

Reti di Calcolatori  
**Canonico Roberto**  
a.a. 2023-2024

**Author**  
Alessio Romano

March 7, 2025

# Contents

<b>1 Modello a strati</b>	<b>7</b>
1.1 Definizione . . . . .	7
1.2 Il modello OSI . . . . .	8
<b>2 L3: rete</b>	<b>9</b>
2.1 Packet switching . . . . .	9
2.1.1 Datagram . . . . .	10
2.1.2 QoS . . . . .	10
2.2 Protocolli . . . . .	11
2.2.1 It.png Pv4 . . . . .	11
<b>3 L4: Trasporto</b>	<b>12</b>
3.1 Multiplexing-Demultiplexing . . . . .	12
3.1.1 UDP . . . . .	13
3.1.2 TCP . . . . .	13
3.2 Client-Server . . . . .	13
<b>4 HTTP (applicativo)</b>	<b>13</b>
4.1 Funzionamento . . . . .	13
4.1.1 URL . . . . .	14
4.2 Richiesta . . . . .	14
4.2.1 GET . . . . .	15
4.2.2 HEAD . . . . .	15
4.2.3 POST . . . . .	15
4.2.4 PUT . . . . .	15
4.3 Risposta . . . . .	15
4.3.1 Status Code . . . . .	16
4.4 Header . . . . .	16
4.4.1 Header generali . . . . .	16
4.4.2 Header risposta . . . . .	16
4.4.3 Header identità . . . . .	17
4.5 Cookies . . . . .	17
4.6 Connessioni . . . . .	18
4.7 Web Caching . . . . .	18
4.8 Browser Web . . . . .	19
<b>5 DNS (applicativo)</b>	<b>19</b>
5.1 Soluzioni . . . . .	19
5.2 Attori . . . . .	20
5.3 Tipologie di DNS . . . . .	20
5.3.1 Root Name Server . . . . .	20
5.3.2 Top Level Domain . . . . .	20
5.3.3 Authoritative . . . . .	21
5.3.4 Local Name Server . . . . .	21
5.4 Risoluzioni DNS . . . . .	21
5.4.1 Query Iterative . . . . .	21
5.4.2 Query Recursive . . . . .	22
5.5 Caching . . . . .	22
5.5.1 Resource Records . . . . .	22

5.6	Protocollo DNS . . . . .	23
<b>6</b>	<b>FTP (applicativo/trasporto)</b>	<b>23</b>
6.1	Funzionamento . . . . .	23
6.1.1	Controllo . . . . .	24
6.2	FTP Attivo/Passivo . . . . .	24
<b>7</b>	<b>SMTP (applicativo)</b>	<b>25</b>
7.1	Funzionamento . . . . .	25
7.2	Protocollo FTP . . . . .	25
7.2.1	MIME . . . . .	26
7.3	POP3 . . . . .	26
7.3.1	Caratteristiche principali di POP3 . . . . .	26
7.3.2	Esempi di utilizzo . . . . .	27
7.3.3	Limiti di POP3 . . . . .	27
7.3.4	Differenze rispetto ad altri protocolli . . . . .	27
<b>8</b>	<b>CDN (applicativo)</b>	<b>27</b>
<b>9</b>	<b>P2P</b>	<b>27</b>
9.1	Directory centralizzata/decentralizzata . . . . .	28
9.1.1	Directory centralizzata . . . . .	28
9.1.2	Directory decentralizzata . . . . .	29
<b>10</b>	<b>IPv4 (rete)</b>	<b>29</b>
10.1	Header IPv4 . . . . .	30
10.1.1	Identification, Flags, Fragment offset . . . . .	30
10.1.2	Frammentazione e riassemblaggio . . . . .	31
10.1.3	Opzioni dell'header IPv4 . . . . .	31
10.2	Indirizzi IP . . . . .	32
10.2.1	Indirizzi IP e classi . . . . .	32
10.3	Netmask . . . . .	33
10.3.1	Fixed Length Subnet Mask . . . . .	34
10.3.2	Variable Length Subnet Mask . . . . .	34
10.4	Router . . . . .	35
10.4.1	Forwarding/Routing . . . . .	35
10.4.2	Routing Statico e Dinamico . . . . .	36
<b>11</b>	<b>Protocollo ARP (data link)</b>	<b>36</b>
11.1	Ethernet . . . . .	37
11.2	ARP . . . . .	37
11.2.1	ARP Packet Fomat . . . . .	39
11.2.2	Gratuitos ARP . . . . .	39
11.2.3	ARP Probe . . . . .	40
11.2.4	ARP Announcement . . . . .	40
<b>12</b>	<b>Internet Control Message Protocol (ICMP Applicativo)</b>	<b>41</b>
12.1	Scopo di ICMP . . . . .	41
12.2	Funzionamento di ICMP . . . . .	41
12.2.1	Struttura di un messaggio ICMP . . . . .	41
12.2.2	Esempi di messaggi ICMP . . . . .	42

12.3 Sicurezza e ICMP . . . . .	42
<b>13 NAT (rete)</b>	<b>42</b>
13.1 Funzionamento . . . . .	42
<b>14 Protocollo DHCP (applicativo)</b>	<b>43</b>
14.1 Cos'è il DHCP . . . . .	43
14.2 A cosa serve il DHCP? . . . . .	43
14.3 Funzionamento del DHCP . . . . .	44
14.4 Vantaggi del DHCP . . . . .	44
<b>15 Routing (rete)</b>	<b>45</b>
15.1 Tipologie di routing . . . . .	45
15.2 Tecniche di routing . . . . .	46
<b>16 Routing Distance Vector</b>	<b>46</b>
16.1 Equazione di Bellman-Ford . . . . .	47
16.2 Velocità di Convergenza . . . . .	47
16.3 Poisoned Reverse . . . . .	47
<b>17 Link-State e Dijkstra</b>	<b>47</b>
17.1 Routing Link-State . . . . .	47
17.1.1 LSP flooding . . . . .	48
17.1.2 Gestione LSP . . . . .	49
17.1.3 Architettura Link-State . . . . .	49
17.2 Algoritmo di Dijkstra . . . . .	50
17.2.1 Descrizione dell'Algoritmo . . . . .	50
<b>18 Protocollo OSPF (rete)</b>	<b>50</b>
<b>19 Routing gerarchico (rete)</b>	<b>51</b>
19.1 Tipi di AS . . . . .	52
19.2 Instradamento gerarchico . . . . .	53
<b>20 Multicast</b>	<b>53</b>
20.1 Indirizzo Multicast IPv4 . . . . .	53
20.2 Multicast ethernet . . . . .	54
20.3 Multicast Router . . . . .	54
20.4 Protocolli multicast . . . . .	54
20.4.1 IGMP . . . . .	55
20.5 Routing Multicast . . . . .	56
20.5.1 Group-Shared Tree . . . . .	57
20.5.2 Source-based Tree . . . . .	57
<b>21 MBone (rete)</b>	<b>58</b>
<b>22 IPv6</b>	<b>58</b>
22.1 Differenze . . . . .	58
22.2 Header . . . . .	59
22.2.1 Header principale . . . . .	59
22.2.2 Extension Header . . . . .	59
22.3 Indirizzi IPv6 . . . . .	60

22.4 Indirizzi IPv6 Unicast . . . . .	60
22.4.1 Indirizzi Link-Local . . . . .	61
22.4.2 Indirizzi IPv6 Global Unicast . . . . .	61
22.4.3 Indirizzi IPv6 Multicast . . . . .	61
22.5 Assegnazione ad Host . . . . .	62
22.6 ICMPv6 . . . . .	62
22.7 Mapping IPv6 - IPv4 . . . . .	62
<b>23 Multimedia (trasporto (trasporto))</b>	<b>63</b>
23.1 Live/Pre-registrate . . . . .	63
23.1.1 Pre-registrate . . . . .	63
23.1.2 Live . . . . .	63
<b>24 RTP/RTCP (applicativo/trasporto)</b>	<b>64</b>
24.1 RTP . . . . .	64
24.2 Struttura . . . . .	64
24.2.1 Header RTP . . . . .	65
24.3 Sessione RTP . . . . .	65
24.4 RTCP . . . . .	65
<b>25 Trasmissione affidabile (trasporto)</b>	<b>66</b>
25.1 Soluzioni . . . . .	66
25.2 Stop and Wait . . . . .	67
25.3 Pipelining . . . . .	68
25.3.1 Go Back-N . . . . .	68
25.3.2 Ripetizione selettiva . . . . .	69
<b>26 TCP</b>	<b>69</b>
26.1 Struttura . . . . .	70
26.1.1 Opzioni header TCP . . . . .	70
26.2 Affidabilità . . . . .	71
26.3 Connection Establishment . . . . .	72
26.4 Schema riassuntivo ACK . . . . .	73
26.5 Flow Control . . . . .	73
26.6 Congestion Control . . . . .	74
26.6.1 TCP congestion control . . . . .	75
<b>27 Ethernet (data link)</b>	<b>76</b>
27.1 Indirizzi LAN . . . . .	76
27.2 Ethernet . . . . .	77
27.2.1 CSMA/CD . . . . .	77
27.3 Standard Ethernet Storici e Moderni . . . . .	78
27.4 Differenze tra Cavi UTP . . . . .	79
<b>28 Dispositivi LAN (data link)</b>	<b>79</b>
28.1 Hub . . . . .	79
28.2 Bridge . . . . .	80
28.3 Switch . . . . .	80
<b>29 VLAN (data link)</b>	<b>81</b>
29.1 Funzionamento . . . . .	82

29.2 Comunicazione tra VLAN . . . . .	83
29.3 VLAN trunking . . . . .	84
<b>30 Crittografia</b>	<b>85</b>
30.1 Chiave simmetrica . . . . .	85
30.1.1 Cipher block-chaining . . . . .	87
30.2 Chiave pubblica . . . . .	87
<b>31 Sicurezza</b>	<b>88</b>
31.1 Hashing functions . . . . .	88
31.1.1 Collisioni . . . . .	89
31.1.2 Message Authentication Code . . . . .	89
31.2 Firma Digitale . . . . .	89
31.3 Certificazione della chiave pubblica . . . . .	90
31.4 Autenticazione . . . . .	91
31.5 Email sicura . . . . .	92
<b>32 Wireless</b>	<b>93</b>
32.1 Wireless LAN 802.11 . . . . .	94
32.1.1 Associazione . . . . .	94
32.1.2 Metodo di accesso . . . . .	95
32.2 CSMA/CA . . . . .	95

# 1 Modello a strati

Cos'è un computer network? Per rete di calcolatori si intende un insieme di calcolatori connessi in vari modi, con l'obiettivo di comunicare e condividere risorse. Il problema della comunicazione tra computer, tuttavia, è molto complesso ed incontra una serie di problemi quali:

- Ricezione e trasmissione fisica
- Controllo degli errori
- Controllo di flusso
- Conversione dei dati
- Crittografia e sicurezza
- Sincronizzazione

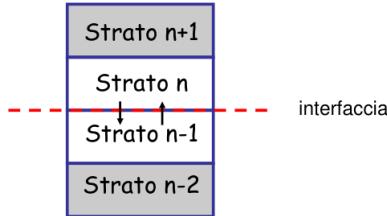
L'approccio logico di analizzare singolarmente i vari problemi ha condotto al **modello a strati**

## 1.1 Definizione

La suddivisione delle funzionalità secondo questo tipo di modello agevola la gestione della complessità, dato che ciascuno stato

- È responsabile di un sottoinsieme definito e limitato di compiti
- Interagisce solo con gli strati immediatamente superiori o inferiori
- Fa affidamento su servizi forniti dallo strato immediatamente inferiore
- fornisce servizi allo stato immediatamente superiore

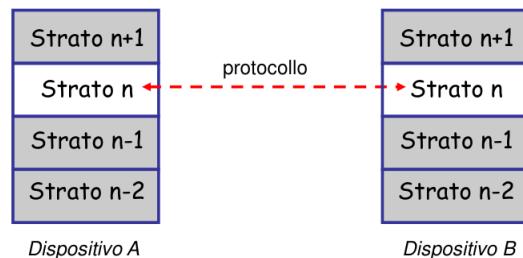
Ci sono diversi vantaggi in questo tipo di modello, ad esempio l'indipendenza degli strati e le funzionalità limitate di ogni strato che ne semplificano la realizzazione. Tuttavia bisogna tener a mente che un numero eccessivo di strati può portare ad inefficienze.



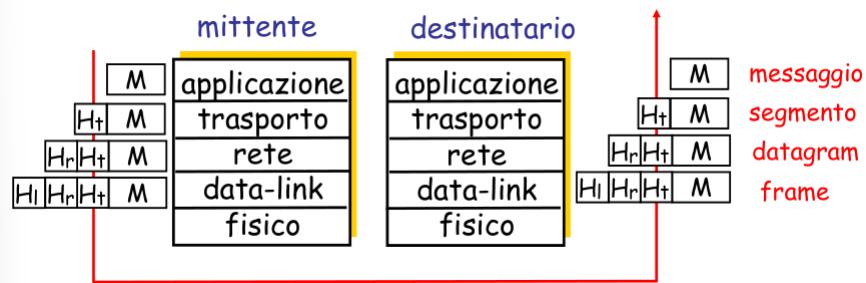
All'interno di ciascun dispositivo di rete, lo scambio di informazioni tra due strati adiacenti avviene attraverso un'**interfaccia**, che definisce i servizi offerti dallo strato inferiore allo stato superiore.

Lo strato n-esimo di un dispositivo comunica con lo strato n-esimo di un'altra entità secondo un **protocollo**

Per protocollo di comunicazione si intende un insieme di regole che permette la corretta istaurazione, mantenimento e terminazione di una comunicazione. Un protocollo definisce il formato e l'ordine dello scambio di messaggi tra le entità comunicanti



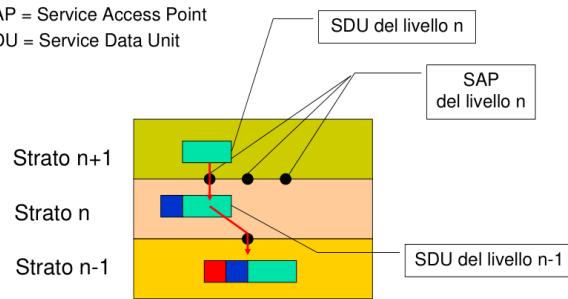
Ipotizziamo di voler scambiare un messaggio con un altro calcolatore nel modello a strati. Questo avviene nel seguente modo



Ogni strato del modello antepone una intestazione (header) al messaggio ricevuto dallo strato soprastante. L'insieme messaggio + header viene passato allo strato sottostante, a destinazione il messaggio risale la pila e in ricezione ogni strato rimuove l'header dello strato corrispettivo. Definiamo i componenti di questo sistema in modo rigoroso

- L'**SDU** (Service Data Unit) è l'unità di dati scambiata tra due livelli adiacenti in un sistema di rete (i dati grezzi che un livello deve elaborare)
- L'**SAP** (Service Access Point) è il punto di accesso al servizio che un livello fornisce al livello immediatamente superiore, permette al livello  $n + 1$  di interfacciarsi con il livello  $n$

SAP = Service Access Point  
SDU = Service Data Unit



In seguito al passaggio di un SDU attraverso un livello, il livello ricevente i dati, li incapsula in un **PDU** (Protocol Data Unit) contenente il dato grezzo, header e footer dei livelli.

Un livello della pila protocolare può essere costretto a frammentare il pacchetto ricevuto dallo strato superiore prima di passarlo allo strato inferiore. Il corrispondente livello del ricevente si occuperà di riassemblare il dato.

## 1.2 Il modello OSI

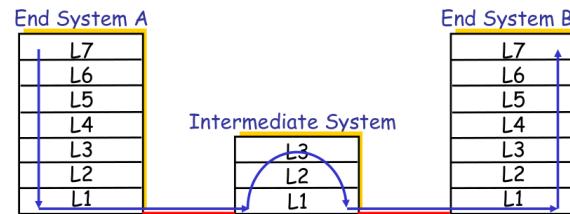
Negli anni 80 viene definito il modello **Open System Interconnection** (OSI). Si tratta di un modello a strati su 7 livelli.



OSI: Open Systems Interconnection

Oltre alla struttura del sistema mittente/ricevente descritta sopra, viene definita una struttura intermedia su 3 strati. i dispositivi di rete si differenziano per il numero di livelli fino a cui operano

- fino a L1 operano i **ripetitori**
- fino a L2 operano i **bridge/switch** di rete locale
- fino a L3 operano i **router**



Di seguito uno schema riassuntivo di cosa si occupa ciascun strato

- **L1 Fisico:** si occupa della trasmissione di sequenze binarie sul canale di comunicazione. Si specificano in questo livello le caratteristiche elettriche del segnale, le tecniche di codifica e decodifica, le tecniche dei mezzi trasmissivi e i tipi di connettori.
- **L2 Data Link:** ha come scopo la trasmissione affidabile di pacchetti di dati (frame). Accetta in input i frame e li trasmette sequenzialmente. Verifica la presenza di errori di trasmissione aggiungendo delle informazioni di controllo (**Frame control sequence**)
- **L3 Rete:** Si occupa dell'intradamento dei messaggi
  - Determina i sistemi intermedi da attraversare per giungere a destinazione
  - Gestisce le tabelle di intradamento per ottimizzare il traffico sulla rete
- **L4 Trasporto:** Fornisce servizi per il trasferimento dei dati da terminale a terminale (end-to-end)
  - Frammenta i pacchetti in modo che abbiano dimensioni idonee per il livello 3
  - rileva e corregge gli errori
  - controlla il flusso
  - gestisce le congestioni
- **L5 Sessione:** è responsabile dell'organizzazione del dialogo e della sincronizzazione tra due programmi applicativi e del conseguente scambio di dati. Si occupa cioè di stabilire la sessione.
- **L6 Presentazione:** il livello di presentazione gestisce la sintassi dell'informazione da trasferire (Es. ASCII o EBCDIC).
- **L7 Applicazione:** È il livello dei programmi applicativi, cioè tutti i programmi appartenenti al sistema operativo o scritti da utenti attraverso i quali l'utente finale utilizza la rete

## 2 L3: rete

Il compito del livello rete è quello di definire i percorsi dei pacchetti nel loro transito da host mittente a host destinazione. Nel L3, oltre agli end-system operano i **router**.

### 2.1 Packet switching

Le reti di calcolatori operano secondo il modello detto **packet switching**. In una rete a commutazione i pacchetti sono composti da **header**, e **payload**.

- L'header contiene informazioni di controllo, tra cui l'indirizzo di destinazione

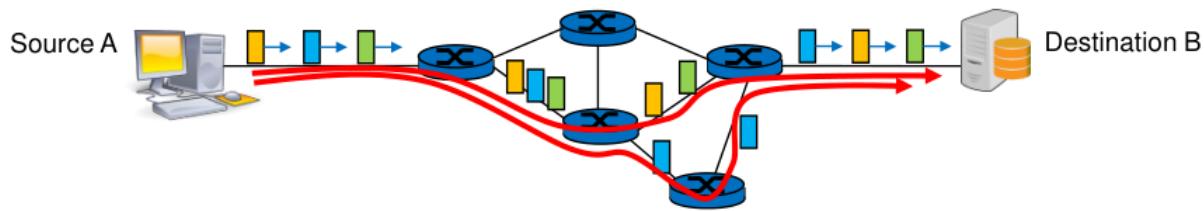
- Il payload contiene le informazioni del messaggio

I dispositivi intermedi che operano al livello rete funzionano in una modalità detta **store-and-forward**. Ogni pacchetto è ricevuto interamente, viene controllata l'assenza di errori e poi ritrasmesso in uscita. All'interno dei dispositivi intermedi, i pacchetti vengono conservati in un buffer di memoria gestiti come delle code.

### 2.1.1 Datagram

In una rete a commutazione di pacchetto basata su **modello datagram**, ciascun pacchetto è inoltrato verso la sua destinazione indipendentemente dagli altri.

- Ogni volta che un pacchetto arriva ad un router, quest'ultimo inoltra il pacchetto verso un successivo dispositivo intermedio, o il destinatario finale se disponibile.
- Pacchetti inviati da *A* a *B* in momenti successivi possono seguire percorsi differenti nella rete ed arrivare alla destinazione in ordine diverso dall'ordine di trasmissione



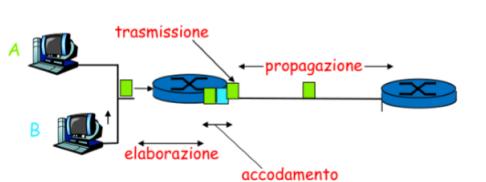
### 2.1.2 QoS

Lo scopo di una rete a commutazione di pacchetto consiste nel recapitare pacchetti da un qualunque mittente ad un qualunque destinatario. I metodi per analizzare le prestazioni e la qualità del servizio di una rete sono diversi, alcuni dei più utilizzati sono

- **End-to-end Delay**: ritardo nella consegna dei pacchetti
- **Packet delay variation PDV**: variazione temporale del ritardo one-way
- **Throughput**: quantità di bit al secondo che la rete è in grado di trasferire tra i due terminali [b/s]
- **Loss-Rate**: probabilità che un pacchetto non venga consegnato a destinazione

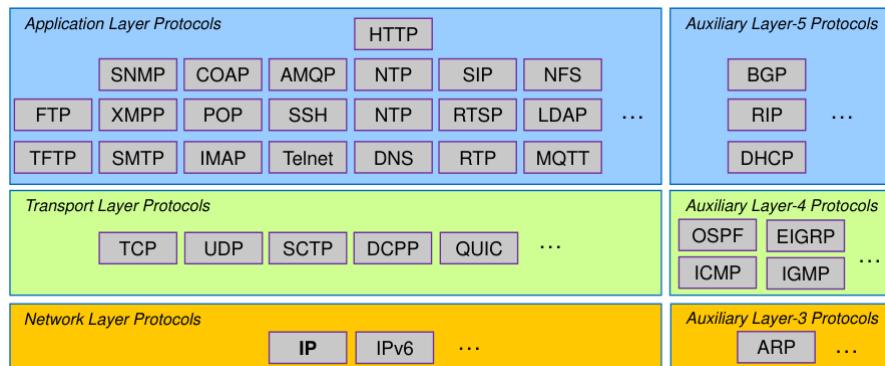
I ritardi di un pacchetto alla destinazione è determinato da vari fattori:

- Tempo di elaborazione nel nodo
- Tempo di trasmissione su ciascun link (lunghezza in bit/velocità in bps)
- Tempo di attesa nelle code dei router
- Tempo di propagazione sulle linee (lunghezza linea/velocità in bps)



## 2.2 Protocolli

I protocolli definiti per la comunicazione variano a seconda del layer in cui operano, uno schema riassuntivo dell'attuale suite di protocolli è il seguente

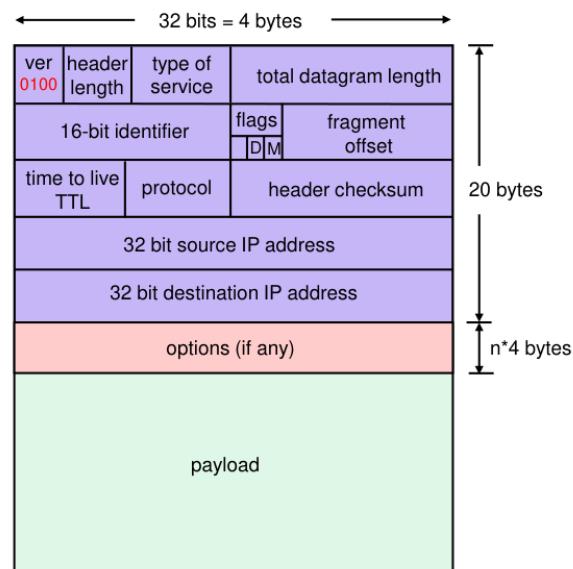


Concentrandoci sui protocolli di rete, il principale è il protocollo **IP**. Attualmente si utilizza ancora la versione 4 del protocollo IP anche se la 6 sta venendo gradualmente introdotta e utilizzata. Il protocollo IP fornisce un servizio di consegna elementare e senza garanzie (best effort) di pacchetti, la sua semplicità lo rende estremamente adattabile.

### 2.2.1 It.png Pv4

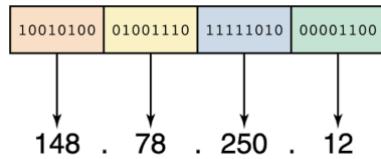
Il protocollo IPv4, realizza un servizio di consegna best-effort di pacchetti singoli (datagram). Opera nello stack TCP/IP che comprende i protocolli di trasporto UDP e TCP. Vediamo di cosa si occupa nello specifico.

- IP gestisce indirizzamento, frammentazione, ri-assemblaggio e multiplexing dei datagram
- È implementato negli end terminal e nei router e si occupa dell'instradamento dei pacchetti.
- Un datagram IPv4 può avere una dimensione massima di 65535 byte ( $2^{16} - 1$ )
- È composto da un header e un payload, l'header è costituito da una struttura fissa (20 byte) ed una opzionale e il payload è il dato creato dal protocollo di trasporto TCP o UDP



Il protocollo IP necessita di un indirizzo da cui trasmettere e un indirizzo a cui recapitare i datagram. Questi indirizzi si dicono **indirizzi IP**.

- Permettono di identificare univocamente un'interfaccia di rete di un dispositivo
- Sono composti da 32 bit e possono essere rappresentati per la lettura umana in forma **dotted decimal**
- La notazione dotted decimal associa ad ogni 8 bit il valore intero corrispondente e separa tramite punti i valori



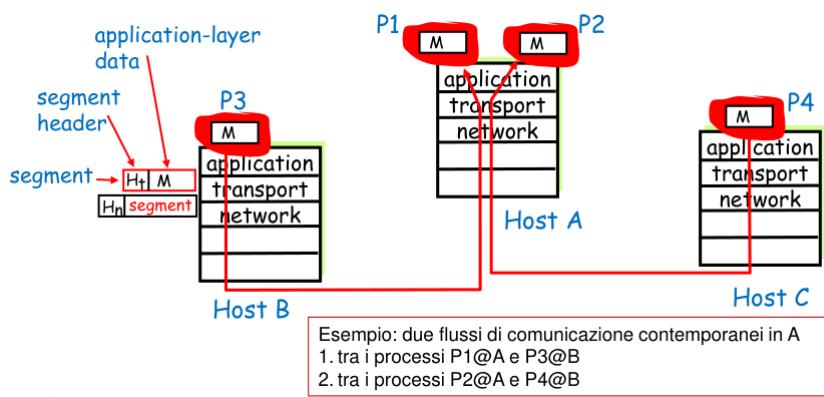
L'assegnazione degli indirizzi IP viene gestita dal IANA - Internet Assigned Numbers Authority

## 3 L4: Trasporto

In una rete di computer, il compito del livello trasporto è quello di consentire alle applicazioni eseguite in un end system lo scambio di informazioni attraverso il servizio offerto dal livello rete. Il livello trasporto agisce unicamente negli end-systems.

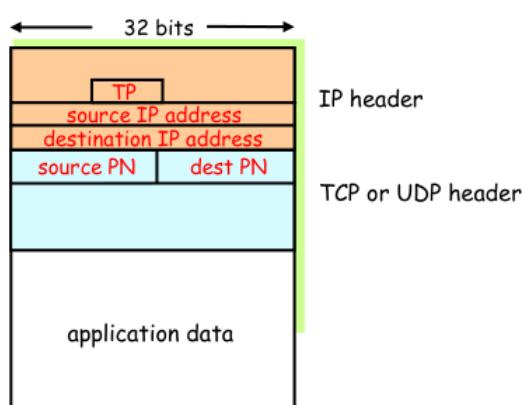
### 3.1 Multiplexing-Demultiplexing

In parole semplici, il livello trasporto si occupa di consegnare i pacchetti provenienti dalla rete ai vari processi di un dispositivo. La funzione di smistamento (**demultiplexing**) è esplicata al livello trasporto attraverso un'informazione ausiliaria contenuta nell'header: i **port number**.



Un pacchetto in arrivo ad un host, è associato ad uno specifico flusso di informazione, mediante la quintupl

- **transport\_protocol**: un numero naturale (8 bit) che identifica il protocollo di trasporto
- **source\_ip\_address** e **destination\_ip\_address**: indirizzo ip (32 bit) del mittente e destinatario
- **source\_port\_number** e **destination\_port\_number**: numeri naturali (16 bit) utili all'identificazione del flusso di informazione



### 3.1.1 UDP

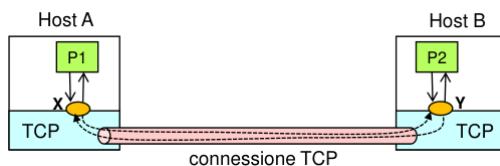
Il protocollo UDP offre alle applicazioni un servizio che non assicura l'ordine o l'arrivo dei pacchetti

- Forma un datagram IP per ciascun messaggio applicativo ricevuto dall'applicazione
- Non è garantita né la consegna né il rispetto dell'ordine di sequenza

In pratica offre solo un servizio di multiplexing/demultiplexing.

### 3.1.2 TCP

Il protocollo TCP offre alle applicazioni un servizio senza perdite e con rispetto dell'ordine. Il servizio è di tipo **connection oriented** il che implica che prima di iniziare la trasmissione è necessario stabilire una connessione tra i due endpoint attraverso una procedura di **connection establishment**



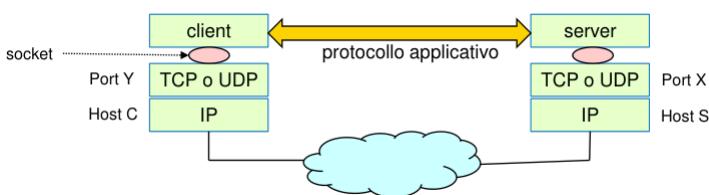
Una connessione TCP tra due endpoint consente un flusso di dati bidirezionale

## 3.2 Client-Server

Il modello a cui si ispira la maggior parte delle applicazioni distribuite è il modello client-server.

- Un processo **Server** esegue su un host S, e si rende disponibile alla comunicazione attraverso il protocollo UDP/TCP ad una porta X nota al client
- Un processo **Client** esegue su un host C, chiede al sistema operativo di rendere disponibile una porta Y e stabilire una connessione tramite TCP/UDP con il processo server S e la porta X

Un protocollo applicativo definisce il formato dei messaggi e i pattern di interazione.



La suite TCP/IP mette a disposizione delle applicazioni due servizi (TCP/UDP) di livello trasporto, e la definizione di protocolli applicativi standard consente di svincolare i servizi da una specifica implementazione

## 4 HTTP (applicativo)

### 4.1 Funzionamento

Il protocollo HTTP si basa su TCP.

- Il client apre un socket verso la porta TCP 80 del server
- Il server accetta la connessione
- Il client manda una richiesta per uno specifico oggetto tramite URL
- Il server risponde e chiude la connessione

Il protocollo HTTP è **stateless**, ne il server ne il client mantengono a livello HTTP informazioni riguardo ai messaggi scambiati in precedenza

#### 4.1.1 URL

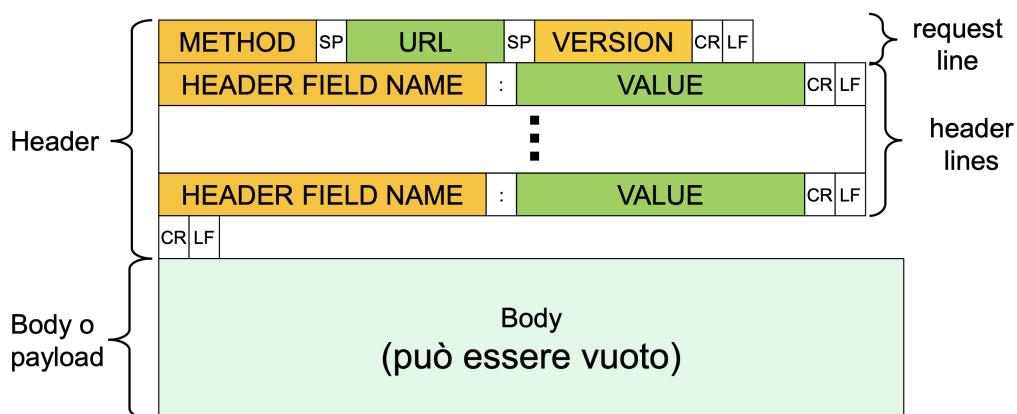
Un URL ha la seguente sintassi

`http://host[:port]/path[#fragment][?query]`

- host: identifica il server, e può essere un indirizzo IP in dotted decimal o un nome assegnato al server
- port: identifica la porta di ascolto del server, è opzionale e di solito la porta 80
- path: identifica la risorsa sul server (/images/cat.png)
- fragment: identifica una sezione della risorsa (paragrafo1)
- query: identifica i parametri passati dal client al server

## 4.2 Richiesta

Il protocollo HTTP si basa sullo scambio di una sequenza di byte. Ogni byte identifica un carattere secondo la tabella ASCII, o una risorsa in formato binario (es: .jpg, .png, .gif). Lo schema sottostante sintetizza una richiesta HTTP



CR = Carriage Return =  $0D_{16} = 13_{10}$   
 LF = Line Feed =  $0A_{16} = 10_{10}$

Le richieste HTTP sono di diverso tipo a seconda del campo method nella request line:

- GET
- HEAD
- OPTIONS
- PUT
- DELETE
- TRACE
- POST

Inoltre tutti i metodi HTTP possono essere:

- **Sicuri:** Non apportano modifiche allo stato del server (GET, HEAD)
- **Idempotenti:** L'effetto sul server di più richieste identiche è lo stesso di quello di una sola richiesta (tutti i metodi sono idempotenti tranne POST)

#### 4.2.1 GET

Il metodo GET è usato per richiedere una risorsa identificata da un URL. In seguito alla richiesta da parte del client, se la risorsa è disponibile il server la invia all'interno del body. GET è un metodo sia sicuro che idempotente, e può essere

- Assoluto: la risorsa viene richiesta senza ulteriori specificazioni
- Condizionale: la risorsa viene richiesta se soddisfatto un dato criterio
- Parziale: si richiede una sottoparte di una risorsa memorizzata

#### 4.2.2 HEAD

Il metodo HEAD è simile a GET e permette di verificare una risorsa nel server tramite URL. La differenza con il metodo GET, è che la risorsa non viene inviata nel messaggio di risposta.

