perde un messaggio

Figura 7.14: Le diverse combinazioni di strategie per client e server.

7.2.7 Protocolli a Finestra Scorrevole

Dopo avere esaminato in dettaglio la costituzione e il rilascio della connessione, vediamo come vengono gestite le connessioni durante il loro utilizzo. I punti chiave del problema sono il controllo degli errori e il controllo di flusso. Il controllo degli errori si assicura che i dati vengano consegnati con il livello di affidabilità desiderato; normalmente ciò vuol dire che tutti i dati siano consegnati senza alcun errore. Il controllo del flusso impedisce che un trasmittente veloce sovraccarichi un ricevente lento.

Entrambi questi problemi si sono già visti prima quando abbiamo studiato il livello data link:

1. Un frame trasporta un codice di rilevamento di errore (come un CRC o un checksum) usato per verificare se l'informazione sia stata ricevuta correttamente.
2. Un frame trasporta un numero di sequenza che lo identifica e viene ritrasmesso dal mittente finché non riceve dal destinatario un acknowledgement di ricezione riuscita. Questo procedimento è chiamato **ARQ** (*automatic repeat request*).
3. Esiste un numero massimo di frame a cui il mittente permetterà di restare in sospeso in ogni momento (finestra), facendo pausa se il destinatario non manderà acknowledgement abbastanza velocemente. Se questo massimo è di un solo pacchetto il protocollo è chiamato stop-and-wait. Finestre più grandi consentono il pipelining e migliorano le prestazioni su linee lunghe e veloci.
4. Un protocollo a **finestra scorrevole** (*sliding window*) combina queste caratteristiche ed è anche utilizzato per trasferimenti di dati bidirezionali.

Visto che questi meccanismi sono utilizzati su frame a livello data link, è naturale chiedersi perché vengano utilizzati anche su segmenti a livello di trasporto. Tuttavia, nella pratica c'è una piccola duplicazione tra i livelli di trasporto e data link; anche se vengono utilizzati gli stessi meccanismi ci sono differenze a livello di funzionalità e grado.

Per quanto riguarda la differenza funzionale consideriamo il rilevamento dell'errore. Il checksum del livello data link protegge un frame mentre percorre un solo collegamento, mentre il checksum del livello di trasporto protegge un segmento mentre attraversa l'intero percorso di rete. Si tratta di un controllo end-to-end, che è diverso da un controllo su ogni collegamento. Saltzer et al. (1984) descrivono una situazione in cui i pacchetti sono stati corrotti in un router. I checksum del livello data link proteggevano

i pacchetti solo mentre passavano attraverso un collegamento, non mentre erano all'interno del router. Perciò i pacchetti erano consegnati in modo errato anche se erano corretti relativamente ai controlli su ogni singolo collegamento.

Questo e altri esempi hanno portato Saltzer et al. ad articolare un **end-to-end argument** (*argomentazione end-to-end*). Secondo questo ragionamento, il controllo del livello di trasporto che funziona in modo end-to-end è essenziale per la correttezza; i controlli a livello data link non sono essenziali, ma sono preziosi per assicurare le prestazioni (poiché senza di loro un pacchetto danneggiato può essere inviato inutilmente lungo l'intero percorso).

Come differenza in grado si considerino le ritrasmissioni e il protocollo a finestra scorrivole. Molte linee wireless e collegamenti satellitari possono avere un solo frame in sospeso da un mittente in un dato istante. Cioè, il prodotto tra ritardo e larghezza di banda per il collegamento è così piccolo che neanche un frame può esservi memorizzato. In questo caso, una finestra piccola è sufficiente per delle buone prestazioni. Per esempio, 802.11 utilizza un protocollo stop-and-wait, trasmettendo o ritrasmettendo ogni frame e aspettando l'acknowledgement prima di spostarsi a quello successivo. Avere una finestra più grande di un frame aggiungerebbe complessità senza migliorare le prestazioni. Per linee cablate, come Ethernet (commutata) o dorsali di ISP, la percentuale d'errore è abbastanza bassa da poter omettere le ritrasmissioni del livello data link, poiché le ritrasmissioni end-to-end porranno rimedio alle minime perdite di frame.

D'altra parte molte connessioni TCP hanno un prodotto banda-ritardo molto maggiore di un singolo segmento. Si consideri una connessione che invia dati attraverso gli Stati Uniti a 1 Mbps con un tempo di andata e ritorno (*round-trip time*) di 100 ms. Anche con questa connessione lenta, 200 Kbit di dati saranno memorizzati dal destinatario nel tempo che occorre per spedire un segmento e ricevere un acknowledgement. Per queste situazioni deve essere usata una finestra grande. Lo stop-and-wait azzopperà le prestazioni. Nel nostro esempio le limiterà a un segmento ogni 200 ms o 5 segmenti al secondo, indipendentemente da quanto sia veloce in realtà la rete.

7.3 Il Protocollo di Trasporto Internet Senza Connessione: UDP

Nel livello di trasporto Internet possiede due protocolli principali: un protocollo non orientato alla connessione e uno orientato alla connessione. I protocolli si completano l'un l'altro. Il protocollo non orientato alla connessione è UDP; non fa praticamente altro che spedire pacchetti tra le applicazioni, permettendo loro di costruirsi sopra i propri protocolli come necessario. Il protocollo orientato alla connessione è TCP. Fa praticamente tutto: crea connessioni e aumenta l'affidabilità con le ritrasmissioni, implementa anche il controllo di flusso e il controllo di congestione, tutto per conto delle applicazioni che lo usano.

La suite di protocolli di Internet supporta un protocollo di trasporto non orientato alla connessione chiamato **UDP** (*user datagram protocol*, protocollo per datagrammi utente), descritto nell'RFC 768. UDP offre alle applicazioni un modo per inviare datagrammi senza dover stabilire una connessione.

UDP trasmette **segmenti** costituiti da un'intestazione di 8 byte seguita dal payload. L'intestazione è mostrata nella Figura 7.15. Le due **porte** servono per identificare gli endpoint all'interno dei computer di sorgente e destinazione. Quando arriva un pacchetto UDP, il suo payload è consegnato al processo associato alla porta di destinazione. Questa associazione avviene con l'utilizzo della primitiva BIND o qualcosa di simile. Pensate alle porte come a delle caselle postali che le applicazioni possono noleggiare per ricevere pacchetti. Avremo molto di più da dire quando descriveremo TCP, visto che anch'esso fa uso di porte. In effetti il vantaggio principale dell'utilizzo di UDP rispetto a IP è l'aggiunta delle porte di sorgente e destinazione. Senza i campi per le porte, il livello di trasporto non saprebbe che cosa farsene dei pacchetti in arrivo. Grazie a tali campi, invece, esso consegna il segmento all'interno del frame alla corretta applicazione.

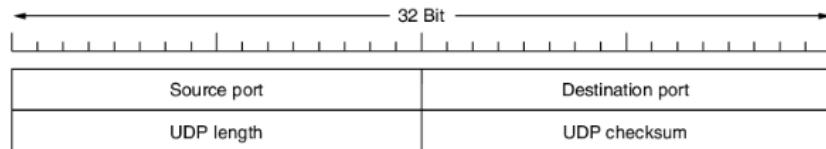


Figura 7.15: L'intestazione di UDP.

La porta sorgente serve principalmente quando si deve inviare una risposta al mittente. Copiando dal segmento in ingresso il campo *Source port* (porta sorgente) nel campo *Destination port* (porta destinazione) del segmento in uscita, il processo che invia la risposta può specificare il processo sulla macchina sorgente che deve riceverla.

Il campo *UDP length* (lunghezza UDP) include l'intestazione di 8 byte e tutti i dati. La lunghezza minima, per comprendere l'intestazione, è di 8 byte. La lunghezza massima è di 65.515 byte, che è inferiore al numero più grande esprimibile con 16 bit a causa del limite di dimensione imposto sui pacchetti IP.

È fornito anche un campo *Checksum* opzionale per migliorare l'affidabilità. Veri ca la somma dell'intestazione, dei dati e di una pseudointestazione concettuale IP. Quando viene fatta questa somma, il campo *Checksum* è impostato a zero e il campo dati viene prolungato con un ulteriore byte a zero se il numero di byte della sua lunghezza è dispari. L'algoritmo per il calcolo del checksum prevede semplicemente di sommare tutti i gruppi di 16 bit in complemento a uno e prenderne il risultato. Di conseguenza, quando un destinatario fa il calcolo sull'intero segmento includendo il campo *Checksum*, il risultato dovrebbe essere 0. Se il checksum non è calcolato, è salvato come uno 0, perché per una felice coincidenza dell'aritmetica in complemento a uno un risultato calcolato che ha valore 0 viene memorizzato come tutti bit a 1. Tuttavia non calcolarlo è stupido, a meno che la qualità dei dati sia poco importante (come per la voce digitalizzata). La pseudointestazione per il caso di IPv4 è mostrata nella Figura 7.16. Contiene l'indirizzo IPv4 di 32 bit delle macchine sorgente e destinazione, il numero di protocollo per UDP (17) e il numero di byte del segmento UDP (inclusa l'intestazione). È diverso ma analogo a IPv6. Includere la pseudointestazione di UDP nel calcolo del checksum aiuta a trovare i pacchetti consegnati in modo scorretto, ma includendola si viola anche la gerarchia dei protocolli, in quanto l'indirizzo IP al suo interno appartiene al livello IP e non al livello UDP. Anche TCP usa la stessa pseudointestazione per il suo checksum.



Figura 7.16: La pseudointestazione IPv4 inclusa nel checksum UDP.

Vale probabilmente la pena menzionare esplicitamente alcune delle cose che UDP non può fare. Il protocollo non si occupa del controllo di usso, del controllo di congestione o della ritrasmissione dopo la ricezione di un segmento errato; questi compiti sono lasciati ai processi utente. UDP si occupa invece di fornire un'interfaccia al protocollo IP, con la caratteristica aggiunta del demultiplexing di più processi utilizzando le porte e optionalmente del rilevamento di errori end-to-end; questo è tutto ciò che fa.

Per le applicazioni che necessitano di esercitare un controllo preciso sul usso di pacchetti, il controllo degli errori o la temporizzazione, UDP fornisce solo ciò che ha ordinato il dottore. Un'area in cui è particolarmente utile è in alcune comunicazioni client-server. Spesso il client spedisce una richiesta breve al server e si aspetta di ritorno una risposta breve. Se la richiesta o la risposta vengono perse, il client può solo aspettare lo scadere di un timer e provare di nuovo. Non solo il codice è più semplice da scrivere, ma sono richiesti meno messaggi (uno in ogni direzione) rispetto a protocolli che richiedono un setup iniziale, come TCP.

Un'applicazione che utilizza UDP in questo modo è **DNS** (*domain name system*), che studieremo nel Capitolo 7. In breve un programma che deve cercare l'indirizzo IP corrispondente a un host, per esempio *www.cs.berkeley.edu*, può inviare un pacchetto UDP contenente il nome dell'host a un server DNS. Il server risponde con un pacchetto UDP contenente l'indirizzo IP dell'host. Non è necessaria una preparazione della connessione e nemmeno un successivo rilascio; sono su canti due messaggi trasmessi sulla rete.

7.3.1 Remote Procedure Call: RCP

In un certo senso, l'invio di un messaggio a un host remoto e la ricezione di una risposta corrispondono all'esecuzione di una chiamata a una funzione in un linguaggio di programmazione. In entrambi i casi si inizia con uno o più parametri, ottenendo alla fine un risultato. Questa osservazione ha portato al tentativo di organizzare le interazioni richiesta/risposta sulle reti per implementarle come chiamate a procedure. Tale disposizione facilita la programmazione e la gestione delle applicazioni di rete. Si immagini per esempio una procedura denominata *get_IP_address (host_name)* che funziona tramite l'invio di un pacchetto UDP a un server DNS, l'attesa di una risposta, il timeout e un nuovo tentativo se la risposta non arriva abbastanza rapidamente. In questo modo tutti i dettagli della rete possono essere nascosti al programmatore.

L'idea alla base di RPC è eseguire una chiamata a procedura remota in maniera più simile possibile a una locale. Nella forma più semplice, per chiamare una procedura remota, il programma client deve essere associato a una piccola procedura di libreria, chiamata client stub, che rappresenta la procedura del server nello spazio di indirizzamento del client. Allo stesso modo il server è associato a una procedura chiamata **server stub**. Queste procedure nascondono il fatto che la chiamata a procedura dal client al server non sia locale.

7.3.2 Protocolli di Trasporto Real-Time

La comunicazione RPC client/server è una delle aree dove UDP è molto di uso; un'altra riguarda le applicazioni multimediali real-time. In particolare, mano a mano che radio e telefonia su Internet, musica e video-on-demand, videoconferenze e altre applicazioni multimediali si diffondevano, la gente scopriva che ogni applicazione reinventava più o meno lo stesso protocollo di trasporto real-time. Gradualmente si è capito che l'utilizzo di un protocollo di trasporto generico real-time per più applicazioni sarebbe stato una buona idea.

Nacque così **RTP** (*real-time transport protocol*).³ È descritto nell'RFC 3350 e ormai è ampiamente utilizzato per le applicazioni multimediali. Descriveremo due aspetti del trasporto real-time. Il primo è il protocollo RTP per il trasporto di dati audio e video. Il secondo è l'elaborazione che viene eseguita, principalmente dal ricevente, per riprodurre l'audio e il video nel momento giusto. Queste funzioni si inseriscono nello stack di protocolli come illustrato nella Figura 7.17.

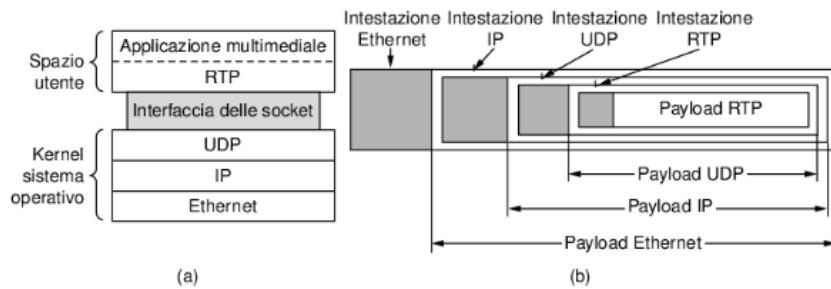


Figura 7.17: (a) La posizione di RTP nello stack dei protocolli. (b) La mappatura dei payload.

Di questa struttura è difficile affermare in quale livello si trovi RTP. Dal momento che viene eseguito nello spazio dell'utente ed è collegato al programma applicativo, certamente è simile a un protocollo del livello applicazione. D'altra parte si tratta di un protocollo generico indipendente dall'applicazione e che

fornisce funzionalità di trasporto; pertanto somiglia anche a un protocollo di trasporto. Probabilmente la descrizione migliore è che si tratta di un protocollo di trasporto implementato nel livello applicazione.

7.3.2.1 RTP

La funzione base di RTP è eseguire il multiplexing di flussi di dati real-time in un singolo flusso di pacchetti UDP. Il flusso UDP può essere inviato a una singola destinazione (unicasting) o a più destinazioni (multicasting). Dal momento che RTP utilizza il normale UDP, i suoi pacchetti non sono trattati in modo speciale dai router, a meno che siano attivate alcune delle normali funzionalità per la qualità del servizio di IP. In particolare non vi sono garanzie speciali sulla consegna e i pacchetti possono essere persi, avere ritardi o essere danneggiati.

Il formato RTP contiene molte caratteristiche che aiutano i destinatari a lavorare con informazioni multimediali. A ogni pacchetto inviato in un flusso è assegnato un numero incrementato di 1 rispetto al suo predecessore; la numerazione serve alla destinazione per scoprire se mancano pacchetti. In assenza di un pacchetto, l'azione migliore che la destinazione può compiere dipende dall'applicazione: potrebbe essere quella di saltare un fotogramma se i pacchetti trasportano dati video o di approssimare il valore mancante per interpolazione in caso di dati audio. La ritrasmissione non è un'opzione praticabile, perché il pacchetto ritrasmesso arriverebbe probabilmente troppo tardi per essere utile. Come conseguenza, RTP non fa uso di acknowledgement o di meccanismi per richiedere la ritrasmissione.

Ogni payload RTP può contenere più campioni, codificati nel modo desiderato dall'applicazione. Per consentire la cooperazione, RTP definisce diversi profili (per esempio un singolo flusso audio) e può consentire l'utilizzo di più formati di codifica per ogni profilo. Per esempio un singolo flusso audio potrebbe essere codificato in campioni PCM di 8 bit a 8 kHz, oppure con codifica delta, predittiva, GSM, MP3 e così via. RTP offre un campo intestazione in cui la sorgente può specificare la codifica, ma non è coinvolto nell'esecuzione della codifica.

L'intestazione RTP è illustrata nella Figura 7.18. Consiste di tre gruppi di 32 bit e di alcune estensioni facoltative. Il primo gruppo contiene il campo *Version*, che attualmente corrisponde a 2. Speriamo che questa versione sia molto vicina a quella definitiva, visto che resta un solo numero di versione a disposizione (anche se in futuro si potrebbe decidere che il numero 3 vuol dire che il numero di versione è in uno dei gruppi di estensione).

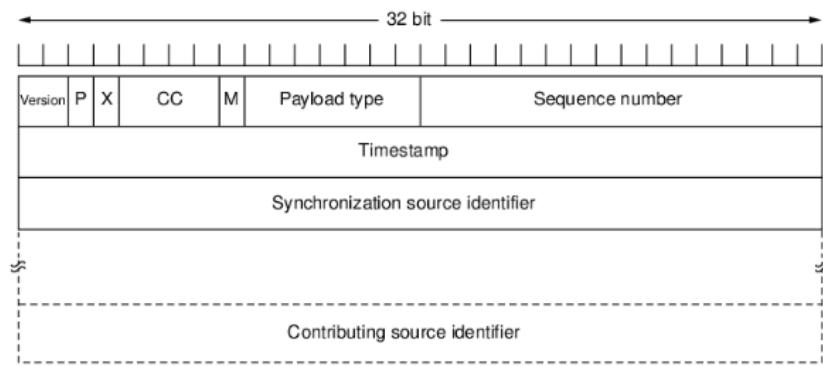


Figura 7.18: L'intestazione RTP.

1 bit *P* indica che il pacchetto è stato riempito no a ottenere un multiplo di 4 byte. L'ultimo byte del riempimento indica quanti byte sono stati aggiunti. Il bit *X* indica che è presente un'intestazione estesa. Il formato e il significato dell'intestazione estesa non sono definiti; l'unica regola è che il primo gruppo dell'estensione indica la lunghezza. Si tratta di una via di fuga per qualsiasi requisito non ancora previsto.

Il campo *CC* indica il numero di sorgenti attive presenti, da 0 a 15 (si veda di seguito). Il bit *M* è un bit di contrassegno specifico dell'applicazione. Può essere utilizzato per contrassegnare l'inizio di un fotogramma per il video, l'inizio di una parola in un canale audio o qualcosa'altro che l'applicazione sia in grado di comprendere. Il campo *Payload type* (tipo di payload) indica quale algoritmo di codifica è stato utilizzato (per esempio audio a 8 bit non compresso, MP3 e così via). Dal momento che ogni pacchetto

contiene questo campo, la codifica può cambiare durante la trasmissione. *Sequence number* (numero di sequenza) è un contatore incrementato a ogni invio di un pacchetto RTP; serve a rilevare i pacchetti persi.

Timestamp è prodotto da chi genera il flusso per indicare quando è stato creato il primo campione nel pacchetto. Questo valore può aiutare a ridurre la variabilità della temporizzazione, disaccoppiando l'istante di riproduzione da quello di arrivo del pacchetto. Il campo *Synchronization source identifier* (identificatore di sincronizzazione di flusso) specifica a quale flusso appartiene il pacchetto e rappresenta l'informazione utilizzata per il multiplexing e il demultiplexing di più flussi dati in un singolo flusso di pacchetti UDP. Per finire, i campi facoltativi *Contributing source identifier* (identificatore di sorgente attiva) sono utilizzati da apparecchiature di studio quali i mixer: il mixer è la sorgente della sincronizzazione e l'elenco dei flussi combinati si trova qui.

