

Reti di Calcolatori 1

A.A. 2021-2022

Docente: G. Ventre

Dispense tratte dalle slide del corso di Informatica a cura dello studente **S. Cerrone**

1. Introduzione alle Reti di Calcolatori	5
Elementi costitutivi delle reti di calcolatori	5
Struttura delle reti di calcolatori	5
Commutazione di circuito	5
Commutazione di pacchetto	6
Le reti di calcolatori	8
2. Modelli a Strati e Protocolli	8
Gestire la complessità delle reti di calcolatori	8
Modelli a strati	9
Protocolli di comunicazione	9
Imbustamento dei messaggi.....	10
Service Access Point (SAP), Service Data Unit (SDU), Protocol Data Unit (PDU)	10
Frammentazione dei messaggi ad un livello	10
3. Modello ISO-OSI.....	11
Il modello OSI	11
Un approccio a livelli per la risoluzione dei problemi	12
Internet	13
4. Protocolli applicativi: HTTP	14
Strato dell'applicazione	14
HTTP	15
HTTP per il trasferimento di pagine web	16
Connessioni HTTP	16
Il protocollo HTTP.....	17
Lo scambio di messaggi	17
Gli header	19
I cookies.....	19
Web caching	20
Server proxy	20
5. Peer-to-Peer Networks (P2P).....	21
Distribuzione dei file.....	21
P2P con directory centralizzata (Napster)	22
P2P con directory decentralizzata (gnutella)	22
Skype	23
6. Protocolli applicativi: FTP ed SMTP	23
File Transfer Protocol (FTP)	23
Il protocollo SMTP	24
Prelievo della posta: Post Office Protocol (POP3)	26
7. Protocolli applicativi: DNS	26
Domain Name System (DNS)	26
Il servizio DNS.....	26
DNS distribuito	27
Tipologie di server DNS	28
Il caching dei nomi.....	29
Il formato dei messaggi	29
Il servizio BIND.....	29
8. Content Delivery Networks (CDN).....	30
Cosa è una CDN	30
Replicazione e aggiornamento	30
Prelievo da parte del client.....	30

9.	Livello di Rete e Protocollo IP	31
	Il livello rete in Internet	31
	Internet Protocol (IP).....	32
	Cenni sui sistemi di numerazione	34
	Indirizzi IP	35
	Classi di indirizzi.....	36
	Il protocollo IP: frammentazione Subnetting	38
10.	ARP – RARP – DHCP – ICMP: ping e traceroute	40
	Address Resolution Protocol (ARP).....	40
	Reverse ARP (RARP).....	42
	Dynamic Host Configuration Protocol (DHCP)	43
	Internet Control Message Protocol (ICMP).....	43
11.	Router NAT	44
	Network Address Translation (NAT)	44
	Un esempio	44
	Implementazione	44
	Problemi di attraversamento	45
12.	Il protocollo IPv6.....	45
	IP Next Generation (IPng o IPv6)	45
	IPv6: modifiche principali rispetto ad IPv4	45
	IPv6: caratteristiche generali.....	46
	L'header IPv6.....	46
	Modalità di scrittura degli indirizzi in IPv6.....	47
	Tunneling: transizione da IPv4 a IPv6	47
13.	Routing: Introduzione	48
	Reti di calcolatori e grafi	48
	Il processo di routing	48
	Tipologie di routing	49
	Problematiche associate al routing	50
14.	Routing Distance Vector e Routing Link State.....	50
	Distance Vector	50
	Link State.....	54
	L'algoritmo di Dijkstra	56
15.	Le socket di Berkeley.....	57
	Le socket.....	57
	Interfacce e protocolli	59
	Il paradigma Client-Server (C/S)	59
	I paradigmi di comunicazione.....	60
	Progettazione di Client e Server	61
16.	Internet e il routing gerarchico	62
	Il routing in Internet	62
	Routing interno e routing esterno	63
	I gateway router	63
17.	Protocolli IGP	64
	Routing Information Protocol (RIP)	64
	Interior Gateway Routing Protocol (IGRP).....	66
	Open Shortest Path First (OSPF)	66

18. Multicasting.....	68
Tecniche di trasmissione broadcast.....	68
Trasmissione multicast in reti IP.....	68
Il multicast router	69
Il protocollo IGMP	69
IP multicast.....	71
Reverse Path Forwarding (RPF)	71
Il routing multicast e l'albero di copertura	72
Protocolli per il multicast in Internet.....	72
La rete MBone	73
19. Il Livello Trasporto	74
Introduzione	74
Il protocollo UDP	75
Tecniche di trasmissione affidabile dei dati.....	76
20. Livello Trasporto: Il protocollo TCP	79
Transmission Control Protocol	79
Struttura del segmento TCP	80
TCP: Caratteristiche	81
Round Trip Time e Timeout	82
Trasmissione affidabile.....	83
Modifiche tipiche del TCP	84
TCP Connection Management.....	85
Sequenza tipica degli stati nel client e nel server	86
Diagramma degli stati del TCP	86
Controllo di flusso e controllo della congestione	87
21. Lo Strato di Collegamento	91
Caratteristiche del livello data link	91
Protocolli di accesso multiplo.....	92
Protocolli di suddivisione del canale	93
Protocolli ad accesso casuale	93
Ethernet	94
Hub.....	95
Bridge	95
Switch	97
22. Reti Wireless.....	97
Wireless LAN: 802.11.....	97
IEEE 802.11: accesso multiplo	99
WLAN/802.11	100
23. Protocolli per la trasmissione di flussi multimediali in Internet.....	101
Trasferimento di informazioni multimediali su rete	101
Sensibilità dello streaming alla qualità del servizio	101
Trasferimento di informazioni multimediali su Internet	102

1. Introduzione alle Reti di Calcolatori

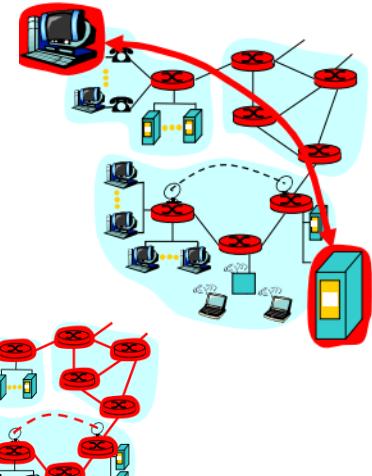
Elementi costitutivi delle reti di calcolatori

Le reti di calcolatori hanno i seguenti scopi: condivisione dell'informazione, condivisione delle risorse, accesso a risorse remote e accesso a servizi.



Alle estremità della rete si trovano gli **end-system** o **host**:

- Sono calcolatori di vario tipo su cui girano i programmi applicativi
- I programmi applicativi possono essere progettati secondo due modelli:
 - **Client-Server**: il client invia una richiesta ed il server risponde
 - **peer-to-peer**: le due entità comunicanti si scambiano informazioni in modo paritetico



L'infrastruttura della rete è fatta di:

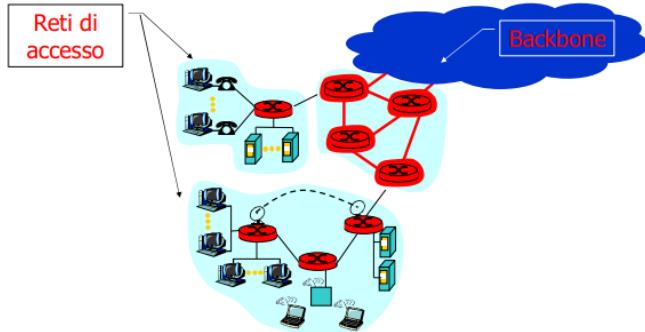
- **apparati** tra loro interconnessi
 - bridge, modem, access point, router, etc...
- **supporti trasmissivi** che realizzano le interconnessioni
 - doppini in rame, cavi coassiali, fibre ottiche, etc...



Struttura delle reti di calcolatori

L'infrastruttura di rete si può dividere grossolanamente in:

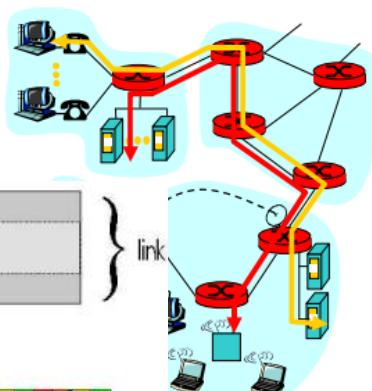
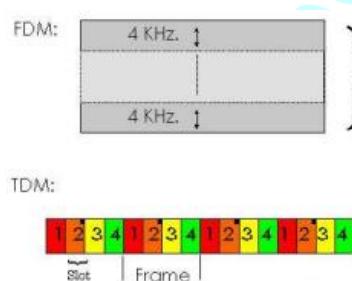
- **Reti di accesso**
 - Forniscono la connettività agli end-system
 - Utilizzano sia tecnologie *Wired* (rete telefonica tradizionale, ethernet, ATM, X.25, frame relay) che tecnologie *Wireless* (WLAN, bluetooth, GPRS, UMTS)
- **Reti backbone**
 - Costituiscono la dorsale della rete vera e propria
 - Sono strutturate in sottoreti tra loro interconnesse
 - Si collegano alle reti di accesso



Commutazione di circuito

Nelle reti a commutazione di circuito, la capacità trasmissiva all'interno della rete è assegnata per ciascuna "chiamata". È definita una porzione di capacità trasmissiva che è allocata in modo esclusivo per servire ciascuna comunicazione. È il modello dell'attuale rete telefonica.

Nella telefonia tradizionale, due coppie di conduttori venivano impegnate per ciascuna conversazione. Successivamente, le coppie "fisiche" sono state sostituite da "**porzioni di banda**" (multiplazione a divisione di frequenza, FDM) o "**porzioni di tempo**" (multiplazione a divisione di tempo, TDM)



Commutazione di pacchetto

La natura discontinua della trasmissione di dati digitali può essere sfruttata per far sì che flussi di dati differenti possano condividere la stessa connessione, a patto di poterli distinguere. Questo principio è alla base della tecnica detta commutazione di pacchetto (*packet switching*).

Nel *packet switching* ciascun flusso di dati è diviso in pacchetti, cioè in entità composte da:

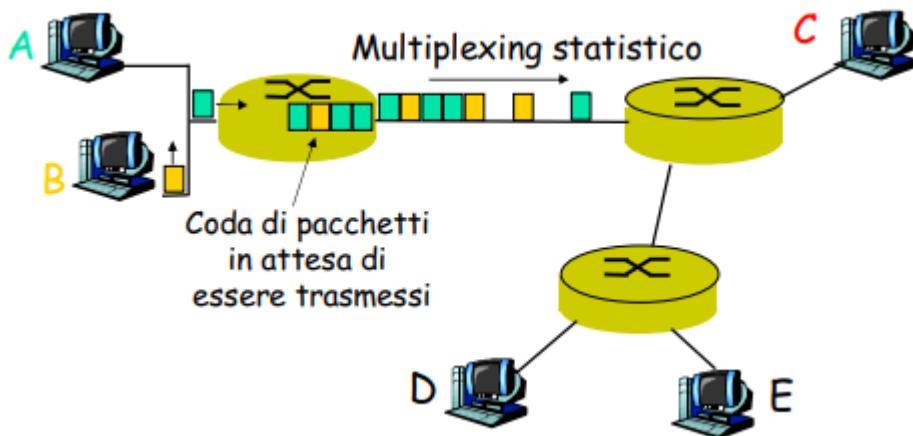


- un'intestazione (**header**), utilizzata ai fini dell'identificazione e gestione
- i dati veri e propri (**payload**)

Una rete a commutazione di pacchetto è composta da:

- sistemi terminali (**end system** o **host**): producono o ricevono dati
- apparati che si occupano dell'instradamento dei pacchetti tra sorgente e destinazione, detti nodi della rete (**network nodes**)

Ogni nodo memorizza i pacchetti in ingresso, per poi instradarli verso il nodo successivo (*store & forward*). I collegamenti fisici tra i nodi sono detti **link**



La qualità del servizio di una rete a commutazione di pacchetto è misurata da una molteplicità di "indici di prestazione".