#### 4.2.3 POST

Il metodo POST è usato per inviare dati dal client al server. Si tratta di un metodo non idempotente e non sicuro. Ci sono 3 tipi di risposte che il server può restituire:

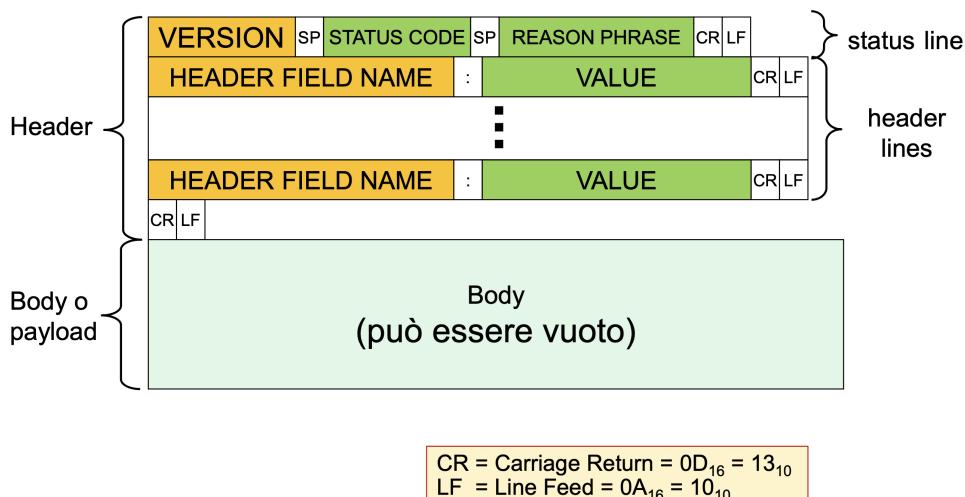
- 200 OK: dati ricevuti e sottomessi alla risorsa specificata
- 201 CREATED: dati ricevuti e risorsa creata
- 204 NO CONTENT: dati ricevuti e sottomessi alla risorsa specificata, non è stata data risposta

#### 4.2.4 PUT

Il metodo PUT, serve per trasmettere informazioni dal server al client, modificando o creando la risorsa se non esiste. PUT è un metodo non sicuro ma idempotente

### 4.3 Risposta

Lo schema sottostante riassume la costruzione di una risposta HTTP



### 4.3.1 Status Code

Lo status code è un numero di 3 cifre di cui la prima indica la classe della risposta, e le altre due la risposta specifica:

- **1xx: Informational:** Una risposta temporanea alla richiesta, durante il suo svolgimento
- **2xx: Successful:** Il server ha ricevuto ed accettato la richiesta
- **3xx: Redirection:** Il server ha ricevuto ed accettato la richiesta ma sono necessarie ulteriori azioni da parte del client
- **4xx: Client error:** La richiesta non può essere soddisfatta per un errore da parte del client
- **5xx: Server error:** Il server non è in grado di accettare la richiesta

## 4.4 Header

### 4.4.1 Header generali

Gli header generali si applicano sia alle richieste che alle risposte e sono

- Date: Data della richiesta
- MIME-Version: Versione MIME
- Transfer-Encoding: Codifica della trasmissione
- Cache-control: Tipo di meccanismo di Caching richiesto o suggerito per la risorsa
- Connection: Il tipo di connessione da usare
- Via: usato da proxy e gateway

### 4.4.2 Header risposta

Gli header della risposta sono posti dal server per specificare informazioni sulla risposta e su se stesso al client

- Server: una stringa che descrive il server
- Accept-range: che tipo di range può accettare

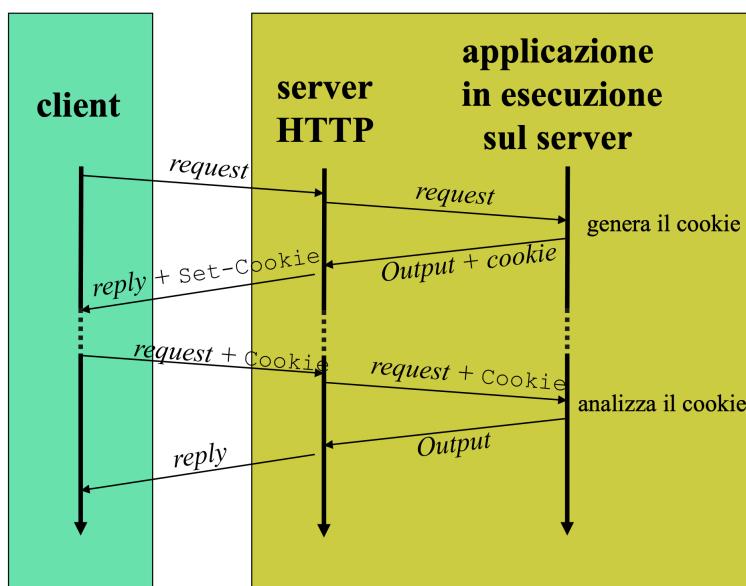
#### 4.4.3 Header identità

Gli header di identità danno informazioni sul contenuto del body o della risorsa inviata:

- Content-Type: Oggetto/Formato
- Content-Length: Lunghezza in byte del body
- Content Base - Encoding - Language - Location - MD5 - Range: URL di base, codifica, linguaggio, URL della risorsa, valore di digest MD5, range richiesto dalla risorsa
- Expires: Una data dopo la quale la risorsa non è più considerata valida
- Last-Modified: data dell'ultima modifica

## 4.5 Cookies

Il protocollo HTTP è stateless, per mantenere brevi informazioni scambiate tra server e client, si utilizzano i **cookie**. Tramite cookie il client mantiene lo stato di precedenti connessioni e lo manda al server di pertinenza ogni volta che richiede una risorsa.



I cookie definiscono dunque due nuovi header possibili:

- **Set-Cookie** (Header della risposta): permette al client di memorizzare cookie e rispedirli alla richiesta successiva
- **Cookie** (Header della richiesta): il client decide se spedirlo sulla base del documento, età del cookie e l'indirizzo del server

Oltre a un nome e a un valore, un cookie contiene dei campi che stabiliscono la durata o lo "scope"

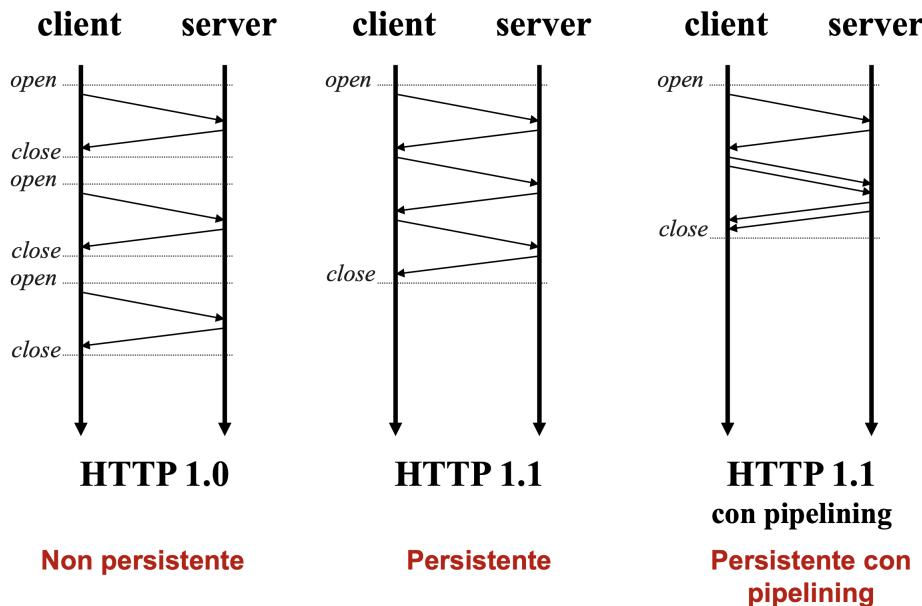
- **expires**: Data entro la quale il browser deve eliminare i cookies
- **max-age**: Tempo in secondi entro il quale il browser deve eliminare i cookie

Un cookie per il quale non è specificata la durata, è detto session-cookie e viene cancellato alla chiusura del browser. I cookie contengono attributi che permettono di ricarne l'appartenenza, quale **domain** (il dominio DNS a cui si applica il cookie) e **path** (URL corrispondente).

## 4.6 Connessioni

Distinguiamo in 3 tipi le connessini HTTP tra client e server:

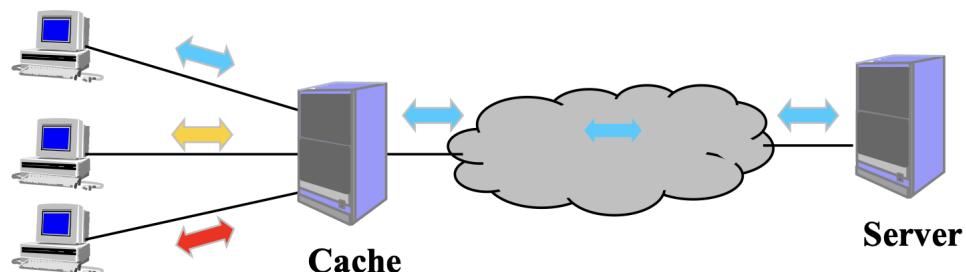
- **Non Persistente:** Ogni richiesta/risposta apre e chiude una connessione separata. Elevata latenza e overhead di connessione.
- **Persistente:** La connessione rimane aperta per più richieste/risposte. Riduce latenza e overhead, migliorando le prestazioni.
- **Persistente con Pipelining:** Permette l'invio di più richieste consecutive senza attendere le risposte. Incrementa ulteriormente l'efficienza, ma richiede un server compatibile.



Una misura del tempo di comunicazione tra client e server è data dal RTT (**round trip time**): è il tempo totale impiegato da un pacchetto di dati per viaggiare da un mittente a un destinatario e tornare indietro al mittente.

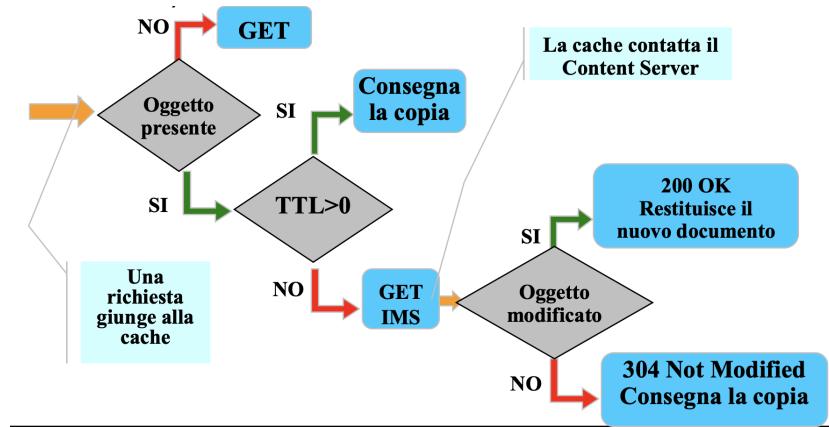
## 4.7 Web Caching

Si parla genericamente di Web caching quando le richieste di un determinato client non raggiungono il Web Server, ma vengono intercettate da una cache. Quest'ultima fornisce con velocità maggiore le risorse necessarie se le ha memorizzate, altrimenti le richiede al server. Ci sono due tipi di interazione HTTP per richiesta, **client-cache** e **cache-server**



Un funzionamento simile è quello dei server proxy, che collegano più client a più server. Ma cosa succede se il server aggiorna una sua risorsa? In che modo la cache previene il problema di coerenza con il client? HTTP fornisce due meccanismi:

- **TTL (Time to live)**: il server fornisce una scadenza per l'oggetto richiesto
- **GET-condizionale**: il client può fare una richiesta al server tramite get-condizionale



## 4.8 Browser Web

Un browser web è costituito da due componenti principali:

- **UI**: l'interfaccia con cui l'utente interagisce
- **Browser engine**: composto a sua volta da
  - **layout engine**: interpreta html e css
  - **js engine**: interpreta javascript

Il ruolo di client HTTP può essere assunto da diversi programmi, non esclusivamente il browser, ad esempio programmi attivabili da command line come **wget**

## 5 DNS (applicativo)

Il DNS **domain name system** si occupa di assegnare a un indirizzo IP in notazione dotted decimal, un nome di facile comprensione. Il servizio si basa su UDP sulla porta 53. Altri dei benefici offerti da un server DNS sono:

- **Alias degli hostname**: Possibilità di assegnare ulteriori alias a host
- **Alias server di posta**: Possibilità di assegnare un alias a un dominio di posta elettronica
- **load balancing**: Possibilità di indirizzare i client ad un server che replica i contenuti del server originale per distribuire il traffico

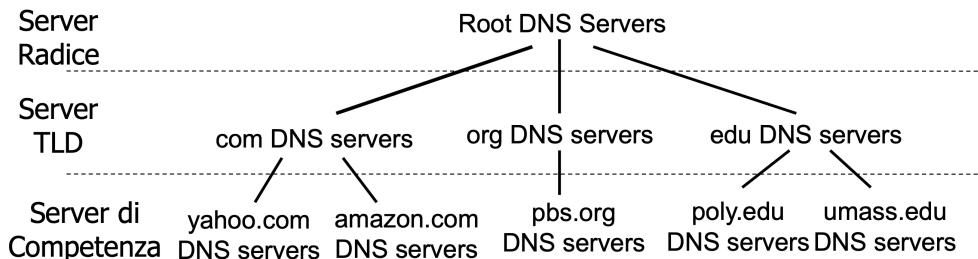
### 5.1 Soluzioni

Il DNS può essere implementato in due modi, di cui una sicuramente più sconveniente dell'altra:

- **DNS centrale**: un unico server mondiale che si occupa di fornire il servizio di dns, le problematiche sarebbero tuttavia
  - single point failure
  - manutenzione
  - volume di traffico

- distanza dal database
- **DNS distribuito:** si distribuiscono le informazioni tra più entità server, e si gestiscono in modo gerarchico, i server più alti in gerarchia hanno più informazioni ma meno dettagliate, quelli più in basso meno informazioni ma più dettagliate

Lo schema sottostante riporta il funzionamento di un servizio di dns distribuito



Un esempio di richiesta client per <https://amazon.com> segue questa route

- Il client richiede al root dns la lista IP dei domini .com del server TLD
- Il TLD di competenza restituisce l'IP del server autorizzato per amazon.
- Il client contatta il server autorizzato per amazon.com che gli restituisce l'indirizzo IP di amazon

## 5.2 Attori

Nel servizio di DNS si alternano 4 attori principali

- **Resolver:** È il client da cui parte la richiesta di risoluzione al sistema DNS
- **Registry:** È il titolare della risoluzione di un namespace, l'organizzazione che si occupa di fare modifiche al database dei nomi di un determinato dominio.
- **Registrar:** È l'agente che sottomette al registry le richieste di modifica di risoluzione per conto del registrant
- **Registrant:** È l'entità che possiede l'uso di un determinato dominio

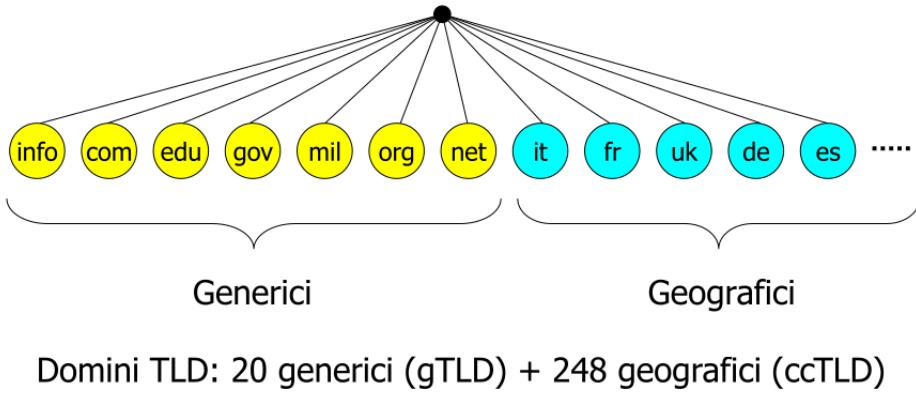
## 5.3 Tipologie di DNS

### 5.3.1 Root Name Server

I **Root Name Server** sono 13 server logici in internet i cui indirizzi IP sono noti. Ad essi si riferiscono i Local Name Server che non possono soddisfare immediatamente una richiesta di risoluzione. Il Local Name Server, si comporta come un client DNS e invia una richiesta di risoluzione al Root Name Server

### 5.3.2 Top Level Domain

I **Top level domain (TLD)** si occupano dei domini di alto livello (generici e geografici)



### 5.3.3 Authoritative

Gli **Authoritative Name Server** sono server capaci di risolvere tutti i nomi all'interno di un determinato dominio. Un server dei nomi assoluto per il dominio unina.it, deve essere in grado di risolvere tutti i nomi del tipo xyz.unina.it

- Ad esso si riferiscono i Name Server TLD quando devono risolvere un indirizzo del dominio

### 5.3.4 Local Name Server

Ciascun operatore di rete ne installa uno nella propria rete, gli host di una rete sono configurati con indirizzo del DNSserver.

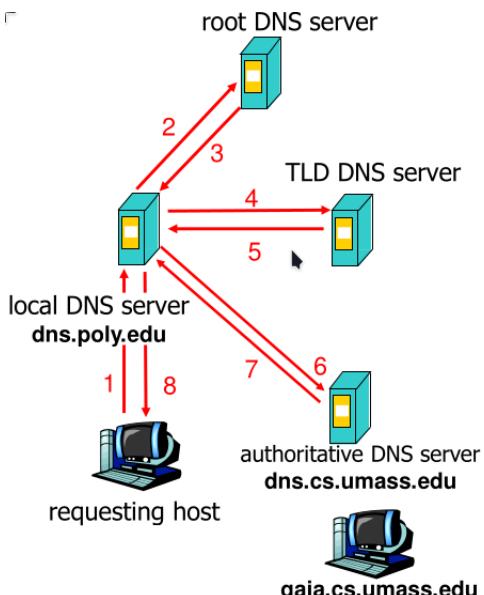
- Funziona similmente a un proxy, ed invia la query alla gerarchia di server DNS restituendo le risposte finali

## 5.4 Risoluzioni DNS

Ci sono due metodi principali con i quali una risoluzione da parte del DNS può avvenire

### 5.4.1 Query Iterative

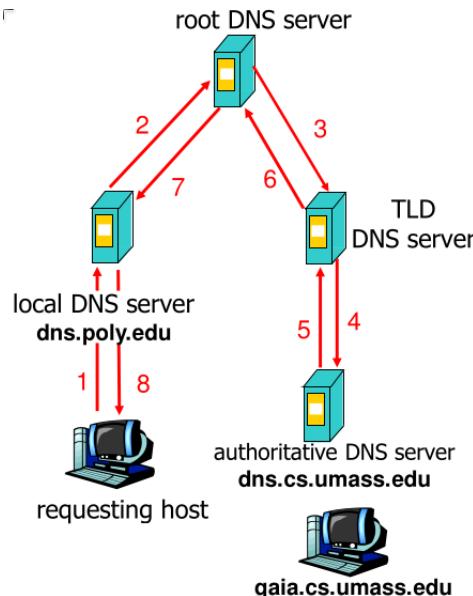
Il server DNS locale, quando riceve una richiesta ripsonde con le informazioni a sua disposizione o indica al richiedente un altro server DNS a cui rivolgersi



Il vantaggio principale di una query di questo tipo è la distribuzione del carico tra i server DNS, tuttavia c'è una maggior complessità per il client che deve eseguire i vari riferimenti

### 5.4.2 Query Recursive

Il server DNS locale si occupa dell'intera risoluzione del nome, contattando tutti i server necessari per ottenere la risposta completa



Il pro di questo tipo di query è il vantaggio per il client, che riceve direttamente la risposta completa, tuttavia c'è un carico maggiore sui DNS locali e superiori che devono gestire l'intero processo

## 5.5 Caching

Per esigenze di efficienza, sia il client DNS presente negli end-system che i server DNS memorizzano localmente un certo numero di corrispondenze, questa operazione è nota come **caching**. Per evitare che informazioni non aggiornate restino nella rete, dopo un certo tempo ( 1 giorno), le associazioni vengono eliminate

### 5.5.1 Resource Records

Per **Resource Records** si intende il formato di memorizzazione delle informazioni in cache

Formato RR: (Nome, Valore, Tipo, TTL)

- **TTL**: tempo di vita residuo
- **Tipo**: Il tipo determina il significato di Nome e Valore
  - Tipo = A
    - \* **Nome**: hostname
    - \* **Valore**: indirizzo IPv4
  - Tipo = AAAA
    - \* **Nome**: hostname
    - \* **Valore**: indirizzo IPv6

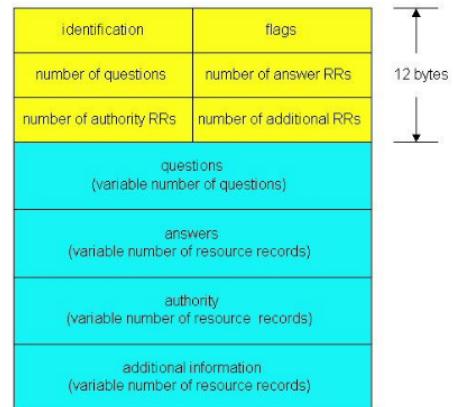
- Tipo = NS
  - \* **Nome:** dominio (es. unina.it)
  - \* **Valore:** IP dell'authoritative NS
- Tipo = CNAME
  - \* **Nome:** alias per il nome canonico
  - \* **Valore:** nome canonico
- Tipo = MX
  - \* **Nome:** dominio di posta
  - \* **Valore:** nome dell'host mailserver

## 5.6 Protocollo DNS

Vediamo ora il formato in cui si comunica con un DNS. Partiamo dicendo che richieste e risposte condividono lo stesso formato di messaggio

l'header del messaggio, è composto da 12 bytes e si divide in **identification**, che è un numero di 2 byte per ogni richiesta, le risposte utilizzano lo stesso identificativo e **flags**, parametri riguardanti il tipo di richiesta, quali:

- Risposta a richiesta
- Ricorsione desiderata
- Ricorsione disponibile
- Risposta authoritative



Il body del messaggio invece è composto da:

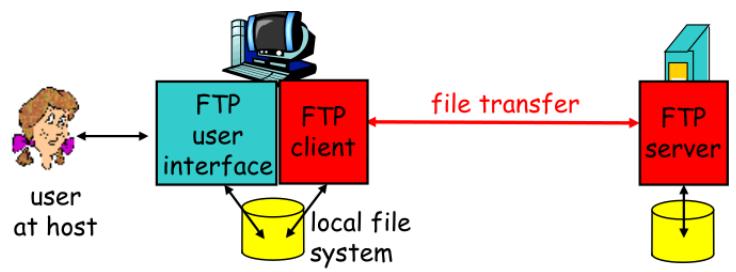
- **Nome e tipo** per una richiesta
- **Resource Record** in risposta ad una richiesta
- **Resource Records** per server authoritative
- **Informazioni aggiuntive**

## 6 FTP (applicativo/trasporto)

Il protocollo FTP (**file transfer protocol**) permette il trasferimento di uno o più file qualsiasi tra due macchine. Lavora utilizzando due connessioni, una dati e una di controllo (out of band)

### 6.1 Funzionamento

FTP si basa sul modello client/server, in cui un client FTP trasferisce file da o verso una macchina remota (server FTP)



Il client FTP contatta il server FTP alla porta 21, e vengono aperte due connessioni parallele

- **Dati:** file che fluiscono dal client al server o viceversa
- **Controllo:** scambio di comandi, messaggi di risposta tra il client e il server

Il server FTP mantiene uno stato, ad esempio la directory corrente, i dati dell'autenticazione etc.

### 6.1.1 Controllo

Nella connessione di controllo, vengono inviati comandi e ricevute risposte come testo ASCII. Esempi di comandi sono:

- **USER** username
- **PASS** password
- **LIST**: restituisce i files presenti nella directory
- **GET** filename

Il tipo di paradigma client/server presuppone che il server venga configurato per accettare connessioni FTP, mentre i client FTP sono invece disponibili pressoché su tutti i sistemi operativi

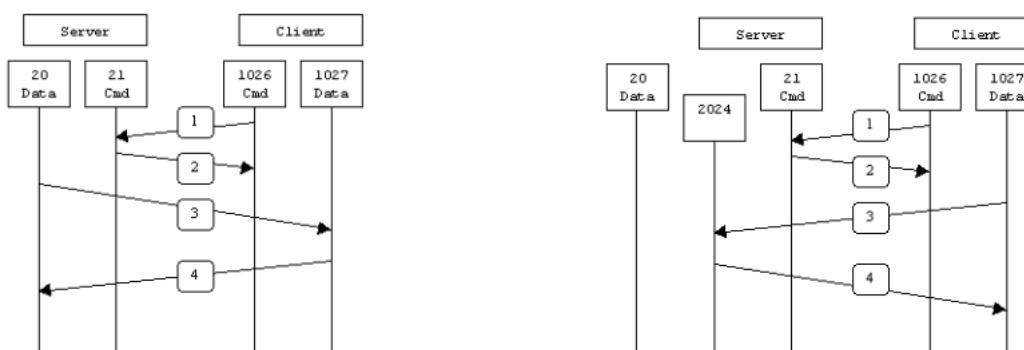
## 6.2 FTP Attivo/Passivo

L'interazione tra client e server FTP può essere di 2 tipi:

- **Modalità attiva:** Il client informa il server su quale porta può ricevere i dati, il server si connette al client per trasferire i dati
- **Modalità passiva:** Il server informa il client su quale porta è in ascolto per il trasferimento dei dati, il client si connette per trasferire i dati

Active FTP

Passive FTP



## 7 SMTP (applicativo)

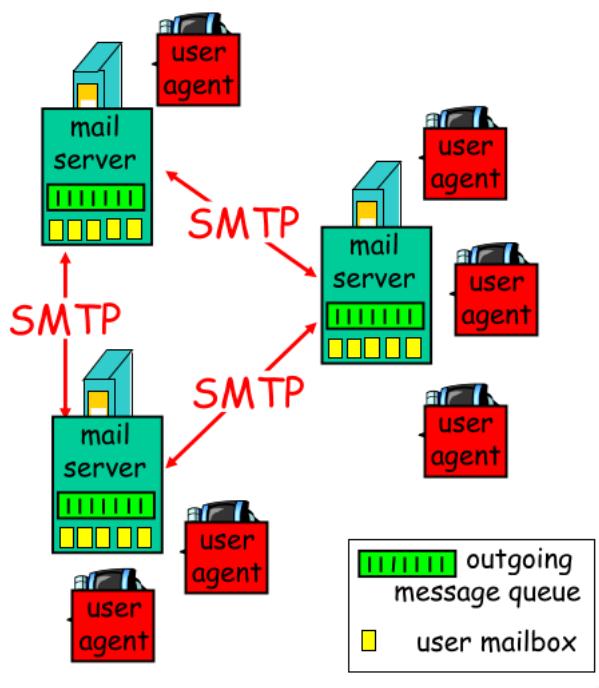
L'SMTP (**S**imple **M**ail **T**ransfer **P**rotocol), viene utilizzato per inviare email da un client a un server di posta elettronica.

### 7.1 Funzionamento

Iniziamo definendo le 3 entità principali in gioco:

- **user agent**: il mail reader, ossia il servizio che permette l'invio di email verso un mail server (es. outlook).
- **mail server**: il mailserver, è il server che fa da intermediario tra l'invio e la ricezione di mail. Contiene una mailbox contenente i messaggi in entrata per l'utente, e una coda dei messaggi in uscita. Il mail server si comporta sia da client che da server a seconda del ruolo che deve ricoprire nello scambio del messaggio
- **protocollo SMTP**: il protocollo definito per la comunicazione

Lo schema sottostante riassume le interazioni tra i vari agenti



SMTP usa il protocollo TCP per consegnare in modo affidabile i messaggi dal client al server.  
Ci sono 3 fasi durante il trasferimento via SMTP

- **handshaking**
- **trasferimento del messaggio**
- **chiusura del messaggio**

### 7.2 Protocollo FTP

Il formato del messaggio SMTP segue la distinzione header, body.

- **header**: linee di intestazione (to:, from:, subject:)

- **body**: caratteri ASCII

### 7.2.1 MIME

MIME (Multipurpose Internet Mail Extensions) è uno standard utilizzato per estendere le funzionalità dei protocolli di posta elettronica come SMTP. Esso permette l'invio di contenuti non testuali, come immagini, video, audio e file allegati, attraverso email.

L'obiettivo principale di MIME è superare i limiti dei protocolli originari, che gestivano solo messaggi di testo semplice in formato ASCII. Grazie a MIME, i messaggi di posta elettronica possono includere contenuti multimediali e strutturati, garantendo una maggiore flessibilità.

MIME aggiunge al messaggio email:

- un'intestazione aggiuntiva (**Content-Type**) per specificare il tipo di contenuto;
- una codifica dei dati (**Content-Transfer-Encoding**) per adattarli al formato richiesto;
- il supporto per la suddivisione di messaggi in più parti (**multipart**).

Esempi di dati che MIME può gestire includono:

- Testi formattati in HTML (**text/html**);
- Immagini, come JPEG (**image/jpeg**) e PNG (**image/png**);
- File audio, come MP3 (**audio/mpeg**);
- Video, come MP4 (**video/mp4**);
- File di testo generico o binario (**text/plain**, **application/octet-stream**);
- Documenti, come PDF (**application/pdf**) o file Microsoft Word (**application/msword**).

Grazie a MIME, i client di posta moderna sono in grado di gestire una varietà di formati di dati, rendendo l'email uno strumento molto versatile per la comunicazione digitale.

## 7.3 POP3

Il protocollo POP3 (Post Office Protocol version 3) è un protocollo di rete utilizzato per recuperare email da un server di posta elettronica. POP3 consente a un client di scaricare i messaggi di posta sul proprio dispositivo locale, rendendoli disponibili offline. Il funzionamento di POP3 si basa su una sequenza di comandi e risposte tra il client di posta e il server. Tipicamente, il protocollo opera attraverso la porta 110 (non sicura) o la porta 995 (POP3S, con crittografia SSL/TLS).

### 7.3.1 Caratteristiche principali di POP3

- I messaggi vengono scaricati dal server al dispositivo locale e, di norma, eliminati dal server dopo il download (salvo configurazioni alternative).
- È progettato per dispositivi con spazio di archiviazione locale e connessioni di rete intermitten.
- Offre un'implementazione semplice rispetto ad altri protocolli come IMAP.

### 7.3.2 Esempi di utilizzo

POP3 è particolarmente adatto nei seguenti scenari:

- Un utente desidera archiviare le email direttamente sul proprio dispositivo, senza mantenerle sul server.
- Ambienti con connessioni internet non persistenti, dove l'accesso offline alle email è prioritario.

### 7.3.3 Limiti di POP3

Nonostante la sua semplicità, POP3 presenta alcune limitazioni:

- Non consente una sincronizzazione efficace tra più dispositivi, poiché i messaggi scaricati non vengono mantenuti sul server (a meno di configurazioni specifiche).
- Non supporta l'organizzazione avanzata delle email direttamente sul server (ad esempio, cartelle o etichette).

### 7.3.4 Differenze rispetto ad altri protocolli

A differenza di IMAP (Internet Message Access Protocol), POP3 si concentra sul trasferimento delle email al dispositivo locale, mentre IMAP è progettato per sincronizzare le email su più dispositivi mantenendole sul server. Grazie alla sua semplicità, POP3 è ancora utilizzato in molte applicazioni, ma sta progressivamente lasciando spazio a protocolli più moderni, come IMAP, che offrono una maggiore flessibilità e funzionalità avanzate.

## 8 CDN (applicativo)

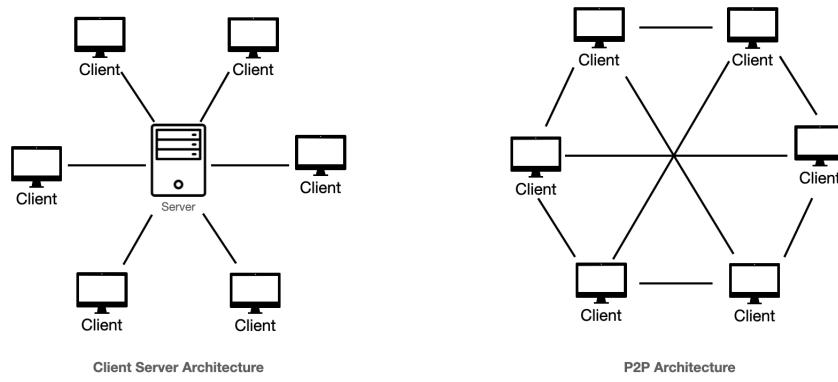
Una CDN (**content delivery network**), è una infrastruttura creata per distribuire efficacemente agli utenti di internet i contenuti dei siti web più popolari. Si basa sulla distribuzione di repliche dei contenuti dal server principale del "Content Provider" ad una molteplicità di server disposti sulla rete da un "content delivery operator". Gli obiettivi di una CDN sono

- Alleviare il server web "master" dal carico degli utenti
- Offrire i contenuti ai singoli utenti tramite server collocati in prossimità degli utenti
- Rendere il sistema di distribuzione dei contenuti più affidabile e robusto ai guasti

## 9 P2P

Il modello P2P (**Peer to Peer**) è un modello alternativo al client/server basato su i seguenti principi

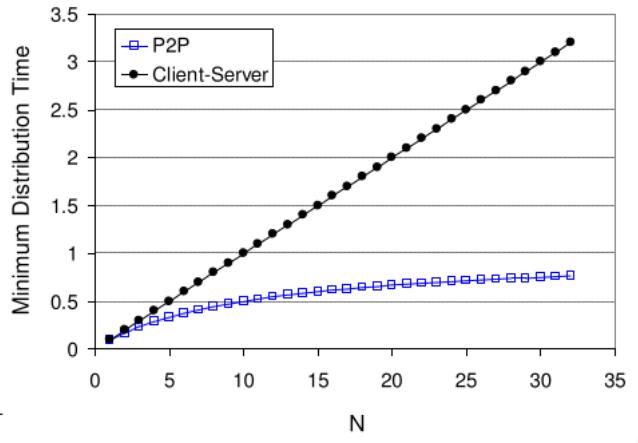
- Non esistono server sempre connessi
- Non esiste una differenziazione tra client e server
- End-system (peer) comunicano direttamente
- I peer sono connessi ad intermittenza e cambiano il proprio IP



La più grande differenza tra client/server e P2P riguarda le prestazioni. Mentre nel caso di un modello client server, il tempo di distribuzione è lineare al crescere dei client, nel metodo P2P, al crescere dei Peer la velocità di distribuzione decresce esponenzialmente

### Ipotesi:

- Tutti i peer hanno lo stesso rate di upload =  $u$
- $F/u = 1$  ora,  $u_s = 10u$ ,  $d_{min} \geq u_s$

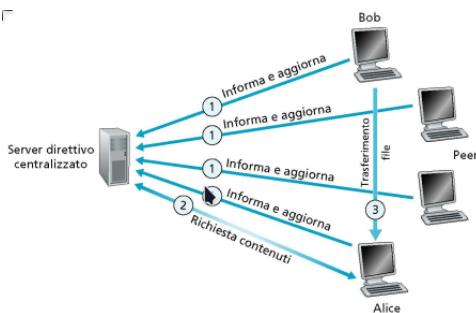


## 9.1 Directory centralizzata/decentralizzata

Le reti P2P si distinguono a loro volta tra reti a directory centralizzata e directory decentralizzata

### 9.1.1 Directory centralizzata

Si tratta di una struttura in cui c'è un server centrale che funge da directory o indice. Questo server tiene traccia dei file condivisi dai vari nodi della rete e delle loro posizioni. I peer comunicano con il server centrale per cercare i file desiderati



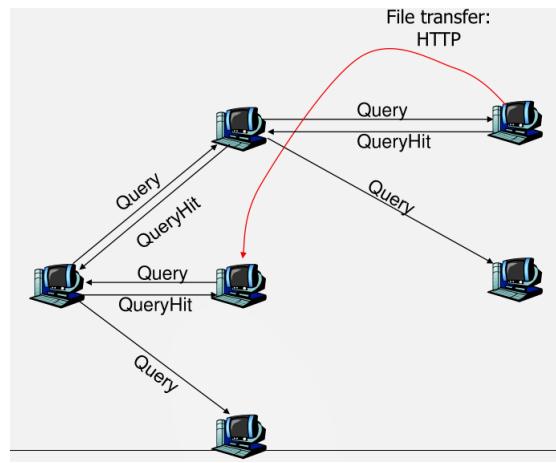
Il vantaggio è la semplicità della struttura, ma tuttavia ci sono problemi quali single point of failure, e scalabilità

### 9.1.2 Directory decentralizzata

Si tratta di un architettura completamente distribuita, realizzata attraverso un'overlay network fatta da connessioni TCP tra peer.

- La rete può essere composta da migliaia di peer, tuttavia ogni peer è connesso al massimo ad altri 10 peer nella overlay

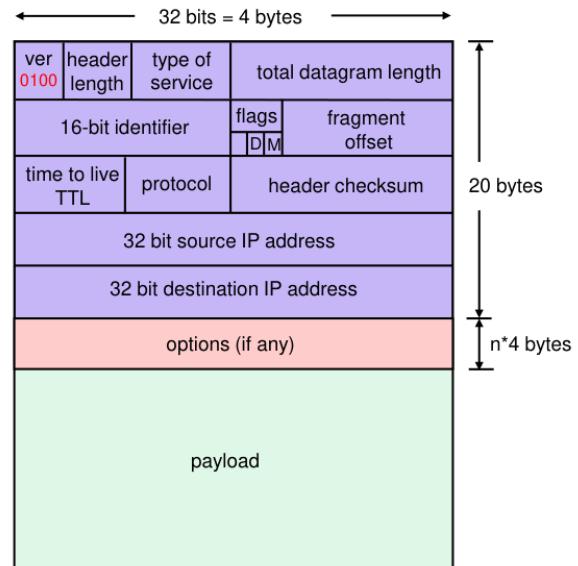
I problemi principali riguardano la costruzione e la gestione della rete di peer, e la localizzazione di un contenuto



## 10 IPv4 (rete)

Come accennato in precedenza, il protocollo IPv4 opera sul Layer 3 (rete). Vediamo di cosa si occupa nello specifico.

- IP gestisce indirizzamento, frammentazione, ri-assemblaggio e multiplexing dei datagram
- È implementato negli end terminal e nei router e si occupa dell'instradamento dei pacchetti.
- Un datagram IPv4 può avere una dimensione massima di  $65535$  byte ( $2^{16} - 1$ )
- È composto da un header e un payload, l'header è costituito da una struttura fissa (20 byte) ed una opzionale e il payload è il dato creato dal protocollo di trasporto TCP o UDP



Il protocollo IPv4 è un servizio **best effort**, il termine indica che la rete non discrimina un pacchetto rispetto ad altri, tuttavia non garantisce di prevenire:

- Pacchetti duplicati
- Consegnata ritardata o fuori ordine
- Corruzione di dati
- Perdita di pacchetti

Il ruolo di rendere la consegna affidabile, viene spostato sui meccanismi di controllo realizzati nei protocolli di livello superiore.