7.3.2.2 RTCP

RTP si accompagna a un protocollo chiamato RTCP (real-time transport control protocol). È de facto insieme a RTP nell'RFC 3550 e gestisce le retroazioni verso la sorgente, la sincronizzazione e l'interfaccia utente. Non trasporta campioni multimediali. La prima funzione può essere utilizzata per fornire alla sorgente del flusso una retroazione (feedback) su ritardi, jitter, larghezza di banda, congestione e altre proprietà di rete. Queste informazioni possono servire al processo di codifica per aumentare la velocità dei dati (e fornire una qualità superiore) quando la rete funziona bene e per ridurla quando vi sono dei problemi. Fornendo un feedback continuo, gli algoritmi di codifica si possono adattare costantemente per fornire la migliore qualità permessa dalle circostanze. Per esempio, se la larghezza di banda aumenta o diminuisce durante la trasmissione, la codifica può passare da MP3 a PCM 8 bit, no alla codifica delta secondo necessità. Il campo *Payload type* viene utilizzato per comunicare alla destinazione l'algoritmo di codifica utilizzato per il pacchetto corrente, in modo che sia possibile variarlo su richiesta. Un problema nel fornire feedback è che i rapporti RTCP vengono spediti a tutti i partecipanti. Per un'applicazione multicast con un folto gruppo di partecipanti, la quantità di banda utilizzata da RTCP potrebbe aumentare velocemente. Per prevenirlo, i mittenti RTCP diminuiscono il tasso dei loro rapporti per consumare collettivamente non più di, più o meno, il 5% conoscere la banda utilizzata che gli viene data dal mittente e il numero dei partecipanti che stima in base agli altri rapporti RTCP. RTCP gestisce anche la sincronizzazione tra ussi. Il problema è che ussi diversi possono utilizzare clock diversi, con granularità e velocità di deriva diverse. RTCP può essere utilizzato per mantenere la sincronia. Inoltre, RTCP fornisce un metodo per denominare le sorgenti (per esempio con testo ASCII). Queste informazioni possono essere visualizzate sullo schermo del destinatario per indicare chi sta parlando al momento.

7.4 Il Protocollo di Trasporto Internet Orientato alla Connessione: TCP

UDP è un protocollo semplice con alcuni utilizzi molto importanti, come le interazioni client-server e le trasmissioni multimediali, ma per la maggior parte delle applicazioni Internet è necessaria una consegna affidabile e in sequenza che UDP non può fornire, pertanto è richiesto un altro protocollo. È chiamato TCP ed è il principale motore di Internet.

TCP (*transmission control protocol*, protocollo di controllo della trasmissione) è stato progettato appositamente per fornire un flusso di byte affidabile end-to-end su una internetwork inaffidabile. Una internetwork differisce da una singola rete perché le diverse parti possono avere topologie, larghezze di banda, ritardi, dimensioni di pacchetti e altri parametri completamente differenti tra loro. TCP è stato progettato per adattarsi dinamicamente alle proprietà della internetwork e per continuare a offrire solide prestazioni in presenza di molti tipi di errore.

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 si interfaccia con il livello IP. Un'entità TCP accetta dai processi locali i flussi dati dell'utente; li suddivide in pezzi di dimensione non superiore a 64 KB (nelle applicazioni pratiche sono quasi sempre 1.460 byte di dati, in modo che stiano in un singolo frame Ethernet con le intestazioni IP e TCP); invia in serie ogni pezzo in un datagramma IP autonomo. I datagrammi contenenti i dati TCP che arrivano a un computer vengono consegnati all'entità TCP che ricostruisce i flussi di byte originali.

Il livello IP non garantisce la consegna senza errori dei datagrammi come pure non fornisce alcuna indicazione sulla velocità con cui è possibile inviarli. È compito di TCP spedire i datagrammi abbastanza velocemente da utilizzare la capacità di trasferimento senza creare congestione e ritrasmettere tutti i datagrammi che non sono stati consegnati. I datagrammi potrebbero anche arrivare nell'ordine sbagliato; è sempre compito di TCP riassemblare i messaggi nella giusta sequenza. Riassumendo, TCP deve fornire delle buone prestazioni con l'affidabilità che la maggior parte delle applicazioni desidera e che IP non offre.

7.4.1 Il Modello di Servizi

Il servizio TCP è ottenuto con la creazione di punti terminali di un sistema di comunicazione da parte di mittente e ricevente, chiamati **socket**, come già visto nel Paragrafo 6.1.3. Ogni socket possiede un numero (un indirizzo) composto dall'indirizzo IP dell'host e da un numero di 16 bit locale all'host, chiamato **porta**. Una porta è il nome TCP per un TSAP. Per ottenere il servizio TCP si deve stabilire esplicitamente una connessione tra una socket su una macchina e una socket su un'altra macchina. Le chiamate alle socket sono elencate nella Figura 7.5.

Una socket può essere usata per più connessioni contemporaneamente. In altre parole, due o più connessioni possono terminare alla stessa socket. Le connessioni sono individuate dagli identificatori di socket a entrambe le estremità, vale a dire dalla coppia (*socket1, socket2*). Non vengono utilizzati numeri di circuito virtuale o altri identificatori.

I numeri di porta minori di 1024 sono riservati per servizi standard, che normalmente possono essere erogati solo da utenti privilegiati (come root nei sistemi UNIX). Le porte in questo sottoinsieme prendono il nome di **well-known port** (*porte ben note*). Per esempio, qualsiasi processo che desidera stabilire una connessione a un host per recuperare la mail può connettersi alla porta 143 per contattare il server (che, in gergo, prende il nome di demone, *daemon*) IMAP. L'elenco delle porte ben note si può ottenere da www.iana.org; ne sono state assegnate più di 700. Alcune tra le più conosciute sono elencate nella Figura 7.19.

Porta	Protocollo	Utilizzo
20, 21	FTP	Trasferimento di file
22	SSH	Login remoto, rimpiazzamento di Telnet
25	SMTP	Posta elettronica
80	HTTP	World Wide Web
110	POP-3	Accesso remoto alla posta elettronica
143	IMAP	Accesso remoto alla posta elettronica
443	HTTPS	Web sicuro (HTTP su SSL/TLS)
543	RTSP	Controllo di riproduttori multimediali
631	IPP	Condivisione di stampanti

Figura 7.19: Alcune porte assegnate

In fase di avvio del sistema operativo sarebbe certamente possibile associare il demone FTP alla porta 21, il demone SSH alla porta 22 e così via; tuttavia in questo modo si occupa la memoria con demoni che rimangono inattivi per la maggior parte del tempo. In genere si preferisce invece avviare un singolo demone che, per motivi storici, nei sistemi UNIX viene chiamato **inetd** (*internet daemon*), associato a più porte per attendere la prima connessione in ingresso. Quando questa si verifica, *inetd* genera un nuovo processo ed esegue il demone appropriato, consentendogli di gestire la richiesta. In questo modo i demoni diversi da *inetd* sono attivi solo quando hanno del lavoro da compiere. Inetd apprendeva quali porte utilizzare da un file di configurazione; di conseguenza l'amministratore di sistema poteva configurare il computer perché disponesse di demoni permanenti sulle porte più usate (per esempio la porta 80) mentre *inetd* si occupava delle altre.

Tutte le connessioni TCP sono di tipo full-duplex punto a punto. Full-duplex indica che il traffico può procedere in entrambe le direzioni contemporaneamente. Punto a punto significa che ogni connessione ha esattamente due punti terminali. TCP non supporta il multicast o il broadcast.

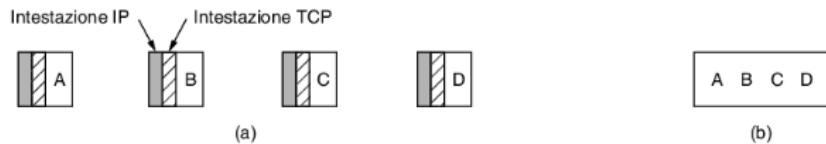


Figura 7.20: (a) Quattro segmenti di 512 byte inviati come datagrammi IP separati. (b) I 2.048 byte di dati consegnati all'applicazione in risposta a una singola chiamata READ.

TCP, quando un'applicazione gli passa i dati a sua discrezione può inviarli immediatamente o inserirli in un buffer per raccoglierne una quantità maggiore da inviare tutta insieme. A volte può succedere che l'applicazione desideri realmente un invio immediato dei dati. Si supponga, per esempio, che l'utente di un gioco interattivo voglia mandare un flusso di aggiornamenti. È essenziale che gli aggiornamenti siano spediti immediatamente e non inseriti in un buffer. Per forzare l'uscita dei dati, TCP possiede un flag PUSH in ogni pacchetto il cui intento originale era permettere alle applicazioni di indicare alle varie implementazioni di TCP di non ritardare la trasmissione; tuttavia le applicazioni non sono in grado di impostare il flag PUSH quando inviano i dati e vari sistemi operativi hanno sviluppato diverse opzioni per accelerare la trasmissione (come TCP_NODELAY in Windows e Linux).

Per gli archeologi di Internet menzioneremo anche una caratteristica interessante del servizio TCP prevista nel protocollo, ma usata raramente: i **dati urgenti**. Quando un'applicazione ha dati ad alta priorità che devono essere elaborati immediatamente, come accade, per esempio, se un utente interattivo digita CTRL-C per interrompere un processo di calcolo remoto già iniziato, l'applicazione di spedizione può mettere delle informazioni di controllo nel flusso di dati e darlo a TCP con il flag URGENT. Questa azione ferma l'accumulo di dati di TCP e fa trasmettere immediatamente tutto ciò che ha per quella connessione. La fine dei dati urgenti è contrassegnata in modo che l'applicazione sappia dove terminano, mentre l'inizio dei dati urgenti non è contrassegnato ed è compito dell'applicazione calcolarlo.

Una funzionalità vitale di TCP, che guida la struttura del protocollo, consiste nel fatto che ogni byte in una connessione TCP ha un proprio numero di sequenza a 32 bit. Le entità TCP di invio e ricezione scambiano i dati sotto forma di segmenti. Un **segmento TCP** consiste di un'intestazione fissa di 20 byte (più una parte facoltativa) seguita da zero o più byte di dati. Il software TCP decide la dimensione dei segmenti e può accumulare in un segmento i dati provenienti da più invii oppure dividere i dati di un invio in più segmenti. I limiti sulla dimensione del segmento sono due: ogni segmento, compresa l'intestazione TCP, deve essere contenuto nel payload IP di 65.535 byte e ogni collegamento ha una **MTU** (*maximum transfer unit*, unità di trasferimento massima). Ogni segmento deve essere contenuto nella MTU sia lato mittente sia lato ricevente, in maniera tale che possa essere spedito e ricevuto in un singolo pacchetto non frammentato.

Il protocollo di base utilizzato dalle entità TCP è il protocollo a finestra scorrevole con una dimensione dinamica della finestra. Quando un mittente trasmette un segmento, avvia anche un timer. Quando il segmento arriva a destinazione, l'entità TCP ricevente invia un segmento (con i dati, se esistono, oppure senza) contrassegnato da un numero di acknowledgement uguale al numero di sequenza successivo che prevede di ricevere la dimensione della finestra disponibile. Se il timer del mittente scade prima della ricezione dell'acknowledgement, questi ritrasmette il segmento.

7.4.2 Intestazione del Segmento TCP

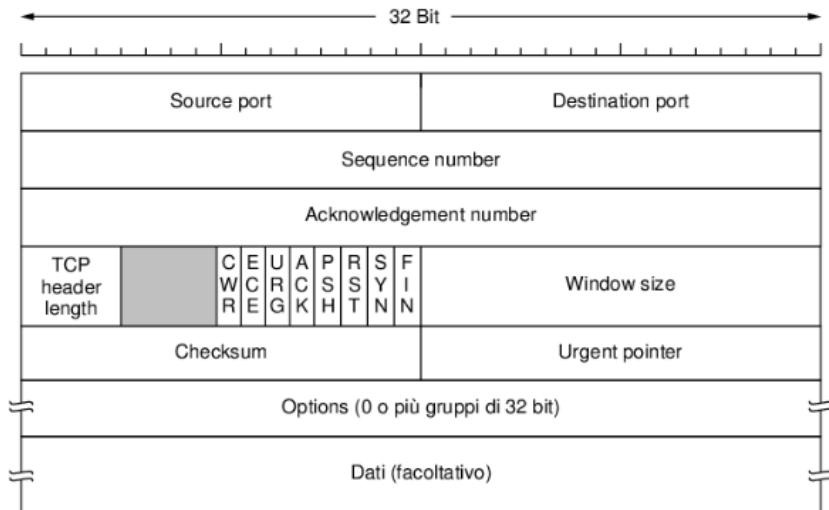


Figura 7.21: L'intestazione TCP.

Analizziamo l'intestazione TCP campo per campo. I campi *Source port* (porta sorgente) e *Destination port* (porta destinazione) identificano gli estremi locali della connessione. Una porta TCP più l'indirizzo IP del suo host formano un unico punto terminale a 48 bit. I punti terminali di sorgente e destinazione insieme identificano la connessione. Questo identificatore di connessione è una quintupla perché consiste di cinque pezzi di informazione: il protocollo (TCP), l'IP e porta della sorgente e l'IP e porta della destinazione.

I campi *Sequence number* (numero di sequenza) e *Acknowledgement number* (numero di acknowledgement) svolgono le loro solite funzioni. Occorre notare che il secondo specifica il successivo byte previsto, non l'ultimo byte ricevuto correttamente; si tratta di un **acknowledgement cumulativo**, perché riassume i dati ricevuti con un solo numero e non va al di là dei dati persi. Entrambi sono numeri di 32 bit, perché ogni byte di dati è numerato un uno stream TCP.

Il campo *TCP header length* (*lunghezza intestazione TCP*) indica quanti gruppi di 32 bit sono contenuti nell'intestazione TCP. L'informazione è necessaria perché il campo Options (opzioni) ha una lunghezza variabile e, di conseguenza, anche l'intestazione. Tecnicamente, questo campo indica l'inizio dei dati all'interno del segmento, misurato in gruppi di 32 bit; ma questo numero rappresenta praticamente la lunghezza dell'intestazione, che è poi la stessa cosa.

Segue un campo di 4 bit inutilizzato. Il fatto che questi bit siano rimasti inutilizzati per 30 anni (e solo 2 dei 6 originali siano stati riutilizzati) testimonia quanto TCP sia ben studiato. Un protocollo studiato male li avrebbe già sfruttati per correggere bug presenti nella struttura originale.

Seguono otto flag di un bit. *CWR* e *ECE* sono usati per segnalare la congestione quando viene utilizzata **ECN** (*explicit congestion notification*), come specificato nell'RFC 3168. *ECE* è regolata per mandare un *ECN-Echo* a un mittente TCP per indicargli di rallentare quando il destinatario TCP riceve un'indicazione di congestione dalla rete. *CWR* è usato per segnalare una condizione di *congestion window reduced* (riduzione della finestra di congestione) dal mittente TCP al destinatario TCP così che esso sappia che il mittente ha rallentato e possa smettere di spedire degli *ECN-Echo*.

URG è impostato a 1 quando si usa *Urgent pointer* (puntatore urgente), che indica lo spiazzamento in byte (partendo dal numero di sequenza corrente) in cui si trovano i dati urgenti. Questa funzionalità è utilizzata al posto dei messaggi di interrupt e, come affermato in precedenza, è un metodo rudimentale per consentire al mittente di inviare segnali al ricevente senza coinvolgere TCP nel motivo dell'interrupt, ma è usato raramente.

Il bit *ACK* è impostato a 1 per indicare che *Acknowledgement number* (numero di acknowledgement) è valido e questo è il caso della quasi totalità dei pacchetti. Se *ACK* è 0, il segmento non contiene un acknowledgement, pertanto il campo *Acknowledgement number* viene ignorato.

Il bit *PSH* segnala la presenza di dati *PUSH*. Al ricevente viene gentilmente chiesto di consegnare i dati all'applicazione all'arrivo e di non archiviarli nel buffer per poi trasmetterle un buffer completo.

Il bit *RST* viene utilizzato per reimpostare una connessione che è diventata confusa a causa di un malfunzionamento dell'host o per altre ragioni. È anche utilizzato per rifiutare un segmento non valido o un tentativo di aprire una connessione. In generale, se si riceve un segmento con il bit *RST* attivato, si è di fronte a un problema.

Il bit *SYN* viene utilizzato per stabilire le connessioni. La richiesta di connessione presenta *SYN* = 1 e *ACK* = 0 per indicare che il campo *Acknowledgement* non è utilizzato. La risposta alla richiesta di connessione porta un acknowledgement, pertanto possiede *SYN* = 1 e *ACK* = 1. In pratica il bit *SYN* è utilizzato per segnalare entrambi CONNECTION REQUEST e CONNECTION ACCEPTED, mentre il bit *ACK* distingue tra le due possibilità.

Il bit *FIN* viene utilizzato per rilasciare una connessione. Specifica che il mittente non ha altri dati da trasmettere. Tuttavia, dopo avere chiuso una connessione, il processo di chiusura potrebbe continuare a ricevere dati all'in nito. Entrambi i segmenti *SYN* e *FIN* possiedono numeri di sequenza ed è quindi garantito che verranno elaborati nell'ordine corretto.

Il controllo di flusso in TCP è gestito con una finestra scorrevole a dimensione variabile. Il campo *Window size* (dimensione finestra) indica quanti byte possono essere inviati a partire da quello che ha ricevuto acknowledgement. Un campo *Window size* con valore 0 è ammesso e a firma che i byte no ad *Acknowledgement number* -1 compreso sono stati ricevuti, ma che il ricevente non ha avuto modo di consumarli e non desidera altri dati per il momento. Il ricevente può in seguito dare l'autorizzazione all'invio trasmettendo un segmento con lo stesso *Acknowledgement number* e un campo *Window size* diverso da zero.

In TCP gli acknowledgement e le autorizzazioni per inviare dati aggiuntivi sono completamente separati. In effetti un ricevente può affermare: "Ho ricevuto i byte fino a k compreso ma non ne voglio altri per ora". Questa divisione (di fatto una finestra a dimensione variabile) offre una flessibilità aggiuntiva, che studieremo in dettaglio in seguito. Per aumentare l'affidabilità viene anche fornito un *Checksum*; questo fa un controllo dell'intestazione, dei dati e di una pseudointestazione concettuale esattamente come UDP, tranne che la pseudointestazione ha un numero di protocollo per TCP (6) e il checksum è obbligatorio.

Il campo *Options (opzioni)* fornisce un modo per aggiungere funzionalità aggiuntive non previste dall'intestazione standard. Sono state definite molte opzioni e diverse di queste sono utilizzate comunemente. Le opzioni sono di lunghezza variabile, riempiono un multiplo di 32 bit occupando lo spazio extra con degli zeri e possono estendersi a 40 byte per accogliere la più lunga intestazione TCP che possa essere specificata. Alcune opzioni sono aggiunte quando viene stabilita una connessione per effettuare la negoziazione o per informare l'altro capo sulle funzionalità disponibili. Altre opzioni sono trasportate nei pacchetti lungo la durata della connessione. Ogni opzione ha una codifica tipo-lunghezza-valore. Un'opzione ampiamente utilizzata è quella che permette a ogni host di specificare il **MSS** (*maximum segment size*) che è disposto ad accettare.

7.4.3 Instaurazione di una Connessione

Le connessioni in TCP vengono stabilite mediante l'handshake a tre vie. Per stabilire una connessione, un lato (per esempio il server) attende in modo passivo una connessione in ingresso eseguendo nell'ordine le primitive LISTEN e ACCEPT che possono indicare una sorgente specifica oppure nessuna in particolare.

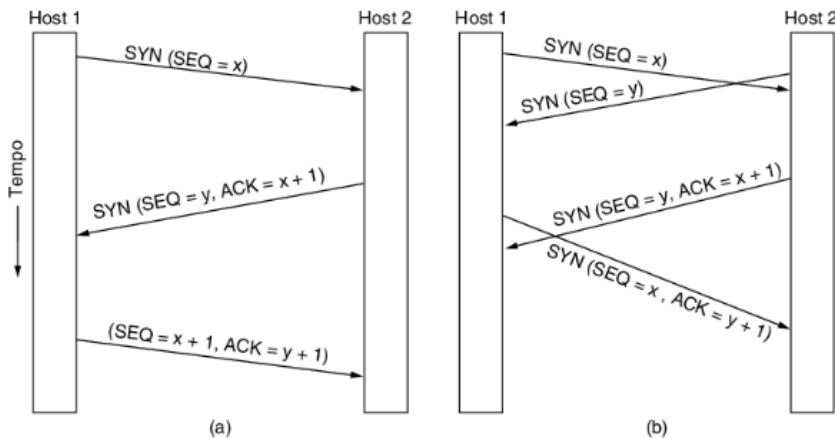


Figura 7.22: (a) Instaurazione di una connessione TCP nel caso normale. (b) Instaurazione simultanea da entrambi i lati.

L'altro lato (diciamo il client) esegue una primitiva CONNECT, specificando l'indirizzo IP e la porta a cui vuole connettersi, la dimensione massima del segmento TCP che è intenzionato ad accettare e, facoltativamente, alcuni dati utente (per esempio una password). La primitiva CONNECT invia un segmento TCP con il bit SYN a 1 e il bit ACK a 0, poi attende una risposta.