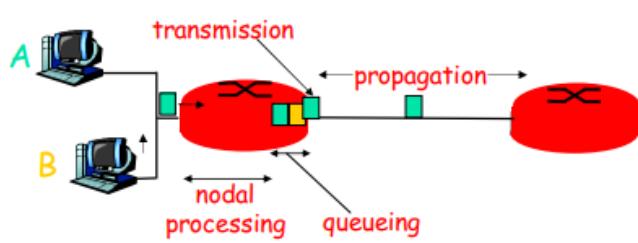
I più importanti sono:

- **Ritardo** nella consegna dei pacchetti [s]
- **Throughput**: quantità di bit al secondo che la rete è in grado di trasferire tra due terminali [b/s]
- **Loss-Rate**: probabilità che un pacchetto non venga consegnato a destinazione
- **Jitter**: variazione temporale del ritardo

Ritardo nelle reti a commutazione di pacchetto

Il ritardo nella consegna di un pacchetto alla destinazione è determinato da:

- **Tempo di elaborazione** nel nodo: controllo di errori, determinazione link di uscita, etc...
- **Tempo di trasmissione** su ciascun link [lunghezza in bit / velocità in bps]
- **Tempo di attesa nelle code dei router** (variabile)
- **Tempo di propagazione** sulle linee [lunghezza della linea / velocità del segnale]

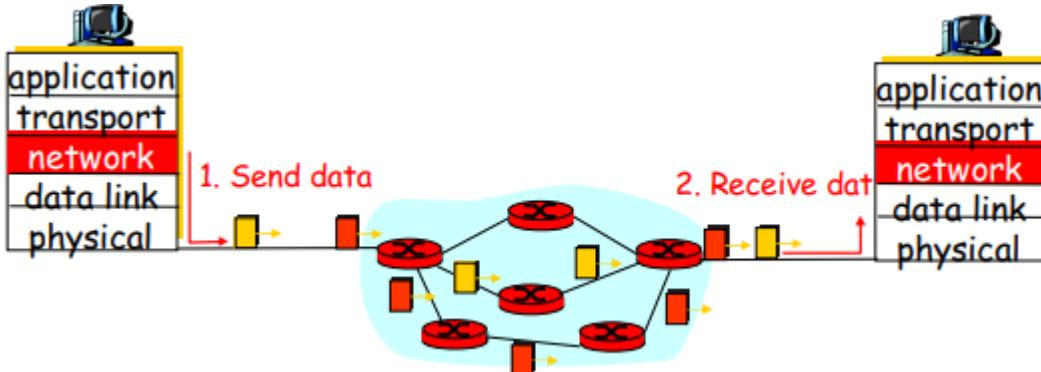


Packet switching: reti a datagrammi

Ogni nodo che riceve un pacchetto decide in maniera **indipendente** a quale altro nodo inoltrarlo, sulla base dell'indirizzo destinazione contenuto nel pacchetto

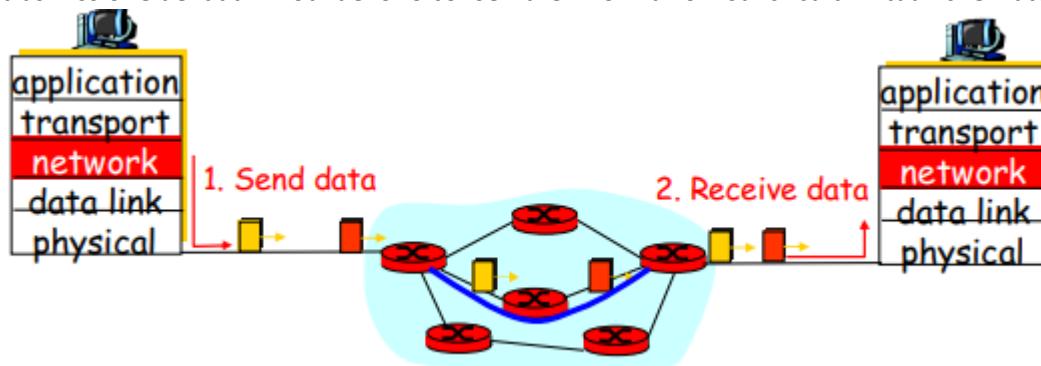
- **Indipendente** rispetto agli altri nodi
- **Indipendente** rispetto agli altri pacchetti passanti per lo stesso nodo

Pacchetti tra la stessa coppia sorgente-destinazione possono seguire percorsi differenti



Packet switching: reti a circuiti virtuali

Ogni pacchetto contiene il numero del circuito virtuale. Il circuito virtuale è stabilito prima della trasmissione dei dati. I nodi devono conservare informazioni sui circuiti virtuali che li attraversano.

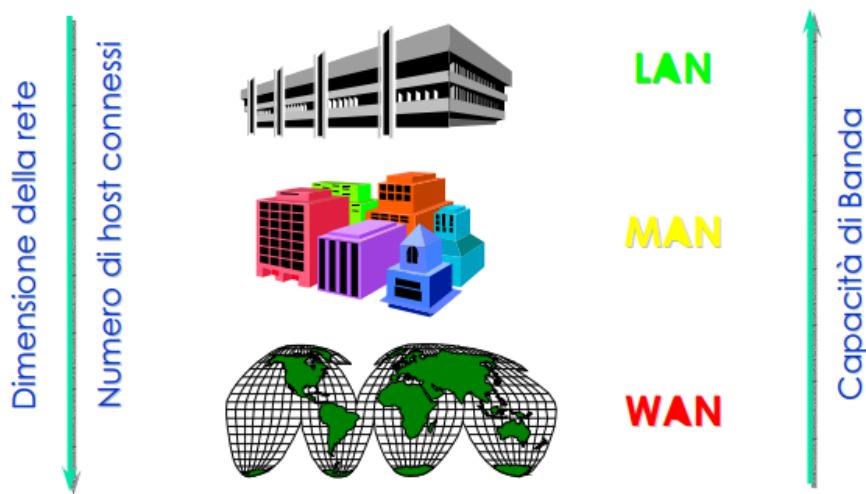


Datagrammi vs circuiti virtuali

Proprietà	Datagrammi	Circuiti virtuali
Creazione del circuito	Non richiesta	Richiesta
Indirizzamento	<i>Ogni pacchetto contiene l'indirizzo della sorgente e della destinazione</i>	Ogni pacchetto contiene un numero di circuiti virtuali
Informazioni sullo stato	I nodi di rete non mantengono informazioni sullo stato	<i>Ogni circuito virtuale richiede uno spazio di memoria sui nodi</i>
Instradamento	<i>Ogni pacchetto è instradato indipendentemente</i>	Percorso pre-calcolato: ogni pacchetto segue questo percorso
Effetti di guasti ai nodi	Nessuno (solo i pacchetti persi durante il guasto)	<i>Tutti i circuiti virtuali che attraversano quel nodo sono chiusi</i>
Controllo di congestione	Complicato	Semplice se possiamo allocare spazio sufficiente per ogni circuito virtuale

Nota: Le proprietà descritte in corsivo sono negative (quindi peggiori rispetto all'altro principio di commutazione)

Le reti di calcolatori



Caratteristiche LAN

- Non attraversano suolo pubblico
- Velocità trasmissiva "V" molto elevata
- Distanze "D" ridotte
- Conformità: conformi a standard emessi da ISO/IEEE/ANSI; non conformi agli standard CCITT

Caratteristiche MAN

- Installazioni in ambito urbano
- Velocità trasmissiva "V" elevata
- Conformità: conformi sia standard CCITT sia ISO/IEEE
- Mezzo trasmissivo tipico: fibra ottica

Caratteristiche WAN

- Installazioni in ambito interurbano
- Velocità trasmissiva "V" inferiore a quella delle LAN
- Conformità: conformi a standard CCITT
- Mezzi trasmissivi: spesso gli stessi usati per la telefonia convenzionale

Reti eterogenee

Gli esempi presentati non sono mutuamente esclusivi. Nella maggior parte dei casi, soluzioni architetturali differenti coesistono in un singolo sistema distribuito complesso. Internet ne è l'esempio per eccellenza.

2. Modelli a Strati e Protocolli

Gestire la complessità delle reti di calcolatori

La comunicazione tra computer richiede soluzioni tecniche complesse riguardanti una serie di problemi:

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

Un approccio logico è quello di analizzare tali problematiche singolarmente. Nelle reti di calcolatori il "*Divide et Impera*" ha condotto a modelli a strati.

Modelli a strati

Come vedremo, la suddivisione delle funzionalità secondo un modello a strati agevola la gestione della complessità.

Ciascuno **strato** (o **livello**):

- è responsabile di un sottoinsieme definito e limitato di compiti
- funziona in maniera non strettamente accoppiata con gli altri
- interagisce solo con gli strati immediatamente superiore e inferiore
- fa affidamento sui servizi forniti dallo strato immediatamente inferiore
- fornisce servizi allo strato immediatamente superiore

Alcuni strati sono realizzati in software, altri in hardware.

Vantaggi:

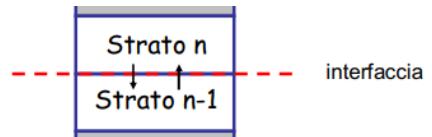
- l'indipendenza tra gli strati consente la sostituzione di uno strato con un altro di pari livello che offre i medesimi servizi allo strato superiore
- limitare le funzionalità di uno strato ne semplifica la realizzazione

Svantaggi:

- L'eccessivo numero di strati può portare ad inefficienze

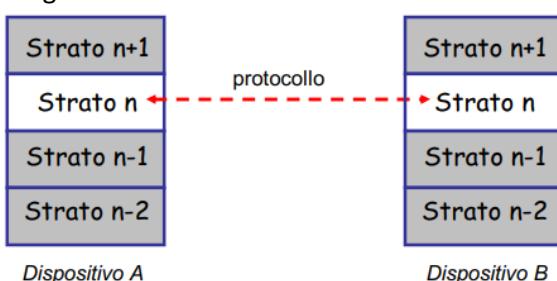
Interfacce

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



Protocolli

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



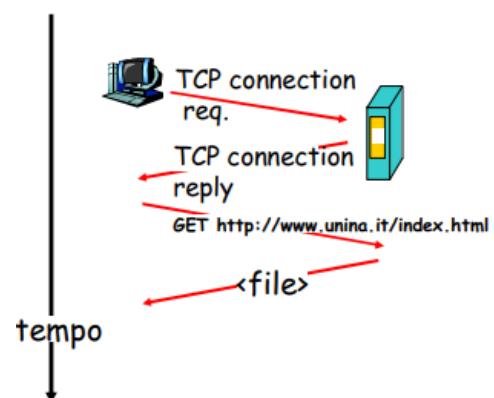
Protocolli di comunicazione

Per **protocollo di comunicazione** si intende un insieme di regole che permette la corretta instaurazione, mantenimento e terminazione di una comunicazione di qualsiasi tipo tra due o più entità.

Un protocollo di comunicazione definisce il formato e l'ordine dello scambio di messaggi tra le entità comunicanti.