## 10.1 Header IPv4

In IPv4 l'header è costituito da una struttura fissa (20 byte) ed una opzionale di lunghezza multipla di 4 byte. I campi che compongono l'header sono

- **IP header length (4 bit)**: lunghezza dell'header in multipli di 32 bit (max 60 byte)
- **Type-of-service (8 bit)**: specifica il tipo di servizio che si richiede alla rete
- **Total length (16 bit)**: indica la lunghezza in byte del pacchetto (header + payload)
- **Protocol (8 bit)**: indica il protocollo di livello superiore associato al payload (6 TCP, 17 UDP)
- **Header checksum (16 bit)**: serve a verificare l'integrità dell'header IP
- **Source IP Address (32 bit)**: indirizzo IP del mittente
- **Destination IP Address (32 bit)**: indirizzo IP del destinatario
- **Identification (16 bit), Flags (3 bit), Fragment Offset (13 bit)**: sono usati in caso di frammentazione del pacchetto da parte di un router

### 10.1.1 Identification, Flags, Fragment offset

Questi campi servono a gestire la frammentazione dei pacchetti IPv4. Un pacchetto IPv4 può essere frammentato da un router in una sequenza di pacchetti che viaggiano singolarmente verso il destinatario, una volta arrivati a destinazione il livello IP del destinatario si occupa di riassemblarli prima di consegnarlo allo strato superiore.

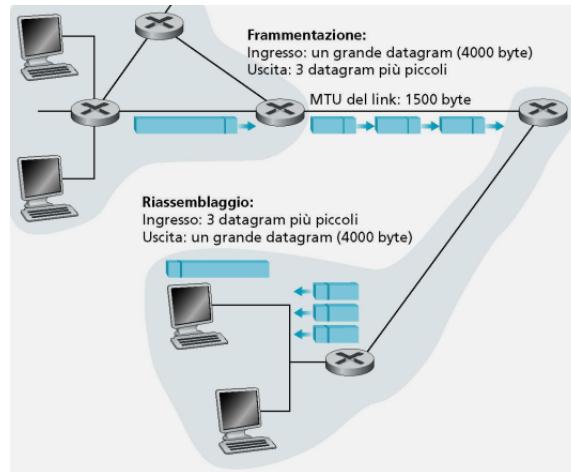
- Un pacchetto può essere frammentato anche più volte lungo il percorso
- La necessità di frammentare un pacchetto si presenta quando la dimensione del pacchetto supera la **MTU (Maximum Transmissible Unit)** sul link di uscita
- Il valore della MTU dipende dalla tecnologia usata al livello 2 (es. ethernet 1500 byte)

Nello specifico, ciascuno di questi campi dell'header si occupano di

- **Identification**: è un identificativo del datagram e serve ad associare diversi frammenti ad un unico pacchetto originario
- **Flags**: Il bit D (don't fragment) indica se il pacchetto può essere frammentato, mentre il bit M (more fragments) indica se il pacchetto è l'ultimo frammento
- **Fragment Offset**: identifica la posizione del frammento all'interno del pacchetto

### 10.1.2 Frammentazione e riassemblaggio

- Se un pacchetto di dimensione N arriva ad un router e deve essere trasmesso su un link di uscita con MTU M<N, il pacchetto è frammentato
- Ogni frammento è trasmesso come singolo pacchetto IP
- La dimensione del payload di ogni frammento è un multiplo di 8 byte
- Tutti i frammenti hanno lo stesso ID number



- N = 4000, MTU = 1500
- Tre frammenti, ciascuno con header 20 byte
- Frammento 1:
  - payload 1480
  - offset 0
- Frammento 2:
  - payload 1480
  - offset (1480/8)=185
- Frammento 3:
  - payload 1020
  - offset (1480+1480)/8=370

NOTA:

$$20 + 1480 + 1480 + 1020 = 4000$$

Note: Datagram size includes an IP header of 20 bytes.  
MTU and Datagram size must be greater than 30, and all values must be less than  $2^{16} - 1$  (65535).

Datagram Size:	4000	MTU:	1500	Datagram ID:	777
<ul style="list-style-type: none"> <li>↳ Fragments           <ul style="list-style-type: none"> <li>↳ Datagram 1               <ul style="list-style-type: none"> <li>1480 byte information field</li> <li>ID: 777</li> <li>Offset: 0</li> <li>Flag: 1 ← More fragments</li> </ul> </li> <li>↳ Datagram 2               <ul style="list-style-type: none"> <li>1480 byte information field</li> <li>ID: 777</li> <li>Offset: 185</li> <li>Flag: 1 ← More fragments</li> </ul> </li> <li>↳ Datagram 3               <ul style="list-style-type: none"> <li>1020 byte information field</li> <li>ID: 777</li> <li>Offset: 370</li> <li>Flag: 0 ← More fragments</li> </ul> </li> </ul> </li> </ul>					
Number of Datagrams: 3 <a href="#">Calculate</a>					

This applet was coded by Ryan Gilbert in 2008, a student at Arizona State University.  
It replaces an applet coded by Albert Huang in 1997 as part of course work at the University of Pennsylvania.

Ci sono tuttavia alcuni problemi relativi all'operazione di frammentazione: il compito di riassemblaggio è oneroso, il destinatario deve collezionare tutti i frammenti del pacchetto originario prima di consegnare il payload al livello superiore, se non li riceve entro un determinato tempo, i frammenti arrivati sono scartati. Un metodo per evitare la frammentazione dei pacchetti lungo il percorso, talvolta si effettua un **Path MTU Discovery**, cioè si determina la più piccola MTU lungo il percorso da un host A ad un host B.

### 10.1.3 Opzioni dell'header IPv4

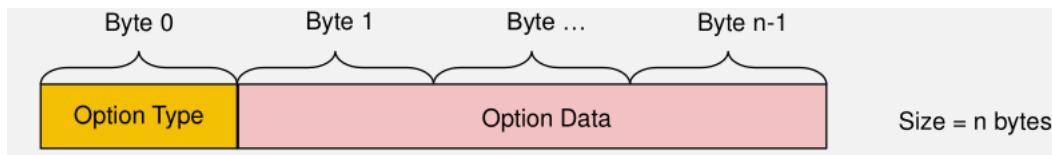
L'header può contenere campi Opzione mediante le quali si richiede una elaborazione "speciale" da parte dei router.

- Security
- Source Routing
- Route Recording
- Stream Identification
- Timestamping

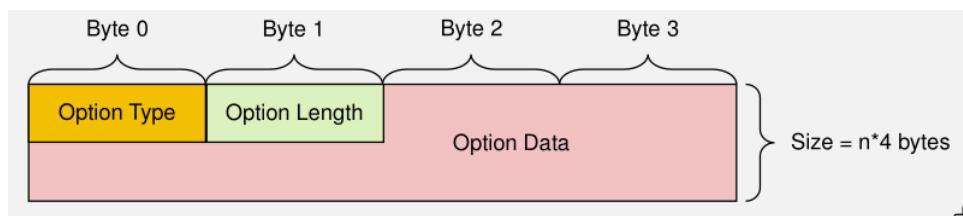
Per la presenza delle opzioni, l'header IP può essere di lunghezza variabile, da questo deriva la

presenza del campo Header Length. Ci sono due formati per segnalare le opzioni

- **Single byte:** Le opzioni di questo formato hanno una lunghezza di n byte definita implicitamente dal valore di Option Type



- **Multiple bytes:** le opzioni di questo formato hanno una lunghezza multipla di 4 byte, esplicitamente indicata nel campo Option Length



## 10.2 Indirizzi IP

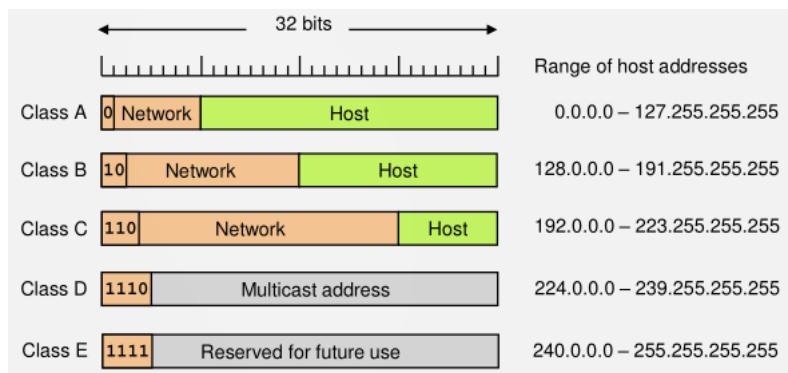
Un indirizzo IP è una sequenza di 32 bit, rappresentata in forma testuale in notazione **dotted decimal**. In una rete IP, un indirizzo IP serve ad identificare univocamente un'interfaccia di rete di un dispositivo.

### 10.2.1 Indirizzi IP e classi

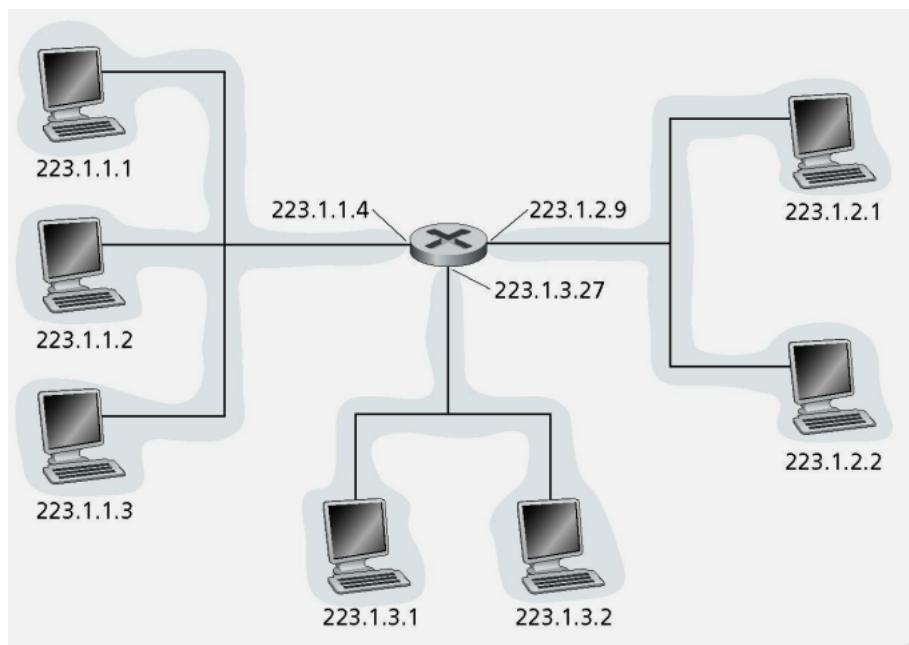
Un indirizzo IP è costituito da due parti:

- **Network id:** un identificatore della rete di appartenenza
- **Host id:** un identificatore del terminale all'interno della rete

Nella rete internet, inizialmente venne adottato un approccio alla gestione degli indirizzi per **classi**. Nella gestione per classe, la demarcazione tra i campi Network e Host è fissa e determinata dal valore dei primi bit.



Si noti che tutti gli host di una stessa rete possono comunicare direttamente a livello 2 senza l'ausilio di un router. Un esempio di tre reti fisiche associate a tre reti di classe C è il seguente:



Esistono alcuni indirizzi IP speciali, ad esempio:

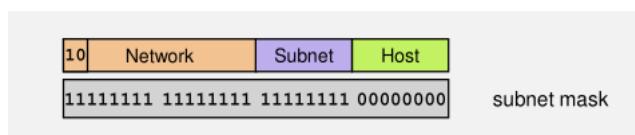
- **0.0.0.0**: è utilizzato per scopi diversi in vari contesti, es. all'interno di un host, identifica "qualunque indirizzo IP assegnato alle sue proprie interfacce"
- **0.X.Y.Z**: è riservato e non può essere assegnato specificamente ad un interfaccia
- **127.X.Y.Z**: sono indirizzi associati ad un'interfaccia virtuale presente in qualunque sistema e che può essere utilizzata per la comunicazione tra processi presenti nella stessa macchina
- **255.255.255.255**: indica il **broadcast** a tutti i dispositivi presenti sulla rete locale del mittente

### 10.3 Netmask

La gestione degli indirizzi IP a classi, comporta un uso inefficiente dello spazio, motivo per il quale dal 1992 venne introdotta una nuova tecnica di gestione degli indirizzi, la **CIDR (classless interdomain routing)**

- La separazione tra campo network e campo host all'interno di una rete è fatta attraverso una stringa di 32 bit ausiliaria, detta **network mask**
- La netmask contiene una sequenza di  $k$  "1" in testa che identificano la parte di bit che costituiscono l'id di rete, ed una restante sequenza di  $32 - k$  "0" che identificano l'host di rete

Ad esempio 255.255.0.0/16 indica che i primi 16 bit sono "1" dunque si riferiscono alla rete. Nella CIDR ciascuna delle reti originariamente definite dalle classi è stata suddivisa in sottoreti, ovvero blocchi di indirizzi consecutivi identificati sottraendo un campo subnet al campo host.



Ci sono inoltre 3 blocchi di indirizzi IP riservati per reti TCP/IP private

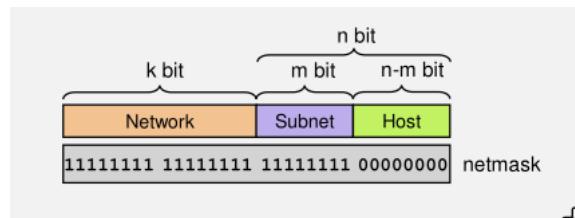
- **10.0.0.0 - 10.255.255.255** ( $10.0.0.0/8$ )
- **172.16.0.0 - 172.255.255.255** ( $172.16.0.0/12$ )
- **192.168.0.0 - 192.168.255.255** ( $192.168.0.0/16$ )

Questi indirizzi vengono utilizzati per **reti private**, ossia reti non collegate a livello 3 alla rete internet.

### 10.3.1 Fixed Length Subnet Mask

Iniziamo definendo un blocco di  $N = 2^n$  indirizzi consecutivi e identificato dal prefisso  $/k$  con  $k = 32 - n$  il termine Fixed Length Subnet Mask (**FLSM**) indica la ripartizione di un blocco di  $N$  indirizzi consecutivi in  $M$  sottoinsiemi disgiunti, ciascuno formato da  $(N/M)$  indirizzi consecutivi (**subnet**). In ciascun blocco di  $(N/M)$  indirizzi consecutivi due sono riservati per scopi speciali

- Campo host tutto a 0  $\Rightarrow$  identifica la subnet
- Campo host tutto a 1  $\Rightarrow$  broadcast alla subnet



All'interno del blocco ciascuna subnet, sarà identificata da

$$m = \log_2(M)$$

bit consecutivi. Tutte le interfacce dei dispositivi della rete saranno configurate con una netmask avente  $k + m$  bit ad 1 per identificare la subnet e  $n - m$  bit a 0 per gli host. En esempio di un blocco di 256 indirizzi ripartito in 8 subnet da  $N/M = 32$  indirizzi ciascuna con netmask /27 è il seguente

Subnet	Subnet Address	Host Range	Broadcast Address
0	192.168.20.0 /27	192.168.20.1 to 192.168.20.30	192.168.20.31
1	192.168.20.32 /27	192.168.20.33 to 192.168.20.62	192.168.20.63
2	192.168.20.64 /27	192.168.20.65 to 192.168.20.94	192.168.20.95
3	192.168.20.96 /27	192.168.20.97 to 192.168.20.126	192.168.20.127
4	192.168.20.128 /27	192.168.20.129 to 192.168.20.158	192.168.20.159
5	192.168.20.160 /27	192.168.20.161 to 192.168.20.190	192.168.20.191
6	192.168.20.192 /27	192.168.20.193 to 192.168.20.222	192.168.20.223
7	192.168.20.224 /27	192.168.20.225 to 192.168.20.254	192.168.20.255

### 10.3.2 Variable Length Subnet Mask

Consideriamo un blocco di  $n = 2$  indirizzi consecutivi con  $k = 32 - n$ . Il termine variable length subnet mask (**VLSM**) indica la ripartizione del blocco di  $N$  indirizzi consecutivi in  $M$  sottoinsiemi disgiunti di differente dimensione. La ripartizione avviene in maniera gerarchica

- Si ripartisce il blocco in  $M_1$  blocchi "grandi" identificati da un prefisso di  $m_1 = \log_2(M_1)$  bit
- Uno o più blocchi ottenuti dalla prima ripartizione sono suddivisi in  $M_2$  blocchi più piccoli identificati da un prefisso di  $m_2 = \log_2(M_2)$  bit
- La ripartizione continua finché necessario

## 10.4 Router

Il router ha due funzioni principali, quella di **forwarding** e quella di **routing**.

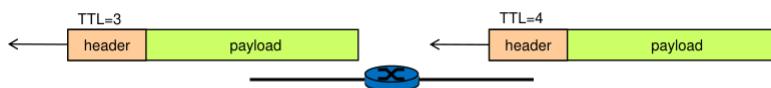
- **Forwarding:** consiste nell'inoltrare ciascun pacchetto che entra da un'interfaccia verso un'altra interfaccia
- **Routing:** consiste nel determinare i percorsi per i pacchetti

Le due funzioni sono svolte contemporaneamente da due parti diverse del router, il **data plane** per quanto riguarda il forwarding e il **control pane** per quanto riguarda il routing.

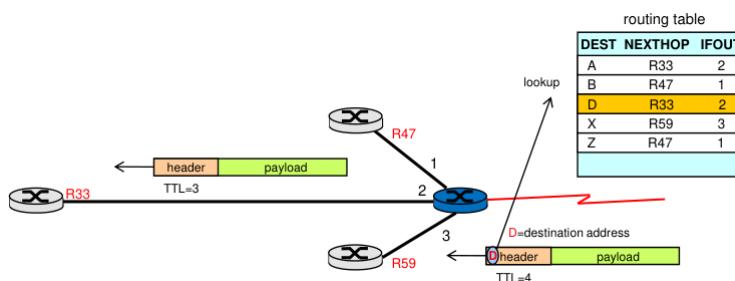
### 10.4.1 Forwarding/Routing

Un router ha diverse interfacce di rete per collegare più reti tra di loro. A ciascuna interfaccia di un router è assegnato un indirizzo IP appartenente alla sottorete associata. La funzione di forwarding del svolta da un router IP è la seguente:

- Per ciascun pacchetto, si determina l'interfaccia di uscita sulla base dell'indirizzo IP destinazione del pacchetto
- Prima della ritrasmissione il campo TTL (time-to-live) nell'header del pacchetto viene decrementato di 1
- Se il TTL diventa zero, il pacchetto non viene inoltrato, ma eliminato



La scelta dell'interfaccia di uscita del pacchetto è determinata sulla base delle regole di stradamento contenute nella **routing table**.



Nella tabella di routing c'è scritto per ogni destinazione, l'indirizzo IP del nexthop router e l'identificativo locale dell'interfaccia tramite la quale si raggiunge il nexthop. In quanto non è possibile avere una regola per ciascun possibile indirizzo IP, ci sono metodi per compattare le regole nelle tabelle di routing.

- Tutti i blocchi di indirizzi consecutivi che hanno lo stesso prefisso e lo stesso nexthop router sono rappresentati nella tabella di routing da una sola regola
- Se non c'è una regola esplicitata si applica una regola di Default

L'operazione di lookup nella tabella di routing viene effettuata tramite **Longest Prefix Match**

#### 10.4.2 Routing Statico e Dinamico

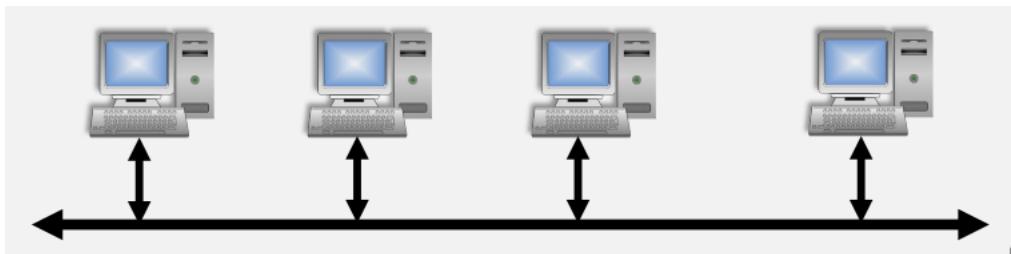
Un router esplica la funzione di forwarding consultando per ogni pacchetto processato la tabella di routing. Quest'ultima può essere costruita in 2 modi:

- **Routing statico**: l'amministratore di rete conoscendo la topologia della rete determina i percorsi tra qualunque coppia sorgente destinazione e conseguentemente configura ciascun router
- **Routing dinamico**: nel control plane, opera un programma il quale mediante scambio di informazioni con i router vicini, determina (attraverso un algoritmo) i percorsi verso quale destinazione e conseguentemente crea nella tabella di routing le regole corrispondenti

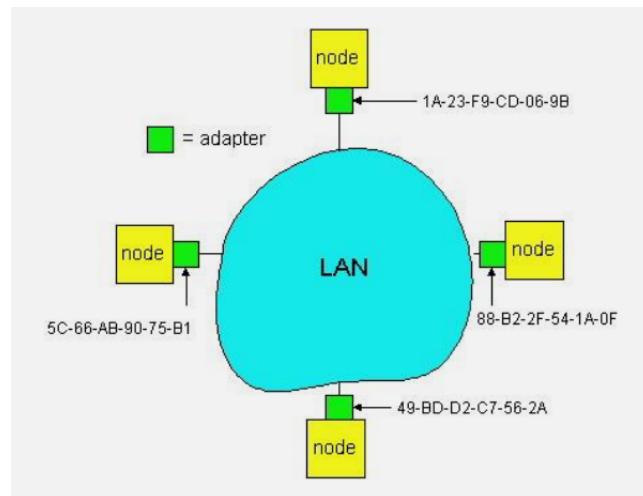
Lo scambio di informazioni tra i router necessario all'esecuzione dell'algoritmo di routing è regolato da appositi protocolli di comunicazione detti **protocolli di routing**

## 11 Protocollo ARP (data link)

Prima di poter introdurre il protocollo ARP, bisogna introdurre il concetto di indirizzo MAC. Quasi tutte le tecnologie di rete locale realizzano in modi diversi l'equivalente logico di un bus



Ciascuna stazione è collegata alla rete mediante una scheda (NIC) identificata univocamente da un indirizzo MAC di 48 bit configurato dal costruttore dell'hardware. Gli indirizzi MAC sono rappresentati comunemente in notazione esadecimale.



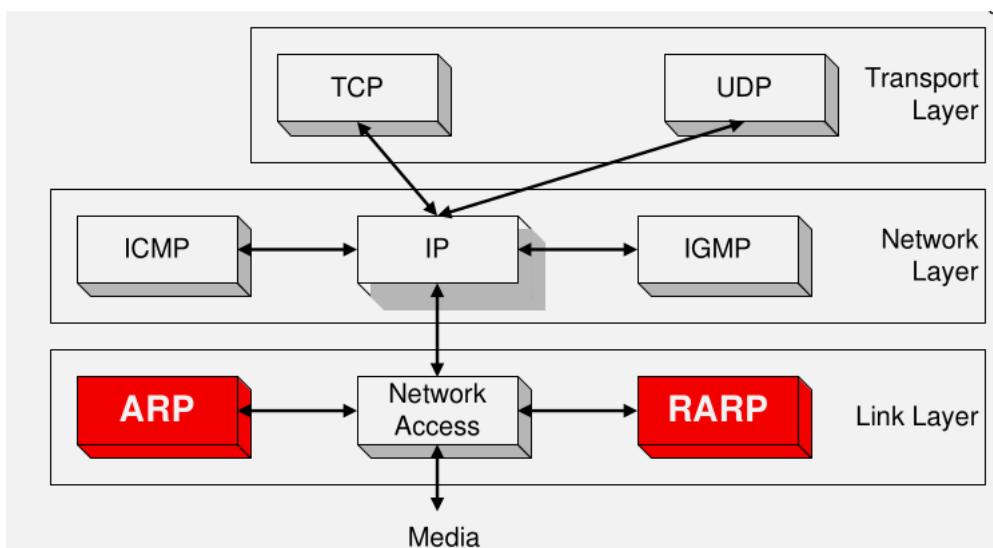
## 11.1 Ethernet

Ipotizziamo l'utilizzo di una PDU di livello 2 (frame) Ethernet, presente nell'header:



- **SA:** indirizzo MAC di 48 bit che identifica la scheda che ha trasmesso la frame
- **DA:** indirizzo MAC di 48 bit che identifica la destinazione della frame. La destinazione può essere:
  - **Unicast:** ad una specifica scheda collegata nella rete locale
  - **Broadcast:** a tutte le schede collegate alla rete locale
  - **Multicast:** un sottoinsieme di schede collegate alla rete locale
- **Type:** indica il protocollo del pacchetto trasportato dalla frame nella parte Data (IPv6, IPv4, ARP)

I protocollo ARP e RARP svolgono funzioni ausiliarie a supporto della trasmissione di datagramma IP in reti locali con capacità di trasmissione broadcast.

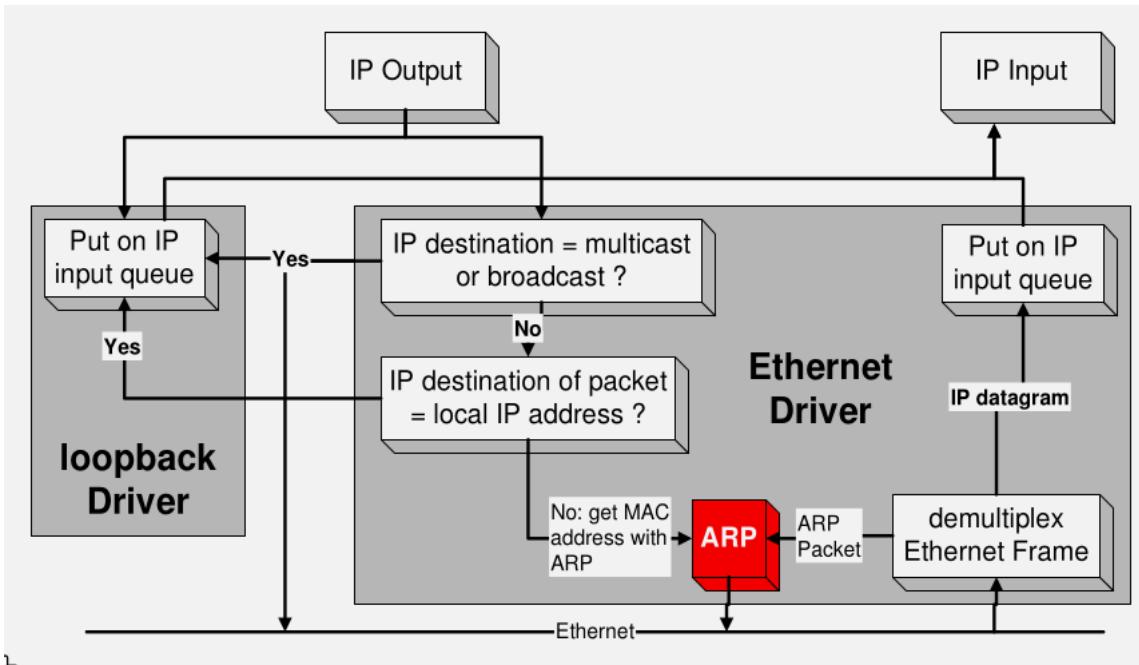


## 11.2 ARP

Il protocollo ARP serve ad associare ad un indirizzo di rete il corrispondente indirizzo MAC

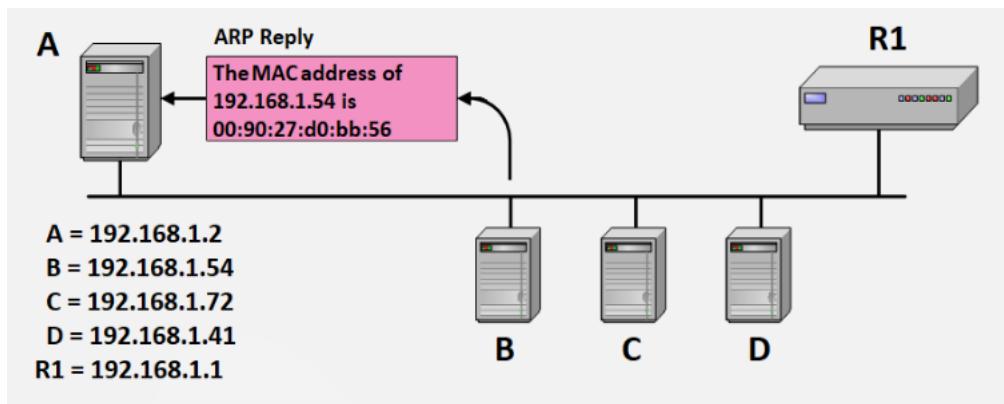
- La sua funzione è necessaria quando un host vuole trasmettere un pacchetto IP ad una certa destinazione presente sulla rete locale e non ne conosce il corrispondente indirizzo MAC.

Il protocollo RARP serve ad associare un indirizzo MAC al corrispondente indirizzo di rete.

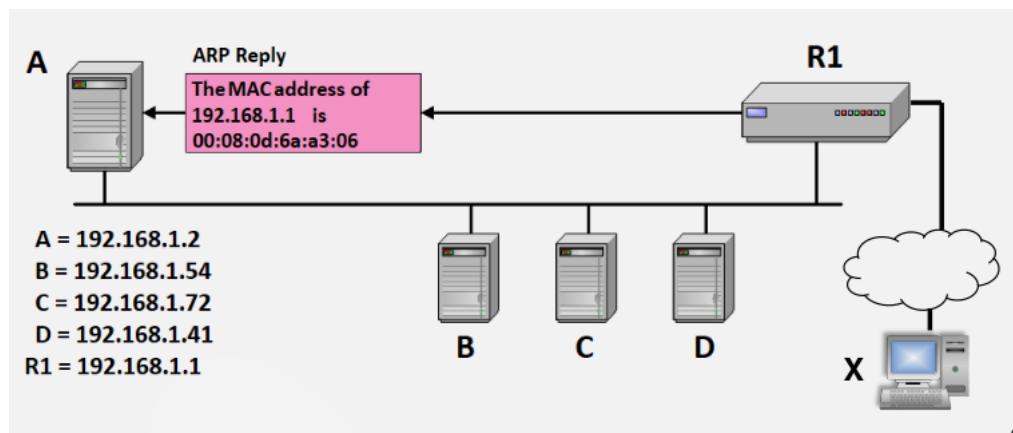


Distinguiamo due scenari di uso per ARP:

- A vuole trasmettere un pacchetto IP a B, la destinazione è nella stessa LAN (subnet IP) del mittente.
  - A chiede mediante ARP di conoscere il MAC address associato alla destinazione B (target IP) direttamente raggiungibile
  - La richiesta ARP è trasmessa in una frame con indirizzo MAC destinazione broadcast (FF:FF:FF:FF:FF:FF)
  - La risposta ARP è inviata da B direttamente ad A

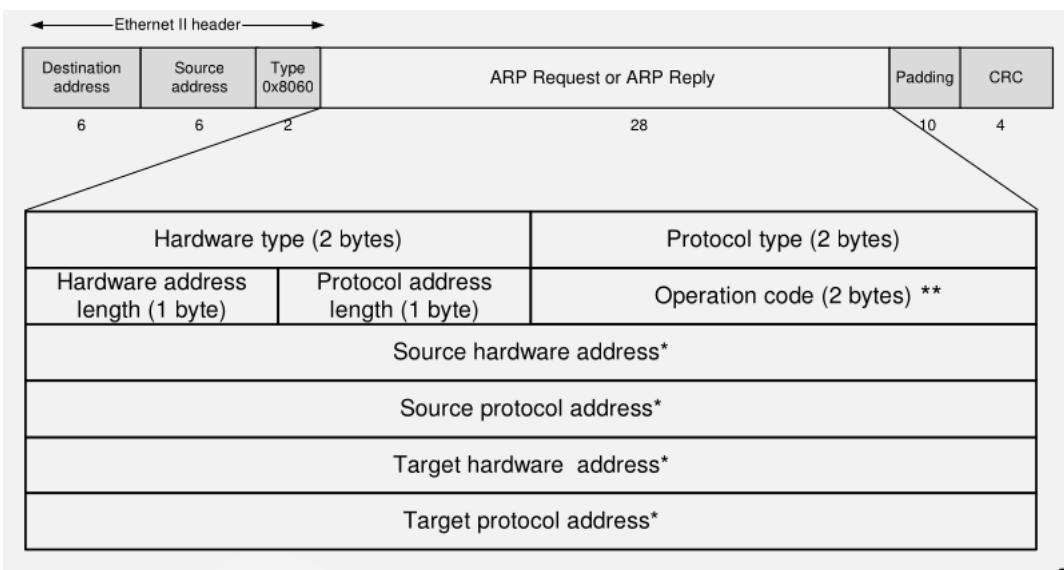


- A vuole trasmettere un pacchetto ad X fuori dalla LAN (subnet IP) del mittente
  - In base alla sua tabella di routing, A determina l'indirizzo di R1 del gateway associato alla destinazione X
  - A chiede mediante ARP di conoscere il MAC address associato al gateway R1
  - La richiesta è trasmessa in una frame con indirizzo MAC destinazione broadcast
  - La risposta ARP è inviata dal gateway R1 direttamente ad A



### 11.2.1 ARP Packet Format

Il formato di una ARP è il seguente



Dove

- **Protocol:** per riferirsi a indirizzi di rete (tipicamente IPv4)
- **Hardware:** per riferirsi ad indirizzi fisici (tipicamente MAC)
- **Source:** per riferirsi al mittente del messaggio ARP
- **Target:** per riferirsi al destinatario del messaggio ARP
- **Operation Code:** discrimina tra due tipi di messaggi
  - ARP request: con OpCode = 0x0001
  - ARP reply: con OpCode = 0x0002

Per ridurre il traffico sulla rete e ridurre il tempo necessario all'invio di nuovi pacchetti IP, ogni host mantiene in una cache le corrispondenze tra indirizzi logici e fisici precedentemente apprese. Ciascuna corrispondenza viene mantenuta per un tempo limitato

### 11.2.2 Gratuitous ARP

Un gratuitous ARP o GARP è un messaggio ARP reply trasmesso in una frame con MAC address FF:FF:FF:FF:FF:FF.

Address Resolution Protocol (reply/gratuitous ARP)	
Hardware type:	Ethernet (1)
Protocol type:	IPv4 (0x0800)
Hardware size:	6
Protocol size:	4
Opcode:	reply (2)
[Is gratuitous: True]	
Sender MAC address:	00:53:ff:ff:bb:bb
Sender IP address:	10.0.0.22
Target MAC address:	ff:ff:ff:ff:ff:ff
Target IP address:	10.0.0.22

Un GARP è un messaggio di risposta ARP unsolicited, ossia non generato per effetto di una precedente richiesta. I GARP sono generati se:

- L'host desidera popolare le ARP cache degli altri host della rete locale
- L'host ha modificato il proprio indirizzo IP oppure ha subito una disconnessione dalla rete e desidera aggiornare del cambiamento le altre interfacce nella rete locale

### 11.2.3 ARP Probe

Un ARP Probe è un messaggio ARP request trasmesso in una frame con MAC destinazione FF:FF:FF:FF:FF:FF

Address Resolution Protocol (request)	
Hardware type:	Ethernet (1)
Protocol type:	IPv4 (0x0800)
Hardware size:	6
Protocol size:	4
Opcode:	request (1)
Sender MAC address:	00:50:56:c0:00:01
Sender IP address:	0.0.0.0
Target MAC address:	00:00:00:00:00:00
Target IP address:	192.168.174.111

Un ARP Probe può essere usato per verificare che nella rete locale non ci siano altri host con lo stesso indirizzo IP target

### 11.2.4 ARP Announcement

Un ARP announcement è un messaggio ARP request in una frame con MAC destinazione FF:FF:FF:FF:FF:FF

Address Resolution Protocol (request/gratuitous ARP)	
Hardware type:	Ethernet (1)
Protocol type:	IPv4 (0x0800)
Hardware size:	6
Protocol size:	4
Opcode:	request (1)
[Is gratuitous: True]	
Sender MAC address:	00:50:56:c0:00:01
Sender IP address:	192.168.174.111
Target MAC address:	00:00:00:00:00:00
Target IP address:	192.168.174.111

Un ARP announcement è simile ad un gratuitous ARP con la differenza di essere un messaggio di richiesta

## 12 Internet Control Message Protocol (ICMP Applicativo)

L'**Internet Control Message Protocol** (ICMP) è un protocollo di rete utilizzato per inviare messaggi di controllo e di errore tra dispositivi di rete, come router e host. Fa parte della suite di protocolli Internet ed è definito nella RFC 792. ICMP è principalmente utilizzato per diagnosticare e gestire problemi di comunicazione nella rete, ma svolge anche un ruolo cruciale in strumenti di diagnostica come **ping** e **traceroute**.