Quando questo segmento arriva a destinazione, l'entità TCP controlla se esiste un processo che ha eseguito una LISTEN sulla porta indicata nel campo Destination port e in caso negativo invia una risposta con il bit RST a 1 per rifiutare la connessione.

Se un processo è in ascolto sulla porta, gli viene dato il segmento TCP in ingresso; quindi può accettare o rifiutare la connessione. Se accetta, viene restituito al mittente un segmento di acknowledgement. La sequenza dei segmenti TCP inviata nel caso normale è mostrata nella Figura 7.22 (a). Si noti che un segmento *SYN* consuma un byte nello spazio delle sequenze, così da poter ricevere un acknowledgement non ambiguo.

Nel caso che due host tentino contemporaneamente di stabilire una connessione tra le stesse due socket, la sequenza di eventi è illustrata nella Figura 7.22 (b). Il risultato di questi eventi è la costituzione di una sola connessione, non due, perché le connessioni sono identificate dai loro punti terminali. Se sia la prima attivazione che la seconda generano una connessione identificata da (x, y) , viene comunque creata una sola voce nella tabella per (x, y) .

7.4.4 Rilascio di una Connessione

Anche se le connessioni TCP sono full-duplex, per comprendere come vengono rilasciate è meglio immaginarle come una coppia di connessioni simplex, in cui ogni connessione è rilasciata in modo indipendente dalla sua gemella. Per rilasciare una connessione, entrambe le parti possono inviare un segmento TCP con il bit *FIN* a 1, per indicare che non hanno più dati da trasmettere. Quando il *FIN* riceve l'acknowledgement, la direzione viene chiusa ai nuovi dati; tuttavia i dati possono continuare affluire indefinitamente nell'altra direzione. Quando entrambe le direzioni saranno state chiuse la connessione sarà rilasciata. Normalmente per rilasciare una connessione sono necessari quattro segmenti TCP, un *FIN* e un *ACK* per ogni direzione; tuttavia è possibile inserire il primo *ACK* e il secondo *FIN* nello stesso segmento, riducendo il totale a tre.

Per evitare il problema dei due eserciti si usano i timer: se una risposta a *FIN* non arriva entro il tempo di vita massimo di due pacchetti, il mittente del *FIN* rilascia la connessione.

7.4.5 Modello di Gestione della Connessione

Stato	Descrizione
CLOSED	Nessuna connessione è attiva o in sospeso
LISTEN	Il server è in attesa di una chiamata in ingresso
SYN RCVD	È arrivata una richiesta di connessione; in attesa di ACK
SYN SENT	L'applicazione ha iniziato ad aprire una connessione
<hr/>	
Stato	Descrizione
ESTABLISHED	Il normale stato di trasferimento dei dati
FIN WAIT 1	L'applicazione ha detto di aver terminato
FIN WAIT 2	L'altro lato ha accettato il rilascio
TIME WAIT	Attende la scadenza di tutti i pacchetti
CLOSING	Entrambi i lati hanno cercato di chiudere contemporaneamente
CLOSE WAIT	L'altro lato ha iniziato il rilascio
LAST ACK	Attende la scadenza di tutti i pacchetti

Figura 7.23: Gli stati utilizzati nella macchina a stati finiti per la gestione della connessione TCP.

I passi richiesti per stabilire e rilasciare le connessioni possono essere rappresentati in una macchina a stati finiti con gli 11 stati elencati nella Figura 7.23. In ogni stato sono ammessi alcuni eventi; quando si verifica un evento ammesso è possibile svolgere alcune azioni, mentre se si verificano altri eventi viene segnalato un errore.

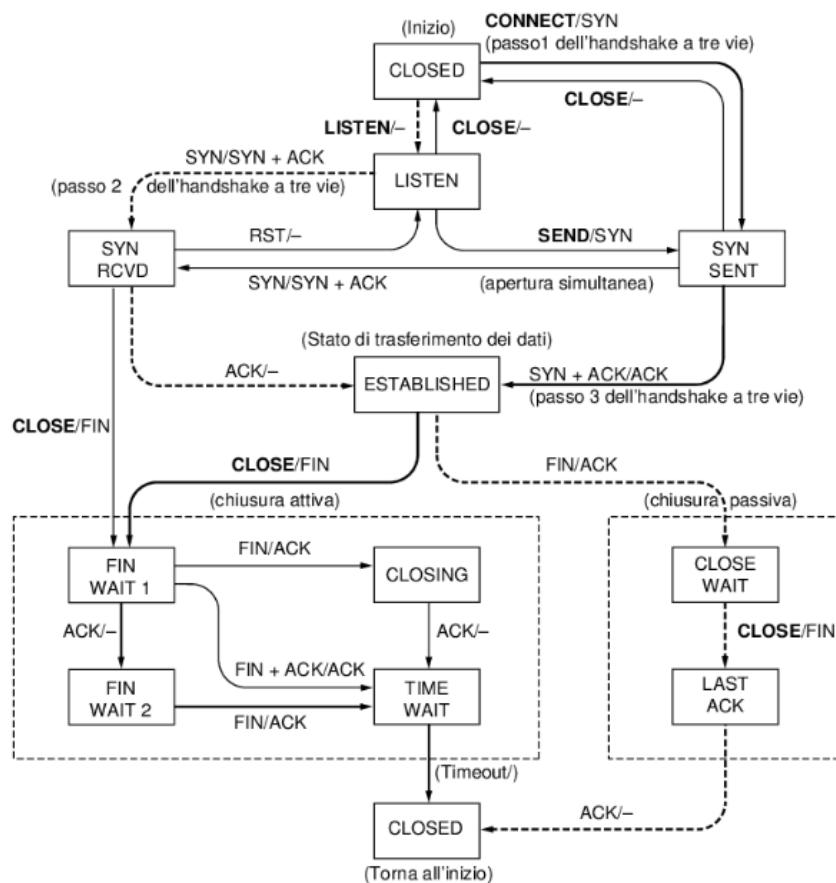


Figura 7.24: La macchina a stati finiti per la gestione di una connessione TCP.

La linea continua spessa è il percorso normale per un client; la linea tratteggiata spessa è il percorso normale per un server. Le linee sottili rappresentano eventi insoliti. Ogni transizione è etichettata con l'evento che la provoca e l'azione risultante, separati da una barra.

7.4.6 La Finestra Scorrevole

Si consideri una connessione a un terminale remoto tramite SSH o telnet che reagisce a ogni pressione di un tasto. Nel caso peggiore, quando un carattere arriva all'entità TCP di invio, TCP crea un segmento TCP di 21 byte che passa a IP affinché lo invii come datagramma IP di 41 byte. Dal lato ricevente TCP invia immediatamente un acknowledgement di 40 byte (20 byte di intestazione TCP e 20 byte di intestazione IP). In seguito, quando il terminale remoto ha letto il byte, TCP invia un aggiornamento della finestra, spostandola di 1 byte a destra; anche questo pacchetto è di 40 byte. Per finire, il terminale remoto, quando ha elaborato il carattere, lo restituisce per poterlo visualizzare localmente tramite un pacchetto di 41 byte. In tutto vengono utilizzati 162 byte di banda e quattro segmenti per ogni carattere digitato.

Un approccio usato da molte implementazioni di TCP per ottimizzare questa situazione prende il nome di **delayed acknowledgement** (*acknowledgement ritardato*). Utilizzato da molte implementazioni di TCP, prevede di ritardare gli acknowledgement e gli aggiornamenti della finestra di 500 ms, nella speranza di acquisire alcuni dati che possano “scroccare un passaggio”. Supponendo che il terminale visualizzi il carattere entro i 500 ms, occorre spedire un solo pacchetto di 41 byte verso il mittente, dimezzando il numero di pacchetti e l'utilizzo della banda.

Anche se l'acknowledgement ritardato riduce il carico di rete al destinatario, un mittente che spedisce molti pacchetti piccoli (come pacchetti da 41 byte contenenti 1 byte di dati) fa ancora un uso inefficiente della rete. Un modo per ridurre questo spreco è chiamato **algoritmo di Nagle** (Nagle, 1984). Ciò che Nagle suggerì è semplice: quando i dati arrivano al mittente in piccoli gruppi, è sufficiente inviare il primo gruppo e inserire il resto nel buffer finché arriva l'acknowledgement. A questo punto si possono inviare

tutte le informazioni nel buffer usando un solo segmento TCP e iniziare di nuovo a inserire i dati nel buffer no all'arrivo del prossimo acknowledgement. Questo vuol dire che solo un pacchetto (di piccole dimensioni) può essere in circolazione in ogni dato momento. Se molti dati vengono spediti dall'applicazione in un round-trip time, l'algoritmo di Nagle li metterà tutti in un segmento, riducendo notevolmente la banda utilizzata.

Un altro problema che può degradare le prestazioni di TCP è la **silly window syndrome** (sindrome della finestra stupida) (Clark, 1982). Questo problema si verifica quando i dati vengono passati all'entità TCP di invio in blocchi grandi, ma un'applicazione interattiva dal lato ricevente legge i dati 1 byte alla volta. Per comprendere il problema si osservi la Figura 6.41. Inizialmente il buffer TCP lato ricevente è pieno (ha una finestra con dimensione 0) e il mittente ne è a conoscenza. L'applicazione interattiva legge quindi un carattere dal usso TCP. Questa azione fa felice il TCP di ricezione, che invia un aggiornamento della finestra al mittente comunicandogli che può inviare 1 byte. Il mittente obbedisce e invia 1 byte. Ora il buffer è pieno, pertanto il ricevente manda un acknowledgement per il segmento di 1 byte, ma imposta la finestra a 0. Questo comportamento può andare avanti per sempre.

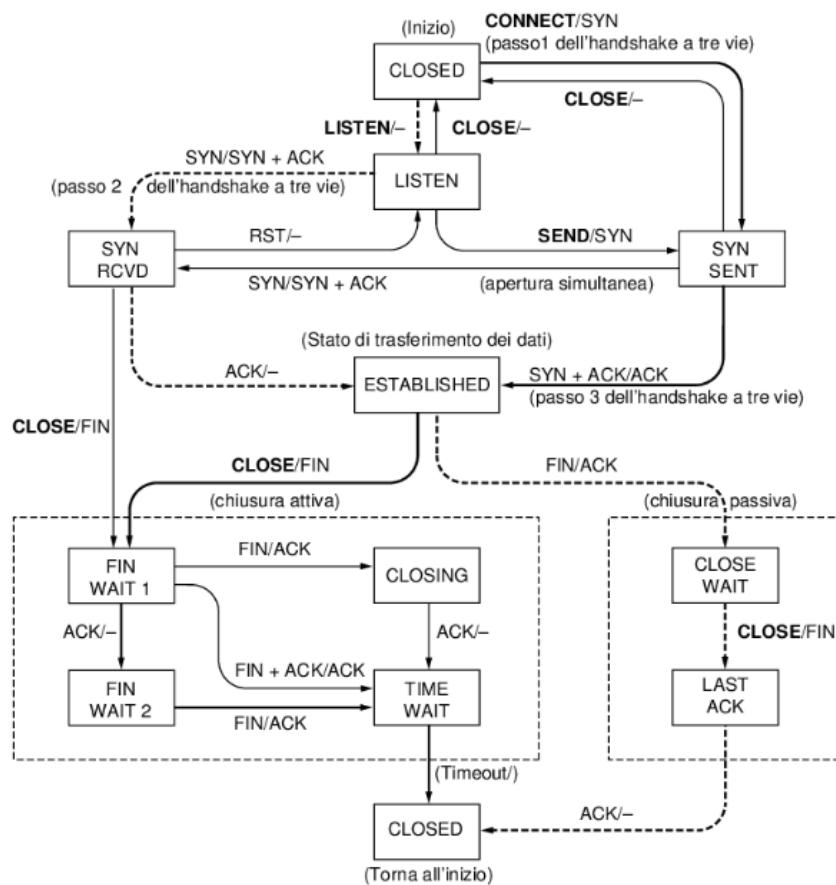


Figura 7.25: Silly window syndrome.

La soluzione di Clark sta nell'impedire che il ricevente invii un aggiornamento della finestra per 1 byte; invece è obbligato ad attendere la disponibilità di una certa quantità di spazio. Nello specifico il ricevente può inviare un aggiornamento della finestra solo quando è in grado di gestire la dimensione massima del segmento (concordata quando la connessione è stata instaurata) o quando il suo buffer è vuoto per metà. Inoltre il mittente può essere di aiuto se non invia segmenti di piccole dimensioni; dovrebbe invece provare ad attendere no a quando ha accumulato un segmento completo o almeno un segmento contenente dati per la metà della dimensione del buffer del ricevente.

7.4.7 Controllo della Congestione

Il controllo della congestione di TCP è basato sull'implementazione di questo approccio, utilizzando una finestra e interpretando la perdita di pacchetti come segnali binari di congestione. Per farlo, TCP man-

tiene una finestra di congestione la cui dimensione è pari al numero di byte che il mittente può avere sulla rete in qualsiasi momento. Il tasso corrispondente è la dimensione della finestra diviso per il round-trip time della connessione. TCP aggiusta la dimensione della finestra secondo le regole AIMD.

Si ricordi che la finestra di congestione viene mantenuta in aggiunta alla finestra di controllo del flusso che specifica il numero di byte che il destinatario può inserire nel suo buffer. Entrambe le finestre vengono seguite in parallelo e il numero di byte che possono essere spediti è il più piccolo delle due finestre. Perciò la finestra effettiva è la più piccola di quelle che il mittente e il destinatario credono vada bene.

Quando viene stabilita una connessione, il mittente inizializza la finestra di congestione a un valore iniziale basso, di al massimo quattro segmenti, l'uso di quattro segmenti è un incremento, sulla base dell'esperienza, da un primo valore iniziale di un solo segmento. Il mittente, quindi, spedisce la finestra iniziale. I pacchetti impiegheranno un round-trip time per ottenere i relativi acknowledgement. Per ogni segmento che ha ricevuto l'acknowledgement prima dello scadere del timer di ritrasmissione il mittente aggiunge alla finestra di congestione il valore in byte di un segmento; in più, quando quel segmento ha ricevuto un acknowledgement ci sarà un segmento in meno in rete. Il risultato è che ogni acknowledgement, permette a due segmenti di essere inviati. La finestra di congestione raddoppia a ogni round-trip time.

Questo algoritmo è chiamato **slow start** (*partenza lenta*), ma non è lento per niente (ha una crescita esponenziale) tranne se confrontato con l'algoritmo precedente che permette a un'intera finestra di controllo di flusso di essere spedita in una volta. Lo slow start è illustrato nella Figura 7.26; nel primo round-trip time il mittente immette un pacchetto nella rete (e il destinatario riceve un pacchetto), due pacchetti sono spediti nel successivo round-trip time e quindi quattro nel terzo round-trip time.

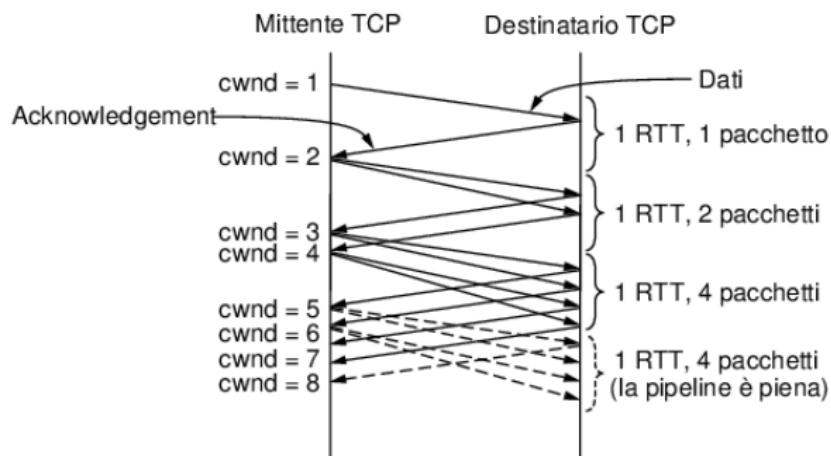


Figura 7.26: Slow start a partire da una finestra di congestione iniziale di un segmento.

Poiché lo slow start causa una crescita esponenziale, prima o poi (più prima che poi) spedirà in rete troppi pacchetti e troppo velocemente. Quando ciò avviene, si creano code nella rete e quando le code saranno piene uno o più pacchetti saranno persi. Dopo che questo accade, il mittente TCP andrà in timeout quando un acknowledgement non riuscirà ad arrivare in tempo. Per tenere sotto controllo lo slow start, il mittente mantiene una soglia per il collegamento chiamata **slow start threshold** (soglia di *slow start*). Inizialmente questo valore è fissato, in modo arbitrariamente elevato, alla dimensione della finestra di controllo di flusso, in maniera tale da non limitare la connessione. TCP continua a far crescere la finestra di congestione nello slow start finché non si verificherà un timeout o la finestra di congestione supererà la soglia (o si è riempita la finestra del destinatario).

Ogni volta che viene rilevata la perdita di un pacchetto, per esempio attraverso un timeout, la soglia di slow start è fissata per essere la metà della finestra di congestione e l'intero processo ricomincia. L'idea è che la finestra corrente sia troppo grande perché ha già causato una congestione che solo ora viene rilevata da un timeout. Metà della finestra, usata con successo in un momento precedente, è probabilmente una stima migliore per una finestra di congestione vicina alla capacità del percorso, ma che non causerà perdite.

Ogni volta che viene oltrepassata la soglia dello slow start, TCP passa dallo slow start all'incremento additivo; in questa modalità, la finestra di congestione viene aumentata di un segmento per ogni round-trip time. Questo, però, viene solitamente implementato con un aumento (più piccolo) per ogni segmento, per cui si riceve un acknowledgement invece di crescere a ogni RTT, come nel caso dello slow start. Indichiamo la finestra di congestione con $cwnd$ e la grandezza massima di segmento con MSS . Un'approssimazione di uso comune consiste nell'aumentare $cwnd$ di $(MSS \times MSS)/cwnd$ per ognuno dei $cwnd/MSS$ pacchetti che potrebbero ricevere un acknowledgement. Questo incremento non ha bisogno di essere veloce; l'idea è che una connessione TCP passi molto tempo con la propria finestra di connessione vicina al valore ottimale (non così piccola da rendere il throughput basso e non così grande da provocare congestione).

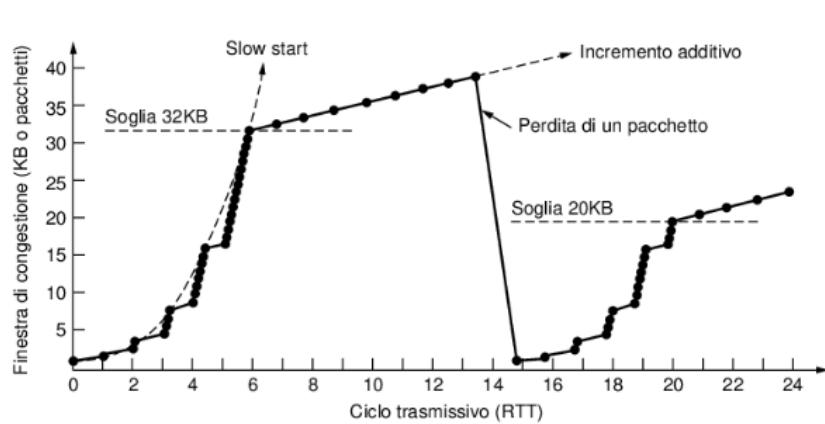


Figura 7.27: Slow start seguito da incremento additivo in TCP Tahoe.

TCP Tahoe (che include buoni timer di ritrasmissione) fornisce un algoritmo di controllo della congestione funzionante che risolve il problema del collasso per congestione. Jacobson capì che era possibile fare anche di meglio. Al momento della ritrasmissione veloce la connessione ha a disposizione una finestra di congestione troppo grande, ma anche un ack clock attivo. L'arrivo di ogni ulteriore acknowledgement duplicato vuol dire che un altro pacchetto ha lasciato la rete. Usare gli acknowledgement duplicati per contare i pacchetti in rete rende possibile spedire un nuovo pacchetto per ogni acknowledgement duplicato aggiuntivo.