Nelle reti di calcolatori, un protocollo regola la comunicazione tra entità di pari livello esistenti in due dispositivi della rete tra loro comunicanti.

Nell'ambito delle reti di computer un notevole sforzo è stato compiuto per definire protocolli standard, allo scopo di consentire l'integrazione di reti differenti.



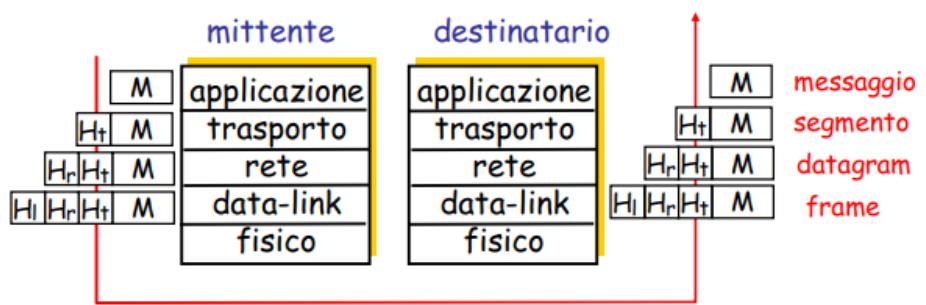
Imbustamento dei messaggi

In trasmissione, ogni strato antepone una intestazione (*header*) al messaggio ricevuto dallo strato soprastrante (paragonabile alla busta di una lettera).

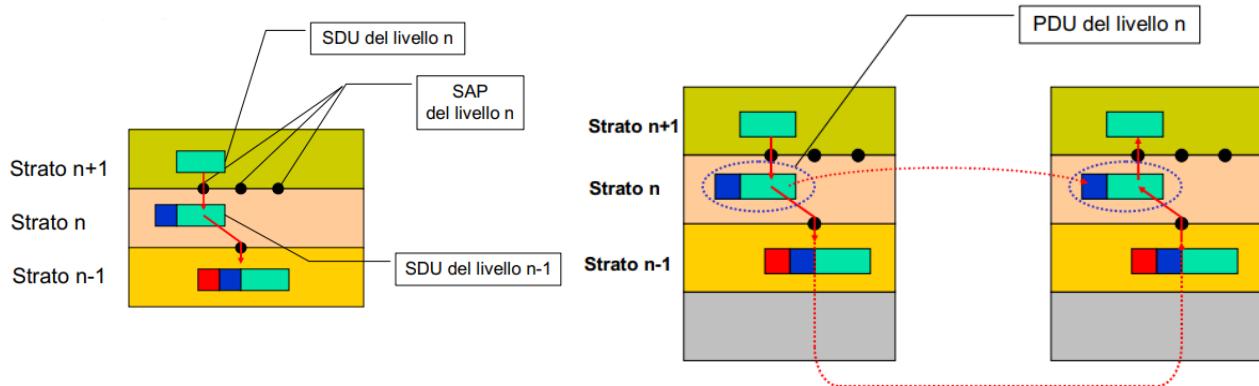
L'insieme messaggio+header viene passato allo strato sottostante.

A destinazione il messaggio risale la pila

In ricezione, ad ogni strato l'header viene rimosso



Service Access Point (SAP), Service Data Unit (SDU), Protocol Data Unit (PDU)

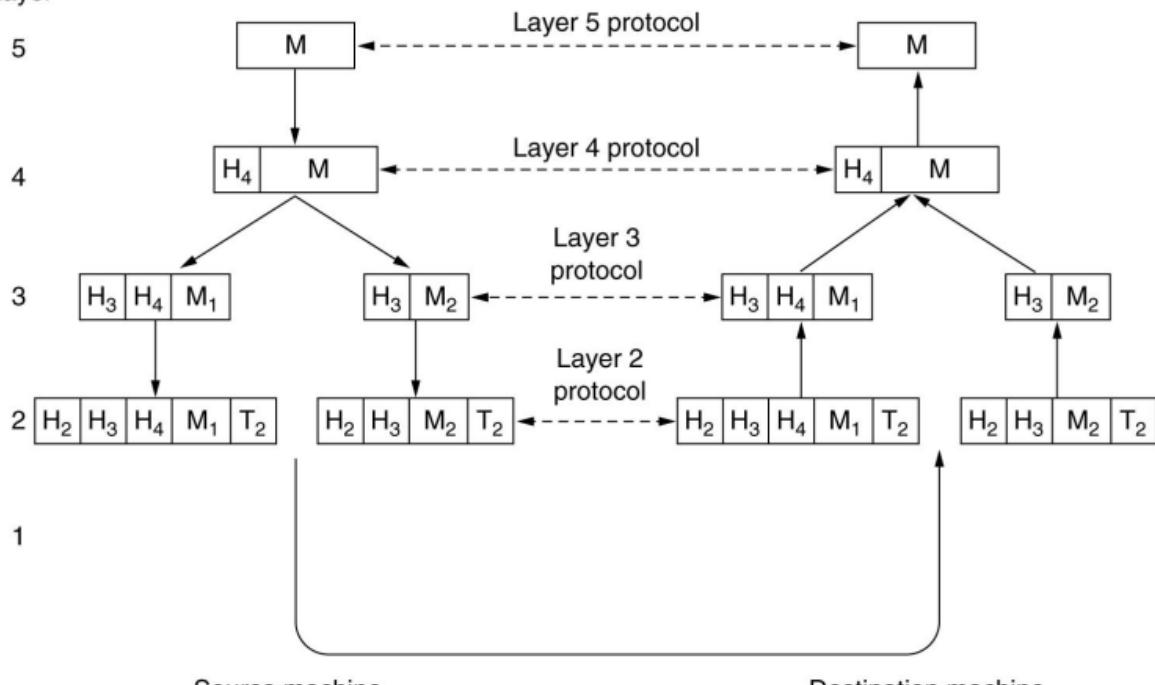


Frammentazione dei messaggi ad un livello

Un livello della pila protocollare può essere costretto a frammentare il pacchetto ricevuto dallo strato superiore prima di passarlo allo strato inferiore.

Si rende necessaria una operazione di ricostruzione mediante riassemblaggio.

Layer



3. Modello ISO-OSI

Il modello OSI

Negli anni '80 l'ISO (International Standards Organization) ha definito un modello di riferimento per reti di calcolatori a commutazione di pacchetto: il modello OSI (Open System Interconnection).

Il modello OSI è un modello a strati su 7 livelli:

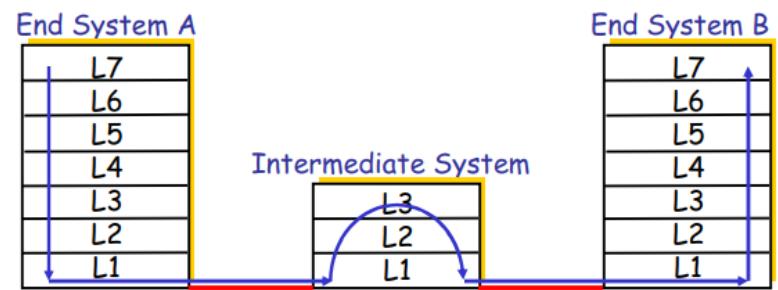
- Applicazione
- Presentazione
- Sessione
- Trasporto
- Rete
- Data link
- Fisico



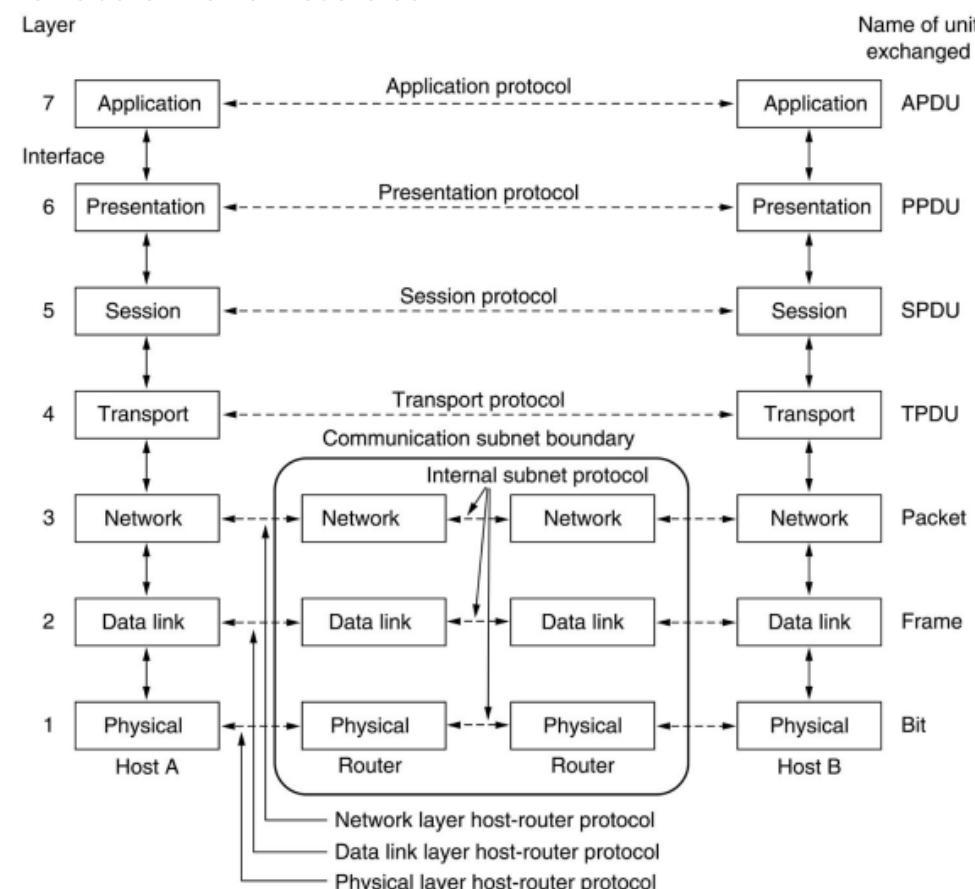
Questo modello non è risultato vincente, a causa della sua eccessiva complessità.