### 12.1 Scopo di ICMP

ICMP è progettato per fornire un meccanismo standardizzato per la segnalazione di errori e la gestione di situazioni anomale nella rete. I suoi principali scopi includono:

- **Segnalazione di errori:** ICMP invia messaggi di errore quando un pacchetto IP non può essere consegnato correttamente. Ad esempio, se un router non può inoltrare un pacchetto perché la destinazione è irraggiungibile, invierà un messaggio ICMP di tipo "Destination Unreachable" al mittente.
- **Diagnostica di rete:** ICMP è alla base di strumenti di diagnostica come **ping**, che utilizza messaggi ICMP di tipo "Echo Request" e "Echo Reply" per verificare la raggiungibilità di un host e misurare il tempo di risposta.
- **Gestione del traffico:** ICMP può essere utilizzato per gestire il flusso di traffico nella rete. Ad esempio, un router può inviare un messaggio ICMP di tipo "Source Quench" per chiedere a un host di ridurre la velocità di invio dei pacchetti.
- **Supporto al routing:** ICMP supporta il routing dinamico attraverso messaggi come "Redirect", che informano un host di un percorso più efficiente verso una destinazione.

### 12.2 Funzionamento di ICMP

ICMP opera a livello di rete (Layer 3) del modello OSI, insieme al protocollo IP. I messaggi ICMP sono incapsulati all'interno di pacchetti IP, ma a differenza di altri protocolli come TCP o UDP, ICMP non è utilizzato per il trasporto di dati applicativi. I messaggi ICMP sono generalmente generati dai dispositivi di rete, come router o host, in risposta a condizioni specifiche.

#### 12.2.1 Struttura di un messaggio ICMP

Un messaggio ICMP è composto da un **header** e un **payload**. L'header contiene i seguenti campi:

- **Tipo (8 bit):** Specifica il tipo di messaggio ICMP (ad esempio, Echo Request, Destination Unreachable, Time Exceeded).
- **Codice (8 bit):** Fornisce ulteriori dettagli sul tipo di messaggio. Ad esempio, nel caso di un messaggio "Destination Unreachable", il codice può indicare se la destinazione è irraggiungibile a causa di un problema di rete, di un host o di un protocollo.
- **Checksum (16 bit):** Utilizzato per verificare l'integrità del messaggio ICMP.
- **Campi specifici (variabile):** A seconda del tipo di messaggio, possono essere presenti ulteriori campi, come l'identificatore e il numero di sequenza per i messaggi di Echo

Request/Reply.

### 12.2.2 Esempi di messaggi ICMP

Alcuni dei messaggi ICMP più comuni includono:

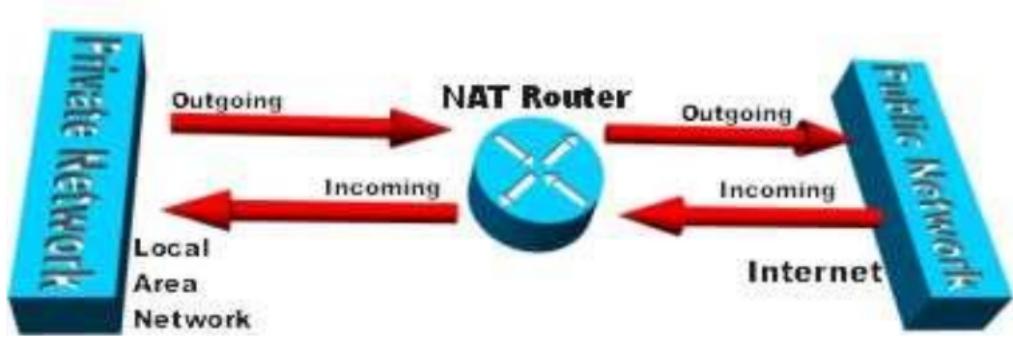
- **Echo Request / Echo Reply:** Utilizzati dal comando ping per verificare la raggiungibilità di un host. Un host invia un messaggio "Echo Request" e attende una risposta "Echo Reply".
- **Destination Unreachable:** Indica che un pacchetto IP non può essere consegnato alla destinazione. Il codice specifica il motivo, come "Network Unreachable" o "Host Unreachable".
- **Time Exceeded:** Generato quando un pacchetto supera il Time To Live (TTL) definito nel suo header IP. Questo messaggio è utilizzato da traceroute per determinare i router lungo il percorso verso una destinazione.
- **Redirect:** Inviato da un router per informare un host di un percorso più efficiente verso una destinazione.

## 12.3 Sicurezza e ICMP

ICMP può essere utilizzato anche per scopi malevoli, come nel caso di attacchi di tipo **ICMP Flood** o **Smurf Attack**, che sfruttano messaggi ICMP per sovraccaricare una rete o un host. Per questo motivo, molti firewall e sistemi di sicurezza limitano o filtrano il traffico ICMP per prevenire abusi.

## 13 NAT (rete)

La **Network Address Translation** è una tecnica che consente ad un dispositivo (router) di agire come intermediario tra Internet (rete pubblica) e una rete privata. In questo modo un unico indirizzo IP può rappresentare un intero gruppo di computer di una rete privata.



l'uso più comune del nat è quello di mappare un insieme di indirizzi privati su di un unico indirizzo pubblico utilizzando differenti porti.

### 13.1 Funzionamento

Un router NAT deve:

- Sostituire nei pacchetti in uscita dalla rete privata

- Indirizzo IP privato del mittente A, numero di port sorgente P con l'indirizzo IP pubblico del NAT N, e numero di port sorgente X
- Ricordare in una tabella le corrispondenze
  - assegnare ad un indirizzo privato e una porta l'indirizzo pubblico del nat e la porta di destinazione X

Source Computer	Source Computer's IP Address	Source Computer's Port	NAT Router's IP Address	NAT Router's Assigned Port Number
A	192.168.32.10	400	215.37.32.203	1
B	192.168.32.13	50	215.37.32.203	2
C	192.168.32.15	3750	215.37.32.203	3
D	192.168.32.18	206	215.37.32.203	4

Tabella  
NAT per  
TCP

- sostituire nei pacchetti in entrata dalla rete pubblica
  - indirizzo IP pubblico del NAT, numero di porta destinazione X con indirizzo IP privato del destinatario e porta di destinazione P

## 14 Protocollo DHCP (applicativo)

Il protocollo DHCP (Dynamic Host Configuration Protocol) è un protocollo di rete utilizzato per assegnare automaticamente indirizzi IP e altri parametri di configurazione di rete ai dispositivi collegati a una rete. DHCP semplifica la gestione delle reti, specialmente in ambienti di grandi dimensioni, eliminando la necessità di configurare manualmente ogni dispositivo.

### 14.1 Cos'è il DHCP

- Il DHCP è un protocollo di livello applicazione (Layer 7 del modello OSI) che opera su UDP.
- Viene utilizzato per assegnare dinamicamente indirizzi IP ai dispositivi in una rete, evitando conflitti di indirizzi e riducendo il carico di lavoro amministrativo.
- Oltre all'indirizzo IP, il DHCP può fornire altri parametri di configurazione, come:
  - Subnet mask
  - Gateway predefinito
  - Indirizzi dei server DNS
  - Tempo di lease (durata dell'assegnazione dell'indirizzo IP)

### 14.2 A cosa serve il DHCP?

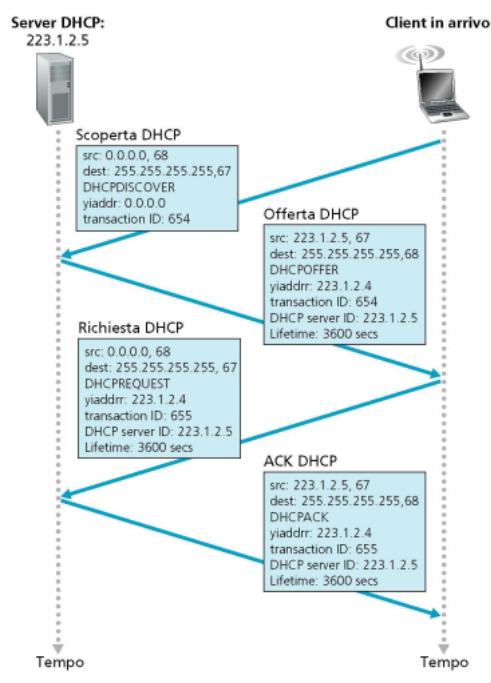
- **Assegnazione automatica degli indirizzi IP:** Il DHCP assegna automaticamente un indirizzo IP a un dispositivo quando si connette alla rete, senza che l'utente debba configurarlo manualmente.
- **Gestione centralizzata:** Gli amministratori di rete possono gestire e monitorare gli indirizzi IP da un server centrale, semplificando la configurazione e la manutenzione della rete.

- **Riduzione degli errori:** Poiché gli indirizzi IP vengono assegnati automaticamente, si riduce il rischio di conflitti di indirizzi o errori di configurazione manuale.
- **Ottimizzazione dell'uso degli indirizzi IP:** Il DHCP gestisce dinamicamente il pool di indirizzi IP disponibili, assegnandoli solo quando necessario e rilasciandoli quando non sono più in uso.
- **Flessibilità:** Il DHCP supporta reti di diverse dimensioni, dalle piccole reti domestiche alle grandi reti aziendali.

### 14.3 Funzionamento del DHCP

Il funzionamento del DHCP si basa su un processo a quattro fasi, noto come **DORA**:

1. **Discover:** Il client DHCP invia un messaggio di broadcast (DHCPDISCOVER) per trovare un server DHCP disponibile nella rete.
2. **Offer:** Il server DHCP risponde con un messaggio DHCPOFFER, offrendo un indirizzo IP e altri parametri di configurazione.
3. **Request:** Il client accetta l'offerta inviando un messaggio DHCPREQUEST al server.
4. **Acknowledge:** Il server conferma l'assegnazione con un messaggio DHCPACK, completando il processo di configurazione.



### 14.4 Vantaggi del DHCP

- **Facilità di configurazione:** Gli utenti non devono configurare manualmente i parametri di rete.
- **Riduzione degli errori:** Meno probabilità di conflitti di indirizzi IP.
- **Scalabilità:** Ideale per reti di grandi dimensioni con molti dispositivi.
- **Risparmio di tempo:** Gli amministratori di rete risparmiano tempo nella gestione degli indirizzi IP.

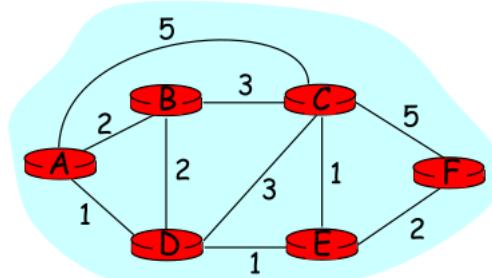
## 15 Routing (rete)

Il problema del routing si presenta al layer 3 dell'architettura OSU. Per routing si intende il processo decisionale per la scelta del percorso ottimale di un pacchetto da un mittente ad una destinazione. Gli algoritmi per la gestione di una rete sono basati sulla teoria dei grafi in cui la rete viene modellata come un grafo

- **nodi:** rappresentano i router

- **archi:** rappresentano i link fisici

- Ogni link ha un costo, relativo al ritardo, il costo di trasmissione, la congestione ecc...



▲

La scelta del cammino di solito si basa sul cammino a costo minimo, ma è possibile calcolarlo anche in base ad altri parametri, quali vincoli specifici ecc. I parametri del processo decisionale sono

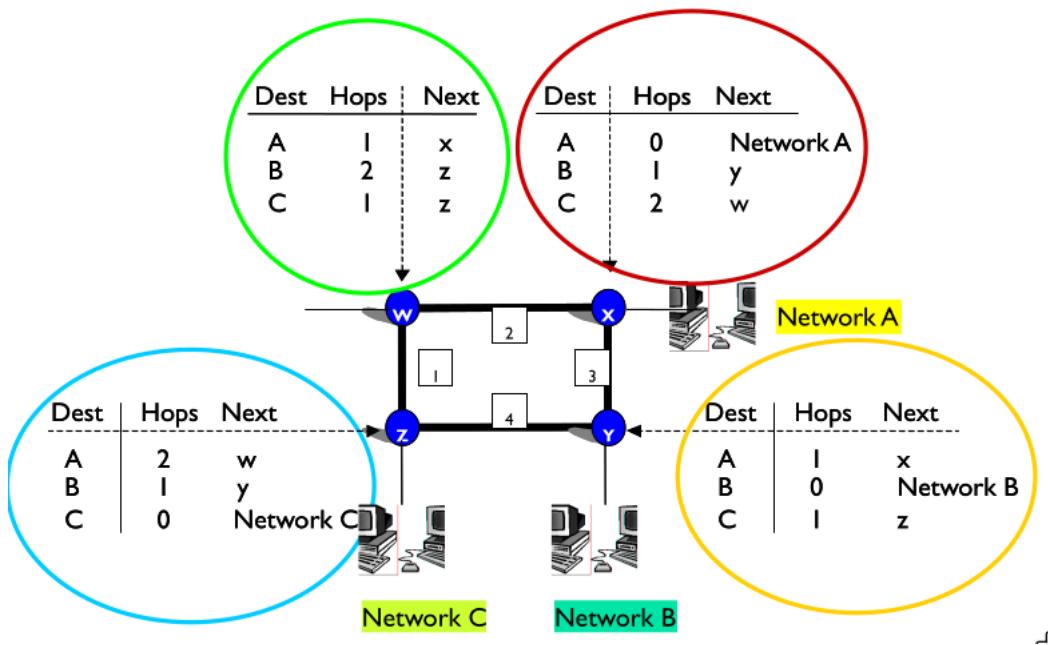
- **Bandwidth:** capacità di un link (bit per secondo)
- **Delay:** tempo necessario per spedire un pacchetto da una sorgente ad una destinazione
- **Load:** carico di un link
- **Reliability:** affidabilità di un link (basata sull'error rate)
- **Hop count:** il numero di router da attraversare nel percorso dalla sorgente alla destinazione
- **Cost:** valore arbitrario che definisce il costo di un link, costituito su vari parametri come packet loss, delay, MTU...

### 15.1 Tipologie di routing

Lo scopo ultimo del routing consiste nel creare una **tavella di instradamento** in ciascun nodo (router) della rete. Ci possono essere 3 approcci differenti

- **routing statico:** i percorsi sono calcolati una tantum sulla base della conoscenza della topologia della rete, e conseguentemente, l'amministratore della rete impone le tabelle di routing
- **routing dinamico a controllo centralizzato:** un'entità centralizzata detta controller acquisisce dai router informazioni circa la topologia e lo stato della rete e calcola i percorsi configurando i router
- **routing dinamico a controllo locale:** i router si scambiano informazioni circa lo stato della rete e sulla base delle informazioni acquisite ciascun router determina per ogni possibile destinazione il next-hop router

Un esempio di routing è il seguente



## 15.2 Tecniche di routing

Esistono diverse tecniche di routing, vediamo il routing by network address e il label swapping:

- **Network Address Routing:** ogni pacchetto contiene l'indirizzo del nodo destinatario, che viene usato come chiave di accesso alle tabelle di instradamento. Si utilizza nei protocolli non orientati alla connessione come (IPv4, IPv6)
- **Label Swapping:** ogni pacchetto è marcato con una label che identifica la connessione
  - identifica la connessione
  - viene utilizzata come chiave di accesso per le tabelle di instradamento

Si utilizza solitamente in protocolli orientati alla connessione (X.25, ATM, MPLS)

# 16 Routing Distance Vector

Ogni router mantiene una tabella di tutti gli instradamenti noti, in cui ogni entry della tabella indica:

- Una rete raggiungibile
- Il next hop
- Il numero di hop necessari per raggiungere la destinazione

Periodicamente ogni router invia a tutti i vicini (router collegati alla stessa rete fisica), un messaggio di aggiornamento contenente tutte le informazioni della propria tabella (**vettore delle distanze**). I router che ricevono tale messaggio aggiornano la tabella nel seguente modo:

- eventuale modifica di informazioni relative a cammini già noti
- eventuale aggiunta di nuovi cammini
- eventuale eliminazione di cammini non più disponibili,

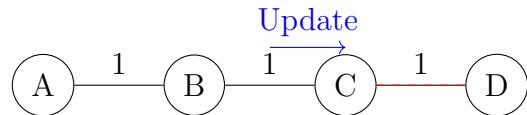
## 16.1 Equazione di Bellman-Ford

L'algoritmo si basa sull'equazione fondamentale:

$$D_x(y) = \min_v \{c(x, v) + D_v(y)\} \quad (1)$$

dove:

- $D_x(y)$ : costo del percorso ottimo da  $x$  a  $y$
- $c(x, v)$ : costo del link diretto tra  $x$  e il vicino  $v$
- $D_v(y)$ : costo annunciato dal vicino  $v$  per raggiungere  $y$



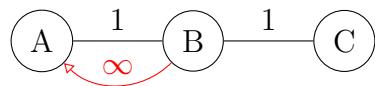
## 16.2 Velocità di Convergenza

La velocità di convergenza, è la velocità con cui un protocollo giunge a convergenza, ossia tutti i router di una rete stanno operando con la stessa visione della rete stessa. Problema principale: **count-to-infinity** (Se un collegamento si interrompe (es. tra B e C), i router potrebbero incrementare all'infinito la distanza (es. A pensa che C sia raggiungibile via B, e B via A)) in reti con loop.

- Tempo di convergenza  $O(n)$  per reti con  $n$  nodi
- Aggiornamenti periodici (tipicamente 30 secondi in RIP)
- Esempio classico:  
A-B-C-D (link C-D si rompe)  
B pensa di raggiungere D via A → A pensa di raggiungere D via B

## 16.3 Poisoned Reverse

Tecnica per prevenire loop di routing: Logica di funzionamento:



- Se il percorso ottimo verso  $y$  passa attraverso  $v$
- Annuncia a  $v$  costo infinito per  $y$
- Previene loop ma riduce l'ottimalità delle rotte

# 17 Link-State e Dijkstra

## 17.1 Routing Link-State

Nell'algoritmo di routing Link-State, ogni router

- impara il suo ambito locale (linee e nodi adiacenti)

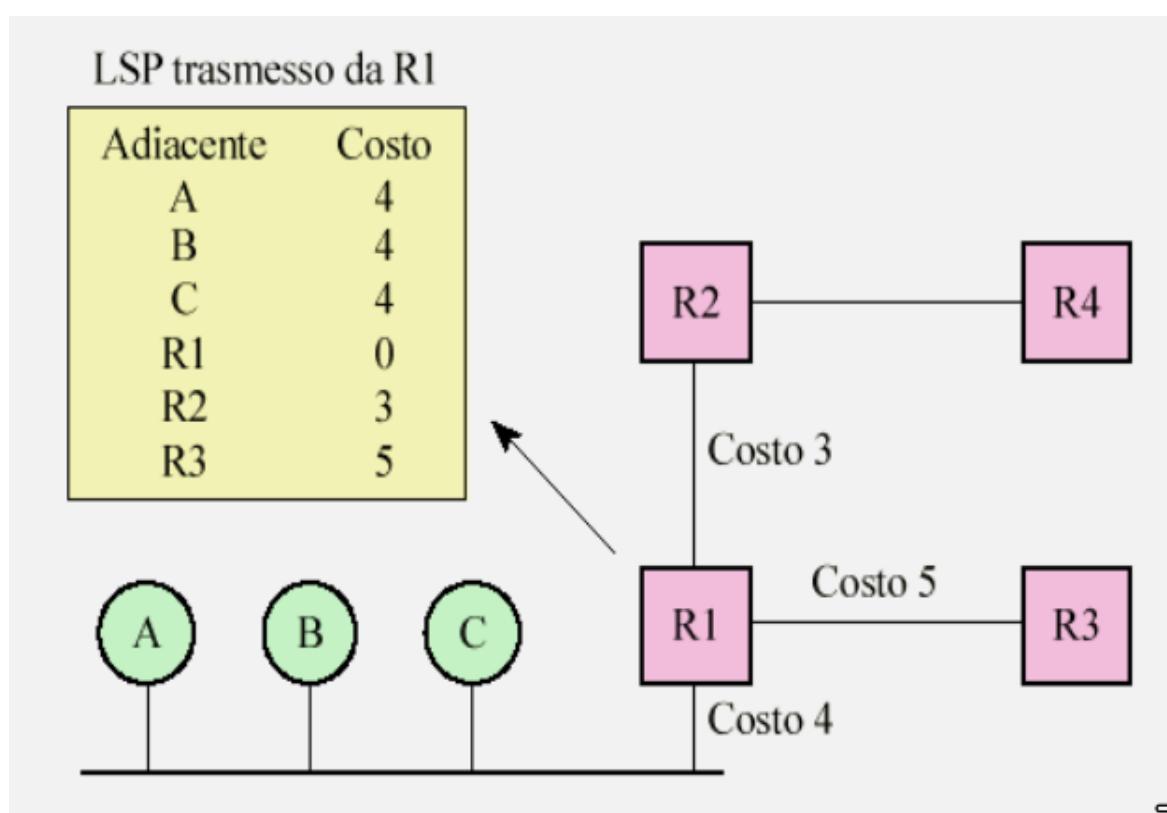
- trasmette queste informazioni a tutti gli altri router della rete tramite un **Link State Packet** (LSP)
- memorizza gli LSP trasmessi dagli altri router e costruisce una mappa della rete
- calcola in maniera indipendente, le sue tabelle di instradamento applicando alla mappa della rete l'algoritmo di Dijkstra

LSP contiene diversi campi

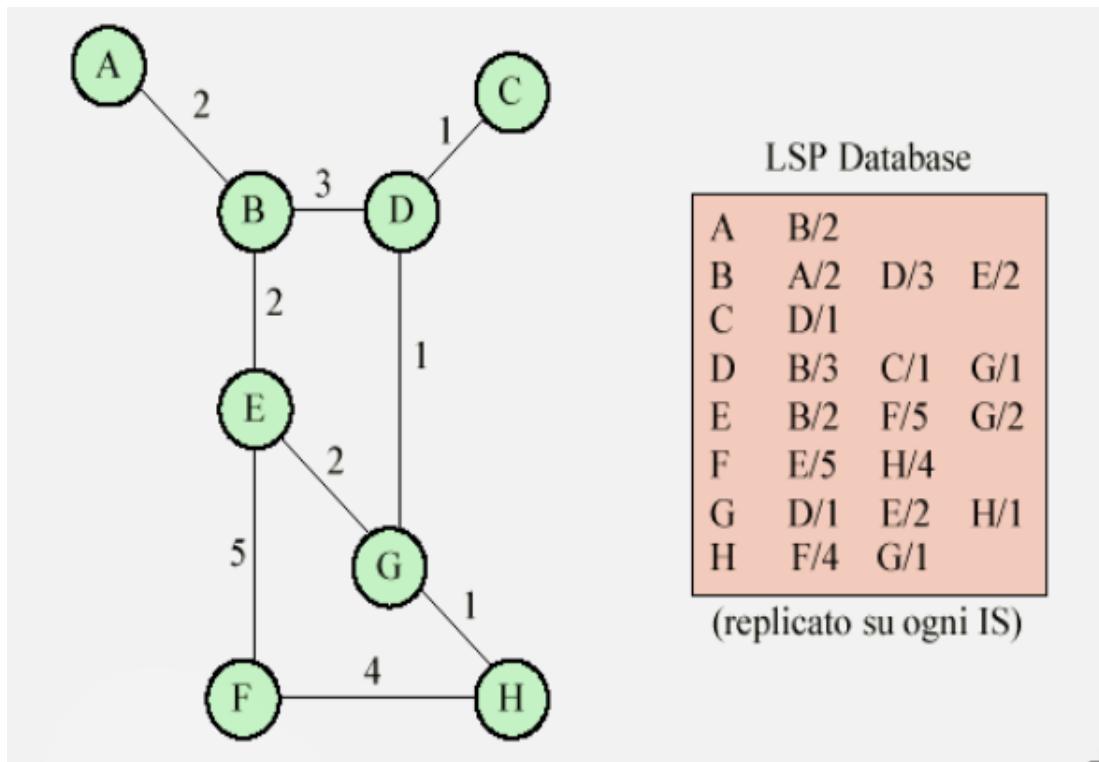
- stato di ogni link connesso al router
- identità di ogni vicino connesso all'altro estremo del link
- costo del link
- numero di sequenza per l'LSP (onde evitare informazioni passate)
- checksum
- lifetime

### 17.1.1 LSP flooding

Un LSP viene generato periodicamente, oppure se viene rilevato un cambiamento nella topologia locale (nuovo vicino, cambiamento del costo verso un vicino, persa connettività verso un vicino).



Successivamente alla generazione dell'LSP avviene un operazione di **flooding**, in cui l'LSP è trasmesso su tutti i link del router, che a loro volta lo trasmetteranno a tutti i loro link. I pacchetti memorizzati nei router formano una mappa completa e aggiornata della rete detta **Link State Database**



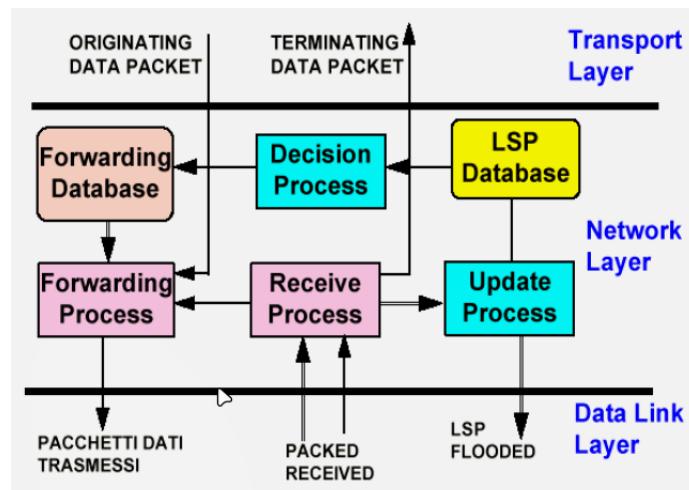
### 17.1.2 Gestione LSP

Alla ricezione di un LSP ci sono 3 casi possibili:

- **LSP nuovo/aggiornato:** il router memorizza il pacchetto e lo ritrasmette in flooding su tutte le linee eccetto quella da cui l'ha spedito
- **LSP con stesso numero di sequenza di quello posseduto:** il router lo ignora e non fa nulla
- **LSP vecchio:** trasmette al mittente il pacchetto più recente

### 17.1.3 Architettura Link-State

Un ulteriore componente fondamentale per comporre l'architettura di un router link state, è il **Forwarding Database**. Ogni router ne produce uno basandosi sul contenuto del Link State Database, e il forwarding database contiene l'insieme delle coppie (path, vicino) per ogni nodo



Ci sono svariati pregi quale la gestione di reti di grandi dimensioni, la rapida convergenza, la minor probabilità di loop, tuttavia è molto complesso da realizzare

## 17.2 Algoritmo di Dijkstra

L'algoritmo di Dijkstra è un metodo per calcolare il percorso più breve da un nodo sorgente a tutti gli altri nodi in un grafo con pesi **non negativi**. È ampiamente utilizzato nei protocolli di routing Link-State, come OSPF.

### 17.2.1 Descrizione dell'Algoritmo

L'algoritmo funziona come segue:

**1. Inizializzazione:**

- Assegna distanza 0 al nodo sorgente.
- Assegna distanza  $\infty$  a tutti gli altri nodi.
- Crea una lista di nodi non visitati.

**2. Iterazione:**

- Seleziona il nodo non visitato con la distanza minima.
- Per ogni vicino del nodo selezionato, calcola la nuova distanza:

$$d(y) = \min\{d(y), d(x) + c(x, y)\}$$

dove:

- $d(y)$ : distanza corrente verso il nodo  $y$ .
- $d(x)$ : distanza dal nodo sorgente al nodo  $x$ .
- $c(x, y)$ : costo del collegamento tra  $x$  e  $y$ .

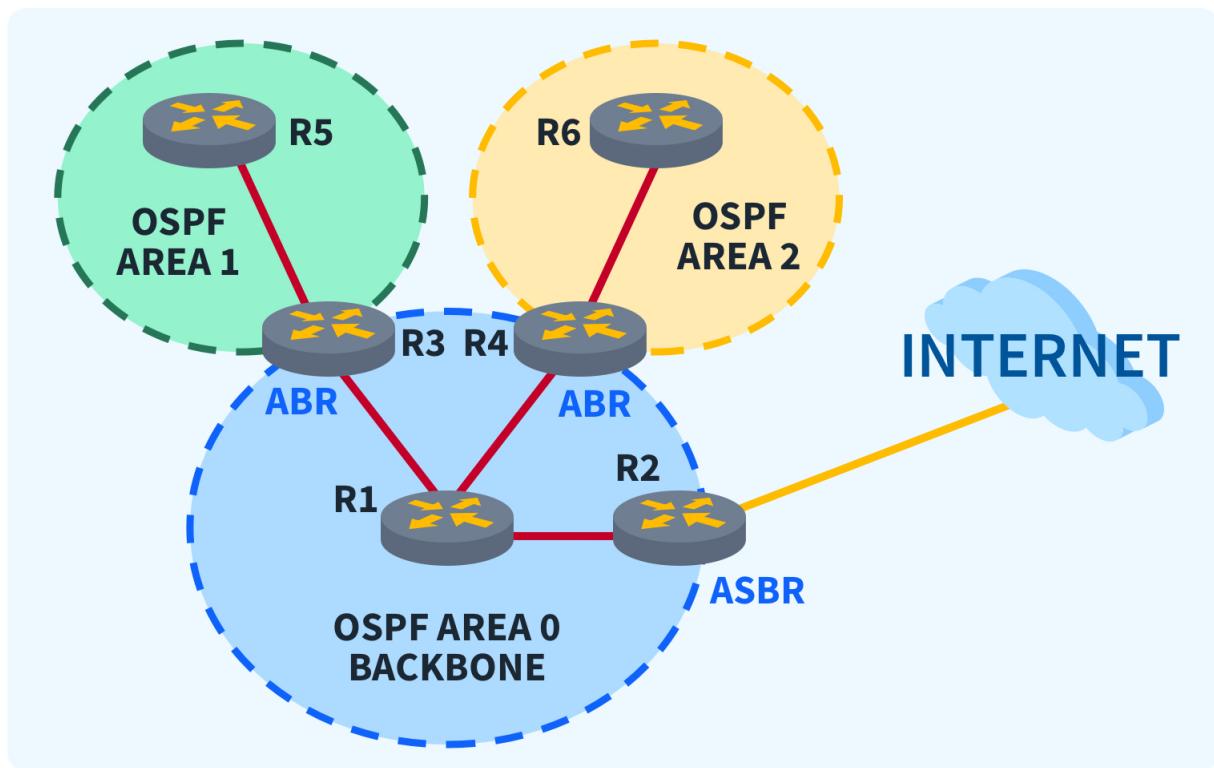
- Segna il nodo selezionato come visitato.

**3. Ripeti** finché tutti i nodi non sono visitati.

## 18 Protocollo OSPF (rete)

Il **OSPF (Open Shortest Path First)** è un protocollo link-state utilizzato all'interno di sistemi autonomi (IGP). Opera al layer 3 del modello OSI e utilizza l'algoritmo di Dijkstra per calcolare il percorso più breve verso una destinazione. Caratteristiche principali:

- Supporta **VLSM** (Variable Length Subnet Mask) e **CIDR**
- Utilizza multicast (224.0.0.5 e 224.0.0.6) per la comunicazione
- Offre autenticazione integrata
- Divide le reti in **aree** per ridurre il traffico di routing



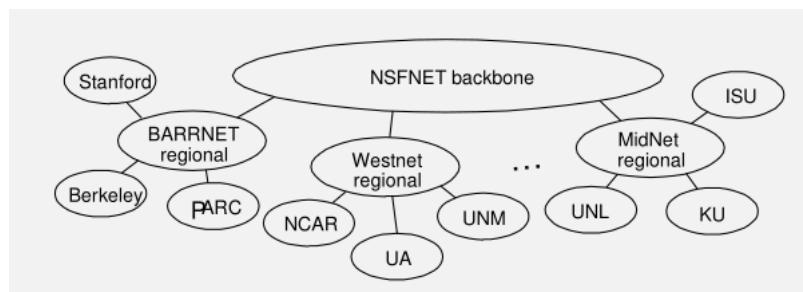
OSPF prevede 4 tipi di router:

- **Internal router**: tutte le sue interfacce appartengono alla stessa area
- **Area border router**: possiede interfacce in due o più aree distinte
- **Backbone router**: possiede almeno un'interfaccia appartenente all'area 0
- **Autonomous system boundary router**: almeno una delle sue interfacce utilizza un diverso protocollo di routing o appartiene ad un altro AS

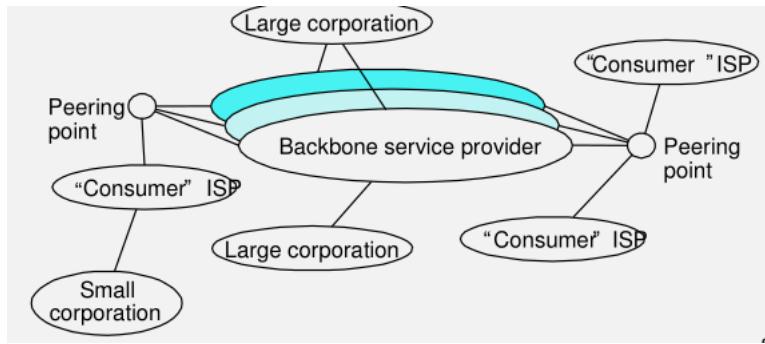
OSPF è organizzato su una gerarchia ad aree a due livelli, la **local area** e la **backbone (area 0)**. Gli Advertisement Link-State non lasciano le rispettive aree, dunque i nodi in ogni area hanno una topologia dettagliata dell'area ma conoscono solo la direzione verso reti in altre aree. Gli Area Border router riassumono distanze a reti nell'area di competenza e le comunicano ad altri router di tipo Area Border.

## 19 Routing gerarchico (rete)

Nel corso degli anni l'architettura di internet, si è evoluta da un'architettura molto semplice basata su un'unica rete backbone, in cui ogni rete fisica era collegata alla backbone da un core router ad un'architettura più scalabile



Al giorno d'oggi, il routing in internet si basa su reti con **peer backbone** dando vita al **routing gerarchico**.



Internet è strutturato come un'insieme di **autonomous systems** (AS). Un AS è una collezione limitata di reti amministrate da un'unica autorità con un identificativo AS number di 16 bit. Ogni AS è responsabile del routing all'interno delle sue reti (**routing interno**) e dello scambio di informazioni di raggiungibilità con altri AS (**routing esterno**). Abbiamo dunque due tipi di routing:

- Routing interno ad un AS (**intra-domain routing**), gestito dall'**Interior Gateway Protocol** (IGP). I messaggi IGP sono scambiati tra router interni all'AS e contengono solo informazioni sulle reti dell'AS. Questo tipo di routing si basa su protocolli quali:
  - RIP (distance vector)
  - OSPF (link state)
  - IGRP (Interior Gateway Routing Protocol - Cisco)
- Routing esterno tra AS (**inter-domain routing**), gestito dall'**Exterior Gateway Protocol** (EGP). I messaggi sono scambiati tra router designati dai rispettivi AS come router di confine (Border Router) e contengono informazioni sulle rotte conosciute dai due AS. Questo tipo di routing si basa su due protocolli:
  - EGP (Exterior Gateway Protocol), ormai obsoleto
  - BGP (Border Gateway Protocol), approccio path vector

## 19.1 Tipi di AS

Disinguiamo gli AS in base ad una serie di caratteristiche. La prima distinzione possibile è quella per dimensione.