Il **fast recovery** (*recupero veloce*) è l'euristica che implementa questo comportamento. È una modalità temporanea che cerca di mantenere l'ack clock in funzione con una finestra di congestione grande quanto la nuova soglia o nel momento della ritrasmissione veloce quanto metà del valore della finestra di congestione. Si rimane in modalità temporanea finché non ci si accorge che il numero di pacchetti in rete è sceso al di sotto della nuova soglia contando gli acknowledgement duplicati (inclusi i tre che hanno fatto scattare la ritrasmissione veloce); questa procedura impiega circa metà del round-trip time. Da quel momento in poi, un nuovo pacchetto può essere spedito per ogni acknowledgement duplicato ricevuto. Trascorso un round-trip time dopo la ritrasmissione veloce il pacchetto perduto avrà ricevuto un acknowledgement. A quel punto il uso di acknowledgement duplicati cesserà e la modalità di fast recovery sarà abbandonata. La finestra di congestione sarà impostata alla nuova soglia di slow start e aumenterà con crescita lineare.

Il risultato di questa euristica è che TCP evita lo slow start, tranne all'inizio della connessione e quando si verificano un timeout. Quest'ultimo può ancora verificarsi quando va perso più di un pacchetto e la ritrasmissione veloce non recupera adeguatamente. Invece di slow start multipli, la finestra di congestione della connessione in atto segue un prolo a dente di sega per gli incrementi additivi (di un segmento ogni RTT) e i decrementi moltiplicativi (di metà in un RTT). Questa è esattamente la regola AIMD che cercavamo di implementare

Due grandi cambiamenti hanno ulteriormente influenzato le implementazioni di TCP. Primo, molta della complessità di TCP è la deduzione da un flusso di acknowledgement duplicati di quali pacchetti siano arrivati e quali andati persi; il numero di acknowledgement cumulativo non fornisce questa infor-

mazione. Una soluzione semplice è fare uso di **SACK** (*selective acknowledgement*), che elenca no a tre intervalli di byte ricevuti. Con questa informazione il mittente può decidere più direttamente quali pacchetti ritrasmettere e tenere traccia dei pacchetti in viaggio per implementare la finestra di congestione. Quando mittente e destinatario stabiliscono una connessione, ognuno invia l'opzione *SACK permitted* per segnalare che entrambi capiscono gli acknowledgement selettivi. Una volta abilitato, il SACK per una connessione, funziona come mostrato nella Figura 7.28: un destinatario usa il campo TCP *Acknowledgement number* nel solito modo, come un acknowledgement cumulativo del più alto byte che sia stato ricevuto in ordine no a quel momento. Quando riceve il pacchetto 3 fuori sequenza (perché il pacchetto 2 è stato perso), spedisce una SACK *option* per i dati ricevuti insieme all'acknowledgement cumulativo (duplicato) del pacchetto 1. La SACK *option* fornisce gli intervalli di byte ricevuti e superiori al numero dato attraverso l'acknowledgement cumulativo. Il primo intervallo è il pacchetto che ha innescato l'acknowledgement duplicato. Gli intervalli successivi, se presenti, sono blocchi più vecchi e vengono usati comunemente no a tre intervalli. Quando viene ricevuto il pacchetto 6, due intervalli di byte SACK vengono usati per indicare che il pacchetto 6 e i pacchetti da 3 a 4 sono stati ricevuti, in aggiunta a tutti i pacchetti no al pacchetto 1. Dall'informazione di ciascuna SACK *option* che riceve, il mittente può decidere quali pacchetti ritrasmettere. In questo caso ritrasmettere i pacchetti 2 e 5 sarebbe una buona idea.

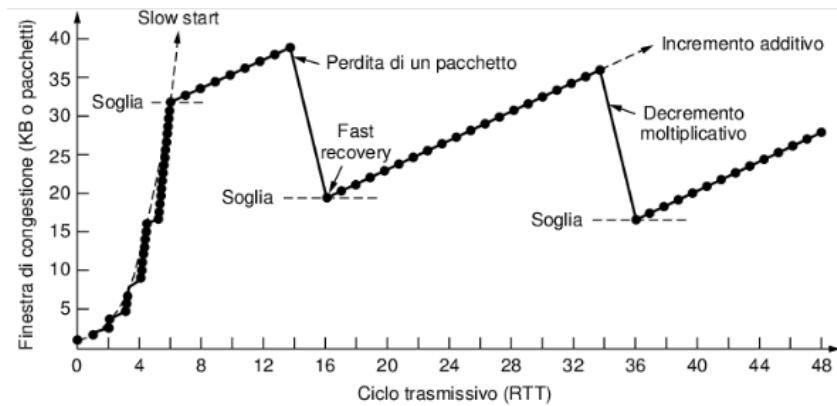


Figura 7.28: Fast recovery e sequenza a dente di sega di TCP Reno.

7.4.8 Gestione de Timer

TCP utilizza più timer (almeno concettualmente) per svolgere il proprio lavoro. Il più importante è quello che fa scattare il RTO (retransmission timeout, timeout di ritrasmissione). Quando viene inviato un segmento, si avvia un timer di ritrasmissione. Se il segmento riceve un acknowledgement prima della scadenza del timer, TCP lo ferma; se invece il timer scade prima dell'arrivo dell'acknowledgement, il segmento viene ritrasmesso (e il timer riavviato). La questione che si pone è: quanto dovrebbe essere lungo l'intervalllo di timeout?

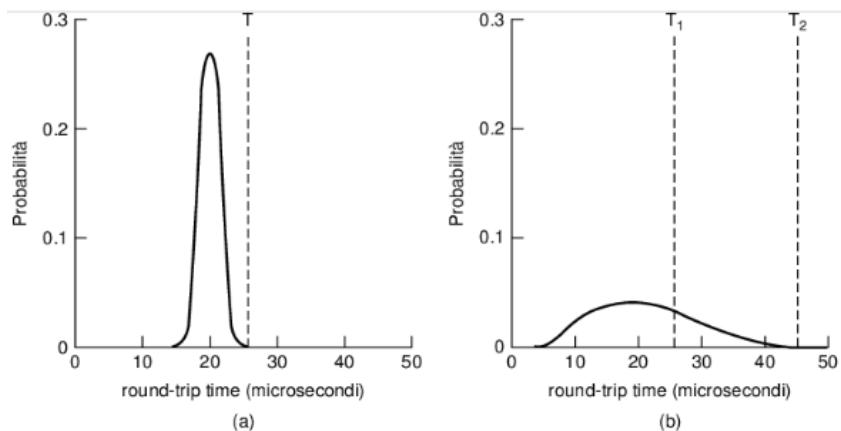


Figura 7.29: (a) Funzione di densità di probabilità dei tempi di arrivo degli acknowledgement nel livello data link. (b) Funzione di densità di probabilità dei tempi di arrivo degli acknowledgement TCP.

La soluzione è utilizzare un algoritmo dinamico che adatta costantemente l'intervallo di timeout in base a continue misurazioni delle prestazioni di rete. L'algoritmo utilizzato generalmente da TCP è dovuto a Jacobson (1988) e funziona come segue: per ogni connessione TCP mantiene una variabile, **SRTT** (*smoothed round-trip time*), che rappresenta la migliore stima corrente del round-trip time per la destinazione in questione. Quando viene inviato un segmento, si avvia un timer che serve a due scopi: per sapere quanto tempo richiede l'acknowledgement e per innescare un'eventuale ritrasmissione. Se l'acknowledgement torna indietro prima della scadenza del timer, TCP misura il tempo richiesto, diciamo R, e poi aggiorna SRTT secondo la formula

$$SRT = \alpha SRTT + (1 - \alpha)R$$

dove α è un fattore di perequazione che determina quanto velocemente i vecchi valori vengono dimenticati; generalmente a corrisponde a 7/8. Questo tipo di formula è una **EWMA** (*exponentially weighted moving average*, media mobile a peso esponenziale) o filtro passa basso e serve a togliere il rumore dalle campionature.

7.4.9 WebRTC

WebRTC, acronimo di Web Real-Time Communication, è una tecnologia open-source supportata da Google, Mozilla e Opera, che permette la comunicazione in tempo reale tra browser web e applicazioni mobili. Consente agli sviluppatori di integrare facilmente comunicazioni audio, video e dati direttamente nelle loro applicazioni web senza richiedere plugin o software di terze parti.

WebRTC utilizza TCP come protocollo di trasporto per la comunicazione dati. TCP garantisce affidabilità nella trasmissione dei dati, garantendo che tutti i pacchetti inviati vengano ricevuti correttamente e nell'ordine corretto. Questa affidabilità è essenziale per applicazioni in tempo reale come videoconferenze, dove anche una breve interruzione potrebbe compromettere l'esperienza dell'utente.

Pregi:

- *Bassa Latenza*: WebRTC è importante per la sua velocità di trasmissione, offre uno dei metodi più veloci di trasporto video attraverso internet.
- *Architettura P2P*: Grazie alla sua architettura peer-to-peer (P2P), WebRTC riduce la latenza e migliora la qualità della comunicazione, consentendo una connessione diretta tra i dispositivi senza passare attraverso un server centrale.
- *Compatibilità con i Browser*: WebRTC è supportato da una vasta gamma di browser web moderni, inclusi Google Chrome, Mozilla Firefox, e Opera, garantendo una maggiore interoperabilità e accessibilità per gli utenti.

Difetti:

- *Complessità della Configurazione*: Anche se WebRTC semplifica l'implementazione della comunicazione in tempo reale, la configurazione e l'ottimizzazione delle connessioni P2P possono risultare complesse, specialmente per gli sviluppatori meno esperti.
- *Limitazioni di Sicurezza*: Anche se WebRTC incorpora protocolli di sicurezza come DTLS e SRTP, alcune vulnerabilità e minacce alla sicurezza possono ancora essere presenti, specialmente nelle implementazioni personalizzate o non correttamente configurate.
- *Requisiti di Rete*: WebRTC richiede una connessione Internet stabile e ad alta velocità per garantire una comunicazione fluida e senza interruzioni. Le prestazioni possono essere compromesse in presenza di connessioni lente o instabili.

7.5 LAB - Socket

7.6 LAB - Implementazione Sistemi Client-Server

Capitolo 8

Livello Applicazione

Terminate le necessarie premesse si può passare al livello in cui si trovano tutte le applicazioni. I livelli al di sotto del livello applicazione forniscono il trasporto affidabile, ma non svolgono alcun lavoro per gli utenti. In questo capitolo studieremo alcune applicazioni reali delle reti. Tuttavia anche nel livello applicazione sussiste l'esigenza dei protocolli di supporto che permettano alle applicazioni di funzionare. Per questo motivo studieremo uno di questi protocolli prima di iniziare l'esame delle applicazioni vere e proprie: il DNS, che gestisce i nomi all'interno di Internet. Successivamente esamineremo tre applicazioni reali: posta elettronica, World Wide Web e contenuti multimediali.

8.1 Posta Elettronica: MIME, SMTP, IMAP, POP3

In questo paragrafo forniremo una panoramica di ciò che possono fare i sistemi di posta elettronica e del modo in cui sono organizzati. L'architettura del sistema è mostrata nella Figura 7.7. Normalmente è composta da due sottosistemi: gli **user agent** (*agenti utente*), che consentono alle persone di leggere e inviare la posta elettronica, e i **message transfer agent** (*agenti di trasferimento dei messaggi*), che spostano i messaggi dalla sorgente alla destinazione. Questi ultimi sono informalmente chiamati **mail server**.

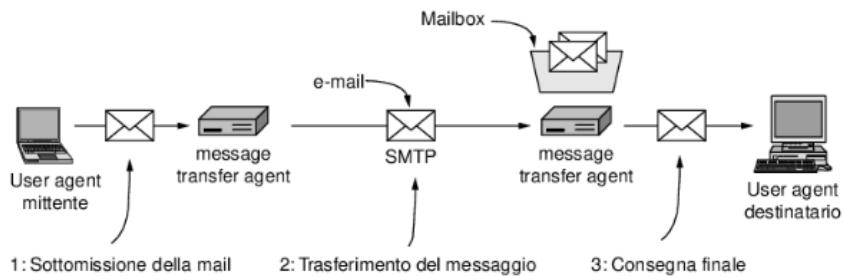


Figura 8.1: Architettura di un sistema di posta elettronica.

Uno user agent è un programma che fornisce un'interfaccia grafica o, talvolta, un'interfaccia testuale o basata su comandi, per interagire con il sistema di posta elettronica. Include gli strumenti per comporre, rispondere a, visualizzare e organizzare i messaggi riempiendoli, cercandoli e scartandoli.

Alcune elaborazioni dello user agent possono essere fatte automaticamente, anticipando i desideri degli utenti. Per esempio, le e-mail in ingresso possono essere filtrate per estrarre o attribuire una minore priorità ai messaggi che potrebbero essere spam. Alcuni user agent includono caratteristiche avanzate, come la compilazione di una e-mail automatica di risposta. Lo user agent è semplicemente un altro programma che può essere sporadicamente seguito.

I message transfer agent sono tipicamente processi di sistema. Vengono eseguiti come servizi sui mail server e sono pensati per essere sempre disponibili. Il loro compito è di spostare automaticamente le e-mail attraverso il sistema dalla sorgente al destinatario facendo uso di **SMTP** (*simple mail transfer protocol*).

SMTP venne originariamente specificato nell'RFC 821 e quindi rivisto per diventare quello attuale nell'RFC 5321. Invia le e-mail con un approccio orientato alla connessione e noti ca lo stato della consegna e gli errori. Esistono numerose applicazioni in cui la conferma della consegna è importante e può anche avere signi cato legale (“Vostro onore, il mio sistema di e-mail non è molto a dabile, per cui immagino che la citazione elettronica si sia persa da qualche parte.”).

I message transfer agent implementano anche le **mailing list**, in cui copie identiche di un messaggio sono consegnate a ogni iscritto alla lista. Altre funzionalità avanzate sono le copie per conoscenza (*carbon copy, cc*), le copie per conoscenza nascosta (*blind carbon copy, bcc*), i messaggi di posta elettronica ad alta priorità, la posta segreta cifrata, destinatari alternativi se il primo non è disponibile, la capacità da parte delle segretarie di leggere e rispondere alla posta del capo.

I concetti di mailbox e di formato standard dei messaggi e-mail sono il collegamento tra user agent e message transfer agent. Le **mailbox** (*caselle di posta*) memorizzano le e-mail ricevute dall'utente e sono mantenute dai mail server, mentre gli user agent semplicemente presentano agli utenti una visione dei contenuti delle loro mailbox. Per fare ciò, gli user agent inviano ai mail server i comandi per manipolare le mailbox, ispezionando i loro contenuti, cancellando messaggi e così via. Chi ritira la mail è il destinatario finale (passo 3) nella Figura 8.1. Con questa architettura per accedere a una mailbox un utente può usare differenti user agent su diversi computer.

Un'idea basilare nei sistemi di posta elettronica è la distinzione tra l'*involturlo* (**envelope**) e il suo contenuto. L'involturlo incapsula il messaggio e contiene tutte le informazioni necessarie per il suo trasporto, come l'indirizzo di destinazione, la priorità e il livello di protezione, che sono distinte dal messaggio stesso. I message transfer agent utilizzano l'involturlo per il routing, come gli u ci postali tradizionali utilizzano le buste.

Il messaggio all'interno dell'involturlo consiste di due parti: l'**intestazione** e il **corpo**. L'intestazione contiene le informazioni di controllo per gli user agent, mentre il corpo è dedicato interamente al destinatario umano.

8.1.1 Formato dei Messaggi

Passiamo ora dall'interfaccia utente al formato dei messaggi di posta elettronica. I messaggi inviati dallo user agent devono essere in un formato standard gestibile dal message transfer agent. Per prima cosa osserveremo i messaggi di posta ASCII che utilizzano RFC 5322, l'ultima revisione del formato originale dei messaggi Internet descritto in RFC 822. In ne studieremo le estensioni multimediali.

8.1.1.1 RFC 5322

L'RFC 5321 descrive la struttura dei messaggi di posta elettronica, delineando tre componenti principali: l'involturlo, i campi di intestazione e il corpo del messaggio.

- **Involturlo:** È la parte iniziale del messaggio, contenente informazioni fondamentali per la consegna, come il mittente e i destinatari. Questo segmento viene costruito dal message transfer agent (MTA) basandosi sui campi di intestazione.
- **Campi di intestazione:** Sono informazioni aggiuntive che forniscono dettagli sul messaggio. I campi di intestazione includono:
 - *To*: Indica gli indirizzi DNS dei destinatari principali del messaggio.
 - *Cc*: Elenca gli indirizzi dei destinatari secondari.
 - *Bcc*: Simile a Cc, ma nascosto ai destinatari principali e secondari.
 - *From*: Identifica chi ha scritto il messaggio.
 - *Sender*: Indica chi ha effettivamente inviato il messaggio, che può essere diverso dal mittente.
 - *Received*: Aggiunto da ogni MTA lungo il percorso, contiene informazioni sulla ricezione del messaggio.
 - *Return-Path*: Fornisce un modo per ricontrattare il mittente.
 - *Message-Id*: Un numero univoco per identificare il messaggio e prevenire consegne duplicate.

- **Reply-To:** Specifica a chi rispondere, utile quando il mittente e l'inviatore sono diversi.
 - **Corpo del messaggio:** Contiene il contenuto effettivo del messaggio, che può essere testo, allegati, immagini, ecc.
- Inoltre, l'RFC 5321 riconosce la possibilità di creare intestazioni personalizzate che iniziano con "X-", consentendo agli utenti di aggiungere informazioni specifiche o personalizzate.

8.1.1.2 MIME

MIME (*Multipurpose Internet Mail Extension*) è stato sviluppato per superare le limitazioni dei messaggi di posta elettronica basati esclusivamente su testo ASCII, consentendo l'invio di contenuti più complessi, come lingue con accenti, alfabeti non latini, ideogrammi e file non testuali come immagini, audio e documenti binari.

MIME è descritto negli RFC 2045-2047, 4288, 4289 e 2049. Estende il formato RFC 822, aggiungendo una struttura al corpo del messaggio e definendo regole di codifica per gestire messaggi non ASCII.

I messaggi MIME includono cinque nuove intestazioni:

- **MIME-Version:** Indica la versione MIME utilizzata nel messaggio.
- **Content-Description:** Comunica il contenuto del messaggio per aiutare il destinatario a decidere se leggerlo.
- **Content-Id:** Identifica il contenuto, simile a Message-Id.
- **Content-Transfer-Encoding:** Specifica la codifica utilizzata per trasmettere il contenuto tramite SMTP.
- **Content-Type:** Indica la natura del corpo del messaggio e il tipo di dati, come testo, immagini, audio, ecc.

La codifica del contenuto è fondamentale, considerando che SMTP era originariamente progettato per messaggi ASCII con linee di lunghezza limitata. Le estensioni MIME consentono la trasmissione di dati binari, con cinque schemi di codifica comuni: ASCII, 8 bit, binario, base64 e quoted-printable.

I tipi MIME includono testo, immagine, audio, video, modello, applicazione e messaggio, con numerosi sottotipi per gestire una vasta gamma di contenuti. Ad esempio, text/plain per testo semplice, text/html per pagine web, image/jpeg per immagini JPEG, audio/mpeg per file audio MP3, application/pdf per documenti PDF, e così via.

MIME consente anche la composizione di messaggi multipli, inclusi allegati e varianti in diverse lingue o formati, attraverso i tipi multipart, come multipart/mixed, multipart/alternative e multipart/related.

L'uso di MIME permette agli user agent di gestire una vasta gamma di contenuti applicativi, migliorando la flessibilità e l'efficacia delle comunicazioni via email, anche se presenta alcune sfide legate alla sicurezza quando si tratta di eseguire o interpretare contenuti ricevuti.

8.1.2 Consegnna Finale

Oggi lo user agent su PC, computer portatili o dispositivi mobili si trova verosimilmente su una macchina diversa dal server di posta elettronica dell'ISP o dell'azienda e gli utenti vogliono poter accedere alle proprie e-mail da remoto, ovunque siano: dal lavoro, da casa, mentre viaggiano ed agli Internet ca è quando trascorrono le cosiddette vacanze. Vogliono anche lavorare senza essere connessi e quindi connettersi per ricevere i messaggi in arrivo e inviare a loro volta messaggi. Inoltre ogni utente può usare diversi user agent a seconda di quale computer è conveniente usare al momento e più user agent potrebbero essere attivi contemporaneamente. In questo contesto il compito dello user agent è di visualizzare i contenuti della casella di posta e permettere che questa venga gestita da remoto. A questo scopo possono essere usati diversi protocolli differenti, ma non SMTP, che è un protocollo push:6 prende un messaggio e si connette al server remoto per trasferirlo. La consegna finale non può essere eseguita in questo modo, sia perché la casella di posta deve continuare a essere memorizzata sul mail server sia perché lo user agent potrebbe non essere connesso a Internet nel momento in cui SMTP tenta di inoltrare i messaggi.

8.1.2.1 IMAP

IMAP (*Internet Message Access Protocol*) è un protocollo utilizzato per accedere e gestire le email sul server di posta elettronica. La sua versione più recente, definita nell'RFC 3501, prevede che il server IMAP ascolti sulla porta 143. Un client IMAP si connette al server e utilizza una serie di comandi per interagire con le email.