L'intera pila di livelli è realizzata negli end system, mentre, i dispositivi di rete si differenziano per il numero di livelli fino a cui operano:

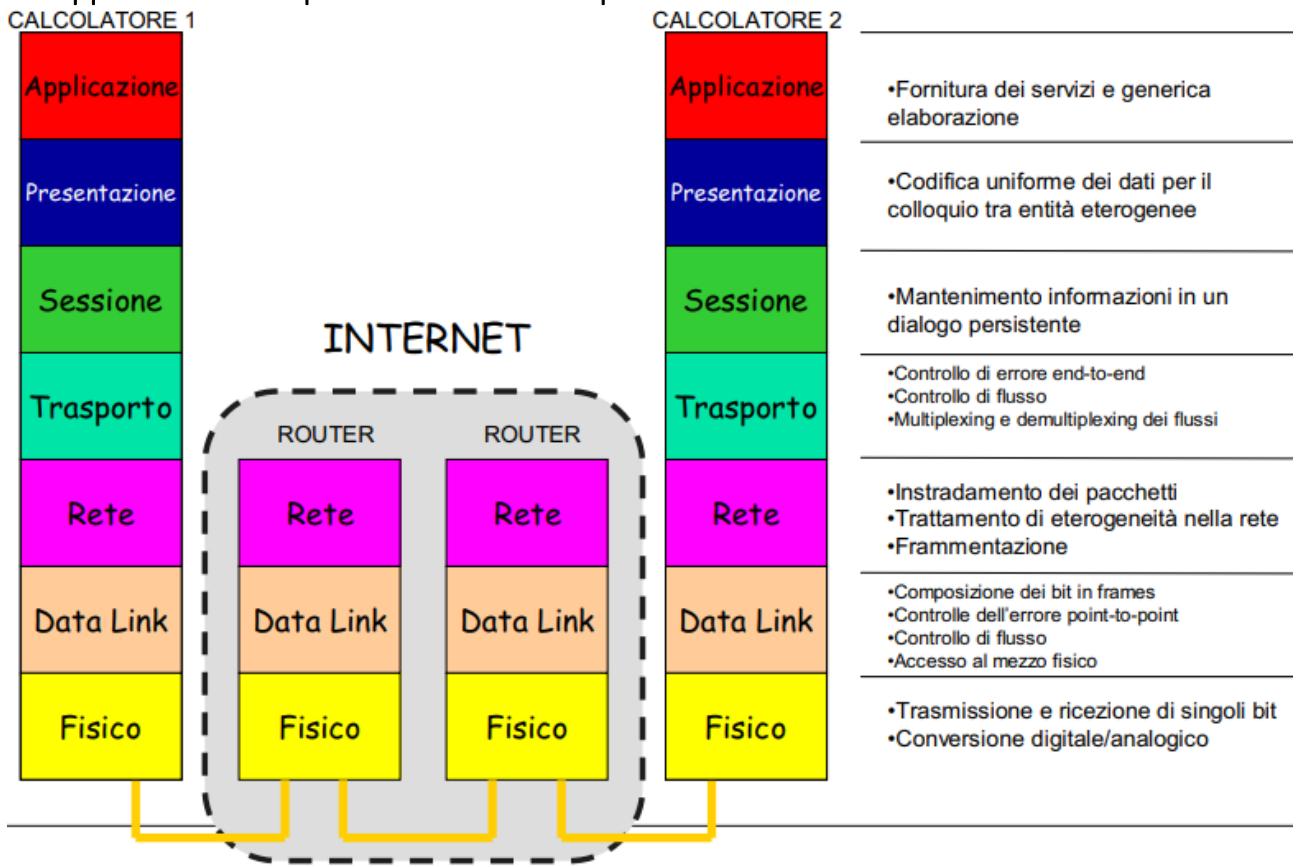
- fino a L1 operano i ripetitori
- fino a L2 operano i bridge/switch di rete locale
- fino a L3 operano i router.



Nome delle PDU nel modello OSI



Un approccio a livelli per la risoluzione dei problemi



Livello 1: Fisico

- Si occupa di trasmettere sequenze binarie sul canale di comunicazione
- A questo livello si specificano:
 - Caratteristiche elettriche dei segnali
 - Tecniche di codifica/decodifica
 - Caratteristiche dei mezzi trasmittivi
 - Tipi di connettori
- Il livello fisico è nel dominio dell'ingegneria elettronica: descrizione elettrico/meccanica dell'interfaccia

Livello 2: Data Link

- Ha come scopo la trasmissione affidabile di pacchetti di dati (*frame*). Affidabile nel senso di “garanzia di inoltro”
- Accetta come input i frame (tipicamente poche centinaia di byte) e li trasmette sequenzialmente
- Verifica la presenza di errori di trasmissione aggiungendo delle informazioni aggiuntive di controllo FCS (Frame Control Sequence)
- Può gestire meccanismi di correzione di errori tramite trasmissione

Livello 3: Rete

- Questo livello gestisce l'instradamento dei messaggi
- Determina quali sistemi intermedi devono essere attraversati da un messaggio per giungere a destinazione
- Il livello 3 gestisce, quindi, delle tabelle di instradamento per ottimizzare il traffico sulla rete

Livello 4: Trasporto

- Fornisce servizi per il trasferimento dei dati da terminale a terminale (ovvero *end-to-end*), indipendentemente dalla rete sottostante
- In particolare, il livello 4 può:
 - frammentare i pacchetti in modo che abbiano dimensioni idonee al livello 3
 - rilevare/correggere gli errori
 - controllare il flusso
 - controllare le congestioni (affollamento del traffico)

Livello 5: Sessione

- Il livello 5 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

Livello 6: Presentazione

- Il livello di presentazione gestisce la sintassi dell'informazione da trasferire
- L'informazione è infatti rappresentata in modi diversi su elaboratori diversi (es. ASCII o EBCDIC)

Livello 7: Applicazione

- È il livello dei programmi applicativi, cioè di quei programmi appartenenti al sistema operativo o scritti dagli utenti, attraverso i quali l'utente finale utilizza la rete
 - Esempi di applicazioni previste dall'OSI sono: Virtual Terminal, connessione interattiva ad un elaborato remoto, File Transfer and Access Management, etc...
 - Nel mondo Internet, le applicazioni sono: WWW, Mail, News, FTP, IRC, NTP, etc...

Internet

Origini ed Evoluzione

Internet nasce come interconnessione di diverse reti di carattere sperimentale; la fattibilità delle reti a commutazione di pacchetto è stata mostrata da Kleinrock attraverso la teoria delle code.

Le prime attività di rilievo come sperimentazione di reti a commutazione di pacchetto finanziate per scopi militari con il protocollo ARPANET, inadeguato per differenti architetture.

Dopo ARPANET, Cerf e Kahn progettano la suite TCP-IP, protocollo dove tutt'ora si basa la rete.

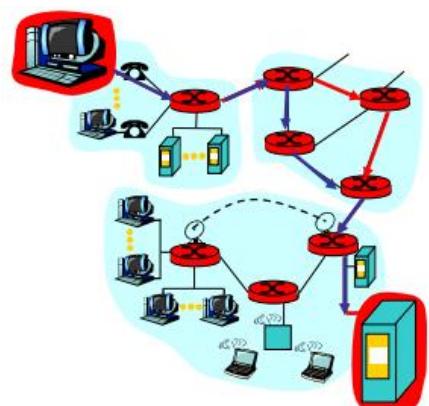
Architettura della rete

Internet è una architettura completa che va dai servizi alle applicazioni. È di dominio pubblico e di enorme diffusione; inoltre, è in continua evoluzione con i requisiti degli utenti.

I protocolli in uso sono standardizzati mediante documenti approvati dall'IETF (uno degli organismi che gestisce l'internet, fa parte dell'*Internet Society ISOC*).

La rete è progettata secondo un modello a **datagram**:

- L'informazione viaggia in pacchetti (datagram) che vengono trattati dalla rete indipendentemente l'uno dagli altri
- Ogni terminale è univocamente individuato da un indirizzo associato alla interfaccia che lo collega alla rete
- Ogni pacchetto contiene l'indirizzo del mittente e l'indirizzo del destinatario
- L'infrastruttura della rete è costituita dai **router** che hanno il compito di instradare i pacchetti e consegnarli a destinazione



- Non c'è garanzia che un pacchetto venga realmente consegnato a destinazione
 - I pacchetti possono andare persi nella rete
 - I pacchetti possono seguire percorsi diversi ed arrivare in un ordine diverso da quello con cui sono stati trasmessi

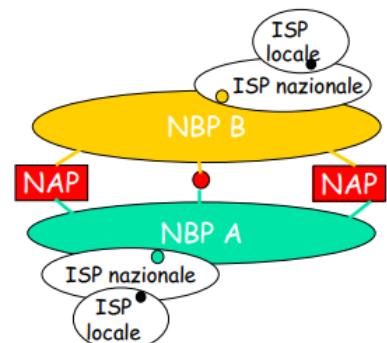
Struttura di Internet: rete di reti

L'accesso ad Internet avviene per mezzo di un fornitore di servizi o *Internet Service Provider (ISP)*.

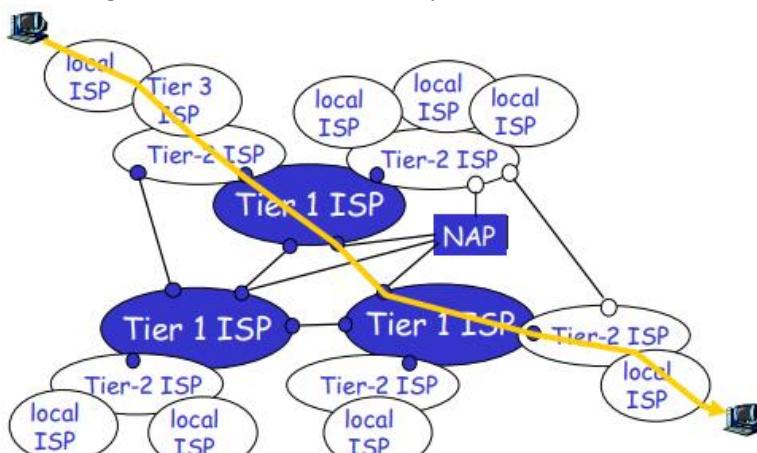
Gli ISP sono collegati tra loro secondo una struttura gerarchica: ISP locali e ISP nazionali.

Gli ISP nazionali si collegano a fornitori di connettività internazionali: i *Network Backbone Provider (NBP)*.

Gli NBP sono tra loro collegati in unità di interscambio detti NAP (*Network Access Point*)



- Dalla sorgente alla destinazione, un pacchetto attraversa diverse reti:



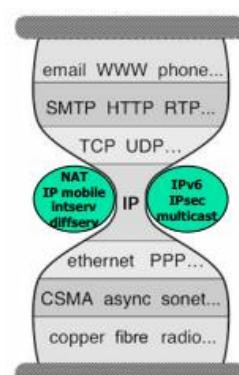
Lo stack TCP/IP

Internet si basa su un modello definito da una collezione di protocolli standardizzati dall'IETF, il modello **TCP/IP**.

Siccome i protocolli sono organizzati secondo una struttura a pila (stack), si parla dello **stack TCP/IP**.

Il modello prende il nome da due protocolli fondamentali:

- TCP, *Transmission Control Protocol*, di livello Trasporto
- IP, *Internet Protocol*, di livello Rete

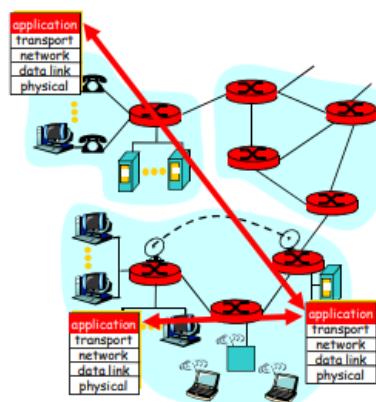


4. Protocolli applicativi: HTTP

Strato dell'applicazione

Comunicazione tra processi:

- API: Application Programming Interface
 - Definisce l'interfaccia tra il livello applicativo e il livello trasporto
 - Socket (Internet API): due processi comunicano mandando dati in un socket, e leggendo dati dal socket



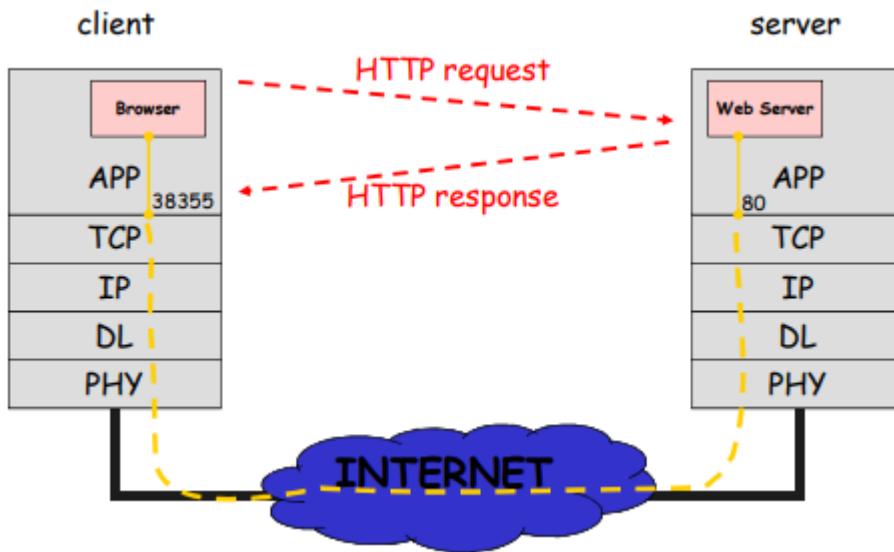
- Come un processo può identificare l'altro processo con cui vuole comunicare?
 - Indirizzo IP dell'host che esegue l'altro processo
 - Numero di porta che permette all'host ricevente di sapere a quale processo locale il messaggio è destinato

Cosa definisce un protocollo dello strato di applicazione:

- Tipo di messaggi scambiati
- Sintassi dei messaggi
- Semantica dei campi
- Regole di trasmissione

Lo strato applicativo riguarda sia applicazioni lato client che lato server (es.: servizi web, FTP).