- **stub o single homed**: AS con un unico border router
- **multi-homed**: AS con più border router

Un ulteriore distinzione è quella riguardante AS transit e non-transit

- **transit AS** (provider): accetta di essere attraversato da traffico diretto ad altri AS
- **non-transit AS** (grandi corporate): non accetta di essere attraversato da traffico diretto ad altri AS

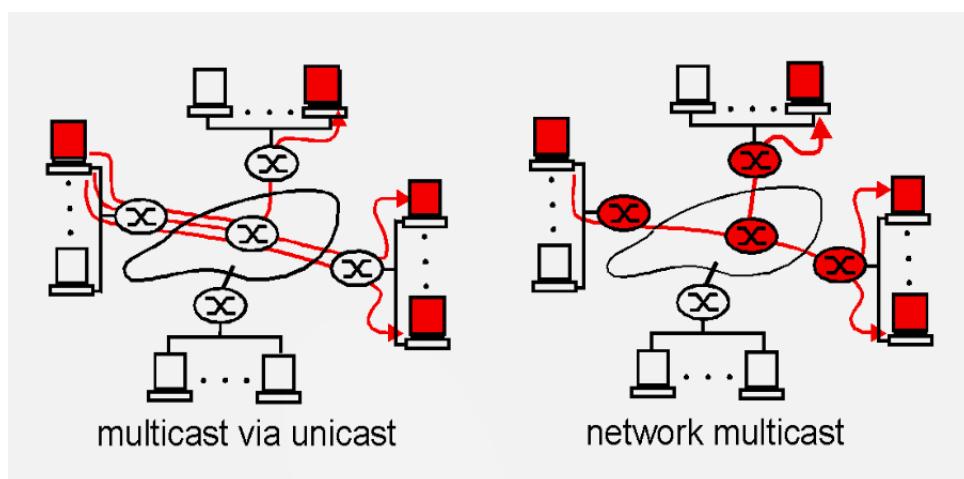
## 19.2 Instradamento gerarchico

L'istradamento gerarchico, organizza la rete in livelli gerarchici per gestire la complessità e migliorare la scalabilità. In BGP gli AS stabiliscono relazioni commerciali che definiscono come il traffico viene scambiato. Ci sono 3 attori principali caratterizzati da diverse relazioni

1. **Customer:** un AS che paga un provider per avere accesso a internet (es: università che paga Telecom).
  - Il provider esporta le routes del customer a tutti gli AS
  - Il customer esporta le routes del provider ad altri customer del provider
2. **Provider:** un AS che vende connettività ad un customer
3. **Peer:** due AS che scambiano traffico gratuitamente (peering settlement-free), per reciprocità (Telecom e Vodafone che si connettono direttamente per scambiare traffico dei rispettivi clienti)

## 20 Multicast

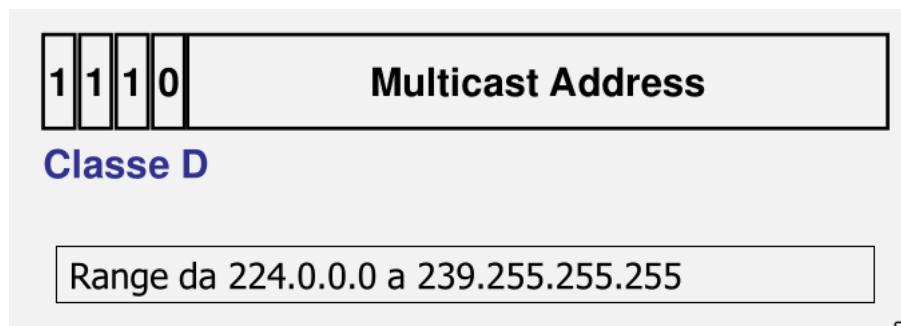
Per **multicast** si intende un'astrazione in cui l'invio di un pacchetto da un sender a molti receiver avviene con una singola operazione. Si pone di risolvere il problema di invio di pacchetti ad un gruppo specifico di destinatari



Sorge il problema di identificazione dei ricevitori di un datagramma multicast e l'invio del datagramma stesso. Questo avviene tramite **address indirection**, ossia si utilizza un identificativo unico per il gruppo di ricevitori e una copia del datagramma è inviata utilizzando tale identificativo a tutti i membri del gruppo

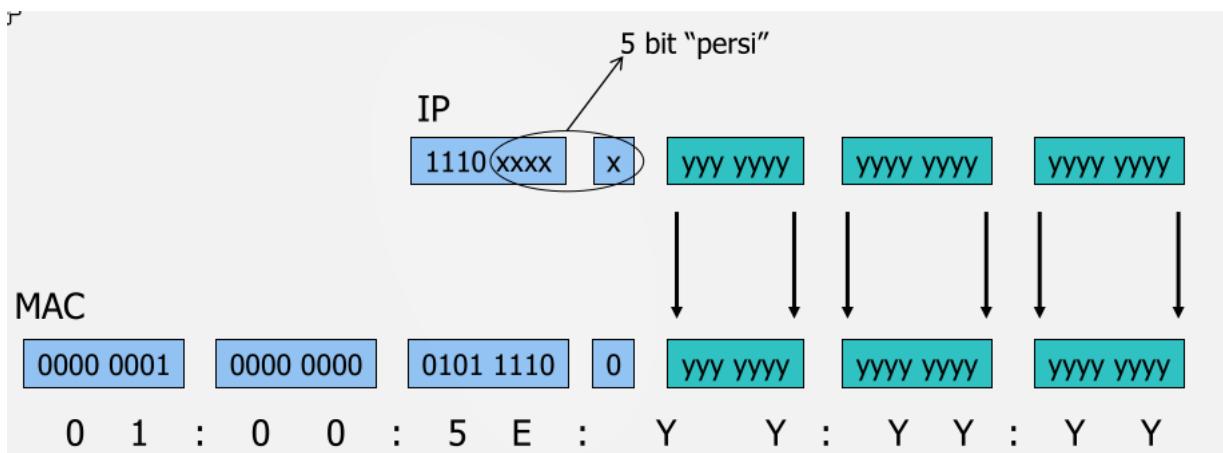
### 20.1 Indirizzo Multicast IPv4

Ad ogni gruppo è associato un indirizzo multicast, ossia un indirizzo IP di classe D



## 20.2 Multicast ethernet

Per la trasmissione di datagram IP multicast su reti LAN ethernet, occorre mappare un indirizzo in classe D su di un indirizzo MAC multicast. Questo non è possibile in maniera univoca, dato il range degli indirizzi MAC riservati al multicast (IP multicast 28 bit, MAC multicast 23 bit), dunque 5 bit vengono persi durante il processo di **aliasing**



## 20.3 Multicast Router

Il multicast router si occupa dello smistamento dei datagrammi multicast, ciò avviene nel seguente modo

- Ogni end system trasmette i datagrammi multicast sfruttando il meccanismo hardware messo a disposizione dalla rete locale su cui si trova
- Se un datagramma giunge al multicast router, quest'ultimo si occupa se necessario di instradarlo verso le altre reti

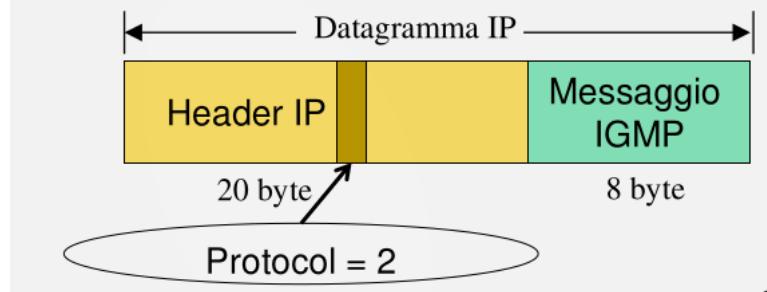
## 20.4 Protocolli multicast

Ci sono due protocolli principali per il multicast:

- **Internet Group Management Protocol (IGMP)**: fornisce ad un host i mezzi per informare il multicast router ad esso più vicino che un'applicazione vuole unirsi ad un determinato gruppo multicast
- **Algoritmi routing**: Coordinano i multicast router all'interno della rete Internet, per permettere l'instradamento dei datagrammi multicast

### 20.4.1 IGMP

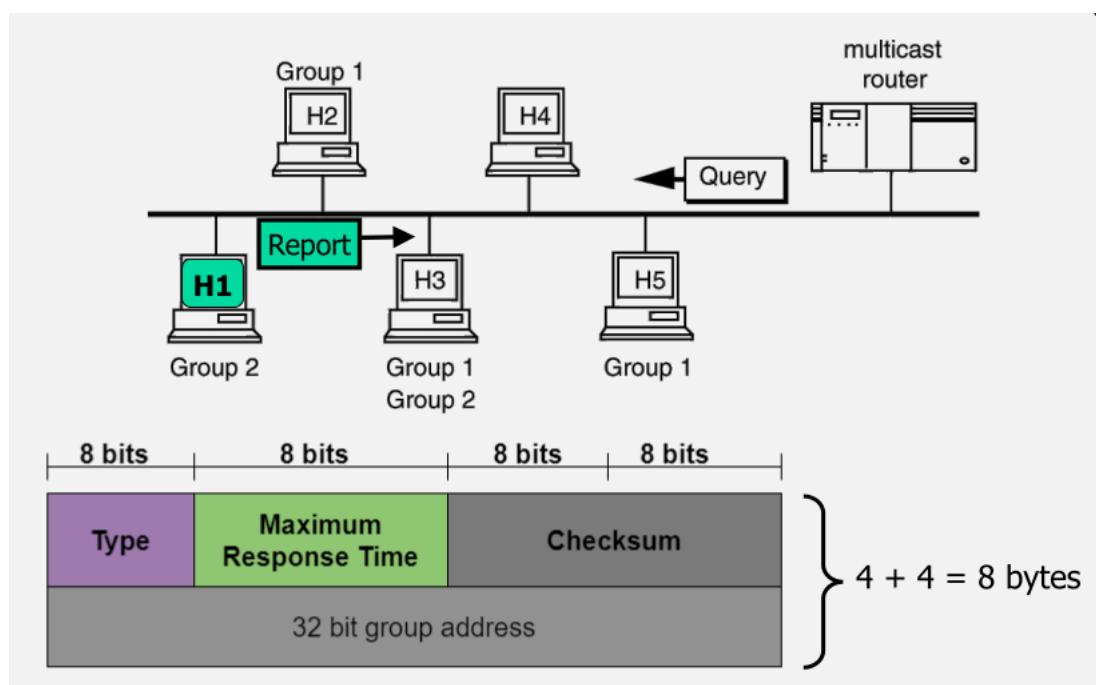
L'IGMP in tutte le sue versioni (IGMPv1, IGMPv2, IGMPv3) opera tra l'host ed il router ad esso direttamente collegato e serve a gestire in ambito locale le informazioni di membership ai vari gruppi multicast. I messaggi IGMP sono trasmessi in datagrammi IP il cui header ha il campo protocol con valore 2 ed il campo TTL con valore 1.



Quando un'applicazione chiede di entrare in un nuovo gruppo:

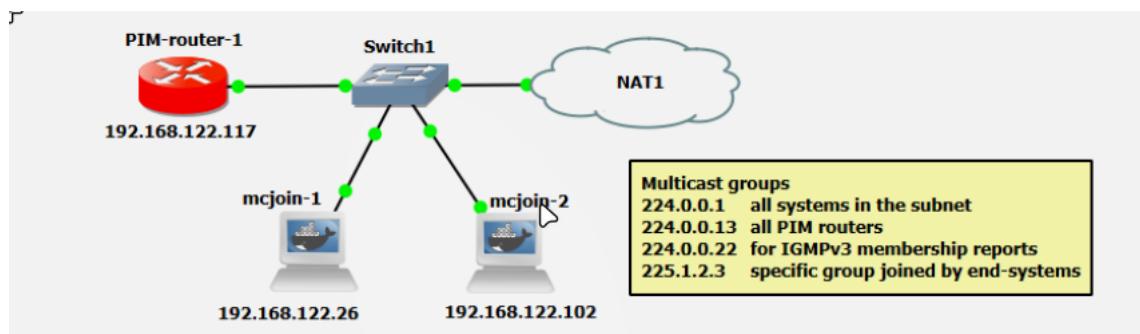
- l'host in cui gira l'applicazione invia un messaggio IGMP **membership report**, il messaggio è **unsolicited** in quanto non generato come risposta ad un messaggio di membership query.
- Gli eventuali router multicast presenti nella rete locale dell'host ricevono il messaggio e stabiliscono i meccanismi di routing atti a ricevere i pacchetti indirizzati al gruppo.

Dovendo gestire dinamicamente i gruppi, i multicast router, effettuano un'operazione di polling periodica, gli host presenti nelle reti locali collegate rispondono ai messaggi di *membership query* da parte del router con messaggi di *membership report*



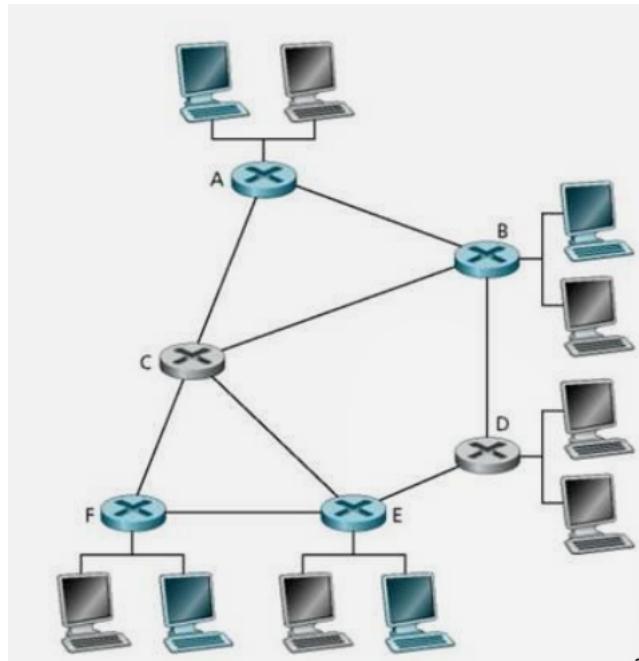
Un ulteriore protocollo di multicast è il **source specific multicast**, in cui un host può manifestare l'intenzione a ricevere pacchetti destinati ad un gruppo multicast esclusivamente da uno specifico host sorgente S.

- Avviene attraverso un'operazione di **source specific join** realizzata mediante messaggi IGMPv3 di source specific membership report



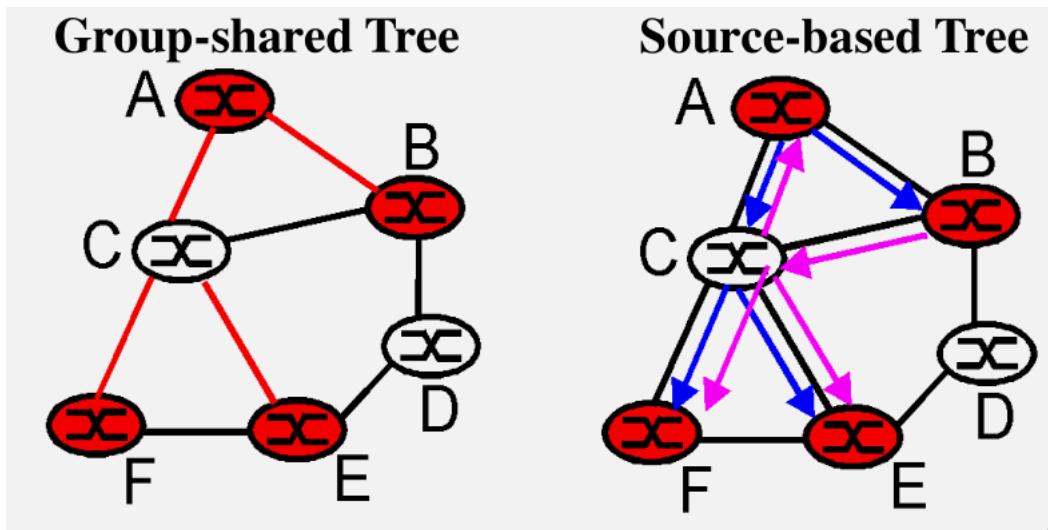
## 20.5 Routing Multicast

L'obiettivo dell'instradamento multicast è trovare un albero di link che colleghi tutti i router cui sono attaccati gli host che appartengono al gruppo multicast. Ovvivamente l'albero può contenere router che non hanno collegati host appartenenti al gruppo multicast.



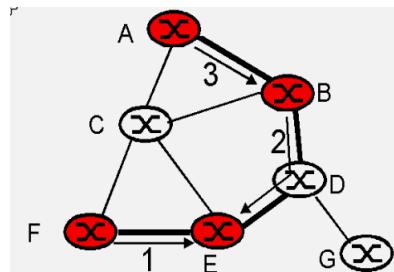
Ci sono due approcci possibili alla costruzione dell'albero di percorrenza.

- **Group Shared Tree:** unico albero condiviso dal gruppo per distribuire il traffico di tutti i mittenti
- **Source-based Tree:** Albero d'instradamento costruito specificamente per ciascun singolo mittente



### 20.5.1 Group-Shared Tree

Ci si pone il problema di risolvere lo **Steiner Tree Problem**, ossia trovare l'albero a costo minimo.



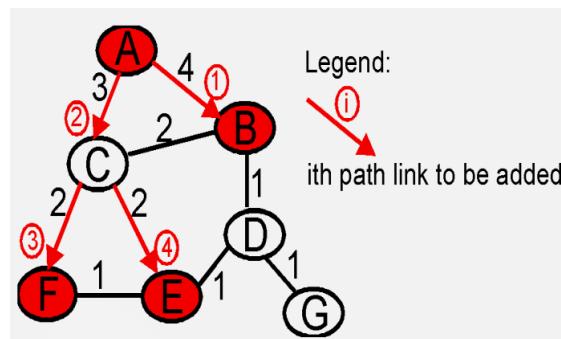
Si utilizza un approccio centralizzato definendo un nodo centrale come punto di **rendezvous/nucleo** (di solito il router con più connessioni adiacenti).

- Tutti i router cui sono collegati host multicast aderiscono al nucleo con messaggi unicast
- Il percorso seguito definisce il ramo dell'albero

Minimizza la somma dei costi dei link dell'albero multicast

### 20.5.2 Source-based Tree

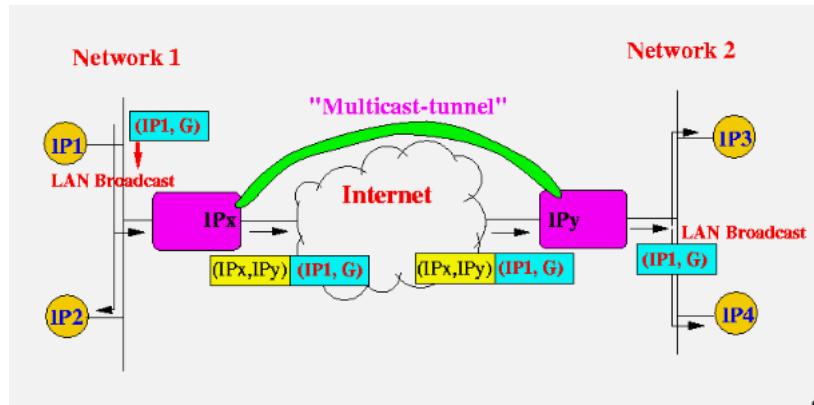
Si utilizza l'unione dei percorsi minimi dalla sorgente a tutte le destinazioni (**Least unicast-cost path tree**).



Minimizza il costo dalla sorgente ad ognuna delle destinazioni

## 21 MBone (rete)

La rete **Multicast BackBone** (MBONE), è una rete virtuale (overlay) composta da "isole" capaci di supportare il forwarding di pacchetti multicast IP collegate mediante link virtuali di tipo punto-punto chiamati **tunnel**.



I pacchetti multicast vengono incapsulati prima di essere trasmessi attraverso i tunnel, in modo da apparire all'esame dei router e delle sottoreti intermedie come normali datagrammi unicast. Un router multicast intenzionato a trasmettere un pacchetto all'altro capo di un tunnel deve aggiungere ad esso un ulteriore header IP in cui sia presente, come indirizzo destinazione, l'indirizzo unicast del router che si trova al capo opposto del tunnel.

## 22 IPv6

La versione IPv6 del protocollo IP, è stata progettata per risolvere alcuni dei problemi che affligono IPv4. principalmente:

- Indirizzamento e routing
- Sicurezza
- Configurazione automatica
- servizi real-time e supporto alla QoS
- elaborazione efficiente dei pacchetti da parte dei router
- coesistenza di IPv4 e IPv6

La motivazione principale che spinse alla progettazione di IPv6, fu l'esaurimento dello spazio di indirizzi supportati da IPv4.

### 22.1 Differenze

Ci sono 5 differenze principali tra il protocollo IPv6 e IPv4:

- **Espansione capacità di indirizzamento e routing:** la dimensione degli indirizzi passa da 32 bit a 128 bit, viene migliorata la scalabilità del routing multicast grazie all'aggiunta di un campo scope agli indirizzi di classe D e viene definito un nuovo tipo di indirizzo **anycast address** che si riferisce ad un insieme di interfacce
- **Semplificazione dell'header:** alcuni campi dell'header IPv4 sono eliminati o resi opzionali
- **Supporto per le opzioni migliorato:** alcuni cambiamenti nel modo di codificare le opzioni permettono uno smistamento più efficiente ed una maggior flessibilità, per introdurre nuove funzionalità

- **Supporto della QoS:** introdotta una funzionalità per etichettare i pacchetti appartenenti a flussi di dati particolari per i quali si chiede un trattamento di tipo non-default
- **Autenticazione e salvaguardia della privacy:** definizione di estensioni che forniscono il supporto per autenticazione, integrità dei dati, e la sicurezza.

## 22.2 Header

L'header IPv6 consiste di due parti:

- **header principale**
- **extension header** (introdotti per le nuove funzioni)

### 22.2.1 Header principale

L'header principale in IPv6 è lungo 40 bit ed è composto nel seguente modo

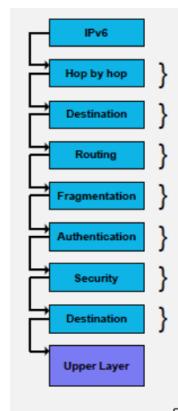
- **Version:** 4bit, numero della versione
- **Traffic Class:** 8bit, priorità del pacchetto
- **Flow Label:** 20bit, associa il pacchetto ad un flusso
- **Payload Length:** 16bit, lunghezza del payload (max 64KB)
- **Next Hdr:** 8bit, il tipo di header che segue il principale
- **Hop Limit:** 8bit
- **Source Address:** 128bit, indirizzo mittente
- **Destination Address:** 128bit, indirizzo destinazione



### 22.2.2 Extension Header

Gli extension header sostituiscono il campo options di IPv4, e permettono di specificare opzioni aggiuntive. Gli extension header se presenti, devono apparire in un ordine preciso

- **Hop by hop**
- **Routing**
- **Fragment**
- **Authentication**
- **Encrypted security payload**
- **Destination Option**



## 22.3 Indirizzi IPv6

Un indirizzo IPv6 è formato da 16 byte (128 bit), per la trascrizione degli indirizzi IPv6 si formano otto gruppi di 2 byte ciascuno in cui ciascun gruppo è identificato da cifre esadecimale. Ad esempio

2340:0023:AABA:0A01:0055:5054:9ABC:ABB0

Valgono due regole di semplificazione della notazione:

1. Una sequenza di zeri all'inizio di un gruppo di 4 cifre può essere omessa

2340:0023:AABA:0A01:0055:5054:9ABC:ABB0

2340:23:AABA:A01:55:5054:9ABC:ABB0

2. Una sequenza di campi successivi uguali a zero può essere rappresentata con il simbolo ":"

2340:0000:0000:0000:0055:5054:9ABC:ABB0

2340::55:5054:9ABC:ABB0

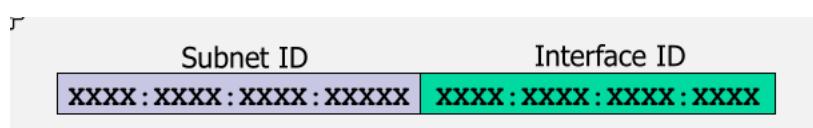
- **Unicast:** Trasmissione 1 ad 1
- **Multicast:** Trasmissione 1 a molti
- **Anycast:** Trasmissione 1 al più vicino (trasmette al più vicino in un gruppo di molteplici interfacce)

Come si nota, non è prevista la trasmissione broadcast

## 22.4 Indirizzi IPv6 Unicast

Gli indirizzi IPv6 unicast si suddividono a loro volta in 3 tipi principali:

- **Global Unicast:** simili agli indirizzi IPv4 pubblici, sono indirizzi pubblici univoci a livello mondiale. Utilizzati per internet o reti pubbliche. Hanno come prefisso **2000::/3**



- **Link Local:** si usano per l'invio di pacchetti nella sottorete locale, hanno come prefisso **FE80::/10**. Ogni interfaccia di rete su cui è abilitato IPv6 è configurata obbligatoriamente con un indirizzo link-local.
- **Unique Local:** simili agli indirizzi IPv4 privati, hanno come prefisso **FD00::/8**. Si usano in reti private, non sono inoltrati dai router internet ma possono essere inoltrati da router all'interno di una organizzazione.

Un ulteriore distinzione è quella sul contesto in cui operano gli indirizzi, infatti gli indirizzi IPv6 identificano due ambiti diversi:

- **Link:** comprende tutti i dispositivi in comunicazione diretta tramite una LAN o un collegamento punto-punto
- **Site:** comprende un insieme di link gestiti da un'unica entità (es: campus universitario)

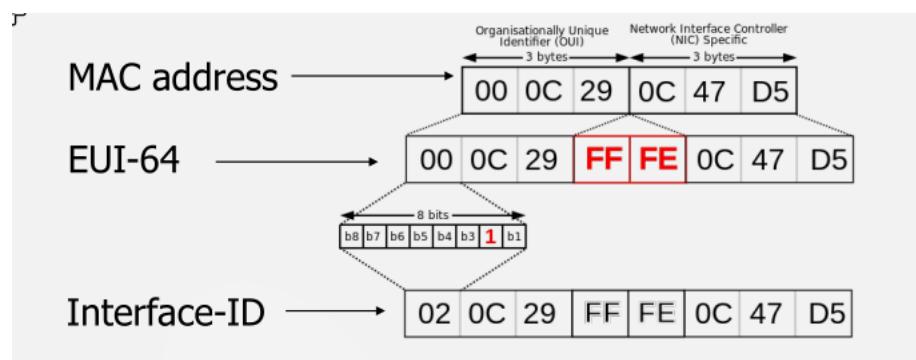
### 22.4.1 Indirizzi Link-Local

Caratterizzati dal prefisso **FE80::/10**, possono essere utilizzati esclusivamente per comunicare con altri host o router direttamente raggiungibili tramite un link. Ogni interfaccia di rete che supporti IPv6 deve essere configurata con almeno un IPv6 Link Local



Il metodo di produzione dell'interface ID varia a seconda del sistema operativo e in generale non vi è garanzia che un indirizzo link local sia globalmente univoco. Con RFC-2373 si definisce un modo per generare un identificativo a 64 bit (EUI-64 extended unique identifier) a partire dal MAC address della scheda di rete.

- Si ottiene inserendo la sequenza di 16 bit FF:FE tra i primi ed i secondi 24 bit del MAC address
- L'interface ID può essere calcolato a partire da EUI-64 invertendo il settimo bit da sinistra



### 22.4.2 Indirizzi IPv6 Global Unicast

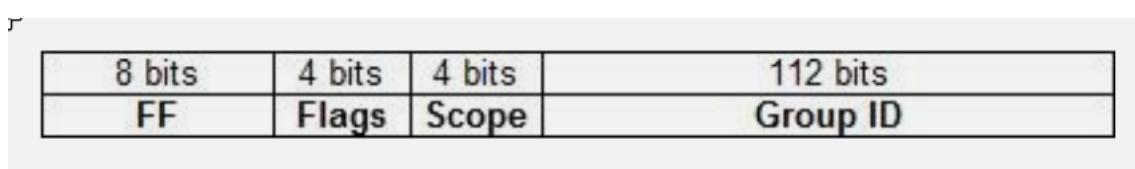
Identificati da un prefisso **2000::/3** identificano un'interfaccia di rete in maniera univoca a livello globale. Sono costituiti da

- Un **subnet ID** da 64 bit, che identifica la sottorete ottenuta aggregando identificatori più piccoli gerarchicamente (Top Level ID, Next Level ID, Site Level ID)
- Un **interface ID** da 64 bit, che identifica l'interfaccia nella sottorete



### 22.4.3 Indirizzi IPv6 Multicast

Iniziano per **FF::/8** e sono usati come indirizzo destinazione per datagrammi che devono essere inviati a gruppi di device



## 22.5 Assegnazione ad Host

Esistono diversi modi per attribuire un indirizzo IPv6 ad una interfaccia di host

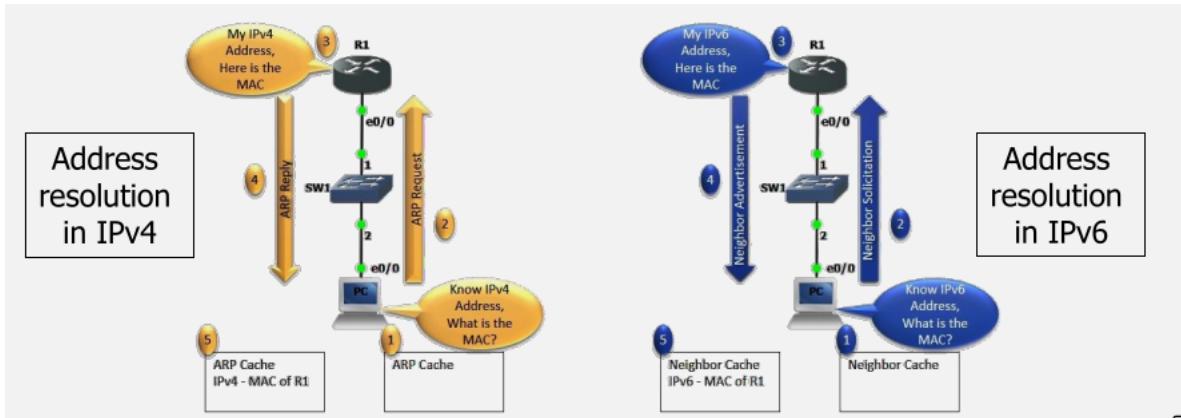
- **Static Address Configuration:** configurazione statica da parte dell'amministratore
- **Stateless Address-Auto Configuration (SLAAC):** un indirizzo IPv6 link local è attribuito automaticamente da parte del sistema operativo dell'host
- **Stateful DHCP address assignment:** configurazione dinamica con indirizzo assegnato da un server DHCPv6

## 22.6 ICMPv6

Si tratta di una variante per IPv6 del protocollo di controllo ICMP. Fornisce funzionalità di controllo, segnalazione errori e diagnostica, ma la vera innovazione risiede nelle funzionalità di **network discovery**:

- **RS - Router Solicitation:** usati da un host per sollecitare i router raggiungibili direttamente tramite le proprie interfacce a inviare un messaggio di tipo router advertisement
- **RA - Router Advertisement:** usati dai router per manifestare la propria presenza
- **NS - Neighbor Solicitation:** usati da host e router per determinare l'indirizzo link-layer di un vicino o per verificare che un vicino sia ancora raggiungibile tramite indirizzo link-layer in cache
- **NA - Neighbor Advertisement:** inviati in risposta ad un NS, o per annunciare un cambiamento di indirizzo link-layer
- **Redirect:** usati dai router per informare gli hosts dell'esistenza di un router migliore

L'address resolution in IPv6 non passa più attraverso il protocollo ARP ma si utilizza ICMPv6

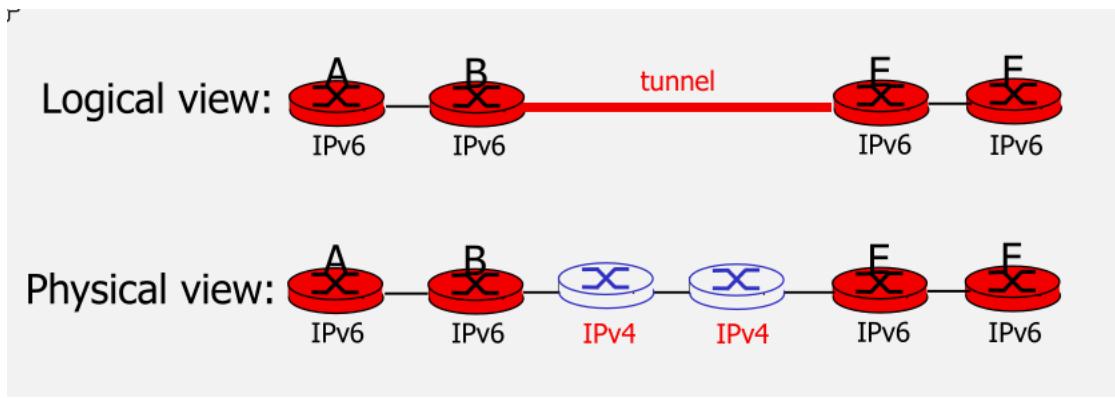


## 22.7 Mapping IPv6 - IPv4

Il mapping degli indirizzi IPv6 su IPv4 avviene attraverso la formazione di indirizzi IPv6 formati da un prefisso di 80 bit "0", 16 bit "1" e il successivo indirizzo IPv4 corrispondente. La notazione è mista, in quanto si usa una notazione esadecimale per la parte IPv6 e la classica dotted decimal per IPv4, dunque un indirizzo del genere ha la seguente forma

0:0:0:0:FFFF:w.x.y.z oppure ::FFFF.w.x.y.z

Per la transizione da IPv4 a IPv6 si introduce la nozione di **tunneling**. I pacchetti IPv6 vengono trasportati come payload all'interno di datagrammi IPv4 tra router IPv4



## 23 Multimedia (trasporto (trasporto))

Si vuole risolvere il problema di trasferire informazioni multimediali da una sorgente ad uno o più ricevitori attraverso una rete. Il processo è teoricamente semplice e composto da 3 step:

- il **trasmettitore** si occupa di effettuare una compressione secondo una opportuna tecnica (MPEG, MJPEG, MP3) per ridurre la quantità di informazioni da trasferire
- la **rete** trasmette l'informazione in pacchetti
- il **ricevitore** recupera l'informazione originaria dalla sequenza di pacchetti ricevuti mediante un'operazione inversa a quella di compressione e una successiva trasformazione in forma sonora o di video

C'è una necessaria differenza tra informazioni **live** e **pre-registrate**

- nelle informazioni live, l'informazione è prodotta e compressa mediante un apposito sistema prima della trasmissione
- nel caso di informazioni pre-registrate, l'informazione è già registrata in formato compresso e memorizzata su memoria di massa prima di effettuare la trasmissione

### 23.1 Live/Pre-registrate

#### 23.1.1 Pre-registrate

Esistono due modalità di fruizione

- **File-Transfer:** Avviene il trasferimento dell'intero file da sorgente a ricevitore e successivamente la riproduzione.
  - La riproduzione può iniziare solo al termine della trasmissione e necessita di capacità di memorizzazione adeguate
- **riproduzione progressiva:** La riproduzione avviene durante il trasferimento
  - Il ricevitore memorizza l'informazione in un buffer (**playout buffer**) che viene progressivamente alimentato e svuotato

#### 23.1.2 Live

Nel caso di informazioni live, la sorgente produce un flusso continuo di informazioni. Questo flusso di informazioni è spezzato in **pacchetti** che sono trasmessi individualmente sulla rete in un processo definito **streaming**. Lo streaming è molto sensile alla QoS:

- il ricevitore riesce a recuperare la continuità del flusso di pacchetti se tutti i pacchetti arrivano a destinazione con la stessa tempificazione relativa
- La rete può alterare la continuità temporale facendo occasionalmente perdere pacchetti o consegnando pacchetti con tempificazione diversa da quella in cui sono stati trasmessi (**jitter**)

Per arginare il problema rispetto alla perdita occasionale di pacchetti, ci si difende mediante tecniche di compressione robuste, per le quali l'informazione audio/video non è sensibilmente degradata quando occasionalmente si perde un pacchetto. Per limitare il problema del jitter, si adotta una strategia di bufferizzazione

## 24 RTP/RTCP (applicativo/trasporto)

Il trasferimento di informazioni pre-registrate avviene mediante file transfer realizzato attraverso HTTP/TCP. Per la trasmissione in streaming vengono invece adottate 2 tecniche, HTTP/TCP o un protocollo ad-hoc RTP su UDP.

### 24.1 RTP

Il **Real-Time Transport Protocol (RTP)** offre un servizio a livello di trasporto specificamente progettato per i requisiti di flussi multimediali. I pacchetti RTP sono incapsulati in datagrammi UDP

- è concepito per essere implementato a livello applicativo e non come uno strato aggiuntivo dello stack protocollare
- offre funzionalità minimali richieste dalla trasmissione di flussi continui tipici delle applicazioni multimediali
- è neutrale rispetto alla codifica utilizzata

Risolve il problema dello jitter. RTP fornisce informazioni di tempificazione (**timestamps**). Esistono due tipi di sincronizzazione

- **Sincronizzazione Intra-Media:** ricostruzione corretta della tempificazione della sequenza di pacchetti ricevuti
- **Sincronizzazione Inter-Media:** con lo scopo di mantenere sincronizzati flussi multimediali trasmessi separatamente (es. audio e video e lip-sync)

Supporta sia la trasmissione unicast che multicast e i suoi meccanismi sono scalabili rispetto al numero di appartenenti al multicast.