Prima di iniziare a utilizzare IMAP, il client può stabilire una sessione di trasporto sicura e quindi autenticarsi sul server. Una volta autenticato, il client può eseguire una serie di operazioni, tra cui elencare le cartelle (o "mailbox"), visualizzare i messaggi, etichettarli, spostarli tra le cartelle e altro ancora.

IMAP è un miglioramento rispetto al protocollo POP3 (Post Office Protocol, versione 3), definito nell'RFC 1939. Mentre POP3 è più semplice e scarica di solito i messaggi sul client, rendendo difficile la gestione delle email su più dispositivi e potenzialmente esponendo i messaggi a rischi di perdita in caso di guasto del client, IMAP consente di mantenere i messaggi sul server e sincronizzarli tra diversi dispositivi. Tuttavia, POP3 è ancora utilizzato, così come alcuni protocolli proprietari, come quelli utilizzati da Microsoft Exchange.

8.1.2.2 Webmail

Il Webmail rappresenta un'alternativa sempre più diffusa ai protocolli tradizionali come IMAP e SMTP per gestire la posta elettronica, consentendo agli utenti di accedere ai propri messaggi tramite un'interfaccia web. I principali provider di Webmail includono Google Gmail, Microsoft Hotmail e Yahoo! Mail. In questo modello, il software per la gestione della posta (lo user agent) è fornito come servizio attraverso il Web.

Nell'architettura del Webmail, il provider gestisce un server di posta elettronica che accetta i messaggi degli utenti tramite SMTP sulla porta 25. Tuttavia, lo user agent è costituito da un'interfaccia utente web, accessibile tramite un browser. Gli utenti possono quindi accedere alle proprie email e inviarne di nuove utilizzando qualsiasi browser e dispositivo connesso a Internet.

Quando un utente accede alla pagina Web del servizio di posta elettronica, inserisce le proprie credenziali di accesso (nome utente e password), che vengono inviate al server per la convalida. Se l'autenticazione ha successo, il server recupera la casella di posta dell'utente e genera una pagina web che mostra il contenuto della casella di posta. Questa pagina viene quindi inviata al browser dell'utente per la visualizzazione.

Gli utenti possono interagire con la pagina web per leggere o eliminare i messaggi, utilizzando spesso elementi interattivi che rispondono ai click del mouse. Molte pagine web includono anche programmi JavaScript per fornire un'interfaccia più dinamica e interattiva, consentendo agli utenti di gestire la posta elettronica in modo più efficiente.

Per inviare un nuovo messaggio, l'utente compila un modulo web e invia i dati al server utilizzando il normale protocollo HTTP. Il server Web si occupa quindi di inviare il messaggio attraverso il sistema di consegna tradizionale, utilizzando i protocolli standard del Web per garantire la sicurezza della comunicazione.

8.2 Protocolli di Trasporto File

FTP (File Transfer Protocol) e TFTP (Trivial File Transfer Protocol) sono entrambi protocolli utilizzati per il trasferimento di file, ma differiscono significativamente nelle loro funzionalità e nel modo in cui operano.

8.2.1 FTP

Il Protocollo di Trasferimento File (FTP) è stato progettato per consentire agli utenti di trasferire file in modo efficiente e affidabile da un host all'altro attraverso Internet. Questo protocollo offre la possibilità di replicare dati, consentendo un backup su larga scala.

Questo protocollo offre due tipi di accesso: *autenticato* e *anonimo*. L'accesso autenticato richiede una coppia di account/password per l'autenticazione dell'utente, mentre quello anonimo di solito è senza restrizioni.

Nel funzionamento dell'FTP, il client FTP stabilisce una connessione TCP con il server FTP remoto. Successivamente, nome utente e password vengono inviati come comandi FTP, permettendo quindi il trasferimento di file da e verso il server.

Una caratteristica distintiva dell'FTP rispetto all'HTTP è che è un protocollo *out-of-band*, il che significa che utilizza due connessioni TCP parallele: una per i dati (non persistente) e una per il controllo (persistente).

Quanto alla modalità di trasferimento, esistono due modalità principali: la *modalità attiva* e la *modalità passiva*. Nella modalità attiva, la connessione di controllo è avviata dal client mentre la connessione dati è avviata dal server. Nella modalità passiva, entrambe le connessioni sono avviate dal client.

8.2.2 TFTP

Il protocollo TFTP (Trivial File Transfer Protocol) è stato progettato per scopi specifici in cui è necessario trasferire file in modo rapido e semplice senza richiedere tutte le funzionalità avanzate offerte da FTP (File Transfer Protocol).

Ecco alcune caratteristiche principali di TFTP:

- **Utilizzo e design:** TFTP viene utilizzato per trasferire file tra processi su una rete. È stato progettato per avere un overhead minimo e non include funzionalità di sicurezza come l'autenticazione degli utenti.
- **Protocollo di trasporto:** Sebbene sia progettato per l'uso con UDP (User Datagram Protocol), potrebbe potenzialmente essere utilizzato con altri protocolli di trasporto.
- **Facilità di implementazione:** TFTP è progettato per essere facile da implementare, il che lo rende adatto anche per dispositivi con risorse limitate o firmware incorporato.
- **Piccole dimensioni:** Il protocollo TFTP è di dimensioni ridotte, il che significa che può essere facilmente incluso nei firmware dei dispositivi.
- **Messaggi del protocollo:** TFTP si basa su cinque tipi di messaggi:
 - *Richiesta di lettura*: inviata dal client per richiedere il trasferimento di un file dal server.
 - *Richiesta di scrittura*: inviata dal client per inviare un file al server.
 - *Dati*: inviati dal server al client e contengono blocchi di dati del file.
 - *Acknowledgment (ACK)*: inviato dal client al server per confermare il ricevimento dei dati.
 - *Errore*: inviato dal server al client in caso di errore durante il trasferimento.

Modalità attiva e passiva: TFTP supporta sia la modalità attiva che quella passiva per stabilire le connessioni. Nella modalità attiva, la connessione di controllo è inizializzata dal client, mentre nella modalità passiva, entrambe le connessioni (controllo e dati) sono inizializzate dal client.

8.3 DNS - Domain Name System

Nel principio, l'Internet era piccolo. Meno di 100 host popolavano la rete. Tutti conoscevano tutti, in un ambiente centralizzato dove il file host era distribuito a tutti. Tuttavia, questo modello non poteva sostenersi con l'espansione della rete.

Poi arrivò il **Domain Name System (DNS)**, un sistema per introdurre nomi in formato ASCII per identificare gli host e separare così i nomi dagli indirizzi IP. Il DNS è un sistema di denominazione gerarchico, come definito nel RFC1035, basato su un database distribuito per la sua implementazione.

Per associare un indirizzo IP a un nome, l'applicazione richiama una procedura di libreria (**resolver**), come ad esempio `gethostbyname()`. Con l'indirizzo IP, l'applicazione può stabilire una connessione TCP o inviare pacchetti UDP.

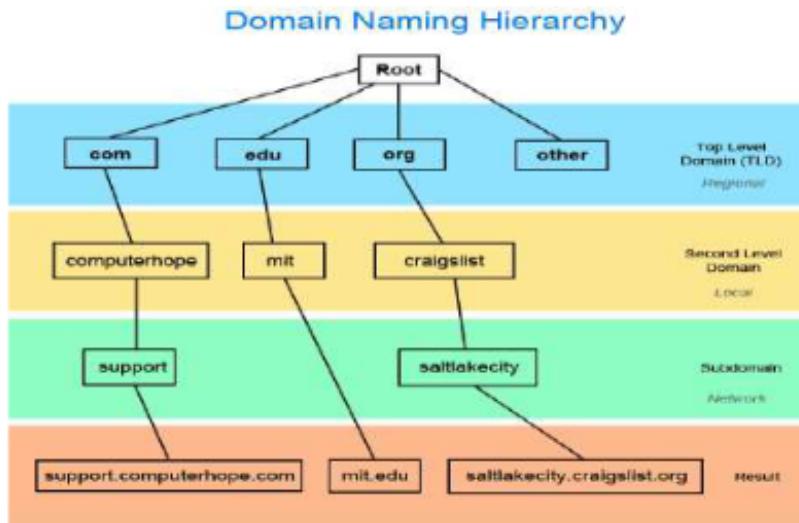


Figura 8.2: DNS Name Space

Lo spazio dei nomi DNS è strutturato gerarchicamente, simile a un indirizzo postale, con una directory radice rappresentata da un punto (.), seguita da due gerarchie di dominio di primo livello: organizzativa e geografica.

Quando un host invia una richiesta DNS, questa viene inoltrata al server DNS locale, che agisce come server proxy. Ogni dominio ha il controllo sull'allocazione dei sottodomini.

Il processo di risoluzione dei nomi avviene attraverso i seguenti passaggi:

1. Un server DNS è configurato con una cache iniziale (chiamata hints) contenente gli indirizzi conosciuti dei server di nome radice. Questo file di hint viene aggiornato periodicamente in modo affidabile e autorevole.
2. Quando un client effettua una richiesta (ricorsiva) a quel server, questo elabora la richiesta utilizzando la cache (se già possiede la risposta da una ricerca precedente) o eseguendo i seguenti passaggi per conto del client.
3. Viene effettuata una query a uno dei server radice per trovare il server autoritativo per il dominio di primo livello richiesto.
4. Viene ricevuta una risposta che indica il nameserver per quella risorsa.
5. Il server "attraversa l'albero" da destra a sinistra, passando da nameserver a nameserver, fino al passo finale che restituisce l'indirizzo IP dell'host in questione.
6. L'indirizzo IP della risorsa viene quindi restituito al client.

Il protocollo DNS è relativamente leggero e utilizza UDP per le query in modo che possano avvenire rapidamente e senza eccessivo overhead. Le query superiori a 512 byte e alcune operazioni più pesanti, come i trasferimenti di zona, passano a utilizzare TCP per evitare problemi di consegna.

8.3.1 Campi del Protocollo

I campi del protocollo DNS sono fondamentali per il funzionamento del sistema, poiché specificano vari dettagli su una richiesta o una risposta. Ecco una panoramica di alcuni dei principali campi:

- **Identifier:** un campo ID a 16 bit che corrisponde alle richieste e alle risposte.

- **Query/Response Flag:** un campo a 4 bit che designa se il pacchetto è una richiesta o una risposta.
- **Opcode:** Specifica il tipo di messaggio trasportato. Le opzioni includono: 0 per una query standard, 1 per una query inversa (obsoleta), 2 per lo stato del server, 3 è riservato e non utilizzato, 4 è un messaggio di notifica e 5 è un aggiornamento (utilizzato per Dynamic DNS).
- **AA:** Questo è un campo a 1 bit che indica una risposta autorevole. Il bit è impostato su 1 se è autorevole, il che significa che il server che ha fornito la risposta è autorevole per il dominio in questione. Se è impostato su 0, è una risposta non autorevole.
- **TC:** Un campo a 1 bit per la troncatura, sì o no. Indica di solito che è stata inviata tramite UDP ma era più lunga di 512 byte.
- **RD:** Un campo a 1 bit chiamato "Ricorsione Desiderata", il che significa che il client sta chiedendo al server di attraversare l'albero per conto del client e di restituire semplicemente la risposta anziché indicare dove cercare.
- **RA:** Un campo a 1 bit chiamato "Ricorsione Disponibile", in cui un server DNS indica a un client se supporta o meno la ricorsione.

Inoltre, ci sono altri campi come **Z** (tre bit riservati impostati sempre a zero), **RCode** (un campo a 4 bit impostato su zero nelle query con le seguenti opzioni: 0 è nessun errore, 1 è errore di formato, 2 è errore del server, 3 è errore di nome, 4 non è implementato, 5 è rifiutato, 6 il nome esiste ma non dovrebbe, 7 esiste un record di risorsa che non dovrebbe, un record di risorsa che dovrebbe esistere non lo fa, 9 la risposta non è autorevole, 10 il nome nella risposta non è all'interno della zona specificata), **QDCount** (quante domande nella sezione delle domande), **ANCount** (quante risposte nella sezione delle risposte), **NSCount** (quanti record di risorse nella sezione di autorità) e **ARCount** (quanti record di risorse nella sezione aggiuntiva).

Una query DNS consiste in diversi elementi:

- **qname:** un nome di dominio (ad esempio www.ripe.net).
- **qtype:** il tipo di query, che può essere A, AAAA, MX, CNAME, PTR, SRV, TXT, NS.
- **qclass:** la classe della query, generalmente IN (Internet), mentre CH e HS sono poco utilizzati oggi.
- **Flags:** includono QR (Response Code), RD (Recursive Desired), EDNS Opt, DO (DNSSEC OK), AD (Authenticated Data), ecc.

I record delle risorse sono associati a ogni dominio e possono includere varie informazioni:

- **Name:** il nome di dominio a cui il record appartiene.
- **TTL:** il tempo di vita del record, che indica quanto tempo il record può essere memorizzato nella cache prima di essere considerato obsoleto.
- **Class:** generalmente IN per le informazioni su Internet.
- **Type:** specifica il tipo di record, come A (indirizzo IP), MX (record di posta), NS (server di nomi), CNAME (alias), PTR (lookups inversi), SRV (servizio), HINFO (informazioni su hardware), TXT (testo libero).
- **Value:** il valore effettivo del record, che può essere un numero, un nome o una stringa ASCII, a seconda del tipo di record.

Ad esempio:

- **SOA:** viene utilizzato per fornire le informazioni principali sul server di nomi della zona.
- **A:** contiene l'indirizzo IP di un host.
- **MX:** specifica il nome dell'host configurato per accettare la posta per quel dominio specifico.
- **NS:** specifica il server di nomi.

- **CNAME**: permette di creare alias.
- **PTR**: utilizzato per le ricerche inverse.
- **HINFO**: utilizzato per conoscere il tipo di macchina e il sistema operativo.
- **TXT**: utilizzato facoltativamente per consentire l'auto-identificazione del dominio.

8.3.2 Server

I server dei nomi DNS gestiscono lo spazio dei nomi DNS, diviso in zone non sovrapposte. Quando un resolver ha una query per un nome di dominio, viene interrogato il server dei nomi locale.

Se il dominio appartiene alla zona del server dei nomi (ad esempio, ai.cs.yale.edu è incluso in cs.yale.edu), viene fornito un record di risorsa autoritativo. Altrimenti, se il dominio è remoto e non ci sono informazioni locali disponibili, il server dei nomi si rivolge al server di dominio di primo livello (TLD) per il dominio richiesto.

Ad esempio, se un resolver su flits.cs.vu.nl desidera l'indirizzo IP di linda.cs.yale.edu, il server dei nomi locale (cs.vu.nl) viene interrogato. Se il server locale non conosce nulla su questo dominio, invia un pacchetto UDP al TLD per edu (edu-server.net) memorizzato nel suo database. Edu-server.net deve conoscere i suoi figli, quindi inoltra la richiesta al server dei nomi per yale.edu. Yale.edu invia quindi la richiesta a cs.yale.edu, che deve avere il record di risorsa autoritativo. Il record di risorsa viene quindi inviato a ciascun cliente dal suo server. I record consegnati a cs.vu.nl vengono memorizzati nella cache per utilizzi futuri (il TTL specifica per quanto tempo devono essere conservati).

I server dei nomi DNS possono essere server veri e propri, contenenti le risposte autoritative per determinate risorse, oppure possono essere server cache che eseguono query ricorsive (e caching). I resolver DNS sono semplicemente client DNS che possono effettuare due tipi principali di query: **iterative** e **ricorsive**.

Le query ricorsive sono quelle in cui il client chiede al server di svolgere tutto il lavoro per lui. Invia nella sua query il flag **RECURSION DESIRED**, e il server DNS lo rispetterà o meno. Le query iterative sono l'opposto delle query ricorsive. Quando vengono utilizzate, il server non cerca la risposta per il client (a meno che non sia la prima domanda e risposta), ma piuttosto indica al client dove guardare successivamente. Quindi, se il client chiede per chat.google.com, il server gli indica di controllare con i server .com e considera il suo lavoro completato.

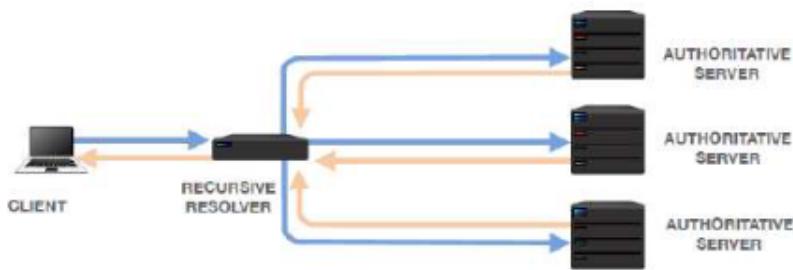


Figura 8.3: Server Autoritativo

Le risposte e le query possono essere **autoritative** o **non autoritative**. Le risposte autoritative provengono direttamente da un server dei nomi che ha autorità sul record in questione. Le risposte non autoritative provengono da un'altra fonte, ad esempio da un altro server o attraverso una cache.

Le query inverse semplicemente invertono il senso delle ricerche DNS, ovvero vanno dall'indirizzo IP al nome invece che dal nome all'indirizzo IP. Le query in avanti sono un altro nome per le normali query di nome-a-IP.

Un server autoritativo contiene i record nel suo file di zona e risponde solo alle query per i dati sotto la sua autorità. Se non può rispondere, indica l'autorità ma non effettua query in modo ricorsivo.

Le cache DNS possono avere diversi significati, il che può essere confusionario: La lista dei nomi e degli indirizzi IP risolti di recente, che vengono "cachati" in modo che se si pone nuovamente la stessa domanda si otterrà la stessa risposta senza generare traffico di rete. Un server DNS che non ha nomi autoritativi propri, ma esegue solo query ricorsive e caching (salvando quelle risposte per future richieste entro un certo periodo di tempo).

Quindi, quando qualcuno dice di dover cancellare la propria cache DNS, probabilmente si riferisce alla cache locale. Se parlano invece di configurare una cache DNS, probabilmente si riferiscono a un server DNS che rende più veloci le query DNS per la rete.

I **trasferimenti di zona** sono il mezzo attraverso il quale i server slave recuperano i record dai server master per scopi di backup e ridondanza. Avvengono tramite TCP perché i dati trasferiti sono solitamente consistenti (e molto probabilmente superiori a 512 byte). Durante l'operazione, il client invia un tipo di query **IXFR** (*Incremental Zone Transfer*) anziché **AXFR** (*Normal Zone Transfer*).

I trasferimenti di zona sono sensibili dal punto di vista della sicurezza perché quando qualcuno sa cosa e dove si trovano le tue risorse, può pianificare un attacco contro di te. I trasferimenti di zona dovrebbero essere consentiti solo da sistemi approvati.

8.3.3 Problemi

Il DNS influenza su diverse aree:

- **Consegna della posta:** Il DNS aiuta gli agenti di consegna della posta a inviare la posta su Internet. Per consegnare la posta su Internet, il DNS utilizza i record di scambio di posta (MX records).
- **Risparmio della latenza:** In alcuni casi specifici, cambiare i server DNS può indirettamente aiutare a ridurre la latenza. Cambiare i server DNS potrebbe migliorare le velocità di upload a tal punto da eliminare la latenza (nel gaming, nello streaming, ecc.).

Problemi di base del DNS includono:

- Il DNS è testo non cifrato.
- Utilizza UDP semplice, senza sessioni.
- Ha una struttura ad albero con deleghe.
- Ogni entità è responsabile di una parte limitata di esso.
- I resolver sono vittime di attacchi, dirottamenti e errori.
- È necessaria fiducia.

Per quanto riguarda la sicurezza del DNS, esiste **DNSSEC** (*DNS Security Extensions*), definito nel RFC4033, che aggiunge strati al DNS per renderlo verificabile. Questo include l'aggiunta di nuovi tipi di record e la creazione di un'infrastruttura a chiave pubblica (PKI) con una catena di fiducia per convalidare i dati.

DNSSEC, in breve, garantisce l'autenticità e l'integrità dei dati firmando i **Resource Record Sets** con la chiave privata **DNSKEY**. Per verificare le firme RRSIG, è necessario disporre delle chiavi pubbliche DNSKEY. I figli firmano le loro zone con la loro chiave privata, mentre il genitore garantisce l'autenticità della chiave del figlio firmando l'hash di essa (DS). Questo processo viene ripetuto per il genitore e il nonno. Nell'ideale, viene distribuita una sola chiave pubblica DNSKEY.