Di seguito un esempio di interazione Client->Server (Web):



HTTP

Il protocollo HTTP/1.0 si basa su TCP:

- Il client apre un socket verso il porto TCP 80 del server (se non diversamente specificato)
- Il server accetta la connessione
- Il client manda una richiesta per uno specifico oggetto identificato mediante una URL
- Il server risponde e chiude la connessione

Il protocollo HTTP è **stateless**: né il server né il client mantengono a livello http informazioni relative ai messaggi precedentemente scambiati.

URL: Uniform Resource Locator

Un URL HTTP ha la seguente sintassi:

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

- **host** identifica il server, può essere sia un nome simbolico che un indirizzo IP in notazione dotted decimal
- **port** è opzionale, di default è 80
- **path** identifica la risorsa sul server (es.: images/sfondo.gif)
- **#fragment** identifica un punto preciso all'interno di un oggetto (es.: #paragrafo1)
- **?query** è usato per passare informazioni dal client al server (es.: dati inseriti nei campi di una form)

HTTP per il trasferimento di pagine web

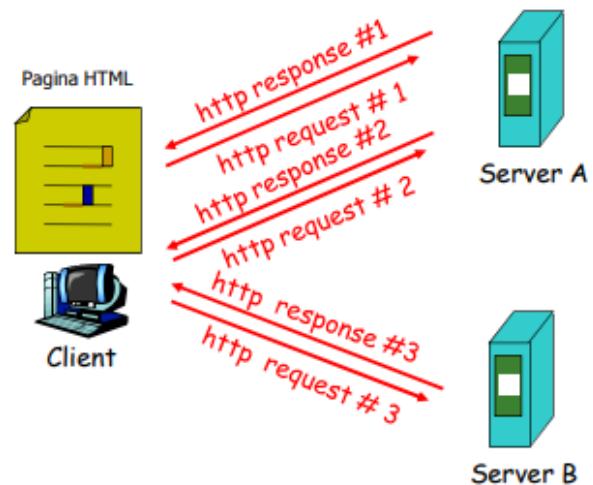
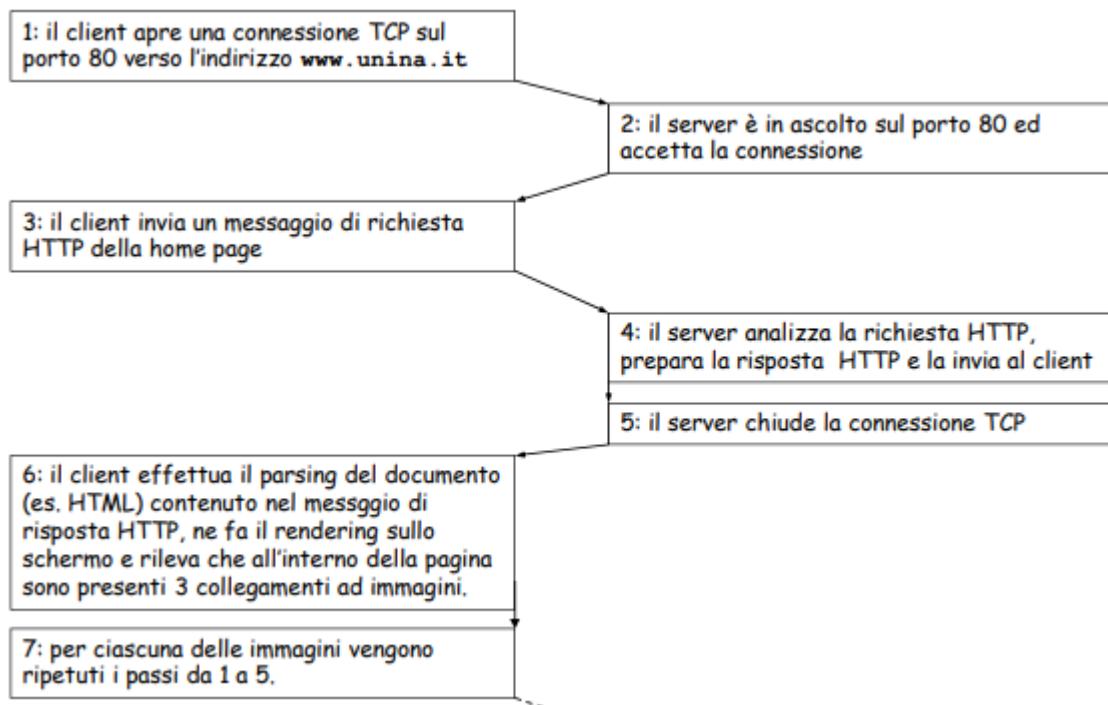
Tipicamente, una pagina web è descritta da un file testuale in formato HTML (*Hypertext Markup Language*) ed è identificata mediante un indirizzo, detto URL.

Un file HTML può contenere riferimenti ad altri oggetti che arricchiscono la pagina con elementi grafici, ad esempio: sfondo, immagini, etc...

Ciascun oggetto è identificato dal proprio URL e tali oggetti possono trovarsi anche su server web diversi.

Una volta ricevuta la pagina HTML, il browser estrae i riferimenti agli altri oggetti che devono essere prelevati e li richiede attraverso una serie di connessioni HTTP.

Esempio di richiesta di una pagina contenente immagini:



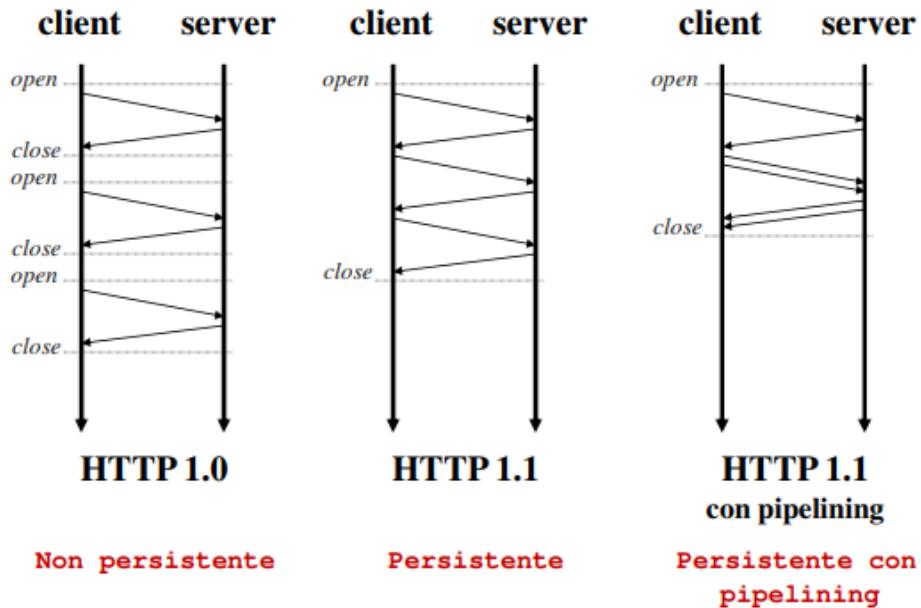
Connessioni HTTP

Connessione non persistente

- HTTP/1.0
- il server analizza una richiesta, la serve e chiude la connessione
- 2 Round Trip Time (RTT) per ciascuna richiesta
- Ogni richiesta subisce lo slow-start TCP

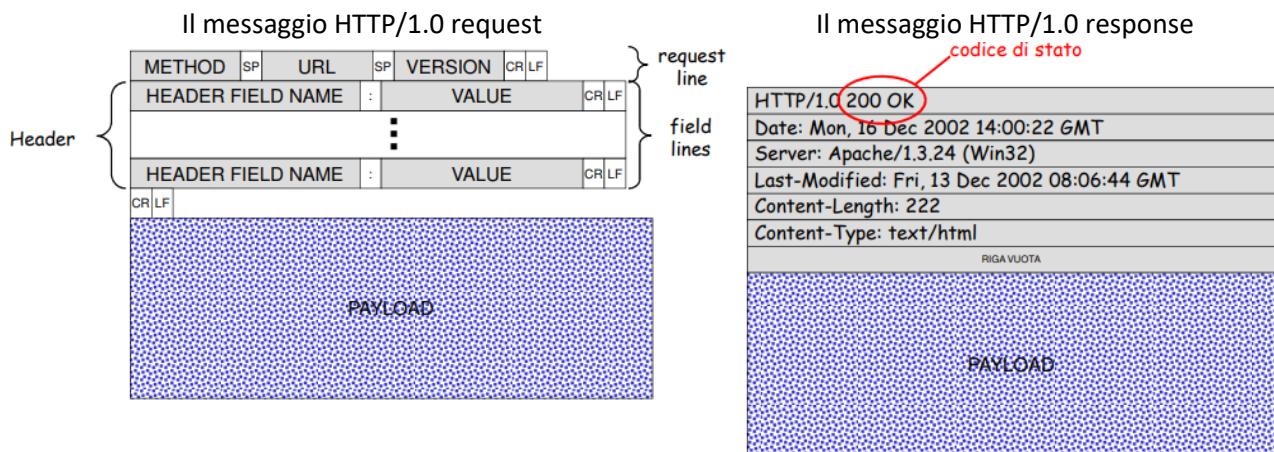
Connessione persistente

- HTTP/1.1
- Sulla stessa connessione il server analizza tutte le richieste e le serve
- Il client riceve la pagina iniziale e invia subito tutte le altre richieste
- Si hanno meno RTT ed un solo slow-start
- Esiste anche una versione con parallelismo (with pipelining)



Il protocollo HTTP

HTTP è un protocollo testuale dove i messaggi sono costituiti da sequenze di byte, dove ogni byte identifica un carattere secondo la tabella ASCII. Il payload dei messaggi può essere comunque anche in formato binario (es.: un immagine GIF, un video, etc...)