### 24.2 Struttura

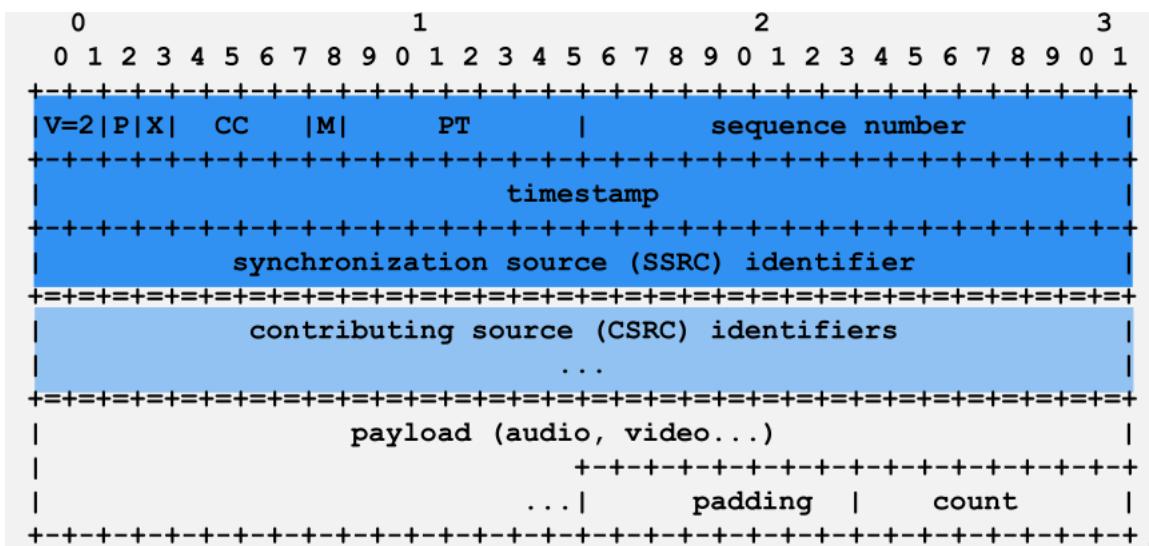
Un pacchetto RTP è trasmesso in un datagramma UDP



L'header UDP contiene i numeri di porto sorgente e destinazione, RTP usa numeri di porto destinazione pari per la trasmissione dei flussi di dati, lasciando il numero dispari successivo al protocollo RTCP per trasmettere informazioni di controllo relative a quel flusso.

#### 24.2.1 Header RTP

L'header RTP è formato nel seguente modo



- **Payload Type:** 7 bit, specifica la codifica utilizzata per i dati
- **Sequence Number:** 16 bit, serve ad identificare le perdite di pacchetti
- **Timestamps:** 32 bit, specifica il tempo di campionamento del primo byte del payload
- **Synchronization Source Identifier (SSRC):** 32 bit, identifica la sorgente del flusso, ed è scelto casualmente dalla sorgente stessa
- **Contributing Source Identifier (CSRC):** sequenza di n campi da 32 bit (n compresa tra 0 e 15), ciascuno dei quali identifica la sorgente originaria in un flusso prodotto dalla "fusione" di flussi diversi mediante un mixer software (es. in una videoconferenza SSRC identifica il mixer, CSRC indica lo speaker corrente)

### 24.3 Sessione RTP

Una associazione tra un gruppo di entità che comunicano mediante RTP è definita **Sessione**. Sessioni differenti vengono distinte da un recipiente mediante il port number di livello trasporto (UDP)

### 24.4 RTCP

Si tratta del protocollo utilizzato congiuntamente ad RTP per la trasmissione di informazioni di controllo. I pacchetti RTCP vengono inviati con una certa periodicità e trasportano informazioni di varia natura:

- feedback sulla qualità della ricezione dei dati
- identificazione dei partecipanti ad una sessione RTP

Il protocollo RTCP definisce cinque tipi diversi di messaggi:

- Source Report - SR
- Receiver Report - RR
- Source Description - SD
- BYE
- APP

I messaggi di tipo report contengono statistiche sul numero di pacchetti inviati, ricevuti... I messaggi di tipo description contengono informazioni sulla sorgente del flusso, BYE serve a notificare l'uscita da una sessione, APP permette di definire la sua funzione all'interno dell'applicazione

## 25 Trasmissione affidabile (trasporto)

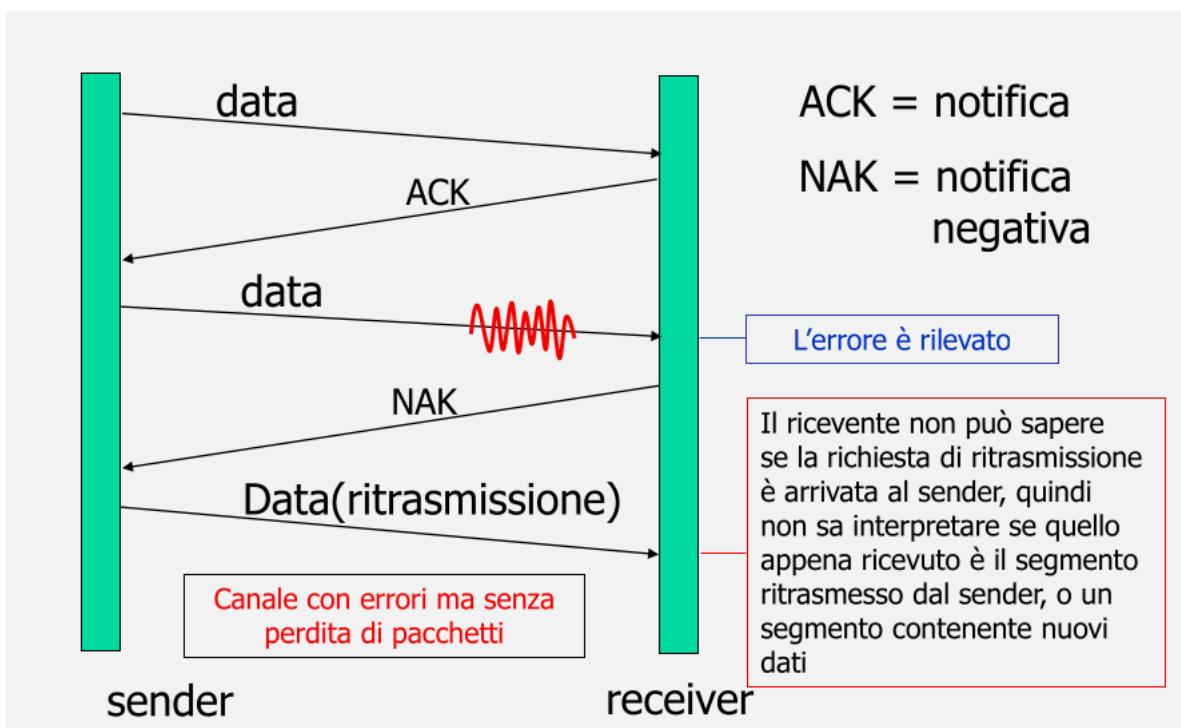
Se il livello rete è inaffidabile, nella comunicazione end-to-end si potrebbero verificare situazioni quali perdita di pacchetti, presenza di errori, ordine dei pacchetti non garantito ecc... È il livello di trasporto che si occupa di mitigare questi problemi.

### 25.1 Soluzioni

Nel caso in cui ci siano errori in trasmissione da parte della rete, il ricevente deve effettuare 2 step:

1. Rilevamento degli errori
2. Correzione degli errori o notifica al mittente richiesta di ritrasmissione

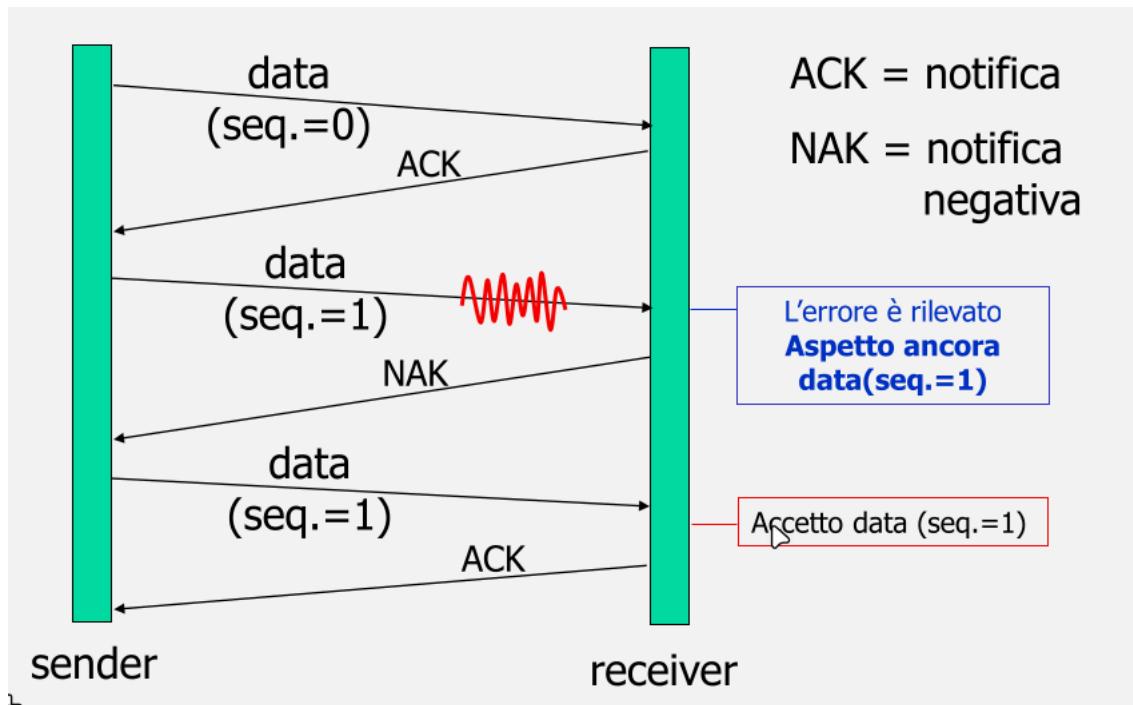
La prima soluzione introduce complicazioni, la seconda introduce possibili duplicazioni sulla rete che il ricevente non è in grado di interpretare.



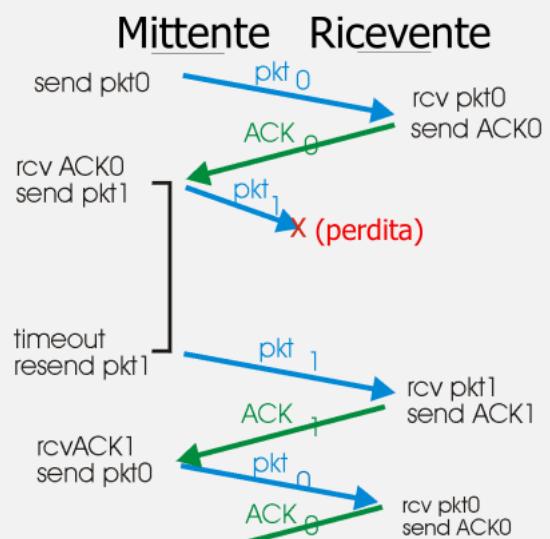
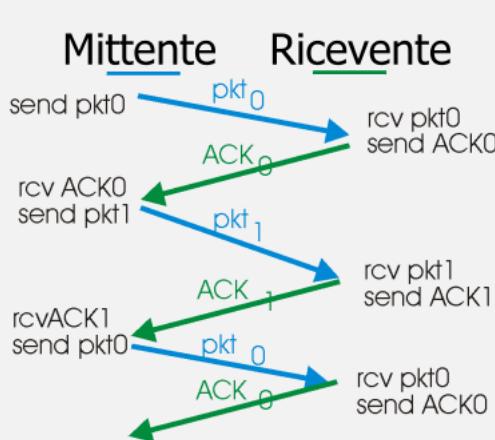
Per risolvere il problema dei duplicati, occorre inserire nell'header del segmento da inviare un **numero di sequenza**.

## 25.2 Stop and Wait

Si definiscono protocolli **Stop & Wait** quei protocolli che inviano un messaggio e attendono un riscontro prima di ritrasmettere un nuovo messaggio. Per questo tipo di protocolli è sufficiente un numero di sequenza ad un bit per realizzare una trasmissione affidabile.

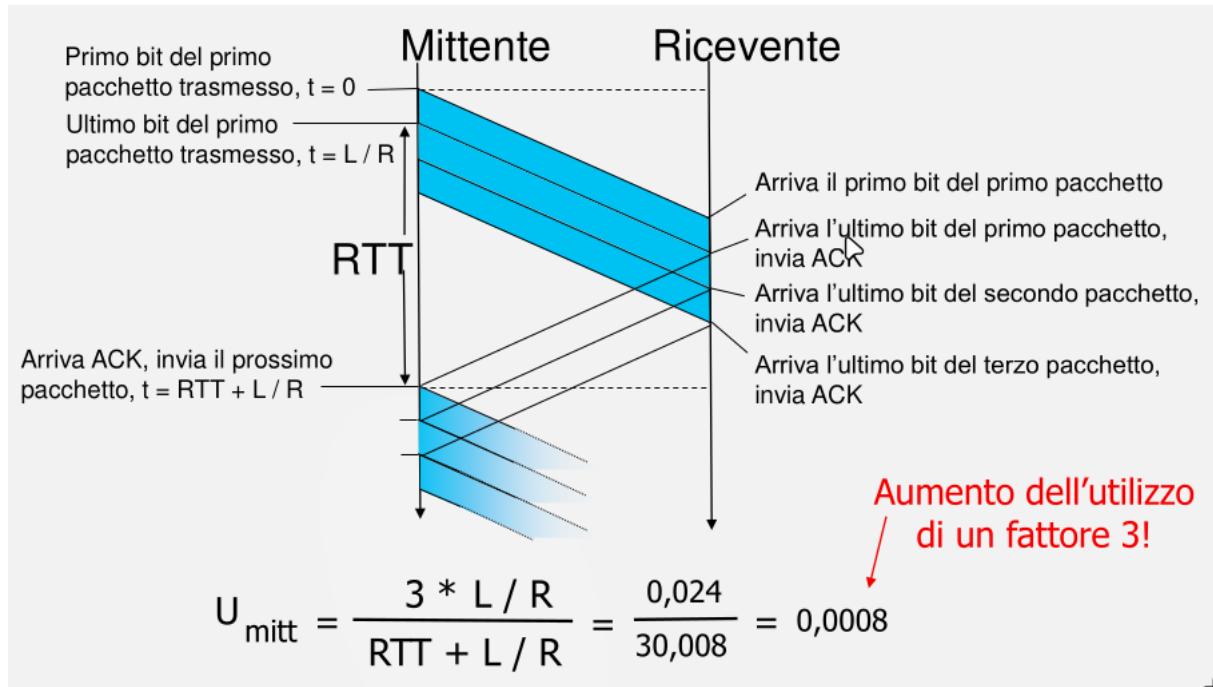


È possibile realizzare un protocollo del genere anche senza NAK, ma facendo in modo che il destinatario invii un ACK per l'ultimo pacchetto ricevuto correttamente che includa il numero di sequenza. Tuttavia in questo modo stiamo gestendo solo il caso dei duplicati, per gestire la perdita di pacchetti è necessario introdurre come parametro un **timeout**, allo scadere del quale, se non è stato ricevuto il segnale di ACK dal destinatario, il pacchetto viene reinviato.



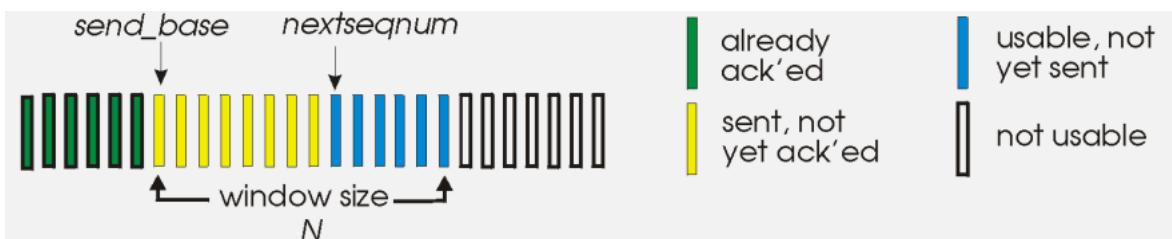
## 25.3 Pipelining

In alternativa allo stop and wait, si può utilizzare la tecnica del **Pipelining**, in cui il mittente invia pacchetti prima di ricevere il riscontro dei precedenti, e successivamente riceve i segnali di ack dei vari pacchetti da parte del destinatario. Ne esistono due versioni: **go-Back-N** e **Selective Repeat**

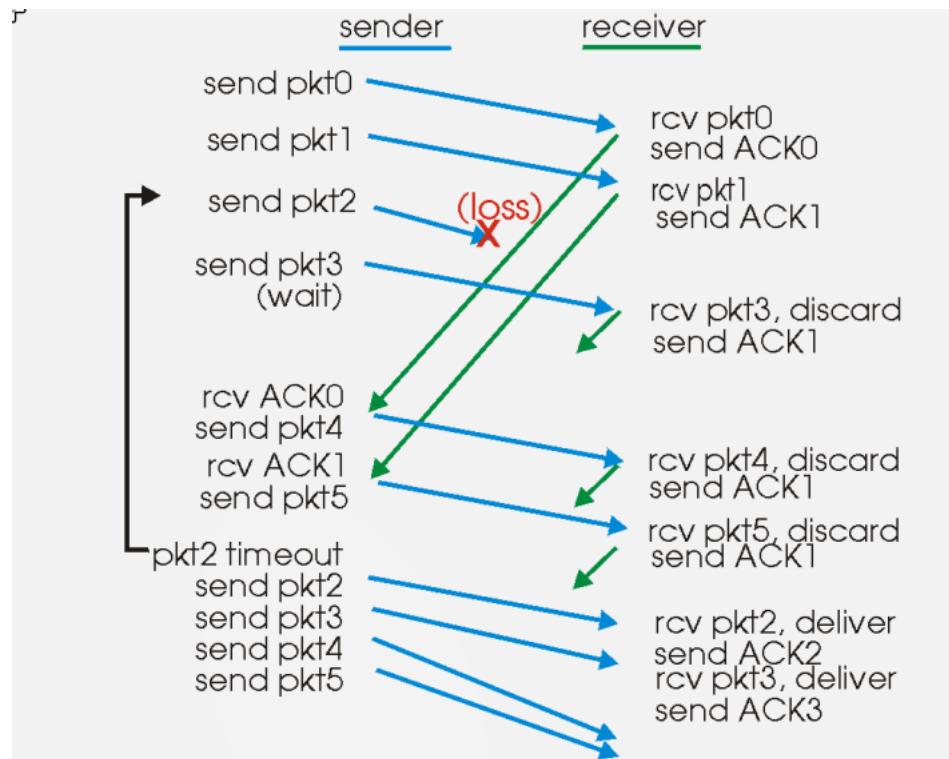


### 25.3.1 Go Back-N

Nel go back-n, vengono inseriti nell'header del segmento k bit per il numero di sequenza, successivamente, si stabilisce una finestra di max N pacchetti senza riscontro.

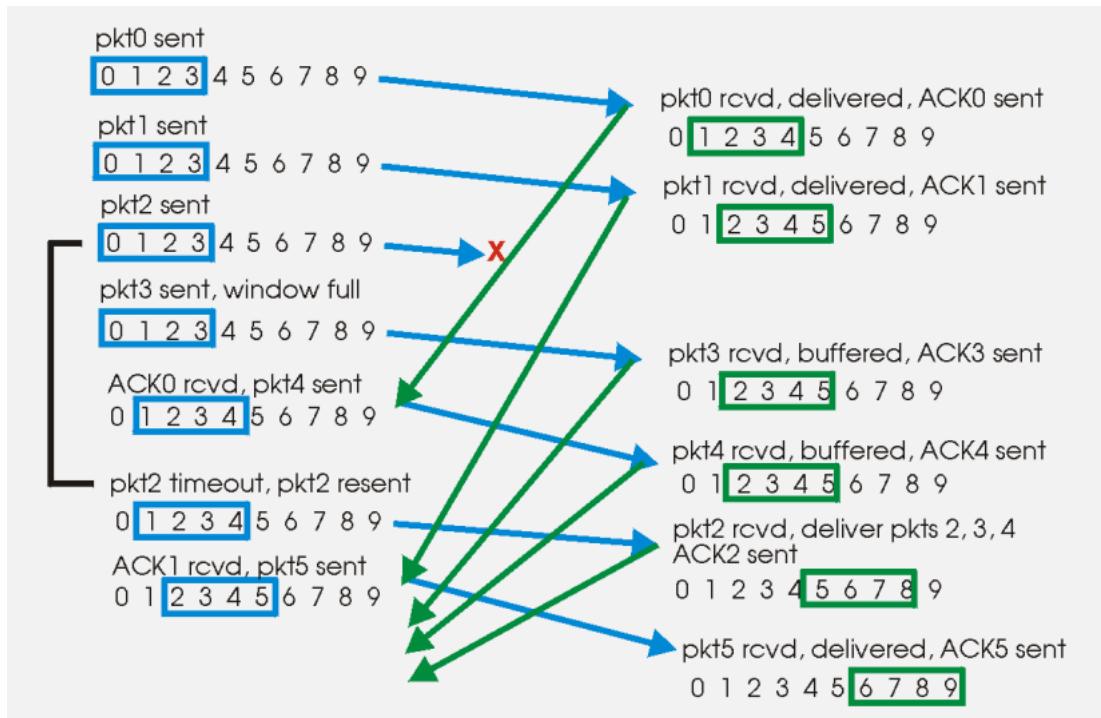


Gli ack ricevuti sono numerati e cumulativi, ciò implica che ricevere ACK(n) vuol dire che tutti i pacchetti precedenti all'ennesimo sono stati ricevuti correttamente.



### 25.3.2 Ripetizione selettiva

Il riceventi invia riscontri specifici per tutti i pacchetti ricevuti correttamente e il mittente ritrasmette soltanto i pacchetti per i quali non ha ricevuto un ACK.

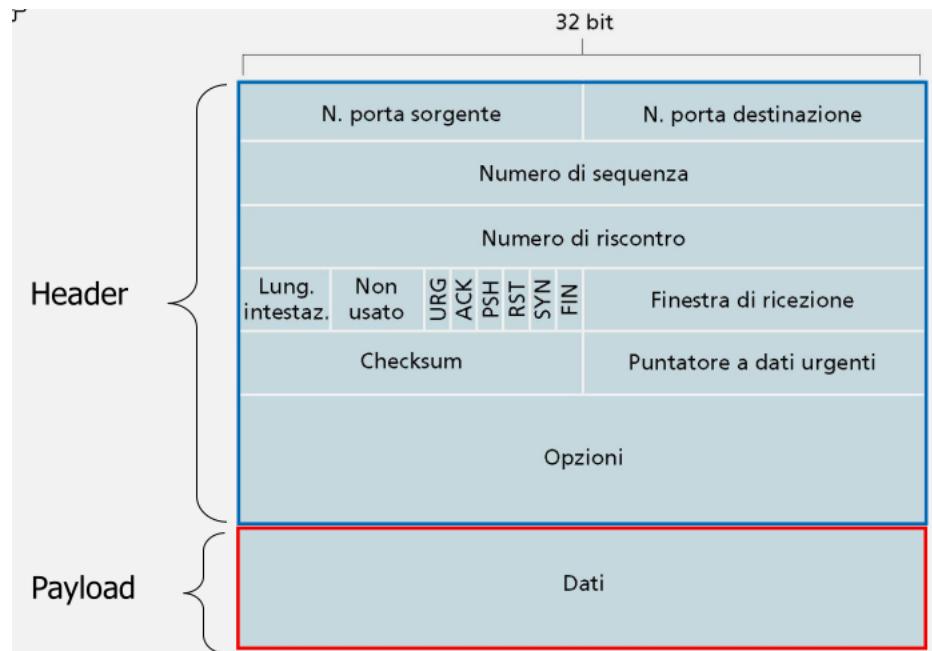


## 26 TCP

Il **Transmission Control Protocol (TCP)** è un protocollo di trasporto connection oriented che consente la trasmissione bidirezionale affidabile di un flusso di byte tra due endpoint.

## 26.1 Struttura

La struttura di un segmento TCP è la seguente

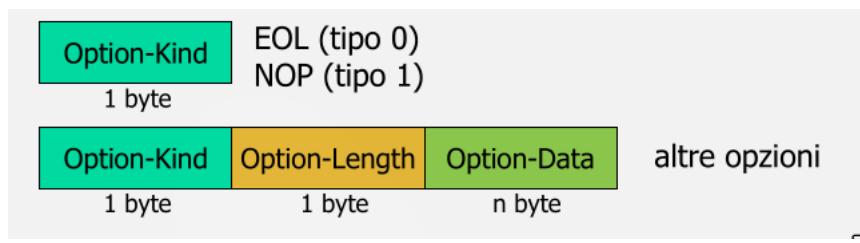


- **N. Porta Sorgente/Destinazione**
- **Lunghezza Header HLEN**: 4 bit, assume valori tra 5 e 15 (da 20byte a 60byte)
- **Numero di sequenza**: identifica nello stream di byte la posizione dei dati nel segmento
- **Numero di riscontro**: contiene il numero sequenziale del byte successivo a quello ricevuto correttamente dalla destinazione (valido solo negli ACK=1)
- **Flag**: per identificare il tipo di informazione contenuta nel segmento
- **Finestra di ricezione**: numero intero di 16 bit che specifica la dimensione del buffer che il TCP ha a disposizione per immagazzinare i dati in arrivo
- **Puntatore ai dati urgenti**: se flag URG=1, contiene il puntatore alla posizione nello stream dei dati non urgenti (ultimo byte dati urgenti)
- **Checksum**: Campo di 16 bit contenente un valore intero per verificare l'integrità del segmento

Alcuni segmenti speciali sono i segmenti con flag ACK, definiti **segmenti di riscontro**. Un segmento inviato da B ad A può contenere o meno dati relativi allo stream, e può contenere o meno dati di riscontro, se li contiene entrambi si definisce (**piggy-backing**).

### 26.1.1 Opzioni header TCP

Le opzioni sono di lunghezza variabile, ma la lunghezza totale dell'header TCP deve essere multipla di 4 byte. Ci sono due formati per le opzioni



Ci sono vari valori che può assumere il campo options dell'header TCP:

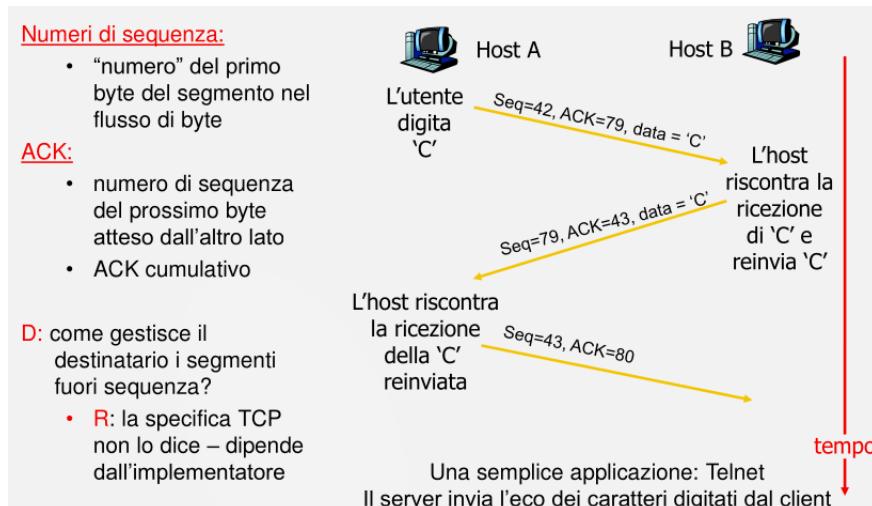
- **MSS**: durante la fase di connessione, ciascun end-point annuncia la massima dimensione di payload (MSS - Maximum Segment Size) che desidera accettare
- **Window Scale**: Per negoziare un fattore di scala per la finestra
- **Selective Repeat**: Nel caso in cui un segmento corrotto sia stato seguito da segmenti corretti, introduce i NAK per permettere al receiver di richiedere la ritrasmissione di quello specifico segmento
- **Timestamps**: utilizza i timestamps per aiutare i due endpoint a determinare il Round Trip Time
- **NOP**: No-Operations, separa opzioni diverse quando non lineate su multipli di lunghezza di quattro byte
- **EOL**: End of Options List, indica la fine delle opzioni necessario se la fine delle opzioni non coincide con la fine dell'header TCP

## 26.2 Affidabilità

Per mantenere standard di affidabilità, TCP si basa su

- **Riscontro e trasmissione**: consiste nella ritrasmissione di un segmento se non è giunta conferma entro un tempo massimo (time out)
- **Time-Out**: Al momento della trasmissione di un segmento, il TCP attiva un timer

TCP vede i dati come un flusso di byte non strutturati ma ordinate in base a **numeri di sequenza progressivi**. Se consideriamo un flusso di 500.000 byte, con MSS 1000 byte, ci saranno 500 segmenti con numeri di sequenza 1000, 2000, ... All'inverso il **numero di riscontro** contiene il numero di sequenza del byte successivo a quello ricevuto correttamente dalla destinazione



Per quanto riguarda il secondo meccanismo, ossia il meccanismo di Time-Out, il problema è a che valore impostare il timeout? Il problema sta nello stimare RTT correttamente. La stima di RTT avviene tramite media esponenziale pesata dei campioni e la stima della deviazione di RTT.

- Valore tipico per  $\alpha$ : 0.125  

$$\text{EstimatedRTT} = (1-\alpha) * \text{EstimatedRTT} + \alpha * \text{SampleRTT}$$
- **Stima della deviazione di RTT:**  

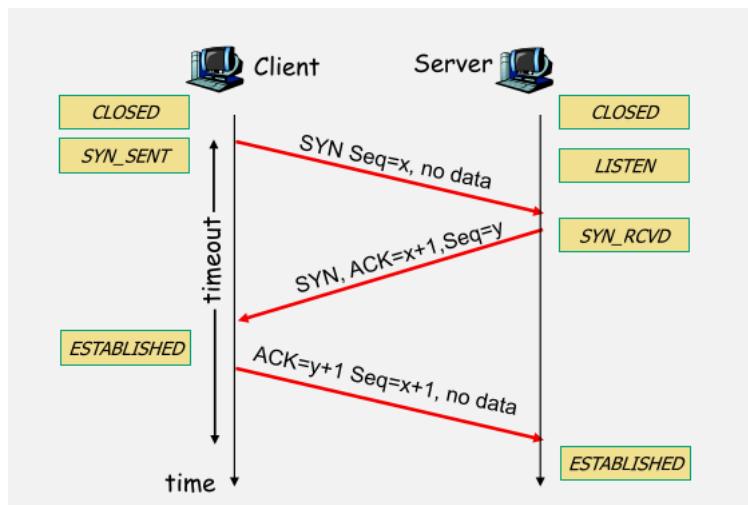
$$\text{DevRTT} = (1-\beta) * \text{DevRTT} + \beta * |\text{SampleRTT} - \text{EstimatedRTT}|$$
  - Valore raccomandato per  $\beta$ : 0,25

il valore del timeout è dato da

$$\text{Timeout} = \text{EsitamatedRTT} + 4 * \text{DevRTT}$$

## 26.3 Connection Establishment

La procedura per stabilire una connessione è detta **Three-way-handshake**. Ed è sintetizzata dal seguente schema



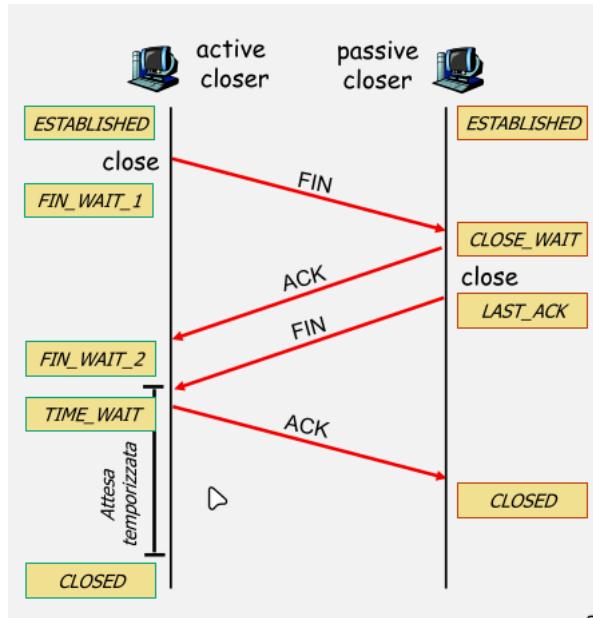
Nella fase three-way handshake, i due endpoint allocano i buffer per i dati e concordano sui valori iniziali dei numeri di sequenza da utilizzare per gli stream dati in entrambi i versi:

1. client invia il segmento di controllo TCP SYN al server
2. cerver riceve SYN e risponde con segmento di controllo SYN/ACK
3. client riceve SYN/ACK e invia ACK al server

In seguito a questo scambio, viene instaurata la connessione, in cui le due parti apprendono il valore iniziale di Recv Window e utilizzano le opzioni TCP per determinare il valore di MSS, RTT ecc... Per quanto riguarda la connection termination, si utilizza la procedura **four-way-handshake** in cui la parte che manifesta la volontà di chiudere la trasmissione si chiama **active closer** e l'altra parte **passive closer**, la procedura segue questi step

1. l'active closer invia un segmento FIN
2. il passive closer invia un segmento ACK/FIN

3. l'active closer attende in uno stato TIME\_WAIT tipicamente di 120s
4. l'active closer invia una ACK di chiusura al passive closer e al termine della TIME\_WAIT chiude



## 26.4 Schema riassuntivo ACK

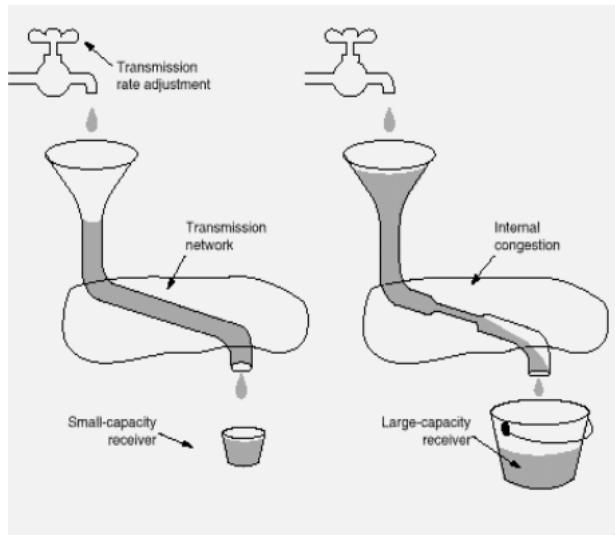
Di seguito uno schema che riassume gli ACK da cosa vengono generati e le azioni conseguenti

Evento nel destinatario	Azione del ricevente TCP
Arrivo ordinato di un segmento con numero di sequenza atteso. Tutti i dati fino al numero di sequenza atteso sono già stati riscontrati.	ACK ritardato. Attende fino a 500 ms l'arrivo del prossimo segmento. Se il segmento non arriva, invia un ACK.
Arrivo ordinato di un segmento con numero di sequenza atteso. Un altro segmento è in attesa di trasmissione dell'ACK.	Invia immediatamente un singolo ACK cumulativo, riscontrando entrambi i segmenti ordinati.
Arrivo non ordinato di un segmento con numero di sequenza superiore a quello atteso. Viene rilevato un buco.	Invia immediatamente un ACK (duplicato), indicando il numero di sequenza del prossimo byte atteso.
Arrivo di un segmento che colma parzialmente o completamente il buco.	Invia immediatamente un ACK, ammesso che il segmento cominci all'estremità inferiore del buco.