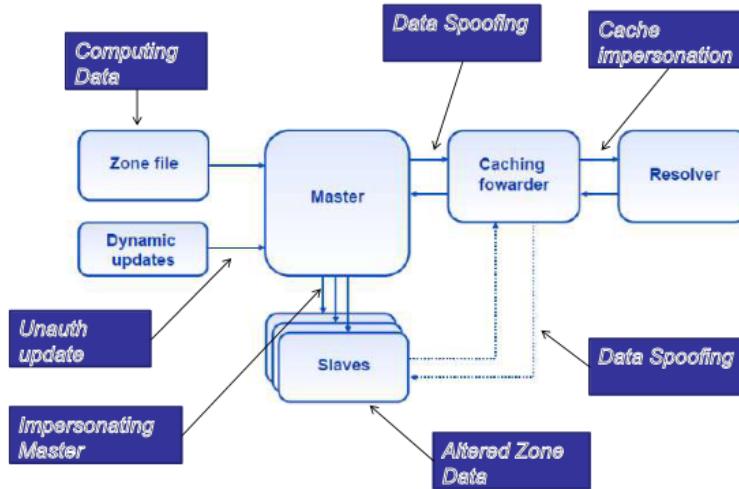


Figura 8.4: Data Flow e Problemi

Ecco un riassunto dei principali software DNS:

- **DNS BIND:** un server DNS autorevole e ricorsivo.
- **Unbound:** un resolver DNS di caching.
- **NSD:** un nameserver solo autorevole.
- **Microsoft DNS:** fornito con Windows Server.
- **Knot DNS:** un nameserver solo autorevole.
- **PowerDNS:** supporta diversi backend di archiviazione dei dati.

BIND (Berkeley Internet Name Domain) è il software DNS open source più ampiamente utilizzato su Internet. La versione attuale è BIND 9.11.4-P2 ed è mantenuta dall'Internet Systems Consortium (ISC).

8.4 World Wide Web

Dal punto di vista degli utenti il Web è una vasta raccolta mondiale di documenti o pagine Web, spesso definite per brevità solamente pagine. Ogni pagina può contenere collegamenti (*link*) ad altre pagine situate ovunque nel mondo. Gli utenti possono seguire un collegamento facendo click su di esso e raggiungendo così la pagina indicata; il processo può essere ripetuto indefinitamente. L'idea di fare in modo che ogni pagina puntasse a un'altra (un ipertesto, *hypertext*).

Le pagine sono visualizzate con un programma chiamato browser, di cui Firefox, Internet Explorer e Chrome sono le forme più popolari. Il browser preleva la pagina richiesta, interpreta il testo e i comandi di formattazione, quindi visualizza la pagina correttamente formattata sullo schermo. Il contenuto può essere un mix di testo, immagini e comandi di formattazione, come in un documento tradizionale oppure altre forme di contenuto come video o programmi che producono un'interfaccia grafica con cui l'utente può interagire.

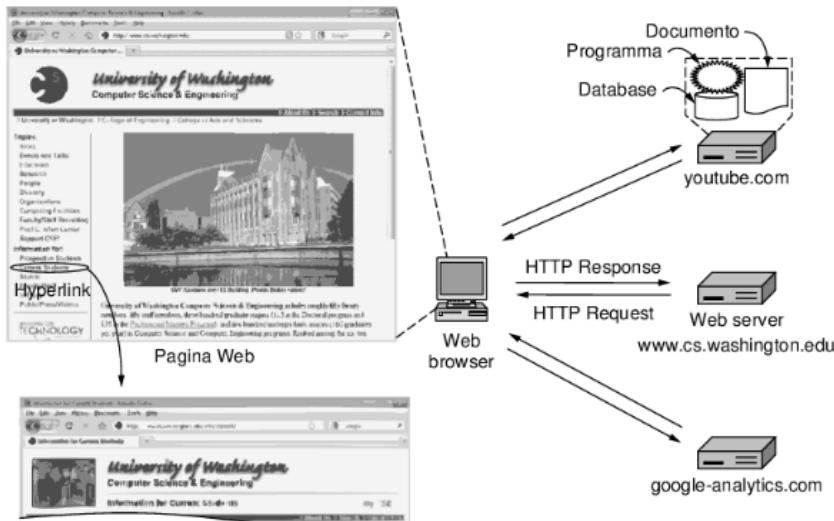


Figura 8.5: Architettura del Web

Il modello di base del funzionamento del Web è mostrato nella Figura 8.5. Il browser visualizza una pagina Web sulla macchina client. Ogni pagina viene presa, inviando una richiesta a uno o più server che rispondono inviando i contenuti della pagina. Il protocollo di domanda e risposta per catturare le pagine è un semplice protocollo basato sul testo che funziona su TCP, esattamente come nel caso di SMTP. È chiamato **HTTP** (*hypertext transfer protocol*). Il contenuto può essere semplicemente un documento che viene letto da un disco, il risultato di una interrogazione a un database o l'esecuzione di un programma. Se il documento è sempre uguale siamo in presenza di una **pagina statica**, mentre se è generato su richiesta da un programma o contiene del codice eseguibile è chiamato **pagina dinamica**.

Una pagina dinamica si presenta in modo diverso ogni volta che viene visualizzata. Per esempio, la prima pagina di un negozio di elettronica può essere differente per ogni visitatore. Se il cliente di una libreria ha comprato nel passato romanzi polizieschi, quando visiterà la pagina principale del negozio probabilmente vedrà visualizzati prevalentemente romanzi polizieschi, mentre un cliente più appassionato di gastronomia vedrà nuovi libri di cucina. Come un sito Web tiene traccia delle nostre passioni è una storia di cui parleremo fra poco. Per il momento diciamo che la risposta coinvolge degli oggetti chiamati *cookie*.

8.4.1 Il Lato Client

Un browser è un programma che può visualizzare una pagina Web e rilevare i click del mouse sugli elementi in essa visualizzati. Quando viene selezionato un elemento, il browser segue il collegamento ipertestuale e preleva la pagina selezionata.

Quando fu creato il Web, divenne immediatamente chiaro che la disponibilità di pagine che facesse riferimento l'una all'altra richiedeva un meccanismo per denominare e individuare le pagine. In particolare occorreva rispondere a tre domande prima di poter visualizzare una pagina selezionata:

1. come si chiama la pagina
2. dov'è situata la pagina
3. com'è possibile accedere alla pagina.

Se a ogni pagina fosse assegnato un nome univoco non vi sarebbero ambiguità nell'identificazione; tuttavia il problema non sarebbe risolto. Si consideri un'analogia tra persone e pagine. In Italia tutti possiedono un codice scale, che è un identificatore univoco (due persone non dovrebbero avere mai lo stesso). Ciononostante, se si dispone solo del codice scale non c'è modo di individuare l'indirizzo del proprietario e ovviamente non c'è modo di sapere se a questa persona sa scrivere in italiano, spagnolo o cinese. Il Web presenta fondamentalmente gli stessi problemi.

La soluzione scelta identifica le pagine in un modo che risolve tutti e tre i problemi contemporaneamente. A ogni pagina è assegnato un **URL** (*uniform resource locator*) che serve effettivamente come nome mondiale per la pagina. Gli URL sono composti di tre parti: il protocollo (noto anche come schema), il nome DNS della macchina su cui è situata la pagina e un percorso che indica in modo univoco la pagina specifica (un file da leggere o un programma da eseguire). Nel caso generale il percorso ha una struttura gerarchica che rispecchia il modello della struttura gerarchica di directory e file. Tuttavia, l'interpretazione del percorso è compito del server; può riflettere o meno la vera struttura delle directory. Ad esempio l'URL della pagina mostrata nella Figura 8.5 è:

`http://www.cs.washington.edu/index.html`

Questo URL è composto di tre parti: il protocollo (*http*), il nome DNS dell'host (*www.cs.washington.edu*) e il nome del percorso (*index.html*).

Quando un utente fa click su un collegamento ipertestuale il browser esegue una serie di passaggi per recuperare la pagina indicata. Vediamo i passaggi seguiti alla selezione di questo collegamento.

1. Il browser determina l'URL (osservando che cosa è stato selezionato).
2. Il browser richiede al DNS l'indirizzo IP del server *www.cs.washington.edu*
3. Il DNS risponde con 128.208.3.88.
4. Il browser esegue una connessione TCP alla porta 80 su 128.208.3.88.
5. Invia una richiesta HTTP per la pagina */index.html*.
6. Il server *www.cs.washington.edu* invia la pagina come risposta HTTP inviando, per esempio, il file */index.html*.
7. Se la pagina include URL necessari alla visualizzazione il browser prende gli altri URL usando lo stesso procedimento. In questo caso, l'URL include più immagini, anche queste prese da *www.cs.washington.edu*, un video integrato da youtube.com e uno script da *google-analytics.com*.
8. Il browser visualizza la pagina */index.html* come appare nella Figura 8.5.
9. La connessione TCP viene rilasciata se non vi sono altre richieste agli stessi server per un breve periodo.

Nome	Utilizzato per	Esempio
<code>http</code>	Ipertesto (HTML)	<code>http://www.ee.uwa.edu/~rob/</code>
<code>https</code>	Ipertesto con sicurezza	<code>https://www.bank.com/accounts/</code>
<code>ftp</code>	Trasferimento file con FTP	<code>ftp://ftp.cs.vu.nl/pub/minix/README</code>
<code>file</code>	File locale	<code>file:///usr/suzanne/prog.c</code>
<code>mailto</code>	Spedire e-mail	<code>mailto:JohnUser@acm.org</code>
<code>rtsp</code>	Streaming multimediale	<code>rtsp://youtube.com/montypython.mpg</code>
<code>sip</code>	Chiamate multimediali	<code>sip:eve@adversary.com</code>
<code>about</code>	Informazioni sul browser	<code>about:plugins</code>

Figura 8.6: Alcuni comuni schemi di URL.

8.4.1.1 Minacce alla Sicurezza

Le minacce alla sicurezza, sia lato server che lato client, sono una preoccupazione costante per chi sviluppa e gestisce pagine web. Queste minacce possono compromettere gravemente la riservatezza, l'integrità e la disponibilità dei dati, oltre a minare la fiducia degli utenti.

Sul lato server, una delle principali minacce è l'**SQL Injection**. Gli attaccanti possono sfruttare vulnerabilità nei moduli di input dell'utente per iniettare codice SQL malevolo nei comandi che il server esegue su un database. Questo tipo di attacco può portare a violazioni di dati, cancellazioni, o modifiche non autorizzate. Un'altra minaccia significativa è il **Cross-Site Scripting** (XSS), che si verifica quando un attaccante riesce a iniettare codice JavaScript dannoso in una pagina web visualizzata da altri utenti. Questo codice può rubare informazioni sensibili, come i cookie di sessione, o eseguire azioni dannose a nome degli utenti. Un'altra minaccia importante è il **Cross-Site Request Forgery** (CSRF), che consente agli attaccanti di indurre un utente autenticato a eseguire azioni non volute su un'applicazione web in cui è autenticato, sfruttando la fiducia del sito verso l'utente. Infine, c'è la minaccia della File Inclusion, dove gli attaccanti possono inserire file dannosi nel server, sfruttando vulnerabilità nei percorsi di inclusione dei file, portando all'esecuzione di codice arbitrario sul server.

Sul lato client, il **Cross-Site Scripting** (XSS) rimane una minaccia rilevante, poiché il codice malevolo viene eseguito nel contesto del browser dell'utente, potenzialmente rubando dati locali o alterando il comportamento della pagina. Un'altra minaccia è il **Clickjacking**, dove gli attaccanti sovrappongono una pagina web con contenuti invisibili o trasparenti per indurre gli utenti a cliccare su elementi diversi da quelli che intendono, come pulsanti nascosti che eseguono azioni non volute. Gli attacchi **Man-in-the-Middle** (MitM) rappresentano un'altra minaccia critica, durante la trasmissione dei dati tra il client e il server, un attaccante può intercettare e modificare la comunicazione, rubando informazioni sensibili o inserendo dati dannosi. Infine, il Phishing è una minaccia costante, dove gli utenti possono essere ingannati da pagine web fraudolente che imitano siti legittimi, inducendoli a fornire informazioni personali, come credenziali di accesso o dettagli di carte di credito.

Per difendersi da queste minacce, è essenziale adottare diverse tecnologie e metodi di difesa. La validazione e sanitizzazione degli input è fondamentale per prevenire l'iniezione di codice malevolo, assicurando che tutti gli input dell'utente siano rigorosamente validati e sanitizzati. L'implementazione di metodi robusti di autenticazione e autorizzazione garantisce che solo gli utenti autorizzati possano accedere a risorse sensibili. L'adozione del protocollo HTTPS è cruciale per criptare i dati in transito tra il client e il server, prevenendo gli attacchi MitM. Le **Content Security Policy** (CSP) aiutano a prevenire attacchi XSS specificando quali script possono essere eseguiti sul sito web, mentre i cookie con l'attributo SameSite limitano le richieste cross-site, riducendo il rischio di attacchi CSRF. Infine, l'uso di intestazioni HTTP come **X-Frame-Options** può prevenire il clickjacking, impedendo l'incorporazione delle pagine in iframe di terze parti.

8.4.2 Il Lato Server

Come affermato in precedenza, quando l'utente digita un URL o fa click su un collegamento ipertestuale, il browser analizza sintatticamente l'URL e interpreta la parte tra `http://` e lo slash successivo come il nome DNS da ricercare. Armato dell'indirizzo IP del server, il browser stabilisce una connessione TCP sulla relativa porta 80. Invia poi un comando contenente il resto dell'URL, vale a dire il nome di un file sul server, che quindi restituisce il file per la visualizzazione da parte del browser.

In prima approssimazione un server Web è simile al server della Figura 6.6. A tale server, come un vero server Web, viene fornito il nome di un file da cercare e restituire. In entrambi i casi, i passaggi eseguiti dal server nel suo ciclo principale sono i seguenti:

1. accettare una connessione TCP da un client (un browser)
2. ottenere il percorso della pagina che è il nome del file richiesto
3. ottenere il file (dal disco)
4. inviare il contenuto del file al client
5. rilasciare la connessione TCP.

Per ovviare al problema di servire una singola richiesta alla volta una strategia è implementare il server con un approccio multithread. In una possibile struttura il server consiste di un modulo di front end che accetta tutte le richieste in ingresso e k moduli di elaborazione come mostrato nella Figura 8.6. I $k + 1$ thread appartengono tutti allo stesso processo, in modo che tutti i moduli di elaborazione abbiano accesso alla cache all'interno dello spazio di indirizzamento del processo.

Oltre ad accettare nomi e a restituire i le i moderni server Web svolgono altre operazioni, per cui l'elaborazione effettiva di ogni richiesta può divenire complessa. Per questo motivo in molti server ogni modulo di elaborazione esegue una serie di passaggi. Il front end passa ogni richiesta in ingresso al primo modulo disponibile, che la soddisfa utilizzando un sottoinsieme dei seguenti passi (che dipende da quali sono necessari per quella particolare richiesta). Questi passi avvengono dopo che è stata stabilita la connessione TCP e i meccanismi di trasporto sicuri (come SSL/TLS):

1. risolvere il nome della pagina Web richiesta
2. eseguire il controllo di accesso sulla pagina Web
3. controllare la cache
4. prelevare la pagina richiesta dal disco o eseguire un programma che la costruisca
5. determinare il resto della risposta (per esempio, il tipo MIME da includere)
6. restituire la risposta al client
7. aggiungere un elemento al registro delle operazioni svolte dal server.

8.4.3 Cookie

Navigare per il Web come descritto finora comporta il recupero di una serie di pagine indipendenti. Non esiste il concetto di sessione: il browser invia una richiesta a un server e ottiene il le, dopo di che il server dimentica di aver mai visto quel particolare client. All'inizio, quando il Web era utilizzato solo per recuperare documenti disponibili pubblicamente, questo modello era perfettamente adeguato. Tuttavia esso non è adatto per fornire pagine differenti a utenti diversi a seconda di quello che hanno già fatto con il server.

Il problema è stato risolto con un meccanismo spesso criticato, che è quello dei **cookie**. Il nome deriva da un antico gergo dei programmatore, in cui un programma chiama una procedura e ottiene qualcosa che potrebbe in seguito dover presentare per svolgere un lavoro.

Un cookie è un piccolo le o stringa (che non supera i 4 KB) che il server può associare a un browser. Questa associazione non è la stessa cosa di un utente, ma è più simile e più utile di un indirizzo IP. I browser archiviano i cookie in un'apposita directory sul disco rigido del client, a meno che l'utente non li abbia disattivati. I cookie sono solo le o stringhe, non programmi eseguibili; potrebbero teoricamente contenere un virus, ma visto che sono trattati come dati, non esiste un modo ufficiale per mandare in esecuzione l'ipotetico virus e quindi causare danni. Tuttavia non si può escludere che qualche hacker sfrutti un bug del browser per provocarne l'attivazione.

Come mostrato nella Figura 8.7, un cookie può contenere no a cinque campi. *Domain* (dominio) indica da dove proviene il cookie. I browser dovrebbero veri care che i server non mentano in merito al loro dominio. Su ogni client non può essere mantenuto un numero superiore a venti cookie provenienti dallo stesso dominio. *Path* (percorso) è un percorso nella struttura delle directory del server che identifica quale parte della struttura di le del server può utilizzare il cookie. Spesso corrisponde a “/”, che indica l'intera struttura.

Domino	Percorso	Contenuto	Scadenza	Sicuro
toms-casino.com	/	CustomerID=497793521	15-10-10 17:00	Sì
jills-store.com	/	Cart=1-00501;1-07031;2-13721	11-1-11 14:22	No
aportal.com	/	Prefs=Stk:CSCO+ORCL;Spt:Jets	31-12-20 23:59	No
sneaky.com	/	UserID=3627239101	31-12-19 23:59	No

Figura 8.7: Alcuni esempi di cookie.

Il campo *Content* (contenuto) assume la forma nome = valore. Nome e valore sono due stringhe a discrezione del server e rappresentano il contenuto del cookie.

Il campo *Expires* (scadenza) specifica quando scade il cookie. Se il campo è assente, il browser scarica il cookie alla sua chiusura; un cookie di questo tipo è definito cookie non persistente. Se vengono fornite una data e un'ora, il cookie è persistente ed è conservato fino alla scadenza. I tempi di scadenza sono forniti secondo l'ora di Greenwich. Per rimuovere un cookie dal disco rigido del client un server deve inviarlo di nuovo, ma con un tempo di scadenza già trascorso.

Per finire, il campo *Secure* (sicuro) può essere impostato per indicare che il browser può restituire il cookie a un server solo usando un sistema di trasporto sicuro come SSL/TLS, che descriveremo nel Capitolo 8. Questa funzionalità è utilizzata per il commercio elettronico, il remote banking e altre applicazioni sicure.

8.4.4 HTTP

Illustrati i contenuti Web e le applicazioni è tempo di studiare il protocollo usato per trasportare tutte queste informazioni tra server Web e client: il protocollo **HTTP** (hypertext transfer protocol), come specificato nell'RFC 2616. HTTP è un protocollo semplice basato sul modello di domanda e risposta (request-response), normalmente implementato su TCP. Specifica quali messaggi i client possono inviare ai server e quali risposte vengono restituite. Esattamente come in SMTP, le intestazioni delle domande e delle risposte sono formulate in ASCII. I contenuti sono dati in un formato simile a quello MIME, ancora come in SMTP. Questo semplice modello è stato parzialmente responsabile dell'iniziale successo del Web, perché semplificava sviluppo e installazione.

All'inizio del Web, con HTTP 1.0, dopo avere stabilito la connessione, veniva inviata una singola richiesta e restituita una singola risposta; a questo punto la connessione TCP veniva rilasciata. In un mondo in cui la tipica pagina Web era composta interamente da testo HTML, questo metodo era adeguato. Nel giro di pochi anni la pagina Web iniziò a contenere sempre più collegamenti a contenuti quali icone e altri elementi per migliorarne l'aspetto, ma stabilire una connessione TCP per trasportare una singola icona diventava un modo molto costoso di operare.

Questa osservazione ha portato ad HTTP 1.1, che supporta **connessioni persistenti**. Con questa tecnologia è possibile stabilire una connessione TCP, inviare una richiesta, ottenere una risposta e poi inviare altre richieste e ottenere nuove risposte. Questa strategia è detta **riutilizzo della connessione**. Ammortizzando l'impostazione e il rilascio di TCP su più richieste il corrispondente overhead è inferiore. È inoltre possibile concatenare le richieste, vale a dire inviare la richiesta 2 prima dell'arrivo della risposta alla richiesta 1.