Lo scambio di messaggi

Il metodo GET

- Uno dei più importanti metodi di HTTP è GET
- Usato per richiedere una risorsa ad un server
- Questo è il metodo più frequente, ed è quello che viene attivato facendo click su un link ipertestuale di un documento HTML, o specificando un URL nell'apposito campo di un browser
- Get può essere:
 - **assoluto**: la risorsa viene richiesta senza altre specificazioni
 - **condizionale**: si richiede la risorsa se è soddisfatto un criterio indicato negli header
 - **parziale**: si richiede una sottoparte di una risorsa memorizzata

Il metodo HEAD

- Simile al metodo GET, ma il server deve rispondere soltanto con gli header relativi, senza il corpo
- Usato per verificare:
 - **la validità di un URI**: la risorsa esiste e non è di lunghezza zero

- **L'accessibilità di un URI:** la risorsa è accessibile presso il server, e non sono richieste procedure di autenticazione del documento
- **la coerenza di cache di un URI:** la risorsa non è stata modificata nel frattempo, non ha cambiato lunghezza, valore hash o data di modifica

Il metodo POST

- Serve per trasmettere delle informazioni dal client al server, ma senza la creazione di una nuova risorsa
- Viene usato, ad esempio, per sottomettere i dati di una form HTML ad un'applicazione sul server
- I dati vengono trasmessi nel body della richiesta
- Il server può rispondere positivamente in tre modi:
 - **200 Ok:** dati ricevuti e sottomessi alla risorsa specificata; è stata data risposta
 - **201 Created:** dati ricevuti, la risorsa non esisteva ed è stata creata
 - **204 No content:** dati ricevuti e sottomessi alla risorsa specificata; non è stata data risposta
- Non è l'unico modo per inviare dati

Il metodo PUT

- Serve per trasmettere delle informazioni dal client al server, creando o sostituendo la risorsa specificata
- In generale, l'argomento del metodo PUT è la risorsa che ci si aspetta di ottenere facendo un GET in seguito con lo stesso nome

La response

- La risposta HTTP è un messaggio testuale formato da una riga iniziale, da header facoltativi ed eventualmente un body (corpo)

Version status-code reason-phrase CRLF } request
 [Header] }
 [Header]
 CRLF

- **Body**

dove:

- **[Header]** indica un elemento opzionale
- **CRLF** indica la sequenza di caratteri di codice ASCII

Status code:

- Lo status code è un numero di tre cifre, di cui la prima indica la classe della risposta, e le altre due la risposta specificata
- Esistono le seguenti classi:
 - **1xx: Informational**
una risposta temporanea alla richiesta, durante il suo svolgimento
 - **2xx: Successful**
il server ha ricevuto, capito e accettato la richiesta
 - **3xx: Redirection**
il server ha ricevuto e capito la richiesta, ma sono necessarie altre azioni da parte del client per portare a termine la richiesta
 - **4xx: Client error**
la richiesta del client non può essere soddisfatta per un errore da parte del client (errore sintattico o richiesta non autorizzata)
 - **5xx: Server error**
la richiesta può anche essere corretta, ma il server non è in grado di soddisfare a richiesta per un problema interno

Gli header

Gli header di risposta

- Gli header della risposta sono posti dal server per specificare informazioni sulla risposta e su sé stesso al client
 - **Server:** una stringa che descrive il server (tipo, sistema operativo e versione)
 - **Accept-ranges:** specifica che tipo di range può accettare (valori previsti: byte e none)

Gli header generali

- Gli header generali si applicano solo al messaggio trasmesso e si applicano sia ad una richiesta che ad una risposta, ma non necessariamente alla risorsa trasmessa
- **Date:** data ed ora della trasmissione
- **MIME-Version:** la versione MIME usata per la trasmissione (sempre 1.0)
- **Transfer-Encoding:** il tipo di formato di codifica usato per la trasmissione
- **Cache-Control:** il tipo di meccanismo di caching richiesto o suggerito per la risorsa
- **Connection:** il tipo di connessione da usare
 - **Keep-Alive:** tenere attiva dopo la risposta
 - **Close:** chiudere dopo la risposta
- **Via:** usato da proxy e gateway

Gli header dell'entità

- Gli header dell'entità danno informazioni sul body del messaggio, o, se non vi è body, sulla risorsa specificata
- **Content-Type:** oggetto/formato
 - Ogni coppia oggetto/formato costituisce un tipo MIME dell'entità acclusa
 - Specifica se è un testo, se un'immagine GIF, un'immagine JPG, un suono WAV, etc...
 - Obbligatorio in ogni messaggio che abbia un body
- **Content-Length:** la lunghezza in byte del body
 - Obbligatorio, soprattutto se la connessione è persistente
- **Content-Base, Content-Encoding, Content-Language, Content-Location, Content-MD5, Content-Range:** l'URL di base, la codifica, il linguaggio, l'URL della risorsa specifica, il valore di digest MD5 e il range richiesto della risorsa, rispettivamente.
- **Expires:** una data dopo la quale la risorsa è considerata non più valida (e quindi va richiesta o cancellata dalla cache)
- **Last-Modified:** la data e l'ora dell'ultima modifica
 - Serve per decidere se la copia posseduta (es. in cache) è ancora valida o no
 - Obbligatorio se possibile

I cookies

Poiché HTTP è stateless, il server non è tenuto a mantenere informazioni su connessioni precedenti.

Un **cookie** è una breve informazione scambiata tra il server ed il client, tramite il cookie il client mantiene lo stato di precedenti connessioni, e lo manda al server di pertinenza ogni volta che richiede un documento (un esempio è quando viene riproposta la propria username all'accesso di un sito)

Il meccanismo dei cookies definisce due nuovi possibili header:

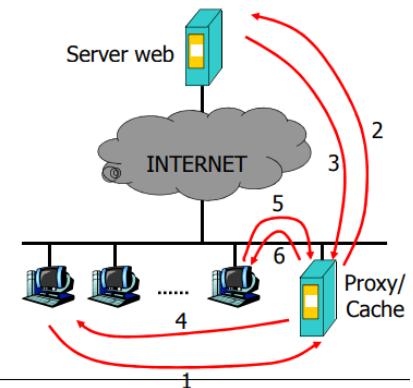
- **Set-Cookie:** header della risposta
 - il client può memorizzarlo (se vuole) e rispedirlo alla prossima richiesta
- **Cookie:** header della richiesta
 - Il client decide se spedirlo sulla base del nome del documento, dell'indirizzo IP del server, e dell'età del cookie.

Un browser può essere configurato per accettare o rifiutare i cookies mentre alcuni siti web richiedono necessariamente la capacità del browser di accettare cookies

Web caching

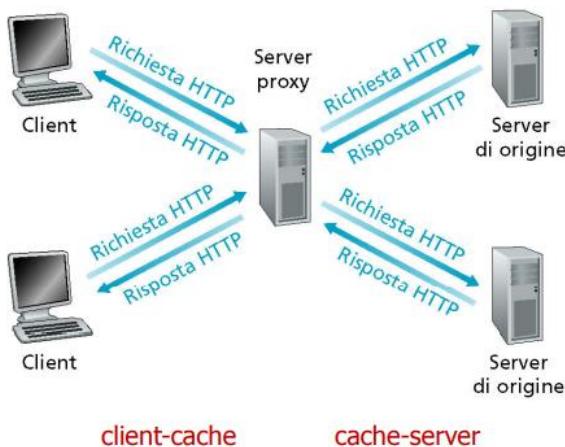
Si parla genericamente di Web caching quando le richieste di un determinato client non raggiungono il web server, ma vengono intercettate da una **cache**. Tipicamente, un certo numero di client di una stessa rete condivide una stessa cache web, posta nelle loro prossimità (es. nella stessa LAN)

Se l'oggetto richiesto non è presente nella cache, questa lo richiede in vece del client conservandone una copia per eventuali richieste successive. Richieste successive alla prima sono servite più rapidamente.

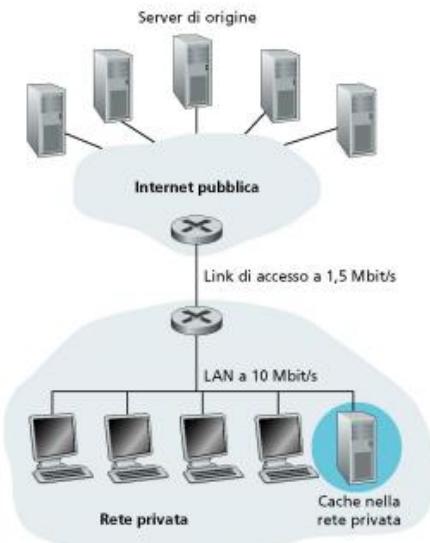


Server proxy

Schema logico:

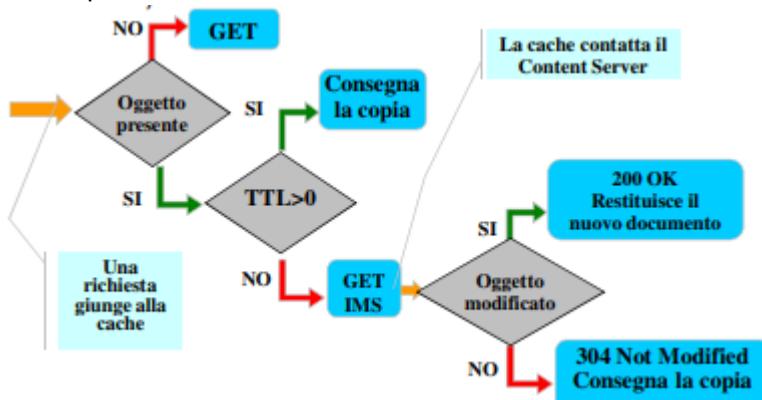


In una rete di accesso:



Gestione della coerenza: cosa succede se l'oggetto presente nel server è aggiornato?

- La copia in cache deve essere aggiornata per mantenersi uguale all'originale
- HTTP fornisce due meccanismi per la gestione della coerenza:
 - **TTL** (Time To Live): il server quando fornisce un oggetto dice anche quando quell'oggetto scade (header Expires). Quando TTL diventa minore di zero, non è detto in realtà che l'oggetto sia stato realmente modificato
 - Il client può fare un ulteriore **controllo mediante una GET condizionale** (*if-Modified-Since*):

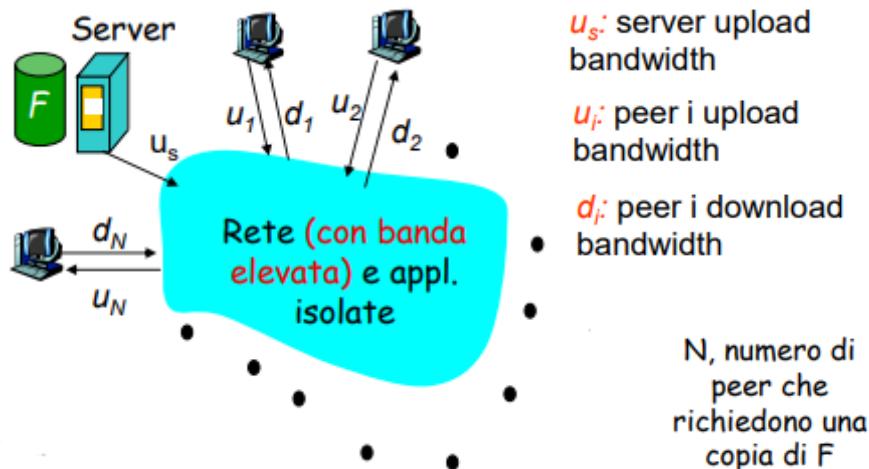


5. Peer-to-Peer Networks (P2P)

Nelle architetture p2p non esistono server sempre connessi (always-on server) ma sono gli end-system (peer) a comunicare direttamente tra loro. I peer sono connessi ad intermittenza e cambiano il proprio IP.

Distribuzione dei file

Quanto tempo serve per distribuire un file da un server ad N peer (tempo di distribuzione, D)?