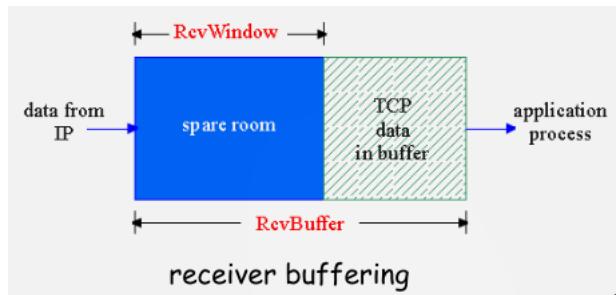
## 26.5 Flow Control

Il controllo di flusso è necessario per risolvere i problemi di sincronizzazione tra invio e lettura di due endpoint. Ci possono essere due bottleneck

- **Receiver Window:** dimensione del buffer di ricezione troppo piccola
- **Congestion Window:** congestione all'interno della rete



Il **receiver buffer** è strutturato così



Il mittente non dovrà sovraccaricare il ricevente inviando dati nel **receiver buffer** ad una velocità troppo elevata ciò avviene nel seguente modo

- **ricevente**: comunica dinamicamente al mittente la dimensione corrente del buffer nel campo **RcvWindow**

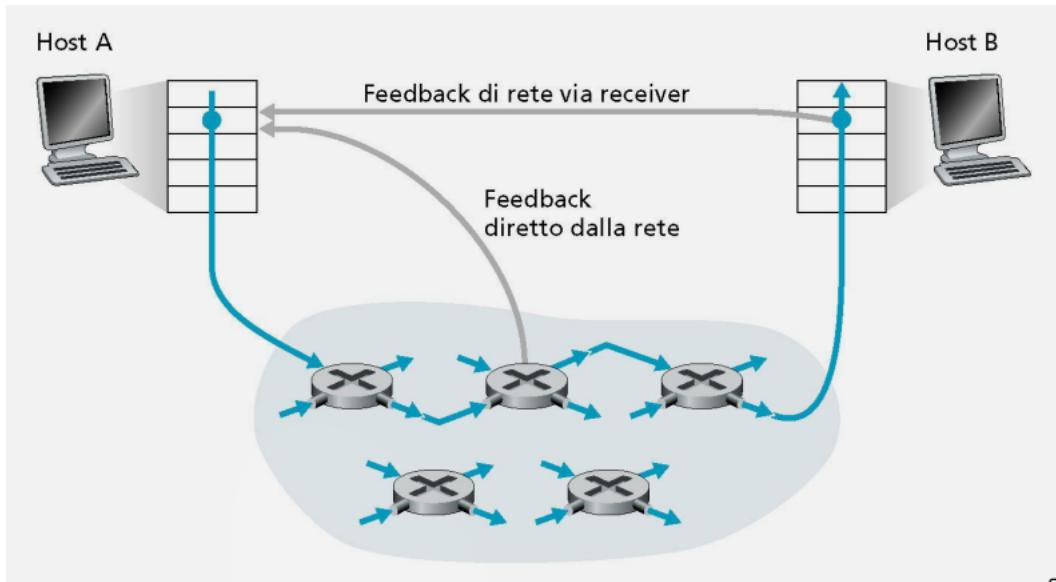
$$\text{RcvWindow} = \text{RcvBuffer} - [\text{LastByteRcvd} - \text{LastByteRead}]$$

- **mittente**: conserva i dati già trasmessi ma non riscontrati e limita tale quantità all'ultima RcvWindow ricevuta

## 26.6 Congestion Control

In caso di un numero elevato di sorgenti di traffico, sorgenti che inviano troppi dati o traffico inviato a frequenza troppo elevata, la rete può andare in contro a un sovraccarico. Questa situazione si definisce come **congestione della rete**. Ci sono due approcci per mitigare questo problema

- **Approccio end-to-end**: non prevede segnalazione esplicita dalla rete, ma gli end-systems a partire dall'osservazione dei ritardi e perdite di pacchetti deducono uno stato di congestione (approccio utilizzato in TCP)
- **Approccio a segnalazione della rete**: i router forniscono informazioni circa lo stato della rete agli end-system



### 26.6.1 TCP congestion control

Il protocollo TCP implementa un controllo end-to-end a frequenza di trasmissione variabile



La frequenza di trasmissione dipende dalla **CongWin**, ossia la finestra di congestione. Il risultato considerando il controllo di flusso e controllo di congestione è

$$\text{LastByteSent} - \text{LastByteAcked} \leq \min\{\text{RcvWindow}, \text{CongWin}\}$$

Si procede per tentativi per stabilire quanti si può trasmettere, con l'obiettivo di trasmettere alla massima velocità possibile senza perdite. L'approccio utilizzato è

- Incrementare CongWin finché non si verifica la perdita di un segmento
- In seguito alla perdita di un segmento si decrementa CongWin e si ricomincia da capo

Le fasi di controllo della gestione si definiscono

- **Slow Start**: si incrementa esponenzialmente la dimensione della CongWin ad ogni RTT fino al verificarsi di un evento di perdita (timeout, tre ACK duplicati consecutivi) e si ritorna a CongWin = 1 MSS
- **Congestion Avoidance**: una volta raggiunta la soglia si entra in questa fase in cui ci si avvicina con cautela al valore della banda disponibile tra le due estremità della connessione incrementando CongWin di

$$\text{CongWin} = \text{CongWin} + \text{MSS} (\text{MSS}/\text{CongWin})$$

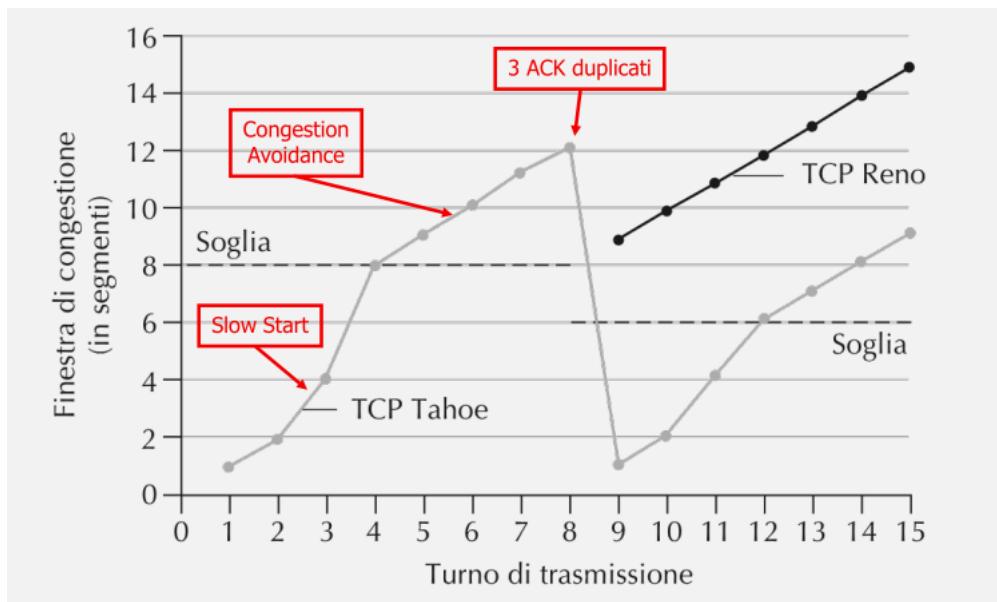
Al sopraggiungere della congestione, se

- scade un timeout: slow start

- tre ACK duplicati consecutivi (TCP Tahoe): slow start
- tre ACK duplicati consecutivi (TCP Reno): fast recovery

Il concetto di **Fast Recovery** è implementato in TCP Reno ed elimina la fase di slow start dopo la ricezione di 3 ACK duplicati. Tale evento indica che nonostante sia perso un pacchetto, almeno 3 segmenti successivi sono stati ricevuti dal destinatario, dunque la rete è in grado di consegnare una certa quantità di dati. La CongWin viene dunque aumentata come

$$\text{Threshold} = \text{CongWin}/2, \text{CongWin} = \text{Threshold} + 3\text{MSS}$$



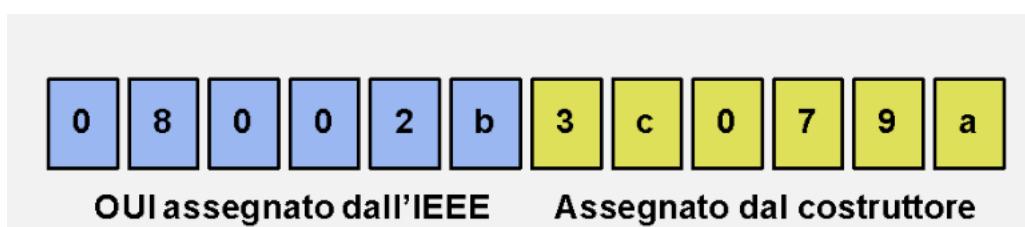
## 27 Ethernet (data link)

Iniziamo sancendo in modo netto la differenza tra indirizzi IP a 32 bit e indirizzi LAN

- **Indirizzi IP:** sfruttati da parte del layer 3 rete per permettere la corretta consegna del pacchetto ad un destinatario collegato alla rete
- **Indirizzi LAN (o MAC o fisici):** usati per permettere la trasmissione di un frame da una scheda di rete ad un'altra con cui sussiste un collegamento diretto (stessa rete fisica). Si tratta di indirizzi MAC di 48 bit per la maggior parte delle LAN cablati nella ROM delle schede di rete

### 27.1 Indirizzi LAN

La distribuzione degli indirizzi MAC è gestita dall'IEEE e i costruttori di schede di rete detengono una porzione degli indirizzi MAC per garantirne l'univocità. Gli indirizzi MAC si compongono di 2 parti grandi 3 byte ciascuna:



I tre byte più significativi indicano il lotto di indirizzi acquistato dall'IEEE e si definisce **Organization Unique Identifier (OUI)**. Gli ultimi 3 byte vengono invece assegnati in base a numerazione progressiva dal costruttore. Gli indirizzi MAC si dividono in 3 tipi

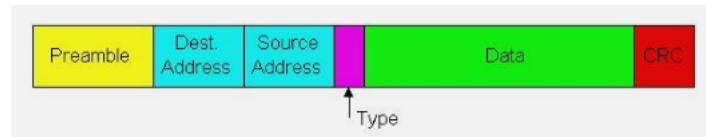
- **single**: di una singola stazione
- **multicast**: di un gruppo di stazioni
- **broadcast**: di tutte le stazioni (ff-ff-ff-ff-ff-ff)

Ogni scheda di rete, quando riceve un pacchetto lo passa ai livelli superiori nei seguenti casi

- broadcast: sempre
- multicast: se ne è stata abilitata la ricezione via software
- single: se il DSAP è uguale a quello hardware della scheda o a quello caricato da software in un apposito buffer

## 27.2 Ethernet

L'Ethernet è la tecnologia dominante per la LAN dato il costo ridotto, vediamo come opera al livello 2. L'interfaccia di rete del mittente incapsula i datagrammi IP in **frame ethernet** strutturati nel seguente modo



- **Preamble (8 byte)**: utilizzato per sincronizzare i clock di mittente e destinatario
- **Indirizzi (6 byte)**: la frame è ricevuta da tutti gli adattatori di rete presenti sulla LAN e scartata se l'indirizzo non coincide con quello della scheda stessa
- **Type (2 byte)**: indica il protocollo di rete sovrastante, principalmente IP
- **CRC (4 byte)**: per il controllo effettuato alla destinazione

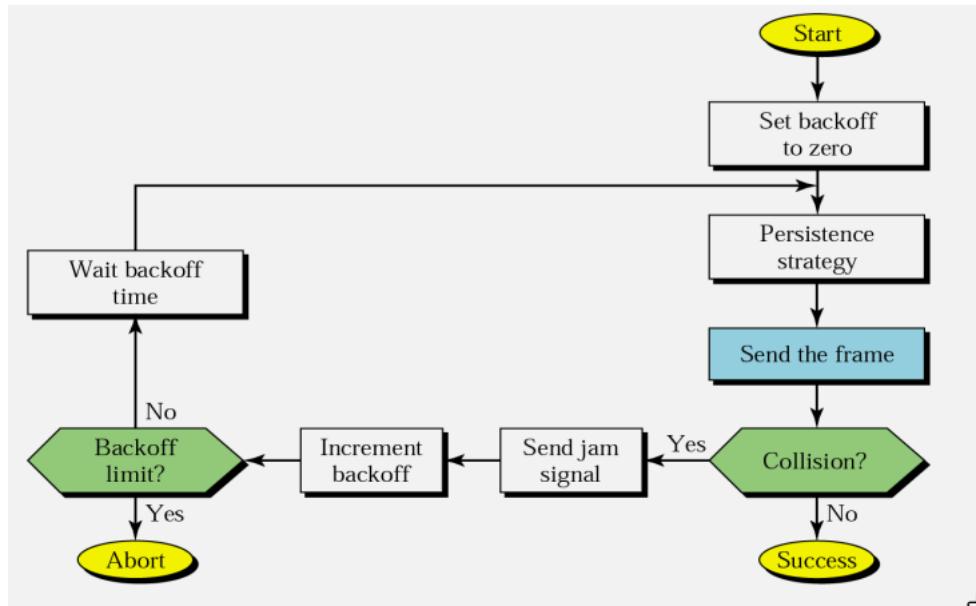
Ogni frame viene trasmessa dopo un **inter frame gap** corrispondente al tempo di trasmissione di 12 Byte. La dimensione minima di una frame Ethernet = 64 byte, 14 Byte di header + un minimo di 46 Byte per la PDU e 4 Byte per la CRC. Tuttavia in trasmissione considerando il Preamble di 8 byte e la interframe gap di 12 si ha una dimensione minima di 84 byte. Con dimensione del PDU massima si ha una dimensione massima di 1538 Byte.

### 27.2.1 CSMA/CD

Il **Carrier Sense Multiple Access w Collision Detection (CSMA/CD)** è un protocollo di accesso multiplo utilizzato nelle reti Ethernet per gestire la trasmissione dei dati su un mezzo condiviso. Lo scopo principale è permettere a più dispositivi di condividere lo stesso canale di comunicazione evitando collisioni o gestendole se si verificano. Opera su 3 attori principali

- **Jam Signal**: consente alle altre stazioni di accorgersi dell'avvenuta collisione (48 bit)
- **Exponential Backoff**: un algoritmo per adattare i successivi tentativi di ri-trasmissione al carico corrente della rete
- **Segnale**: segnale in codifica Manchester

L'algoritmo di backoff fa sì che dopo  $n$  collisioni il dispositivo attende un tempo casuale compreso tra 0 e  $2 - 1$  "slot time". Lo slot time è il tempo minimo per trasmettere 512 bit, dopo 10 collisioni consecutive l'intervallo si stabilizza a un limite massimo di 1023 slot.



## 27.3 Standard Ethernet Storici e Moderni

Gli standard Ethernet si differenziano per velocità, tipo di cavo e topologia. Di seguito i principali:

- **10BASE5 ("Thick Ethernet"):**
  - Cavo coassiale spesso (10 mm), **10 Mbps**, lunghezza massima **500 m**.
  - Topologia a bus, con connessioni a "vampiro" (transceiver agganciati al cavo).
  - Obsoleto, utilizzato negli anni '80.
- **10BASE2 ("Thin Ethernet"):**
  - Cavo coassiale sottile (5 mm), **10 Mbps**, lunghezza massima **185 m**.
  - Connettori BNC a baionetta, topologia a bus senza hub.
  - Più economico del 10BASE5, ma sostituito dal twisted pair.
- **10BASE-T (IEEE 802.3i):**
  - Cavo UTP Cat3, **10 Mbps**, lunghezza massima **100 m**.
  - Topologia a stella con hub, connettori RJ-45.
  - Rivoluzionario per la semplicità di installazione (anni '90).
- **100BASE-TX ("Fast Ethernet"):**
  - Cavo UTP Cat5 o superiore, **100 Mbps**, stessa lunghezza (100 m).
  - Utilizza 2 coppie twisted pair, compatibile con il formato frame 10BASE-T.
  - Dominante negli anni 2000, ancora utilizzato per dispositivi legacy.

## 27.4 Differenze tra Cavi UTP

I cavi UTP (Unshielded Twisted Pair) sono classificati in categorie in base a velocità e frequenza:

Categoria	Banda	Max Velocità	Applicazioni
Cat3	16 MHz	10 Mbps	10BASE-T (obsoleto)
Cat5	100 MHz	100 Mbps	100BASE-TX, Fast Ethernet
Cat5e	100 MHz	1 Gbps	Gigabit Ethernet (base)
Cat6	250 MHz	10 Gbps	Gigabit/10G (fino a 55 m)
Cat6a	500 MHz	10 Gbps	10 Gigabit Ethernet (100 m)

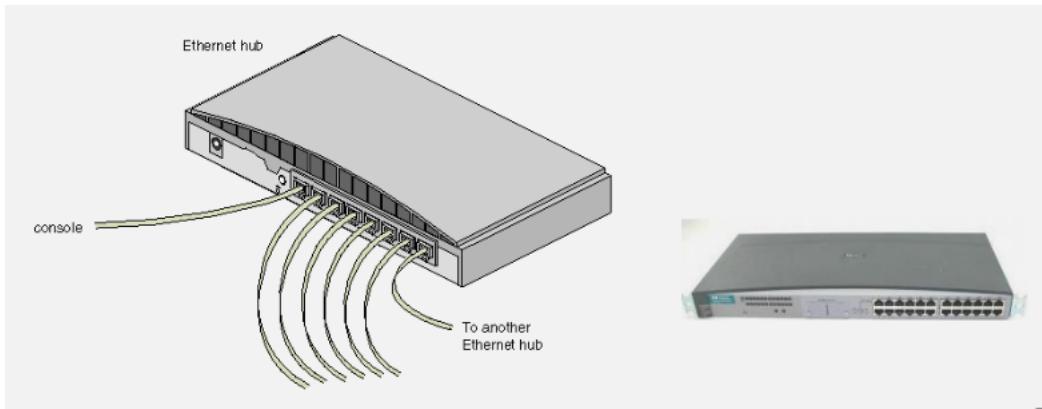
- **Shielding:** I cavi UTP non hanno schermatura (a differenza di STP o FTP), ma usano torsioni delle coppie per ridurre le interferenze.
- **Compatibilità:** Cavi di categoria superiore sono retrocompatibili (es. Cat6 può sostituire Cat5).
- **Distanza:** Tutte le categorie supportano fino a 100 m, tranne Cat6 per 10 Gbps (limitato a 55 m).

## 28 Dispositivi LAN (data link)

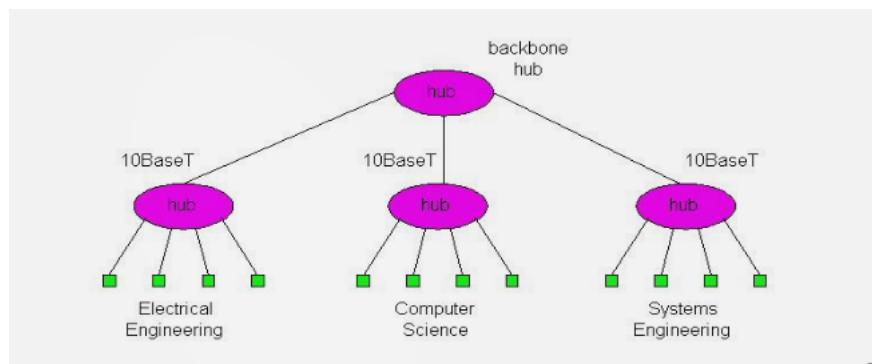
### 28.1 Hub

Gli hub sono dispositivi di livello fisico che fungono sostanzialmente da ripetitori di bit.

- riproducono i bit in ingresso ad un'interfaccia su tutte le altre interfacce



Gli hub sono organizzati in una gerarchia **multi-livello** con un backbone hub al livello più alto. Ogni LAN collegata si dice **segmento di LAN**



Interconnessione tramite hub ha sia pro:

- L'organizzazione multi livello grantisce una parziale tolleranza ai guasti
- Si estende la massima distanza tra i nodi

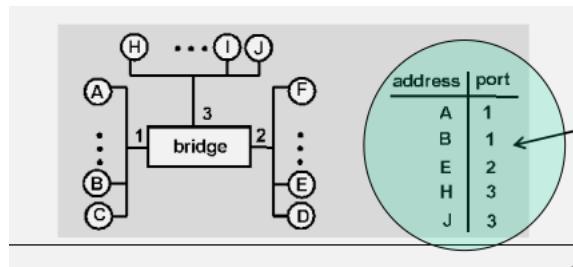
che contro:

- Gli hub non isolano i domini di collisione, le stazioni di un segmento possono subire una collisione per una trasmissione simultanea da parte di una stazione presente su un qualunque altro segmento
- La creazione di un singolo dominio di collisione non comporta aumento del throughput massimo
- La realizzazione di un'unica LAN impone un limite al numero massimo di stazioni che è possibile collegare, nonché all'estensione geografica che è possibile raggiungere

## 28.2 Bridge

I bridge sono dispositivi che permettono il collegamento di due o più LAN tra loro in maniera più efficiente degli Hub. I bridge filtrano i pacchetti, ossia se al destinazione è sullo stesso segmento LAN del mittente, il bridge non compie nessuna azione, se è su un segmento di LAN differente, inoltre il frame sul segmento LAN appropriato. Per scoprire a quali interfacce sono collegati gli host, i bridge eseguono un algoritmo di **autoapprendimento**.

- Salvano informazioni in tabelle dette **filtering tables**
- Quando una frame è ricevuta il bridge salva il segmento di LAN di provenienza in una filtering table
- Una entry nella filtering table è composta da MAC Address, Bridge Port, Time Stamp

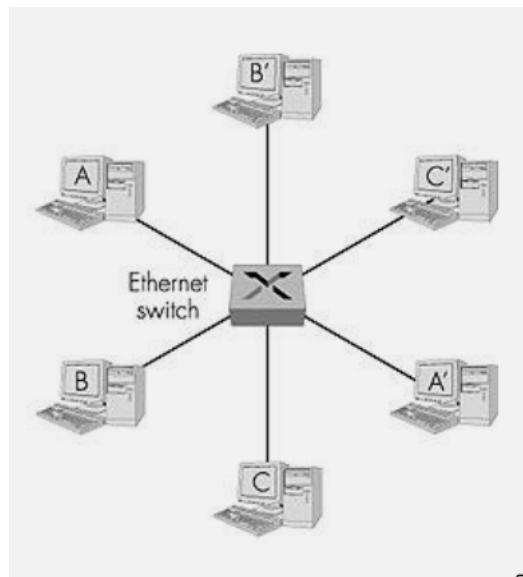


I vantaggi dei bridge sono:

- Isolano i domini di collisione, determinando un aumento complessivo del throughput massimo
- Non introducono limitazioni sul numero massimo delle stazioni
- Possono collegare differenti tecnologie
- Sono trasparenti, ossia non richiedono alcuna modifica negli adattatori del computer né configurazione da parte di un admin

## 28.3 Switch

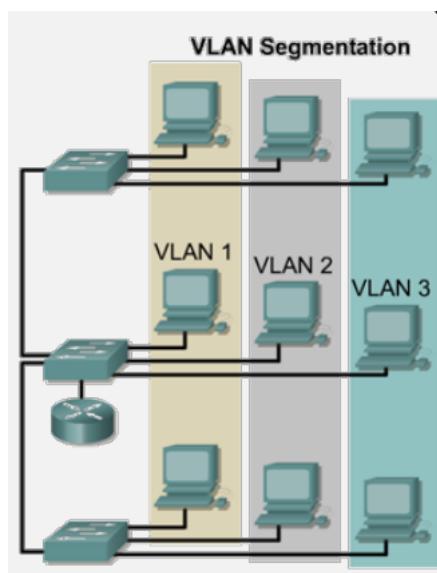
Gli switch ethernet sono dispositivi il cui comportamento è simile a quello di un bridge. Sono dispositivi con diverse porte di rete e fungono da centro stella del cablaggio in modo simile ad un hub, tuttavia la superiorità di uno switch rispetto a bridge e hub è dovuta al fatto che inoltre le frame solo verso la porta del destinatario grazie ad un'azione di filtraggio



Il principale vantaggio di questo tipo di filtraggio, è la totale assenza di collisioni tra porte distinte. Sono molto rapidi ad operare grazie al **Cut-through switching**, ossia l'inoltro delle frame verso la porta di uscita corrispondente prima che il frame sia stato ricevuto completamente (basta ricevere l'header del frame per stabilire la porta di uscita)

## 29 VLAN (data link)

Una **Virtual Local Area Network (VLAN)** è una tecnologia di rete che permette di suddividere una rete fisica in più reti logiche indipendenti. Funziona al livello 2 del modello OSI e offre vantaggi significativi in termini di sicurezza, gestione del traffico e flessibilità organizzativa.

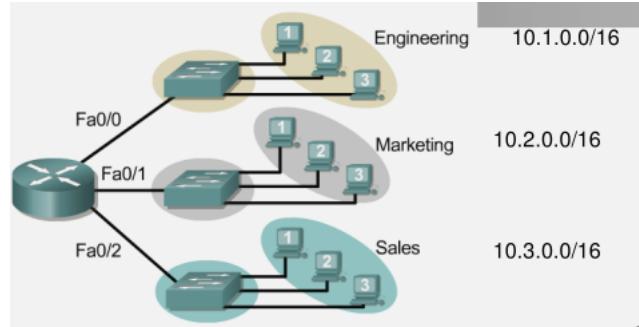


È costituita da switch ethernet che supportano la VLAN e attraverso questi ultimi è possibile raggruppare gruppi di porte tra le quali è possibile una comunicazione diretta e l'inoltro di traffico broadcast. La comunicazione tra VLAN diverse rimane tuttavia possibile solo attraverso una funzione L3 di routing

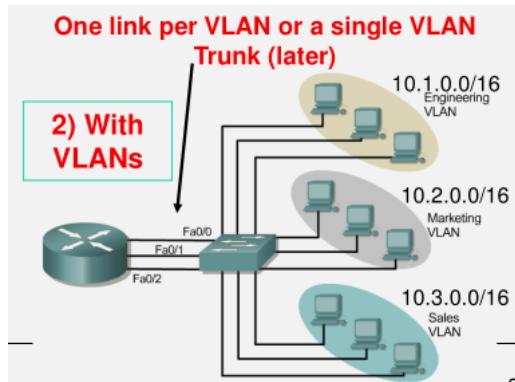
## 29.1 Funzionamento

Come detto in precedenza, una VLAN permette l'interconnessione di switch Ethernet in LAN virtuali, tipicamente coincidenti con una sottorete IP. Vediamo la differenza tra una rete con VLAN e senza.

- **Senza VLAN:** ogni gruppo ha una sottorete IP differente e uno switch ethernet differente



- **Con VLAN:** lo switch è configurato con le porte appropriate per ogni VLAN. Ogni gruppo ha ancora una sottorete IP differente, ma sono tutte collegate allo stesso switch



Esistono 3 tipi principali di VLAN:

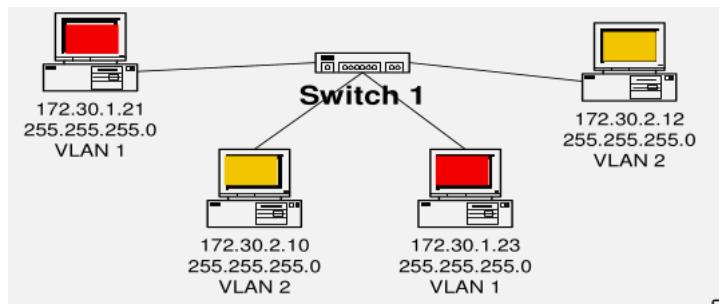
- **Port-based:** è il metodo più comune, le porte sono assegnate individualmente, in gruppi, in righe o attraverso 2 o più switch. Spesso sono implementate con DHCP per assegnare indirizzi IP agli host del network
- **MAC address:** meno comuni, ogni indirizzo MAC deve essere assegnato allo switch e configurato individualmente. Difficili da gestire
- **Protocol Based:** Si configurano come le MAC address ma utilizzano l'indirizzo IP degli host. Non più comuni a causa dei DHCP

La gestione dei messaggi in broadcast, è decisamente più ottimale attraverso l'uso delle VLAN.

- **Senza VLAN:** ogni host a prescindere della sottorete connesso allo switch riceve il messaggio in broadcast (ad esempio ARP message)
- **Con VLAN:** solo gli host all'interno della rete virtuale ricevono il messaggio in broadcast

L'assegnazione degli host, va fatta sulle porte dello switch, e per far si che un host sia parte della VLAN, deve ricevere un indirizzo IP che appartiene alla sottorete appropriata (VLAN = Subnet). L'assegnazione di un host alla corretta VLAN è un processo di due step

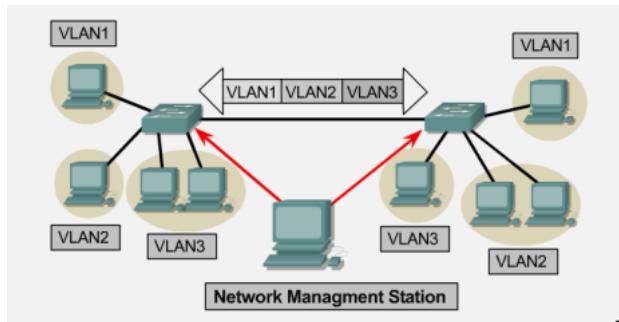
- Collegare l'host alla porta corretta sullo switch
- Assegnare all'host l'indirizzo IP corretto a seconda della appartenenza alla VLAN



La configurazione di una VLAN può essere di 2 tipi:

- **Configurata staticamente:** l'amministratore di rete configura gli switch porta per porta
- **Configurata dinamicamente:** le porte sono in grado di determinare la corretta configurazione VLAN attraverso un database di MAC address to VLAN mappings impostato dall'amministratore

VLAN configurate staticamente sono dette **port-based VLANs**, quando un dispositivo viene connesso, assume la configurazione VLAN della porta di connessione. La VLAN di default per ogni porta è la **Management VLAN** e non può essere eliminata, tutte le altre VLAN possono essere riassegnate ad altre porte



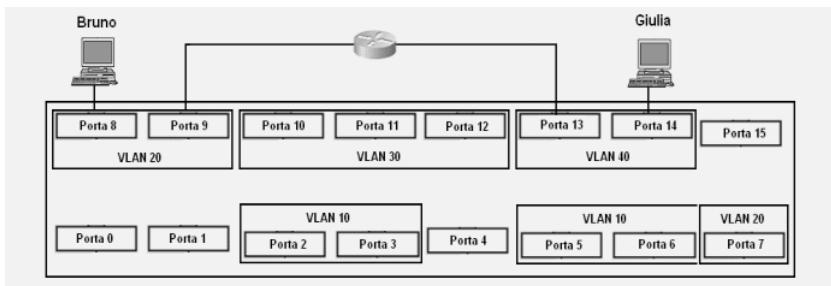
VLAN configurate dinamicamente sono create attraverso un software di network management. Alla connessione di un dispositivo alla rete, viene fatta una query al database interno allo switch per cercare l'appartenenza ad una VLAN in base al MAC address.

## 29.2 Comunicazione tra VLAN

Per poter comunicare tra VLAN diverse occorre creare un ponte attraverso un dispositivo.

- **bridge** se opera la L2
- **router** se opera al L3

molti produttori offrono dispositivi in grado di svolgere contemporaneamente le funzioni di switch a livello 2 e di router al livello 3, questi dispositivi creano la connessione tra VLAN a livello 3

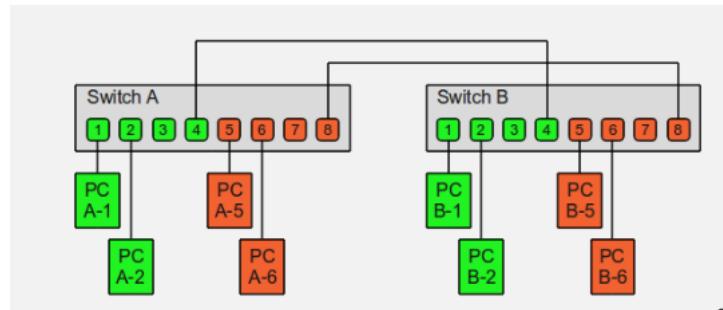


In linea di principio, rimane possibile ottenere lo stesso risultato collegando le interfacce di un router a tutte le coppie di VLAN

### 29.3 VLAN trunking

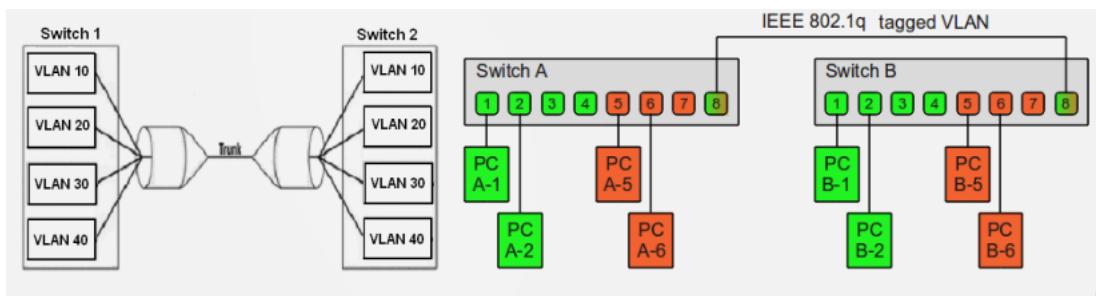
La presenza delle VLAN crea un problema nella connessione tra due o più switch

- Se collego la porta di uno switch a una porta di uno switch, la connessione riguarderà solo le VLAN che comprendono le due porte utilizzate

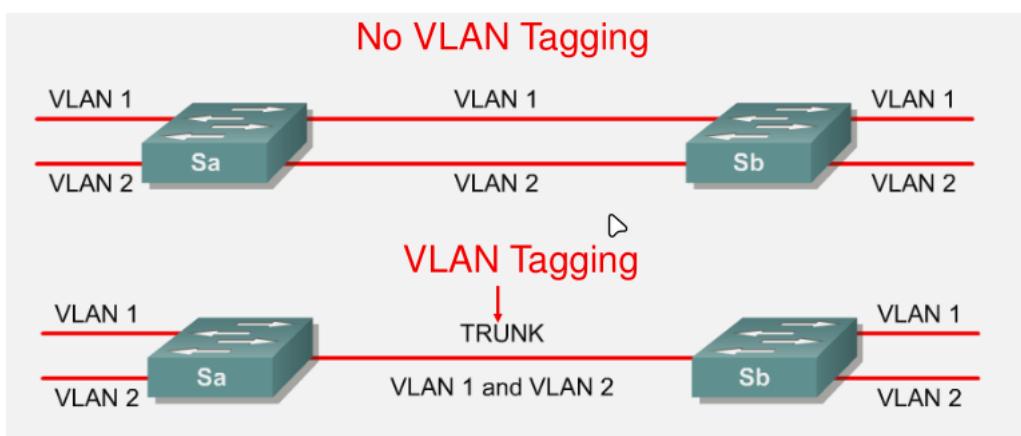


Per ovviare a questo problema, il **trunking** abilita la connessione tra le VLAN di switch diversi.

- lo switch destinazione deve sapere a quale VLAN inoltrare i frame in arrivo su una porta di trunking, occorre dunque **taggare** i frame con l'identificativo della VLAN di destinazione
- il tagging tuttavia non è previsto nel protocollo ethernet originale

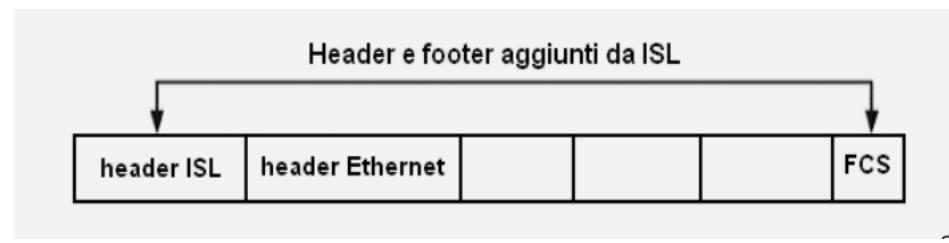


Ciò avviene attraverso **trunk link**, ossia pacchetti che vengono ricevuti dallo switch con un identificativo unico del pacchetto aggiunto in ogni header. L'informazione nell'header stabilisce l'appartenenza a una data VLAN per ogni pacchetto.