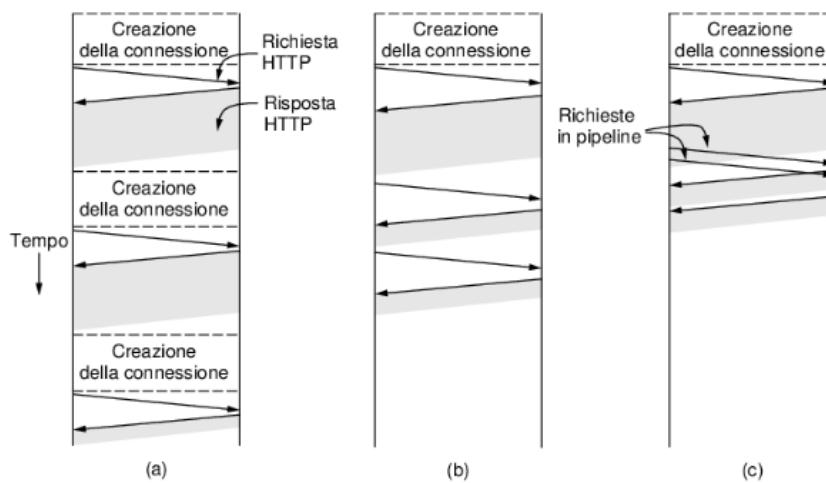


Figura 8.8: HTTP con (a) connessioni multiple e richieste sequenziali. (b) Una connessione persistente e richieste sequenziali. (c) Una connessione persistente e richieste in pipeline.

Anche se HTTP fu progettato per l'utilizzo sul Web è stato intenzionalmente reso più generico del necessario con un occhio alle future applicazioni orientate agli oggetti. Per questo motivo sono supportate

varie operazioni (**metodi**), al di là della richiesta di una pagina Web; chiamate metodi; questa generalità ha permesso la nascita di SOAP.

La riga della richiesta (per esempio la riga con il metodo GET) può essere seguita da altre righe con ulteriori informazioni. Sono chiamate **intestazioni di richiesta** (*request header*). Queste informazioni si possono assimilare ai parametri della chiamata di una procedura. Anche le risposte possono avere **intestazioni di risposta** (*response header*). Alcune intestazioni sono utilizzabili in entrambe le direzioni. Come vedete questa lista è piuttosto lunga; potete quindi ben immaginare che spesso nelle richieste e nelle risposte ci sono molte intestazioni.

8.4.5 Caching

Gli utenti tornano spesso alle pagine Web già viste precedentemente e pagine Web correlate hanno spesso incorporate le stesse risorse. Alcuni esempi sono le immagini usate per la navigazione attraverso un sito, così come fogli di stile comuni e script. Sarebbe un grave spreco dover recuperare tutte queste risorse ogni volta che queste pagine vengono visualizzate, perché il browser ne ha già una copia.

La procedura di memorizzare in una cache delle pagine recuperate per farne un uso successivo viene chiamata **caching**. Il vantaggio è che quando una pagina è memorizzata nella cache, non è necessario ripetere il trasferimento. HTTP ha un supporto integrato che aiuta i client a identificare quando possono riusare le pagine in modo sicuro.

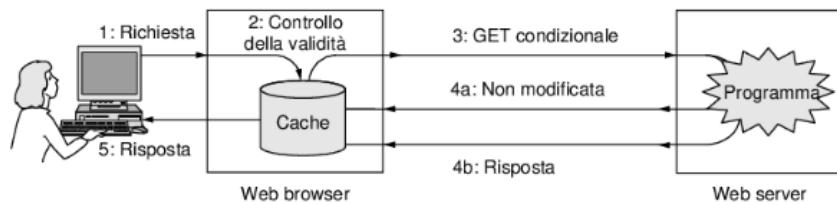


Figura 8.9: Cache HTTP.

Trovare pagine che non siano scadute vuol dire aver fatto un buon uso del caching, perché significa che il server non necessita di essere contattato. Sfortunatamente non sempre succede. I server devono usare l'intestazione *Expires* in modo conservativo, perché non possono essere sicuri di quando una pagina verrà aggiornata; quindi le copie memorizzate nella cache possono essere ancora valide, ma il client non lo sa. In questo caso viene usata la seconda strategia: si richiede al server se la copia memorizzata nella cache sia ancora valida. Questa richiesta è un **GET condizionale** (*conditional GET*) ed è mostrato nella Figura 8.9 al passo 3. Se il server sa che la copia nella cache è ancora valida, invia una breve replica per dirlo (passo 4a), altrimenti invia la risposta completa (passo 4b).

8.5 Distribuzione dei Contenuti

Come la rete telefonica, Internet era storicamente basata unicamente sulle comunicazioni. Nei primi anni gli accademici comunicavano con macchine remote facendo login tramite la rete per effettuare delle attività. Le persone hanno usato le e-mail per comunicare tra di loro per un lungo periodo e adesso usano il video e la voce su IP nello stesso modo. Tuttavia, con la crescita del Web, Internet ha cominciato a riguardare più i contenuti della comunicazione. Molte persone usano il Web per reperire informazioni e c'è un'incredibile attività di condivisione di file peer-to-peer per poter accedere a film, musica e programmi. Il cambio verso i contenuti è stato così evidente che ormai la maggior parte della banda Internet è usata per trasmettere video registrati.

In risposta alle richieste dei consumatori, una gran quantità di banda venne fornita nel core di Internet e connettività a larga banda più veloce ai confini della rete. Questa banda fu fondamentale per aumentare le prestazioni, ma è solo una parte della soluzione. Per ridurre il ritardo in rete i ricercatori svilupparono anche nuove architetture in modo tale da usare la banda per la distribuzione di contenuti. Una di queste architetture è una CDN (content distribution network), nella quale un provider installa

un insieme di macchine distribuito in diversi luoghi di Internet e li usa per distribuire contenuti ai client. Questa è la scelta delle grandi aziende. Un'architettura alternativa è una rete P2P (peer-to-peer), nella quale un insieme di computer usa le proprie risorse per distribuire contenuti al suo interno senza server separati o punti centrali di controllo.

Per lungo tempo si è saputo che c'è un piccolo numero di siti Web con un traffico massiccio e un gran numero di siti Web con un traffico molto più esiguo. Questa caratteristica è diventata parte del linguaggio delle reti. I primi documenti parlavano di traffico in termini di **treni di pacchetti**; l'idea è che i treni espressi con un gran numero di pacchetti in viaggio attraversino improvvisamente un collegamento (Jain e Routhier, 1986). Questo è stato formalizzato con un concetto di **auto-similarità** (self-similarity), che per i nostri scopi può essere pensato come il traffico di rete con molti vuoti brevi e lunghi, anche se osservati su diverse scale temporali (Leland et al., 1994). I lavori successivi parlarono di lunghi flussi di traffico come **elefanti** e di brevi flussi di traffico come **topi**. L'idea è che ci sono solo pochi elefanti e molti topi, ma gli elefanti contano perché sono molto grandi.

Tornando al contenuto del Web, lo stesso tipo di asimmetria è evidente. L'esperienza con negozi di video noleggio, biblioteche pubbliche e altre organizzazioni mostra come non tutti gli articoli siano ugualmente popolari. Sperimentalmente, quando sono disponibili N film, la frazione di tutte le richieste per il k -esimo più popolare è di circa C/k , dove C è una costante di normalizzazione perché la somma dia 1, vale a dire:

$$C = 1/(1 + 1/2 + 1/3 + 1/4 + 1/5 + \dots + 1/N)$$

Così, il film più popolare lo è sette volte di più del film numero sette. Questo risultato è noto come **legge di Zipf** (Zipf, 1949).

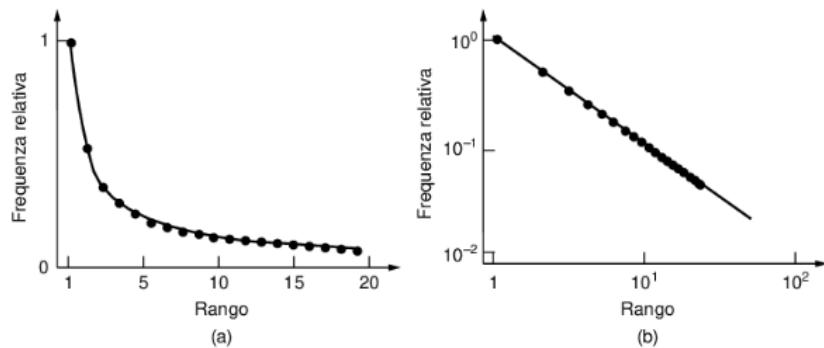


Figura 8.10: Distribuzione di Zipf (a) Su una scala lineare. (b) Su una scala log-log.

8.5.1 Server Farm e Proxy Web

L'architettura Web che abbiamo considerato ora ha un singolo server che dialoga con più client. Per costruire grandi siti Web che lavorino bene dobbiamo velocizzare l'elaborazione sia lato server sia lato client. Lato server, server Web più potenti possono essere costruiti con delle server farm, nelle quali un cluster di computer agisce come un singolo server. Lato client, si possono ottenere prestazioni ottimali usando tecniche migliori per l'uso della cache. In particolare, un proxy Web fornisce cache grande e condivisa per un gruppo di client.

8.5.1.1 Server Farm

Una macchina, indipendentemente da quanta banda usa, può soddisfare richieste Web solo finché il carico non sia troppo elevato. La soluzione in questo caso è usare più computer per fare un server Web. Questo metodo conduce al modello di **server farm** descritto nella Figura 7.65.

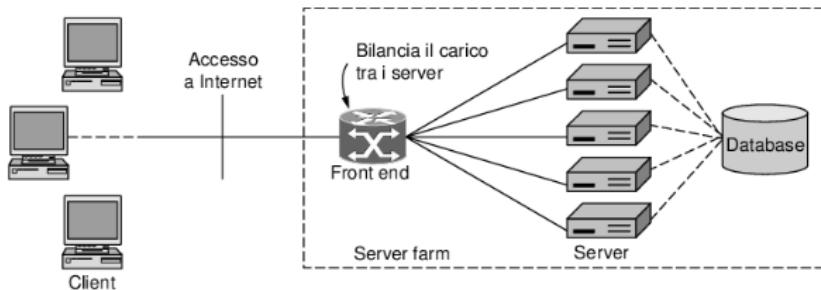


Figura 8.11: Una server farm.

Per fare in modo che un insieme di server appaia come un unico sito Web vi sono alcune possibili soluzioni. Tutte assumono che ogni server possa gestire una richiesta da qualsiasi client. Per fare ciò ogni server deve avere una copia del sito Web. A questo proposito, i server sono mostrati come connessi al database comune da una linea tratteggiata. Una soluzione consiste nell'usare il DNS per distribuire le richieste sui server della server farm.

Le altre soluzioni sono basate su **front end** (di solito uno switch a livello data link o un router IP, cioè un dispositivo che gestisce frame o pacchetti) che diondono le richieste in entrata su un insieme di server nella server farm. Questo accade persino quando il client contatta la server farm usando un singolo indirizzo IP di destinazione. Tutte queste soluzioni si basano sull'idea che il front end (o il server) guardi le intestazioni di rete, di trasporto o applicazione e le usi in modi non convenzionali. Una richiesta o una risposta Web sono trasportate da una connessione TCP; perché il servizio funzioni correttamente, il front end deve distribuire tutti pacchetti di una richiesta allo stesso server.

In uno schema più generale, il front end può ispezionare le intestazioni dei pacchetti IP, TCP e HTTP e associarli arbitrariamente a un server. L'associazione è chiamata politica di **bilanciamento del carico** (*load balancing*), perché l'obiettivo è bilanciare il carico tra i server: può essere semplice o complessa. Una politica semplice consiste nell'usare i server a turno (in *round-robin*). Con questo approccio il front end deve ricordarsi dell'associazione per ogni richiesta in modo che pacchetti successivi, che sono parte della stessa richiesta, vengano spediti al medesimo server. Inoltre, per rendere il sito Web più affidabile rispetto a un singolo server, il front end dovrebbe accorgersi quando i server non stanno più funzionando e smettere di mandar loro richieste.

Il concetto del **middlebox** rappresenta una violazione dei principi fondamentali della stratificazione dei protocolli di rete, rendendo questo schema generale non solo pericoloso ma anche fragile. Ogni livello di protocollo dovrebbe limitarsi a utilizzare solo la propria intestazione per il controllo e non dovrebbe ispezionare né utilizzare informazioni contenute nel payload. Tuttavia, i progettisti continuano a creare sistemi che ignorano queste regole. Quando questi sistemi smetteranno di funzionare a causa di cambiamenti nei livelli superiori, la sorpresa sarà inevitabile.

Un esempio di middlebox è un dispositivo, come uno switch o un router, che prende decisioni basate su informazioni provenienti dal livello di trasporto o da livelli superiori. Questo tipo di dispositivo si colloca nel mezzo del percorso di rete, in una posizione dove, secondo la gerarchia dei protocolli, non dovrebbe avere alcuna azione da compiere. I middlebox violano la separazione dei compiti tra i vari livelli della rete, creando potenziali problemi di compatibilità e funzionamento nel lungo termine.

8.5.1.2 Proxy Web

Le richieste e le risposte Web sono inviate usando HTTP. Abbiamo descritto come i browser possano fare uso di una cache per le risposte e la utilizzino per richieste successive. Il browser usa vari campi delle intestazioni e diverse regole per determinare se una copia di una pagina Web nella cache sia ancora valida. Un **proxy Web** è usato per fare condivisione di una cache tra utenti. Un proxy è un agente che agisce in vece di qualcuno, per esempio l'utente. Ci sono molti tipi di proxy, per esempio un proxy ARP risponde a richieste ARP al posto di un utente che si trova da qualche altra parte e non può rispondere.

Un proxy Web si occupa delle richieste Web al posto dei suoi utenti. Usualmente gestisce una cache delle risposte Web e, poiché le condivide, ha una cache sostanzialmente più grande di un browser.

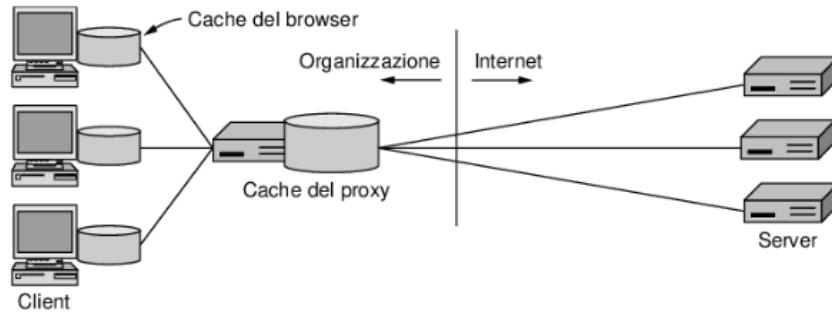


Figura 8.12: Una server farm.

La tipica configurazione per un'organizzazione che fa uso di un proxy è di usarne uno solo per tutti i suoi utenti. L'organizzazione potrebbe essere un'azienda o un ISP, ma entrambi beneficierebbero di una maggiore velocità nelle richieste Web e nella riduzione di banda necessaria. Benché le tariffe fisse, indipendenti dall'uso, siano comuni negli utenti finali, la maggior parte delle aziende e ISP pagano a seconda di quanta banda consumano. Questa configurazione è mostrata nella Figura 8.12.

Ulteriori proxy possono essere aggiunti per fornire livelli addizionali di cache. Ogni proxy (o browser) fa le proprie richieste attraverso il suo **upstream proxy** (*proxy di livello superiore*). Ogni proxy gestisce una cache per i **downstream proxy** (*proxy di livello inferiore*) e i browser. Risulta quindi possibile che un browser di un'azienda usi il proxy dell'azienda, che usa il proxy dell'ISP che contatta direttamente il server Web. Tuttavia, nella pratica, il singolo livello di cache mostrato nella Figura 8.12 è spesso sufficiente per guadagnare la maggior parte dei benefici.

8.6 Content Delivery Network

Server farm e proxy Web aiutano a costruire grandi siti e ad aumentare le prestazioni, ma non sono sufficienti per i siti Web veramente famosi che devono servire contenuti su scala globale. Per questi siti è necessario un approccio diverso.

Le **CDN** (*content delivery network*) sovvertono l'idea della cache Web tradizionale. Invece di avere dei client che cercano una copia della pagina richiesta in una cache vicina, nelle CDN è il provider a mettere una copia della pagina in un insieme di nodi in diversi posizioni e a dirigere il client in modo che usi un nodo a lui vicino come server. Un esempio del percorso ad albero seguito dai dati quando vengono distribuiti da una CDN è mostrato nella Figura 7.67. Il server di origine nella CDN distribuisce una copia del contenuto ad altri nodi nella CDN a Sydney, Boston e Amsterdam in questo esempio, come mostrato dalle linee tratteggiate. I client prendono quindi le pagine dal nodo a loro più vicino nella CDN, come si vede dalle linee continue. In questo modo entrambi i client a Sydney prendono una copia della pagina archiviata a Sydney; non la prendono dal server di origine che potrebbe trovarsi in Europa.

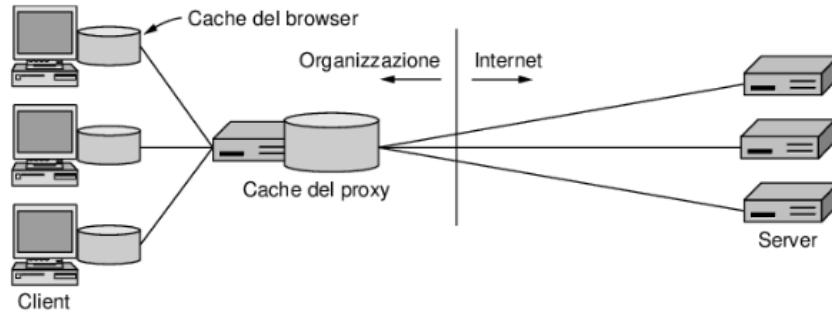


Figura 8.13: Albero di distribuzione CDN.

L'idea di usare un albero di distribuzione è semplice; meno semplice è pensare come organizzare i client in modo che usino questo albero. Per esempio, dei proxy server dovrebbero poter fornire una soluzione.

Tuttavia questa strategia fallisce nella pratica per tre ragioni. La prima è che i client in una certa parte della rete probabilmente appartengono a organizzazioni diverse e usano proxy diversi. Si ricordi che le cache non sono normalmente condivise dalle organizzazioni a causa del bene ciò limitato di usarle con un grande numero di client e anche per motivi di sicurezza. In secondo luogo, ci possono essere più CDN, ma ogni client usa un solo proxy. Quale CDN dovrebbe usare il client come proxy? Infine, forse il problema più pratico di tutti è che i proxy Web sono con gurati dal client. Possono venire con gurati per beneficiare o meno di una distribuzione di contenuti di una CDN e c'è poco che la CDN possa fare in questo senso.

Un altro semplice modo per supportare un albero di distribuzione con un livello è usare il mirroring. In questo approccio il server di origine replica i contenuti nei nodi CDN, come prima. I nodi CDN in diverse regioni della rete sono chiamati **mirror** (letteralmente, *specchi*, visto che riflettono il contenuto del server di origine). Le pagine Web sul server di origine contengono collegamenti esplicativi a mirror distinti, indicando di solito anche la loro locazione.

Il terzo approccio, che risolve le di colta dei primi due, usa il DNS ed è chiamato **DNS redirection (redirezione DNS)**. Si supponga che un client voglia prendere una pagina all'URL <http://www.cdn.com/page.html>. Per prelevare la pagina, il browser userà il DNS per risolvere www.cdn.com in un indirizzo IP. Il DNS procede nel solito modo. Usando il protocollo DNS il browser apprende l'indirizzo IP del name server per *cdn.com*, quindi lo contatta e gli chiede di risolvere *www.cdn.com*. Adesso arriva il colpo di genio: il name server è gestito dalla CDN; invece di restituire lo stesso indirizzo IP per ogni richiesta guarda l'indirizzo IP del client che fa la richiesta e restituisce risposte differenti. La risposta sarà l'indirizzo IP del nodo CDN che è più vicino al client, vale a dire che se un client a Sydney chiede al name server della CDN di risolvere *www.cdn.com*, questo restituirà l'indirizzo IP del nodo di Sydney, ma se un client ad Amsterdam fa la stessa richiesta, il name server restituirà l'indirizzo IP del nodo CDN di Amsterdam.