Caso server-client:

- Il server invia sequenza N copie del file: tempo richiesto NF/u_s
- Il client i impiega un tempo pari a F/d_i per il download

Approccio client/server

$$\text{Tempo per distribuire} \\ F \text{ bit a } N \text{ client} = D_{cs} = \max \{ NF/u_s, F/\min(d_i) \}$$

per N elevato, il termine NF/u_s è dominante: incremento lineare al crescere di N

Caso P2P:

- Dipende dal numero di peer coinvolti...
- Il server deve inviare almeno una copia. Tempo richiesto: F/u_s
- Il client i impiega un tempo pari a F/d_i per il download (F/d_{\min})
- NF bit totali devono essere scaricati. Velocità massima di upload: $u_s + \sum u_i$

Approccio p2p

$$D_{P2P} = \max \{ F/u_s, F/\min(d_i), NF/(u_s + \sum u_i) \}$$

BitTorrent

è un esempio di p2p file distribution

- **tracker**: tiene traccia dei peer che compongono un “torrente”
- **torrente**: gruppo di peer che si scambiano porzioni (chunk) di uno stesso file
- il file è diviso in **chunk** di 256KB
- quando un peer si aggiunge ad un torrente:
 - si registra presso il tracker per avere la lista dei peer e si connette ad un sottoinsieme di tali peer (“neighbors”)
 - non possiede chunk, ma ne accumulerà nel tempo
- durante il download, il peer esegue l’upload di chunk verso altri peer

- i peer possono attivarsi e disattivarsi dinamicamente
- una volta scaricato l'intero file, il peer può (egoisticamente) abbandonare, o (altruisticamente) rimanere nel torrente

1) prelievo di chunk: tecnica “rarest first”

- peer differenti possiedono differenti sottoinsiemi di chunk del file
- periodicamente, un peer chiede a tutti i neighbor la lista dei chunk in loro possesso
- Tale peer invia richieste per i propri chunk mancanti

2) invio di chunks: tecnica “tit-for-tat”

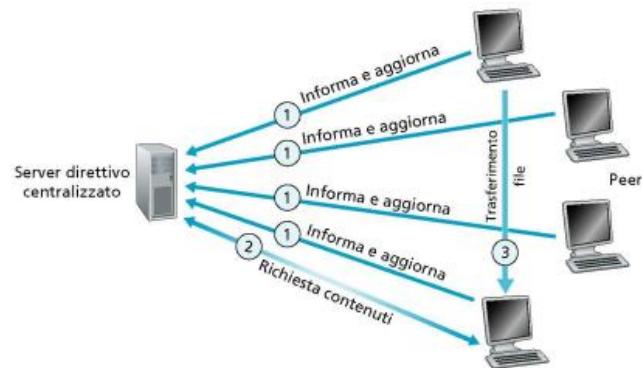
- L’idea di fondo è quella di dare priorità a chi fornisce dati la rate più alto
- Un peer invia chunk ai 4 neighbor attualmente più veloci (che gli inviano chunk al rate più elevato). I top 4 sono ricalcolati ogni 10 secondi
- Ogni 30 secondi si seleziona in maniera casuale un altro peer, e si inizia ad inviargli chunk (il peer appena scelto può essere aggiunto ai top 4)

P2P con directory centralizzata (Napster)

Quando un peer si connette alla rete si collega ad un server centralizzato fornendo il proprio indirizzo IP e il nome degli oggetti resi disponibili per la condivisione. In questo modo il server raccoglie le info sui peer attivi e le aggiorna dinamicamente.

Il trasferimento del file è decentralizzato, ma la localizzazione dei contenuti è pesantemente centralizzata:

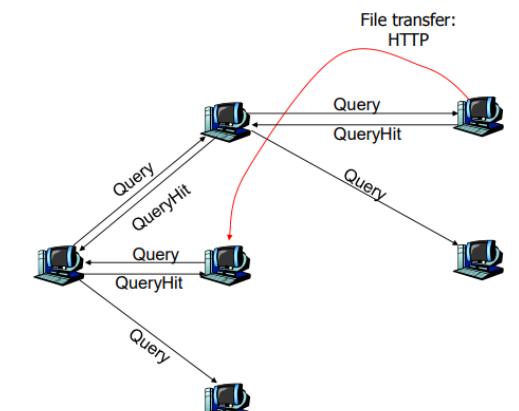
- Singolo punto di fallimento
- Collo di bottiglia per le prestazioni
- Violazione del diritto di autore



P2P con directory decentralizzata (gnutella)

L’architettura è completamente distribuita (senza server centralizzati) e si realizza un’architettura di rete sovrapposta (overlay network, fatta da connessioni TCP in corso). L’overlay network ha una struttura paritetica.

Nonostante la rete possa avere centinaia di migliaia di partecipanti, ogni peer è connesso al massimo a 10 altri peer nella overlay. Causando due problematiche: come viene costruita e gestita la rete di peer e come un peer localizza un contenuto



I peer, una volta unitisi alla rete, inviano richieste mediante la tecnica del flooding (inondazione):

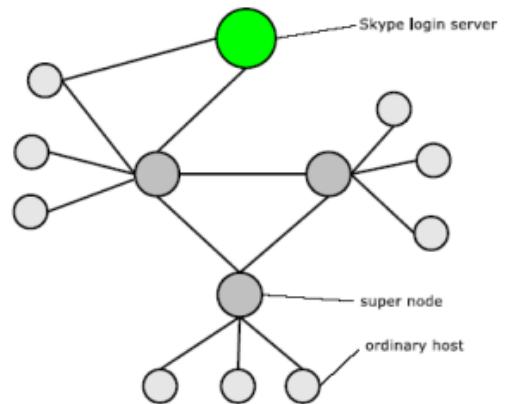
- 1) Il peer X deve trovare altri peer già parte della overlay: mantiene una lista di IP o contatta un sito Gnutella contenente la lista
- 2) Dopo l’accesso alla lista, X tenta di impostare una connessione TCP con i peer della lista; quando si connette a Y si ferma
- 3) X spedisce a Y un ping Gnutella; Y lo inoltra finché il contatore non si azzerà
- 4) Tutti i peer che ricevono un messaggio ping, rispondono con un pong: esso contiene l’indirizzo di chi ha inviato il pong, il numero di file in condivisione, la dimensione totale
- 5) Quando X riceve i messaggi di pong, avendo l’IP, può impostare una connessione TCP con alcuni di essi...
- 6) Ci possono essere più fasi di bootstrap in parallelo

Skype

Skype è una applicazione p2p VoIP sviluppata da KaZaa nel 2003 e supporta anche instant messaging e conferencing. Il protocollo è proprietario. Sfrutta le caratteristiche positive di Napster e Gnutella.

Skype usa una rete overlay (costituita da connessioni TCP tra host ordinari e super nodi), con tre tipi di host:

- host ordinari: utenti Skype
- super nodi: utenti Skype con sufficiente CPU, memoria, banda...
- server di login per l'autenticazione



Ciascun client Skype mantiene una lista (inizialmente vuota) di indirizzi IP di super nodi conosciuti e si connette alla rete attraverso un super nodo.

I super nodi sono responsabili della localizzazione degli utenti, del routing delle chiamate, del mantenimento delle informazioni circa gli host connessi alla rete Skype.

Connessione ai super nodi

- Primo login:
 - Alla prima esecuzione dopo l'installazione un client Skype comunica con il server Skype (skype.com)
 - Durante la comunicazione, la cache dell'host è popolata di 7 indirizzi IP di super nodi da usare per il bootstrap (fase di bootstrap: peer che si connette deve essere associato ad un group leader o deve essere designato group leader)
 - A questo punto l'host può contattare uno di essi per il join
 - Selezionato il super nodo per il join, parte la fase di autenticazione con user name e password con il server Skype
 - L'host viene periodicamente aggiornato con indirizzi IP di nuovi super nodi
- Login successivi:
 - Per i login successivi un client sceglie uno degli indirizzi dei super nodi e stabilisce la connessione

6. Protocolli applicativi: FTP ed SMTP

File Transfer Protocol (FTP)

Internet oggi si presenta come una rete ad estensione globale che connette molti milioni di macchine sparse su tutto il globo. Spesso sorge l'esigenza di copiare un file da una macchina ad un'altra per poterlo utilizzare localmente.

Ciò può accadere sia tra macchine molto distanti tra di loro che tra macchine direttamente connesse, presenti nello stesso locale ed un apposito protocollo chiamato FTP è stato definito a questo scopo.

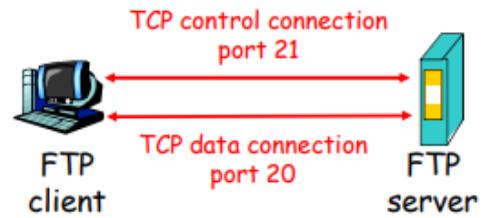
Attraverso tale protocollo è possibile trasferire uno o più files di qualsiasi tipo tra due macchine. Il protocollo FTP lavora utilizzando due connessioni: una dati e una controllo (*out of band*)

Funzionamento del protocollo FTP

- Trasferisce files da o verso una macchina remota
- Usa il modello client/server
 - **client**: è l'entità che dà luogo al trasferimento (sia in un senso che nell'altro)
 - **server**: è l'entità remota che è in continua attesa di connessioni FTP da parte di altre entità
- ftp server: numero di porto 21

Le connessioni di una sessione FTP

- Il client ftp contatta il server ftp al porto 21
- Vengono aperte due connessioni parallele:
 - Controllo:** scambio di comandi, messaggi di risposta tra il client e il server (*controllo out of band, fuori banda*)
 - Dati:** file che fluiscono dal client al server o viceversa
- Un server ftp mantiene uno stato (ad es.: la directory corrente; i dati dell'autenticazione).



Lo scambio delle informazioni avviene tramite codice ASCII: i comandi vengono inviati come testo ASCII sulla connessione di controllo ed anche le risposte sono costituite da testo ASCII.

Server FTP

Non è possibile per un client stabilire una connessione FTP verso una qualsiasi macchina; infatti, il tipo di paradigma adottato (client/server) presuppone che il server debba essere stato opportunamente configurato per accettare connessioni.

Solitamente, per questioni di sicurezza, le macchine non sono configurate per accettare connessioni di tipo FTP; se si tenta di stabilire una connessione verso una macchina non abilitata la sessione fallisce e nessun trasferimento risulta possibile.

I client FTP sono invece disponibili pressocché su tutti i sistemi operativi di rete.

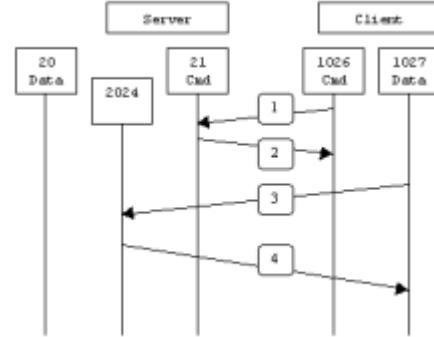
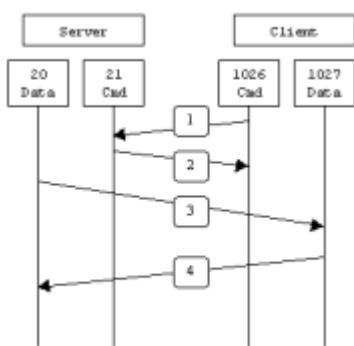
FTP attivo e passivo

Active FTP

- Il client apre una connessione di controllo verso l'indirizzo del server sul porto 21 utilizzando un *ephemeral port*
- Il server apre una connessione dati dal porto 20 verso il client

Passive FTP

- Il client apre una connessione di controllo verso l'indirizzo del server sul porto 21 utilizzando un *ephemeral port*
- Il server sceglie un *ephemeral port* per la connessione dati e la comunica al client
- Il client apre la connessione dati sul porto indicato dal server



Nel caso passive FTP, è il client ad aprire una connessione verso il server, sia per il canale di controllo che per il canale dati.