Ci sono due protocolli per questa procedura

- **protocolli a incapsulamento:** viene aggiunto un header al frame Ethernet per indicare la VLAN di destinazione (Es. Cisco Inter-Switch Link)



- **protocolli a piggyback:** l'identificativo della VLAN (12 bit) è parte di un campo da 4 byte inserito nel frame Ethernet tra i campi indirizzo sorgente e tipo. Comporta il ricalcolo del CRC all'ingresso e all'uscita dal trunk

## 30 Crittografia

Ci sono diversi aspetti da tenere in considerazione quando si parla di sicurezza in rete

- **Riservatezza:** solo il mittente e il destinatario dovrebbero essere in grado di comprendere il contenuto di un messaggio
- **Integrità dei messaggi:** mittente e destinatario di un messaggio devono essere certi che i messaggi non siano alterati da terze parti
- **Autenticazione:** mittente e destinatario di un messaggio devono essere sicuri reciprocamente dell'identità della controparte
- **Accessibilità e disponibilità dei servizi:** i servizi offerti in rete devono essere protetti da eventuali attacchi

Ci sono diversi comportamenti malevoli che possono interferire con la sicurezza online:

- **eavesdropping:** intercettazione dei messaggi
- **spoofing:** inviare pacchetti con il campo source address falso in modo da fingere di essere qualcun altro
- **hijacking:** dirottare la comunicazione tra due sistemi fingendo con il primo di essere il secondo e viceversa
- **denial of service:** impedire al servizio offerto di essere utilizzabile

Per ovviare a questi problemi si utilizza un sistema **crittografico**, ossia un sistema in grado di cifrare e decifrare un messaggio attraverso l'utilizzo di un **algoritmo** e di una **chiave**. Introduciamo la distinzione tra testo in chiaro e testo cifrato:

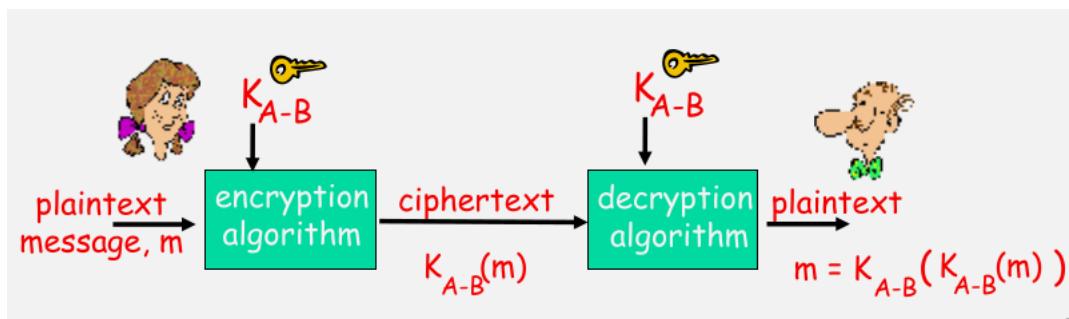
- **plaintext:** il "testo in chiaro" non cifrato
- **ciphertext:** il "testo cifrato" da un algoritmo e una chiave

Esistono due tipi principali di crittografia

- **Crittografia a chiave simmetrica:** mittente e destinatario utilizzano la stessa chiave detta "segreto condiviso" per l'encryption e la decryption
- **Crittografia asimmetrica/chiave pubblica:** la chiave per l'encryption è nota a tutti, la chiave per la decryption è segreta

### 30.1 Chiave simmetrica

Entrambi gli end systems conoscono la chiave crittografica simmetrica  $K_{A-B}$



Il problema di questo tipo di crittografia è il metodo con cui avviene lo scambio della chiave. Un esempio di crittografia a chiave simmetrica è il **cifrario per sostituzione**

- unità di testo del plaintext sono sostituite con corrispondenti sequenze di simboli nel testo cifrato secondo uno schema regolare

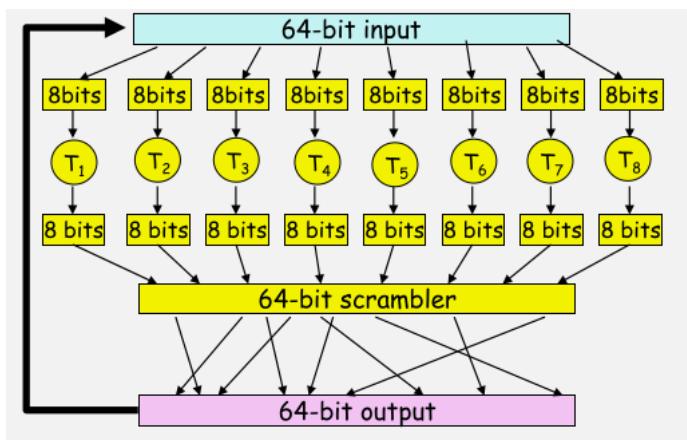
Un esempio è il **cifrario monoalfabetico**, ossia una corrispondenza fissa tra ciascuna lettera dell'alfabeto in chiaro e l'alfabeto cifrato. Si tratta tuttavia di un tipo di crittografia relativamente facile da decifrare.



Un altro esempio è il **cifrario a blocchi**, in cui un blocco di  $k$  bit del testo in chiaro è codificato con altri  $k$  bit nel testo in codice secondo uno schema fisso, per esempio

000 → 110	001 → 111	010 → 101	011 → 100
100 → 011	101 → 010	110 → 000	111 → 001

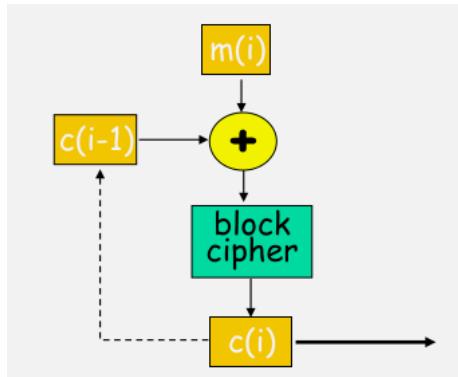
Il messaggio  $(010)(110)(001)(111)$  è cifrato come  $(101)(000)(111)(001)$ . Anche se più sicuro rispetto al cifrario monoalfabetico, risente di problematiche dovute alla facilità di decriptazione per valori piccoli di  $k$  e per valori di  $k$  grandi si hanno problemi di implementazione in quanto cifratura e decifratura richiedono  $2^k$  elementi in memoria. Tuttavia si può ottenere elevati valori di  $k$  per composizione di valori più piccoli ripetendo l'algoritmo più volte



Esempi di cifrari a blocchi sono il **Data Encryption Standard (DES)** che utilizza una chiave simmetrica a 56 bit e un input plaintext a 64 bit, e l'**Advanced Encryption Standard (AES)** che ha sostituito il DES ed elabora dati a blocchi di 128 bit con chiavi da 128, 192, o 256 bit.

### 30.1.1 Cipher block-chaining

La debolezza dei cifrari a blocchi è che input uguali producono lo stesso testo cifrato. Per ovviare a questo inconveniente si utilizza la tecnica detta **cipher block chaining**:



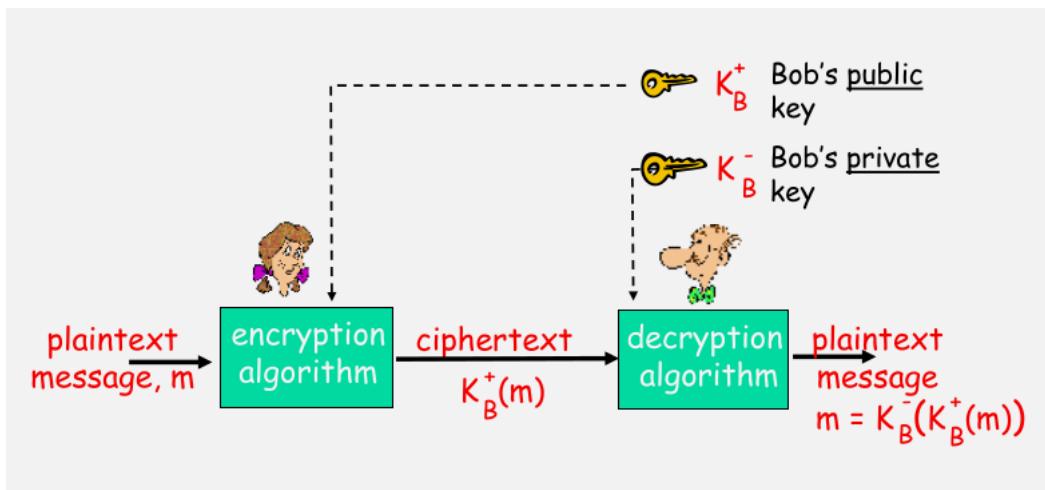
- si calcola  $c(i)$  con l'algoritmo di cifratura applicato al blocco ottenuto tramite XOR del testo in chiaro  $m(i)$  con il blocco di testo cifrato  $c(i - 1)$  calcolato sull'input precedente

## 30.2 Chiave pubblica

Occorre trovare una coppia di chiavi  $K_B^+(\cdot)$  e  $K_B^-(\cdot)$  tali che

$$K_B^-(K_B^+(m)) = m$$

Nota la chiave pubblica, dovrebbe essere impossibile calcolare la chiave privata



Uno degli algoritmi più utilizzati è **RSA**, in cui

- si scelgono due numeri primi  $p, q$  molto grandi
- si calcolano  $n = p \cdot q$  e  $z = (p - 1) \cdot (q - 1)$
- si sceglie un numero  $e$  con  $e < n$  che non abbia fattori comuni con  $z$
- si sceglie un numero  $d$  tale che  $e \cdot d - 1$  sia esattamente divisibile per  $z$
- la chiave pubblica è  $(n, e)$  la chiave privata è  $(n, d)$

Una volta scelte le due chiavi, la cifratura e la decifratura del testo avvengono nel seguente modo

- **cifratura**

$$c = m^e \bmod n$$

- **decifratura**

$$m = c^d \bmod n$$

La robustezza di RSA deriva dal fatto che non sono noti metodi per la rapida fattorizzazione di numeri interi grandi. Dunque anche se  $n$  è noto, i due fattori  $p, q$  sono molto difficili da determinare

## 31 Sicurezza

### 31.1 Hashing functions

Una funzione di hash  $H$  è una funzione non invertibile che mappa una stringa  $m$  di lunghezza arbitraria in una stringa  $h$  di lunghezza prefissata (**digest**)

$$h = H(m)$$

Le funzioni di hash crittografico sono particolari funzioni hash che godono di alcune proprietà che le rendono adatte per l'uso nella crittografia

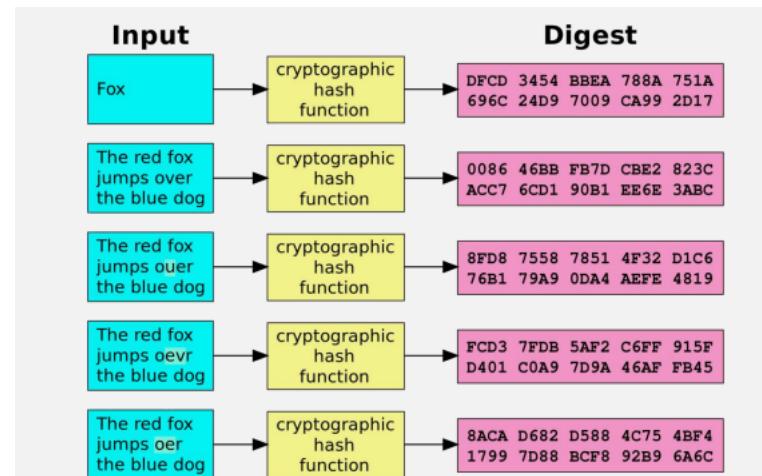
- **resistenza alla preimmagine:** dato un valore di hash  $h$  è impossibile ricostruire l'input del digest
- **resistenza alla seconda preimmagine:** data una stringa  $m_1$  è impossibile trovare una stringa  $m_2$  tale che

$$H(m_1) = H(M_2)$$

- **resistenza alle collisioni:** è impossibile trovare una coppia di input con lo stesso digest

$$H(m_1) = H(m_2)$$

Un esempio di hash crittografico attraverso SHA-1 è il seguente



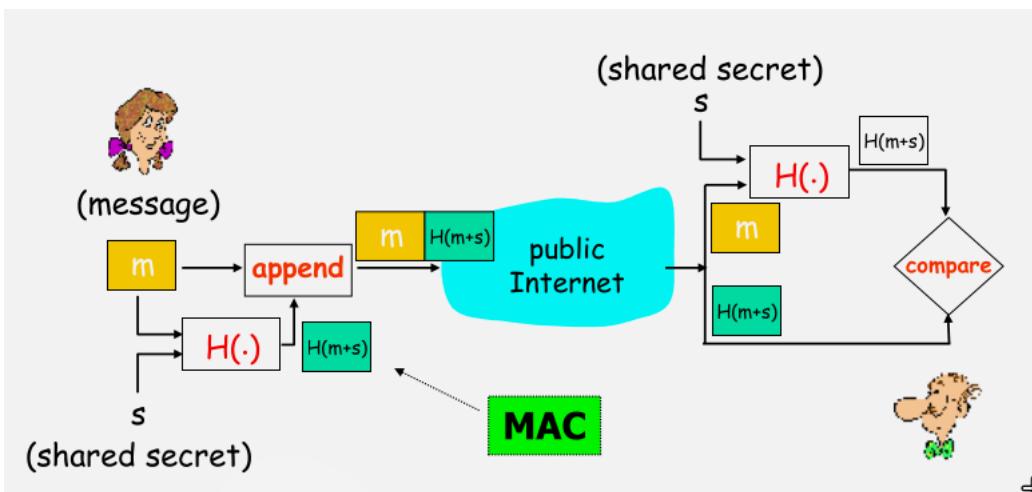
### 31.1.1 Collisioni

Data una funzione di hash  $H()$  si dice che produce una **collisione** quando due input diversi hanno lo stesso valore di hash  $H(\text{input1}) = H(\text{input2})$ . Un esempio di funzione di hash che produce collisioni e dunque non è una funzione di hash crittografica, è quello di una funzione di hash per il calcolo della checksum in protocolli quali TCP e UDP.

- anche se la funzione di hash produce un digest a lunghezza fissa (16 bit) da un messaggio di lunghezza arbitraria, è facile trovare un altro messaggio con lo stesso valore di checksum

### 31.1.2 Message Authentication Code

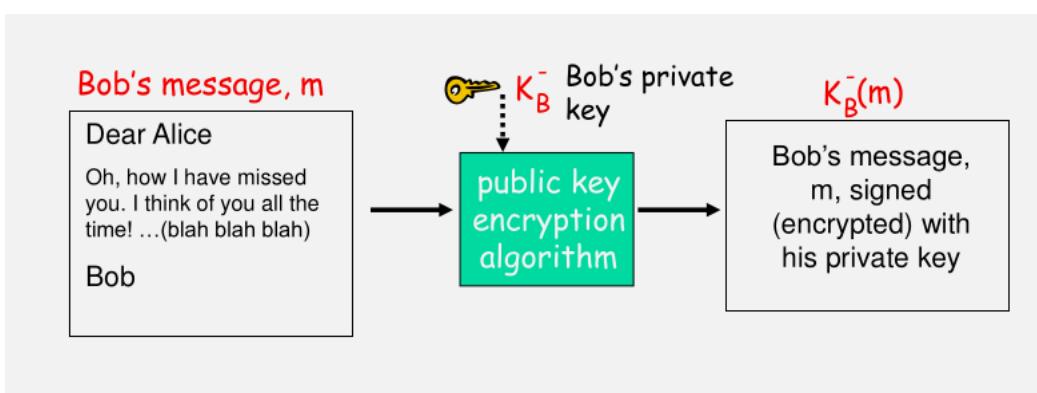
Il MAC (Message Authentication Code), sfrutta una funzione di hash per controllare l'integrità di un messaggio



Sfrutta una funzione di hash e una chiave segreta  $s$  nota ad entrambi. All'inviato del messaggio, si invia anche il digest del messaggio + la chiave segreta, in modo che il ricevente possa ricalcolare la funzione di hash su  $m+s$  e compararla con il messaggio per garantirne l'integrità.

## 31.2 Firma Digitale

La firma digitale, come quella fisica, necessita di essere **verificabile** e **non falsificabile**. Il destinatario del messaggio deve poter provare a un terzo che il mittente del messaggio è stato colui che ha apposto la firma e nessun altro. Ciò avviene tramite hashing crittografico, una semplice tecnica di firma digitale è la seguente

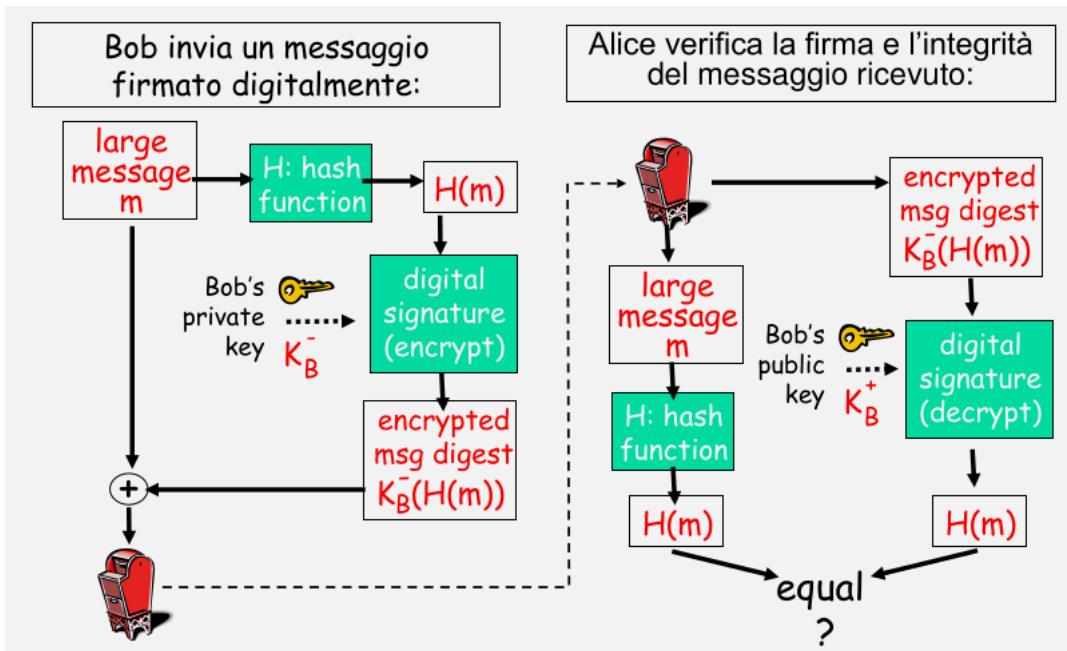


- Si "firma" il messaggio  $m$  crittografandolo con chiave privata  $K_B^-$  e creando un messaggio firmato  $K_B^-(m)$

- Alla ricezione si riceve il messaggio  $m$  con la firma digitale  $K_B^-(m)$
- Si verifica se  $m$  sia stato firmato dal mittente decifrando con la chiave pubblica  $K_B^+$  il testo cifrato, e se

$$K_B^+(K_B^-(m)) = m$$

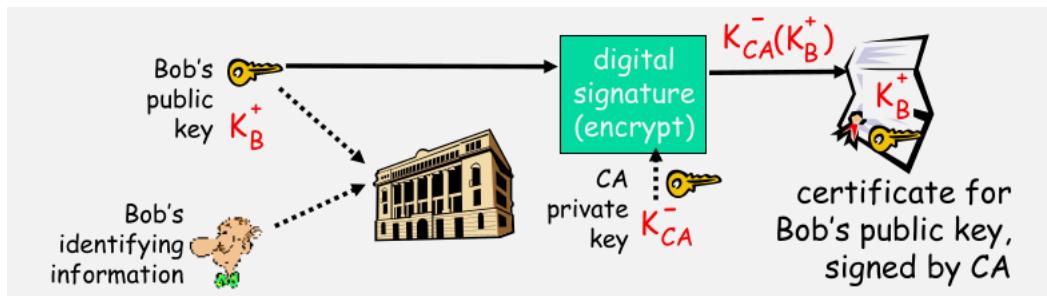
allora si è verificato che il mittente ha firmato  $m$



### 31.3 Certificazione della chiave pubblica

Un problema riguardante la distribuzione di chiavi pubbliche, è l'avere la certezza che la chiave ricevuta sia effettivamente di chi dovrebbe inviarla e non di un terzo. La soluzione a questo problema è l'istituzione di **certification authorities (CA)**. Si tratta di enti abilitati a rilasciare certificati digitali tramite procedure di certificazione basate su standard internazionali.

- Una CA associa in modo sicuro una chiave pubblica ad una particolare entità  $E$
- $E$  registra la sua chiave pubblica presso la CA
- CA crea un certificato che associa  $E$  alla sua chiave pubblica



Un certificato contiene un numero di serie unico emesso dalla CA, informazioni riguardo al possessore del certificato, informazioni su chi ha emesso il certificato, una data di validità, la firma digitale di chi ha emesso il certificato

## 31.4 Autenticazione

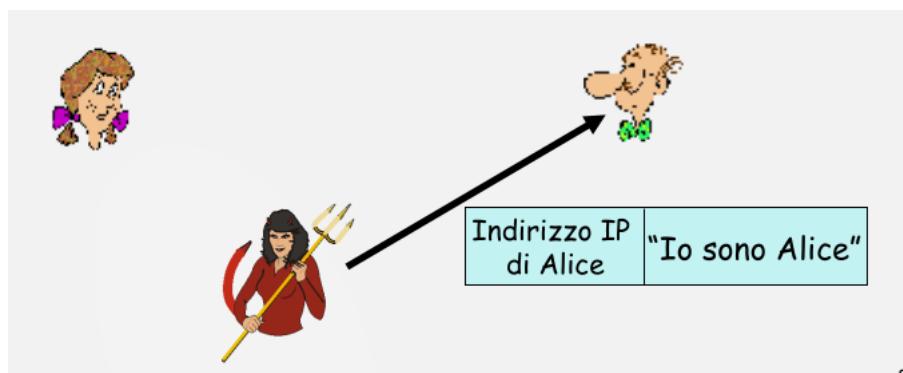
Per quanto riguarda i protocolli di autenticazione, ce ne sono diversi ma i primi 3 che vedremo sono sicuramente troppo poco sicuri per poter essere utilizzati

- **plaintext (ap1)**: A invia un plaintext a B dicendo "sono A"



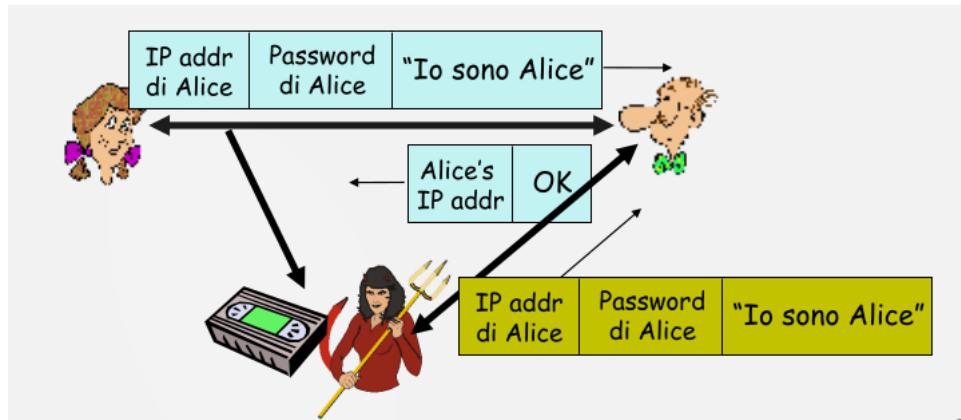
soggetto a terze parti che possono facilmente fingere di essere A

- **Pacchetto IP (ap2)**: A invia un pacchetto IP in cui dice "sono A"



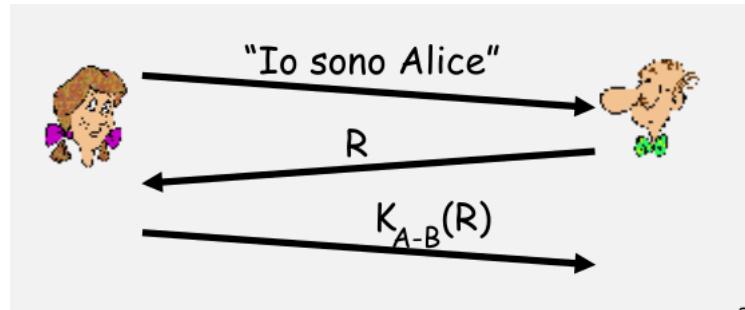
soggetto a terze parti che possono inviare un pacchetto IP contenente come indirizzo IP quello di A

- **IP e password (ap3)**: A invia un pacchetto IP dicendo "sono A" contenente la sua password



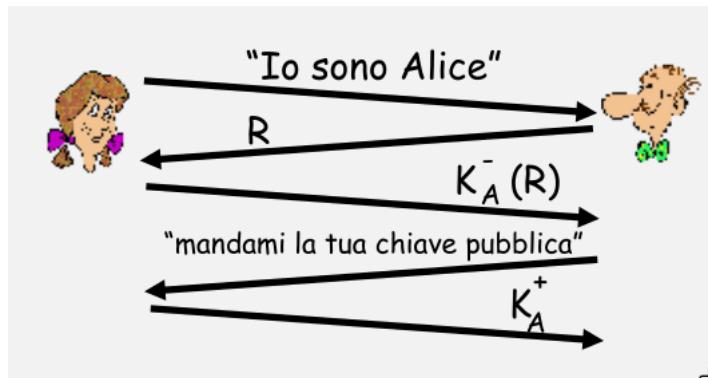
soggetto a **playback**, terzi registrano il pacchetto di autenticazione e lo rinviano

- **IP e password cifrata (ap3.1):** identico ad ap3 ma con password cifrata, sempre soggetto ad attacchi playback
- **Nonce (ap4.0):** per impedire l'attacco playback, si implementa un **nonce**, un numero R usato na sola volta. B invia ad A un nonce R, A deve restituire R cifrato con la sua chiave segreta



Il problema rimane quello della crittografia a chiave simmetrica, ossia lo scambio di chiavi

- **nonce asimmetrico (ap5):** B invia ad A un nonce R, A restituisce R cifrato con chiave privata



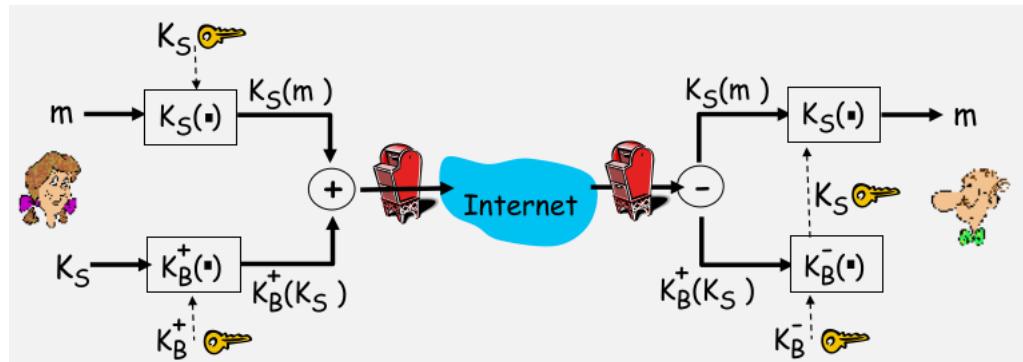
B calcola  $K_A^+(K_A^-(R))$  e sa che A possiede la chiave privata se  $K_A^+(K_A^-(R)) = R$

Si noti che anche ap5 è soggetto a fallo di sicurezza quale quella del **man in the middle**, in cui una terza parte si pone in mezzo tra i messaggi in invio da parte di A e i messaggi di invio da parte di B. L'unica soluzione è legata alla distribuzione delle chiavi pubbliche

### 31.5 Email sicura

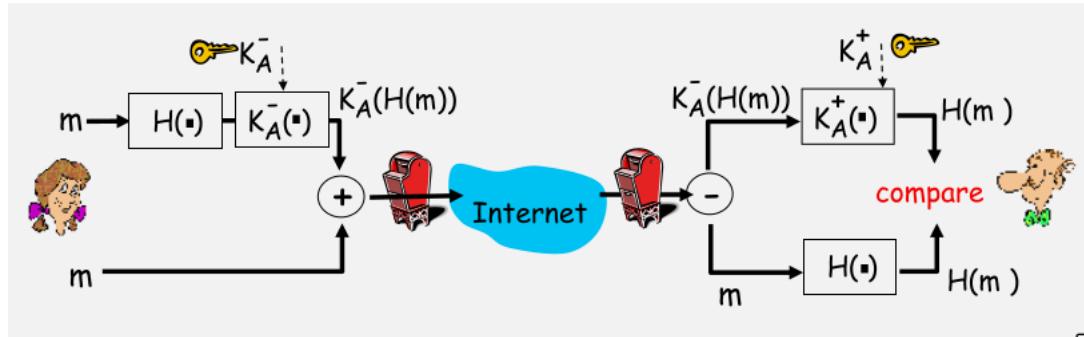
Supponiamo di voler inviare un messaggio m da A a B. Abbiamo 3 casi:

- **Caso 1:** A genera una chiave simmetrica privata  $K_s$ , cifra il messaggio m con  $K_s$  e cifra  $K_s$  con la chiave pubblica di B. Si invia si  $K_s(m)$  che  $K_B(K_s)$



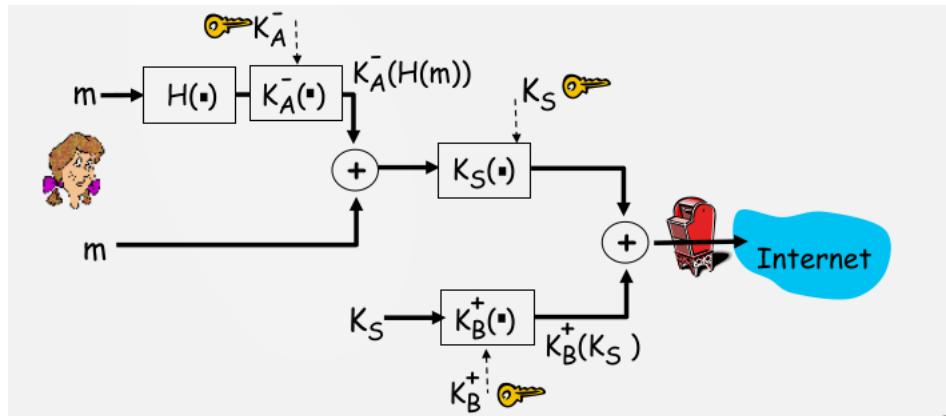
Questo approccio non garantisce l'identità del mittente

- **Caso 2:** A invia a B un messaggio apponendo la firma digitale, tuttavia il messaggio viene trasmesso in plaintext



Questo approccio non garantisce la sicurezza del messaggio  $m$

- **Caso 3:** A invia a B la sua firma digitale e codifica il messaggio con chiave segreta  $K_s$ . Invia successivamente la chiave segreta crittografandola con la chiave pubblica di B



A utilizza 3 chiavi, la sua chiave privata, la chiave pubblica di B e la chiave simmetrica. Si tratta dell'approccio più sicuro

## 32 Wireless

Le due sfide più grandi nella creazione di reti wireless, riguardano:

- **Wireless:** una comunicazione con collegamento senza fili
- **Mobilità:** gestire l'utente mobile che cambia punto di accesso alla rete

Una rete wireless è formata da

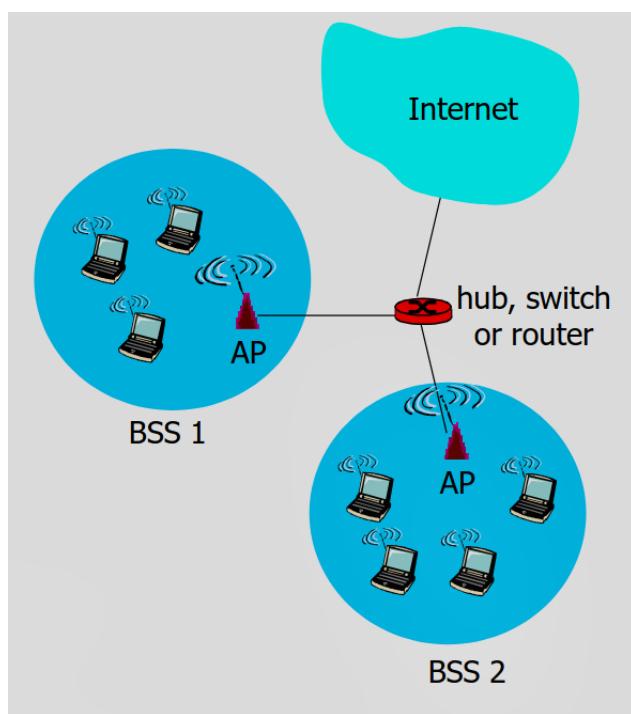
- **wireless hosts:** computer, telefoni ecc. end systems che si collegano alla rete in wireless
- **base stations:** stazioni connesse alla rete wired responsabili per l'invio e l'inoltro di pacchetti tra la rete wireless e la rete wired. Ne sono esempi le celle telefoniche, access points ecc...
- **wireless link:** connessioni di rete che utilizzano segnali radio, infrarossi o altre tecnologie di trasmissione elettromagnetica per la comunicazione tra end system e base stations.

I wireless link differiscono in portata a seconda del meccanismo elettromagnetico che si utilizza per la trasmissione. Alcune caratteristiche comuni sono

- **Diminuzione intensità del segnale:** con il propagarsi attraverso la materia, le onde radio si indeboliscono
- **Interferenze:** reti wireless standardizzate tra più dispositivi potrebbero causare interferenze
- **Multipath propagation:** le onde radio riflettono su oggetti e sul terreno arrivando a destinazione con tempi differenti

## 32.1 Wireless LAN 802.11

Una rete locale wireless, comunica attraverso onde elettromagnetiche che si propagano nello spazio. Come altre tecnologie LAN, la wireless LAN 802.11 è progettata per l'impiego in aree geografiche limitate ed ha lo scopo di fare da mediatore nell'accesso ad un mezzo condiviso di comunicazione (frequenza radio)



Si ha una connessione wireless da parte degli host con la base station (access point AP) che comunica successivamente con la rete wired. 802.11 è progettato per trasmettere dati usando tre tecniche differenti:

- **Frequency Hopping**
- **Direct Sequence**
- **Diffused Infrared**

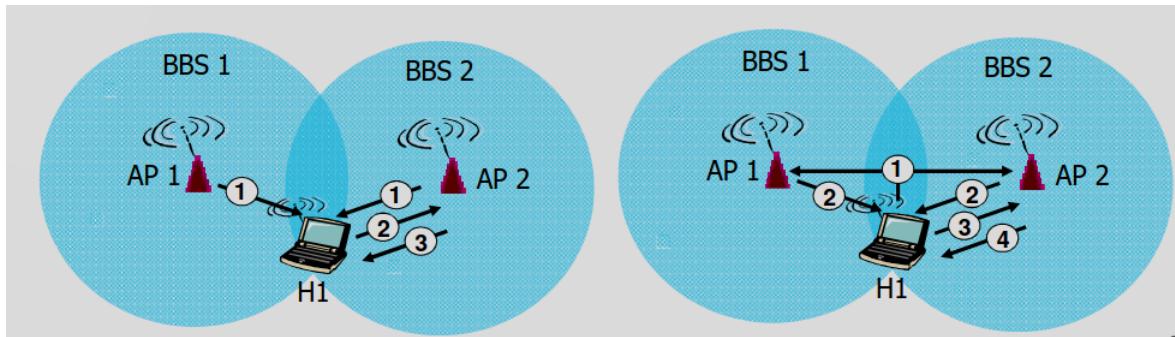
Le prime due tecniche sfruttano il range di frequenza intorno ai 2.4 GHz e sono tecniche del tipo spread spectrum, con l'obiettivo di diffondere il segnale su un intervallo di frequenze ampio in modo da minimizzare gli effetti di interferenza da altri dispositivi

### 32.1.1 Associazione

Un host deve associarsi ad un AP per poter trasmettere, il processo può avvenire in due modi

- **Passive Scanning:** frame inviata dall'AP, l'host invia una request frame all'AP, l'AP accetta l'associazione e invia una response frame

- **Active Scanning:** Probe request frame inviata in broadcast dall'host, probe response frame inviata dall'AP, association request inviata dall'host all'AP, association response inviata dall'AP all'host



### 32.1.2 Metodo di accesso

Il metodo di accesso è simile ad Ethernet:

- l'host prima di trasmettere attende finchè il canale diventa libero
- in caso di collisione ha inizio un algoritmo di binary exponential backoff

Bisogna tuttavia tenere in considerazione che non tutti i nodi sono sempre alla portata l'uno dell'altro, ciò determina due tipi di problemi:

- **Hidden node problem**
- **Exposed node problem**

**Hidden terminal problem:**

□ B, A hear each other  
 □ B, C hear each other  
 □ A, C can not hear each other  
 means A, C unaware of their interference at B

**Signal attenuation:**

□ B, A hear each other  
 □ B, C hear each other  
 □ A, C can not hear each other interfering at B

## 32.2 CSMA/CA