Una domanda complessa all'interno del procedimento è che cosa signi chi trovare il nodo CDN più vicino e come cercarlo. La definizione di "più vicino" non riguarda veramente la parte geografica. Nell'associazione tra client e nodi CDN ci sono almeno due fattori da considerare. Il primo è la distanza di rete; il client dovrebbe avere un percorso di rete verso il nodo CDN breve e ad alta capacità in modo da ottenere download veloci. Le CDN usano una mappa di associazioni elaborata preventivamente per tradurre l'indirizzo IP del client nella sua posizione in rete. Il nodo CDN selezionato potrebbe essere quello a distanza più corta, oppure quello che conta è la combinazione tra lunghezza del percorso di rete e limiti di capacità lungo di esso. Il secondo fattore è il carico già gestito dal nodo CDN. Se i nodi CDN sono sovraccarichi saranno lenti nel rispondere, come i server Web sovraccarichi, cosa che si deve assolutamente evitare. È quindi necessario bilanciare il carico tra i nodi CDN, associando alcuni client a nodi che sono un po' più lontani, ma meno carichi.

La prima che adottò la tecnica di usare il DNS per la distribuzione di contenuti fu Akamai nel 1998, quando il Web stava soffrendo sotto il carico della sua crescita iniziale. Fu la prima grande CDN e divenne un leader industriale. Probabilmente ancora più brillante dell'idea di usare il DNS per connettere client a nodi vicini fu la struttura di incentivazione del loro business. Le aziende pagano Akamai per consegnare i loro contenuti ai client, in modo da avere siti Web veloci molto apprezzati dai clienti. I nodi CDN devono essere posizionati nella rete in luoghi ad alta connettività, che inizialmente significava all'interno di reti di ISP. Per gli ISP il vantaggio di avere un nodo CDN all'interno della loro rete è che questo riduce la banda di rete in uscita (e a pagamento), come un proxy. Inoltre il nodo CDN aumenta la velocità di risposta ai clienti dell'ISP, che quindi lo apprezzano, dando un vantaggio competitivo rispetto a ISP che non hanno nodi CDN. Tali bene ci (che non costano all'ISP) fanno in modo che un ISP non ci pensi due volte a installare un nodo CDN. Quindi i fornitori di contenuti, gli ISP e i clienti si avvantaggiano tutti di questo schema e la CDN fa ottimi affari. Dal 1998 altre aziende si sono buttate in questo business che adesso è un'industria competitiva con molti provider.

I Content Delivery Network (CDN) ottimizzati per il video rappresentano una componente essenziale per la distribuzione efficace e scalabile di contenuti video online. Un esempio ben noto di CDN ottimizzato

per il video è Ustream, che offre servizi specifici per tre principali categorie di streaming video: **Pro Broadcasting, Corporate Internal Communications** (con Ustream Align) e **marketing per la generazione di lead** (con Ustream Demand).

Un altro CDN ottimizzato per il video è **JET-Stream**, precedentemente noto come **Streamzilla**. Questo CDN sta guadagnando quote di mercato in Europa ed è una scelta valida per chi ha un pubblico principalmente europeo. Oltre allo streaming, Streamzilla offre agli affiliati strumenti necessari per generare ricavi e creare una presenza di rete.

Streamzilla ha dato origine a una nuova idea che si è evoluta in JET-Stream. Questo nuovo approccio ha implementato lo streaming adattivo con bitrate variabile tramite HTTP, superando le offerte dei CDN tradizionali. Lo streaming HTTP ha aperto nuove opportunità, consentendo a Jet-Stream di ridefinire chi riceve il traffico e gli affari, integrando tutte le principali CDN nella piattaforma. Inoltre, nuovi algoritmi di bilanciamento del carico hanno dimostrato di poter ridurre i costi di streaming del 50%, aumentando notevolmente il tempo di attività e garantendo prestazioni superiori.

I **MultiCDN**, una tecnologia emergente, offrono nuove capacità tra cui un bilanciamento del carico imposto che distribuisce immediatamente il traffico senza dipendere da sistemi DNS di terze parti. Se una CDN fallisce, viene immediatamente esclusa dal pool. Con il concetto di **CDN stacking**, il traffico viene bilanciato su un pool di CDN, creando di fatto il CDN più grande al mondo sfruttando la capacità combinata di tutte le CDN. Il bilanciamento del carico può essere ottimizzato in modo granulare per singoli ISP e fino agli indirizzi IP. L'integrazione profonda con le CDN globali consente di controllarle come se fossero server edge propri, con funzioni di flushing, monitoraggio, controllo degli accessi, analisi dei log, tutto automatizzato. Infine, si può ottimizzare il traffico su un mix ibrido di risorse, utilizzando sia il proprio CDN che i propri edge server, oltre alle CDN globali.

Capitolo 9

Sicurezza delle Reti

Nei primi decenni della loro esistenza le reti di calcolatori vennero utilizzate prevalentemente dai ricercatori universitari per inviare e ricevere e-mail e dalle aziende per condividere le stampanti. In quelle applicazioni non si prestava particolare attenzione alla sicurezza. Oggi, milioni di persone usano le reti per fare acquisti, home banking oppure compilare la dichiarazione dei redditi, quindi la sicurezza delle reti sta apparendo all'orizzonte come un problema potenzialmente molto vasto. In questo capitolo studieremo la sicurezza delle reti da diversi punti di vista; indicheremo molti punti critici e discuteremo un gran numero di algoritmi e protocolli che servono a rendere le reti più sicure. La sicurezza è un argomento ampio, che copre una molitudine di problemi. Nella sua forma più semplice, riguarda come fare in modo che intrusi non riescano a leggere (o, peggio ancora, modi care di nascosto) i messaggi destinati a terzi. Si occupa inoltre di impedire che determinate persone possano accedere a servizi remoti che non sono autorizzate a usare. La sicurezza si occupa anche di come accertarsi dell'identità dei mittenti dei messaggi, di come impedire l'intercettazione e la ripetizione di messaggi legittimi catturati sulla rete e di come perseguire chi a erma di non aver mai spedito certi messaggi.

9.1 Controllo degli Accessi

Il controllo degli accessi è un processo fondamentale che determina "chi può fare cosa a cosa" basandosi su una specifica politica. Questo processo è essenziale per limitare l'accesso a un sistema o alle risorse del sistema in base a criteri che vanno oltre l'identità dell'utente. Adottare una politica di controllo degli accessi porta diversi benefici, come l'attenzione dell'organizzazione su problemi di sicurezza, che può tradursi in un'allocazione più efficiente delle risorse per la sicurezza del sistema. Inoltre, permette di configurare la sicurezza appropriata per ciascuna risorsa del sistema in base al ruolo e all'importanza, facilitando anche le operazioni di auditing e testing del sistema.

I diritti di accesso regolano le autorizzazioni degli utenti per ogni risorsa del sistema. Il controllo degli accessi coinvolge quattro elementi principali: i soggetti (utenti e gruppi di utenti), gli oggetti (file e risorse come memoria, stampanti e scanner), le operazioni e un monitor di riferimento che verifica le regole per concedere l'accesso a un oggetto a un soggetto. Un insieme di regole che specificano le restrizioni è noto come lista di controllo degli accessi (ACL), dove i soggetti sono utenti o gruppi di utenti e gli oggetti sono file e risorse di sistema.

Le modalità di accesso possono essere di due tipi: **osservare** o **alterare**, e ci sono quattro diritti di accesso: **eseguire**, **leggere**, **aggiungere** e **scrivere**. I diritti di accesso possono essere impostati individualmente per ogni risorsa del sistema e per ciascun utente o gruppo, con i diritti degli utenti che prevalgono sempre sui diritti dei gruppi. Il proprietario della risorsa è responsabile della definizione dei diritti di accesso.

Sono state sviluppate diverse tecniche e tecnologie per gestire i diritti di accesso per ogni utente e per ogni oggetto, tra cui matrici di controllo degli accessi, tabelle di capacità, liste di controllo degli accessi, controllo degli accessi basato sui ruoli, controllo degli accessi basato su regole, interfacce limitate e controllo degli accessi dipendente dal contenuto. La matrice di controllo degli accessi organizza tutte le informazioni necessarie per l'amministrazione del controllo degli accessi in una matrice con righe che rappresentano i soggetti o gruppi di soggetti e colonne che rappresentano gli oggetti, mostrando nel corpo

della matrice i permessi dei soggetti sugli oggetti. Le **liste di controllo degli accessi (ACL)** memorizzano i gruppi con i diritti di accesso associati agli oggetti, mentre le tabelle di capacità specificano le operazioni che un soggetto può eseguire su un oggetto.

Il concetto di **"AAA"** è il pilastro di qualsiasi disciplina sistematica della sicurezza, composta da **autenticazione, autorizzazione e contabilità**. L'autenticazione conferma l'identità degli utenti remoti che accedono alla rete, l'autorizzazione assegna diritti differenziati per l'utilizzo di servizi specifici e la contabilità registra tutte le operazioni di un utente durante il processo di servizio di rete per raccogliere informazioni sull'uso delle risorse di rete.

Il processo di autenticazione coinvolge diverse forme di autenticazione, tra cui sistemi biometrici, dispositivi di sicurezza e autenticazione a più livelli. Ad esempio, un utente può inviare il proprio nome utente e password a un server AAA per essere autenticato; se le credenziali sono corrette, il server AAA invierà un messaggio di accettazione, altrimenti invierà un messaggio di rifiuto, bloccando l'accesso alla rete.

9.2 Crittografia

Storicamente quattro gruppi di persone hanno usato e dato il loro contributo all'arte della crittografia: i militari, il corpo diplomatico, gli scrittori di diari e gli amanti. Fra tutti questi, i militari hanno avuto il ruolo più importante e hanno dato forma alla disciplina nel corso dei secoli. All'interno delle organizzazioni militari, i messaggi da cifrare erano tradizionalmente assegnati ad addetti di basso livello e bassa paga che avevano il compito di cifrarli e poi trasmetterli. Il notevole volume di messaggi faceva sì che non si potesse usare una squadra di specialisti di alto livello per questo lavoro. Fino all'avvento dei computer, uno dei principali limiti della crittografia era la capacità degli addetti alla codifica di riuscire a compiere le operazioni necessarie, spesso anche sul campo di battaglia e utilizzando un equipaggiamento minimo. Un vincolo aggiuntivo era anche la difficoltà nel cambiare velocemente da un metodo crittografico a un altro, in quanto questo significava dover formare una gran quantità di persone. Tuttavia, il pericolo che un addetto alla codifica capotasse essere catturato dal nemico rendeva essenziale la possibilità di cambiare il metodo crittografico all'istante, se necessario. Questi requisiti in comune fra loro hanno dato origine al modello della Figura 8.2.

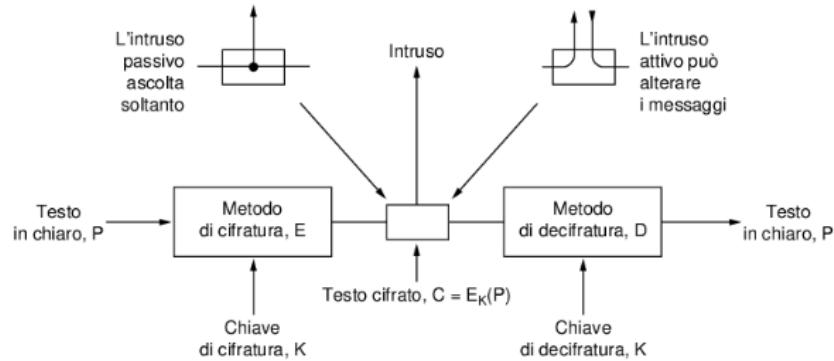


Figura 9.1: Modello di crittografia per un cifrario a chiave simmetrica.

Un messaggio da cifrare, detto **testo in chiaro (plaintext)**, viene trasformato tramite una funzione parametrizzata da una chiave. L'output del processo di cifratura, noto come **testo cifrato (ciphertext)**, viene generalmente trasmesso tramite un messaggero o via radio. Assumiamo che il nemico, o intruso, ascolti accuratamente e trascriva completamente il testo cifrato. Al contrario del destinatario legittimo, l'**intruso** non conosce la chiave di decifrazione e quindi non riesce facilmente a decifrare il testo cifrato. In alcuni casi l'intruso, oltre ad ascoltare il canale di comunicazione (intruso passivo), riesce anche a registrare i messaggi per poi ritrasmetterli aggiungendo delle informazioni oppure per modificare i messaggi legittimi prima che raggiungano il destinatario (intruso attivo). L'arte di far saltare (*o rompere, forzare*) i cifrari, chiamata criptoanalisi e l'arte di inventarli (*crittografia*) sono note sotto il nome collettivo di crittologia. Decrittare è l'attività di decifrazione da parte di un criptoanalista (intruso), mentre decifrare è l'operazione legittima di lettura di un messaggio cifrato. È utile avere una notazione per relazionare

fra loro il testo in chiaro, quello cifrato e le chiavi. Useremo la notazione $C = E_K(P)$ per indicare che la cifratura del testo in chiaro P usando la chiave K genera il testo cifrato C . Analogamente, $P = D_K(C)$ indica la decifrazione di C per estrarre il testo in chiaro. Segue quindi che $D_K(E_K(P)) = P$. Questa notazione suggerisce che E e D sono semplicemente delle funzioni matematiche.

Crittografia simmetrica è un metodo di codifica in cui il messaggio originale (plaintext) viene trasformato in un messaggio cifrato (ciphertext) utilizzando una chiave segreta. Per decifrare il ciphertext e ottenere nuovamente il plaintext, è necessaria la stessa chiave utilizzata per la cifratura. Questo metodo non richiede che l'algoritmo di crittografia sia tenuto segreto; l'unica cosa che deve essere mantenuta riservata è la chiave. Questa caratteristica rende la crittografia simmetrica adatta per un uso diffuso.

Crittografia asimmetrica utilizza un sistema di crittografia in cui le operazioni di cifratura e decifratura sono eseguite con chiavi diverse: una chiave pubblica e una chiave privata. La crittografia asimmetrica trasforma il plaintext in ciphertext utilizzando una delle due chiavi e un algoritmo di crittografia. Successivamente, il plaintext può essere recuperato dal ciphertext utilizzando la chiave associata e un algoritmo di decifratura. Questo tipo di crittografia può essere utilizzato per garantire la riservatezza, l'autenticazione o entrambe. Gli algoritmi a chiave pubblica si basano su funzioni matematiche piuttosto che su sostituzioni e permutazioni, rendendo possibile la sicurezza anche quando l'algoritmo è noto pubblicamente.

9.3 Protocolli di Autenticazione

L'**autenticazione** è la tecnica usata dai processi per verificare che la loro controparte nella comunicazione sia veramente chi indica di essere e non un impostore. La verifica dell'identità di un processo remoto, per combattere gli intrusi attivi e in malafede, è un compito sorprendentemente difficile, che richiede dei protocolli crittografici complessi. In questo paragrafo studieremo alcuni dei molti protocolli di autenticazione usati per le reti insicure. Per inciso, alcune persone confondono l'autorizzazione con l'autenticazione. L'autenticazione riguarda il problema di assicurare l'identità del processo con cui si sta comunicando. L'autorizzazione si occupa, invece, di stabilire che cosa può fare un processo. Per esempio, consideriamo un processo client che contatta un server e gli comunica: sono il processo di Scott e voglio cancellare il file *ricettario.old*. Il server ha bisogno delle risposte a queste due domande:

1. la richiesta viene veramente da un processo di Scott (autenticazione)?
2. Scott è autorizzato a cancellare *ricettario.old* (autorizzazione)?

La richiesta può essere evasa soltanto dopo che entrambe le domande hanno ottenuto una risposta certa e affermativa. La prima domanda è quella veramente cruciale. Una volta che il server sa con chi sta parlando, il controllo dell'autorizzazione consiste solamente in una ricerca dentro tabelle locali o in un database. Per questo motivo, in questo paragrafo ci concentreremo sull'autenticazione.

Un esempio di metodo di autenticazione è l'uso dell'indirizzo IP. Tuttavia, è possibile creare datagrammi IP con indirizzi IP falsi se si ha accesso al kernel del sistema operativo, una tecnica conosciuta come IP spoofing. Un approccio comune per l'autenticazione è l'uso di password, come avviene nei PIN dei dispositivi bancari o nei login dei sistemi operativi. Questo metodo non è completamente sicuro perché un malintenzionato può intercettare la password. Ad esempio, Telnet invia la password in chiaro, quindi se qualcuno monitora la rete locale può rubare le password di login.

Per migliorare la sicurezza, le password possono essere cifrate utilizzando una chiave simmetrica. Questo evita il furto della password ma non risolve completamente il problema dell'autenticazione. Un **attacco di riproduzione** (*playback attack*) può registrare la password cifrata e un malintenzionato può usarla per fingersi un altro utente. Una soluzione è utilizzare una password diversa per ogni autenticazione. Ad esempio, Alice e Bob potrebbero concordare una sequenza di password o un algoritmo di generazione.

Un approccio più generale utilizza un numero univoco chiamato **nonce**, che viene utilizzato una sola volta nel protocollo. Alice invia un messaggio dicendo "Sono Alice". Bob sceglie un nonce R e lo invia ad Alice. Alice quindi cifra il nonce con la sua chiave segreta e lo rimanda a Bob. Se Bob riesce a decifrare il nonce e verificare che è quello che ha inviato, Alice è autenticata.

Un altro metodo utilizza la crittografia a chiave pubblica. Alice invia un messaggio dicendo "Sono Alice". Bob sceglie un nonce e lo invia ad Alice. Alice cifra il nonce con la sua chiave privata e lo invia a Bob. Bob decifra il nonce con la chiave pubblica di Alice e se ottiene il valore corretto, Alice è autenticata. Tuttavia, questo metodo è sicuro quanto il sistema di distribuzione delle chiavi pubbliche, che deve essere affidabile per evitare che un malintenzionato si spacci per Alice.

Tra i vari protocolli di autenticazione, **OAuth2** è noto per essere semplice da implementare e fornire un'autorizzazione lato server del codice, ma è vulnerabile nella gestione di diversi set di codice e può avere seri effetti sui siti connessi a un sistema compromesso. **RADIUS** è un meccanismo efficace per fornire accesso multiplo agli amministratori e assegna un'identità unica a ogni utente in una sessione, ma la sua implementazione iniziale può essere complessa e costosa. SAML riduce i costi amministrativi per gli utenti finali e offre un unico punto di accesso per autenticarsi attraverso vari fornitori di servizi, ma dipende fortemente dal fornitore di identità e gestisce tutti i dati in un unico formato XML.

9.4 Certificati Digitali

Come primo tentativo per distribuire le chiavi pubbliche in modo sicuro, possiamo immaginare l'esistenza di un centro di distribuzione delle chiavi (KDC, *key distribution center*), aperto 24 ore su 24 per fornire le chiavi pubbliche su richiesta. Uno dei problemi associati a questo tipo di soluzione è la mancanza di scalabilità e quindi il fatto che il centro di distribuzione delle chiavi diventerebbe presto il collo di bottiglia del sistema. Inoltre, se questo centro dovesse andare fuori servizio la sicurezza di Internet subirebbe un'improvvisa battuta d'arresto. Per questo motivo è stata sviluppata una soluzione differente, che non richiede un centro di distribuzione delle chiavi on-line a tutte le ore. In effetti non deve essere affatto on-line: quello che deve fare è certi care le chiavi pubbliche che appartengono alle persone, alle aziende e alle altre organizzazioni. Un'organizzazione che certifica le chiavi pubbliche è chiamata una **CA** (*certification authority*, autorità di certificazione). Supponiamo, per esempio, che Bob voglia autorizzare Alice, insieme ad altre persone, a comunicare con lui in modo sicuro. Bob può rivolgersi alla CA portando la sua chiave pubblica e il suo passaporto o un altro documento, e chiedere di essere certi cato. La CA emette un certificato, ne calcola l'hash SHA-1 e lo firma con la chiave privata di CA. Bob paga un compenso alla CA e ottiene un floppy disk con il certificato e l'hash firmato.

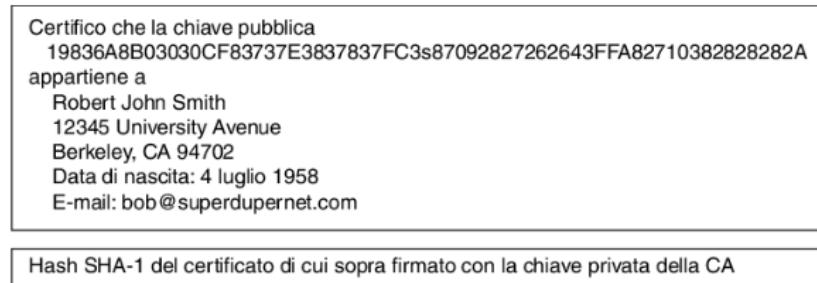


Figura 9.2: Esempio di certificato e del suo hash firmato.

Queste dispense sono state stilate tramite L^AT_EX.

Il PDF è dotato di hyperlinking quindi è possibile cliccare sui nomi delle sezioni nell'indice per spostarsi tra le pagine,
allo stesso modo, per tornare all'indice è sufficiente cliccare sul numero della pagina attuale.

©2024