Il protocollo SMTP

Una volta che una e-mail è stata scritta attraverso l'uso di un programma su un personal computer, è necessario inviarla al destinatario. Come è noto, il destinatario potrebbe non essere in quel momento disponibile ad accettare messaggi di posta (utente impegnato o computer spento)

La posta elettronica sfrutta degli intermediari per il trasferimento delle e-mail tra le parti, alla stregua degli uffici postali che ospitano pacchi nell'attesa che i destinatari passino a ritirarli.

Per trasferire messaggi di posta elettronica tra gli intermediari si utilizza un apposito protocollo chiamato ***Simple Mail Transfer Protocol***.

Ci sono tre entità principali in gioco:

- **User Agent**

- Anche detto mail reader
- Composizione, modifica, lettura di messaggi (Outlook, Mozilla, etc...)
- Messaggi in uscita ed in entrata immagazzinati sul server
- Il protocollo SMTP viene utilizzato anche tra user-agent e server durante l'invio di una mail

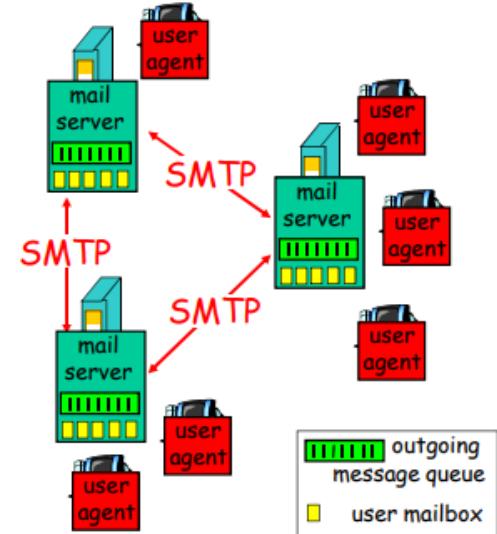
- **Mail Servers**

- Mailbox contenente messaggi in entrata (non letti) per l'utente
- Coda dei messaggi in uscita contenente i messaggi non ancora recapitati
- Protocollo SMTP usato per l'interazione tra due mail server ("client": mail server mittente; "server": mail server destinatario)

- **Protocollo SMTP**

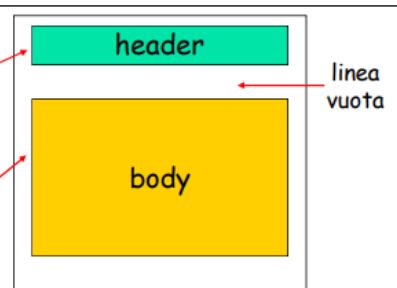
Caratteristiche di SMTP

- Usa il protocollo TCP (porto 25) per consegnare in modo affidabile messaggi dal client al server
- Trasferimento diretto dal server mittente al server destinatario
- Tre fasi durante il trasferimento via SMTP:
 - handshaking ("stretta di mano")
 - trasferimento del messaggio
 - chiusura della connessione
- Interazione comando/risposta (*command/response*)
 - comandi: testo ASCII
 - risposta: codice di stato e descrizione (facoltativa)
- Messaggi codificati con caratteri ASCII a 7-bit
- Usa una connessione persistente
- Richiede che il messaggio, comprensivo del contenuto, sia codificato in caratteri ASCII a 7-bit
- Alcune combinazioni di caratteri non sono ammesse. Quando queste combinazioni si presentano il messaggio deve essere opportunamente codificato
 - Ad esempio, CRLF.CRLF è un carattere non ammesso poiché viene usato da SMTP per determinare la fine di un messaggio



Formato del messaggio

- Linee di intestazione (header):
 - To:
 - From:
 - Subject:
 - ...
 - *differenti* da comandi smtp!
- corpo (body):
 - il "messaggio" vero e proprio
 - solo caratteri ASCII

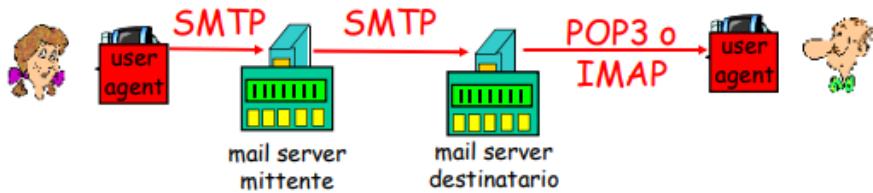


Prelievo della posta: Post Office Protocol (POP3)

Fino ad ora abbiamo visto come sia possibile trasferire messaggi tra i vari mail server ma non abbiamo ancora parlato di come un utente possa, in un momento qualsiasi, accedere alla propria casella di posta elettronica per leggere i propri messaggi.

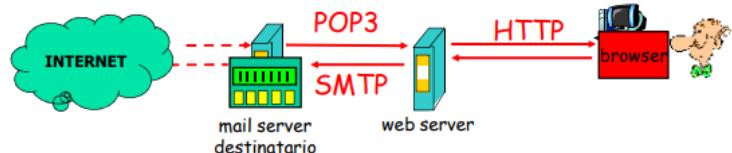
Per questa operazione è previsto un ulteriore protocollo chiamato POP3 e si tratta sempre di un protocollo client server: lo user agent ancora una volta gioca il ruolo di client POP e il mail server gioca il ruolo di server POP.

La catena dei protocolli per la posta



- SMTP: consegna di messaggi
- Protocolli di accesso alla mail: recupero dei messaggi dai server
 - POP: autorizzazione e download
 - IMAP (Internet Mail Access Protocol): più complicato e potente; manipolazione avanzata dei messaggi sul server
 - HTTP: gmail, Hotmail, etc...

Molti siti web forniscono accesso alle proprie caselle di posta (gmail, Hotmail, etc..); in questo caso non serve avere uno user agent installato e correttamente configurato per ricevere ed inviare posta, ma è sufficiente disporre di un qualsiasi browser



7. Protocolli applicativi: DNS

Domain Name System (DNS)

Tutti noi siamo oggi abituati a raggiungere un servizio (e quindi il calcolatore che lo offre) utilizzando nomi simbolici di facile memorizzazione, come ad esempio www.google.com

Questi nomi non sono immediatamente adatti ad essere compresi dai dispositivi che costituiscono la rete Internet; infatti, un nome di questo tipo non dà informazioni esatte sulla dislocazione sul territorio della macchina che si desidera contattare. I router, di conseguenza, non saprebbero come instradare i dati in maniera tale da raggiungere la destinazione.

La rete Internet è stata progettata invece per lavorare con indirizzi di diversa natura (es.: 143.225.229.3), questi indirizzi, detti **indirizzi IP**, sono formati da 4 numeri che vanno da 0 a 255 separati da un punto. Ogni dispositivo nella rete Internet ha un tale indirizzo; esso permette l'identificazione univoca a livello globale e la localizzazione.

A differenza dei nomi simbolici, essendo gli indirizzi IP di lunghezza fissa, sono più facilmente gestibili dalle macchine.

Il servizio DNS

Non volendo rinunciare alla comodità di lavorare con nomi simbolici, è stato necessario progettare un servizio di risoluzione dei nomi simbolici in indirizzi IP. Tale servizio associa ad un nome simbolico univoco (www.grid.unina.it) un indirizzo IP (143.225.229.3) permettendo così di raggiungere la macchina.

Questo servizio si chiama Domain Name System (DNS) e si basa sullo scambio di messaggi UDP sul porto 53.

Altre funzionalità offerte

- Alias degli hostname:
 - ad una macchina con un nome complicato può essere associato un “soprannome” più piccolo e semplice da ricordare.
 - es.: rcsn1.roma.rai.it → www.rai.it
- Alias dei server di posta:
 - permette di associare un server di posta al nome di un dominio per facilitare la memorizzazione dell’indirizzo di posta.
 - es.: pippo@unina.it identifica l’utente pippo sulla macchina mailsrv1.cds.unina.it
 - L’associazione @unina.it → mailsrv1.cds.unina.it è a carico del servizio DNS.
- Distribuzione del carico:
 - quando un server gestisce un carico troppo elevato si suole replicare il suo contenuto su molte macchine differenti.
 - Il servizio DNS distribuisce il carico tra le macchine rilasciando ciclicamente indirizzi appartenenti all’intero pool, senza che gli utenti si accorgano di nulla.

DNS distribuito

Si potrebbe pensare di risolvere il problema piazzando in un unico punto della terra una macchina che realizzhi la risoluzione di tutti i nomi (DNS centralizzato), ma questa soluzione, sebbene teoricamente realizzabile, ha così tanti svantaggi da risultare impraticabile (single point of failure, volume di traffico, database distante, manutenzione)

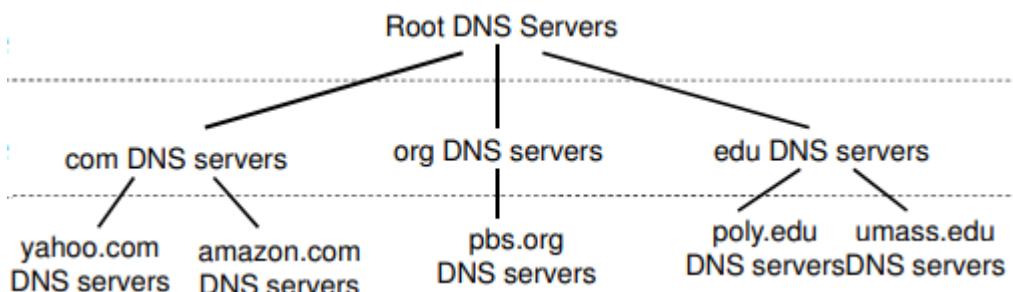
Quello che si fa è distribuire le informazioni sul territorio (DNS distribuito). Ciascuno ha la responsabilità di raccogliere, gestire, aggiornare e divulgare le informazioni che lo riguardano.

In particolare, l’approccio è di tipo gerarchico:

- gli elementi più alti nella gerarchia contengono molte informazioni non dettagliate
- gli elementi più bassi nella gerarchia contengono poche informazioni dettagliate

Attraverso un colloquio stabilito tra le entità (di cui gli utenti non hanno percezione) si riesce a fornire il servizio di risoluzione.

DNS: un database gerarchico e distribuito



Un client richiede l’IP di www.amazon.com:

- Il client dapprima contatta uno dei root server per avere la lista degli indirizzi IP dei TLD per il dominio com
- Il client contatta uno dei TLD server che gli restituisce l’indirizzo IP del server autorizzato per amazon.com
- Infine, il client contatta il server autorizzato per amazon.com che gli restituisce l’indirizzo IP di www.amazon.com

Tipologie di server DNS

Local Name Server (locale)

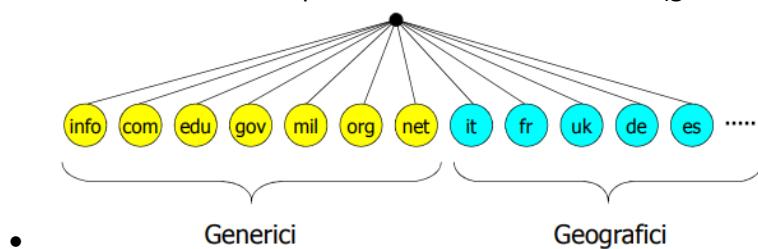
- Non appartengono strettamente alla gerarchia di server
- Ciascun ente (università, società, etc...) ne installa uno nel proprio dominio
- Tutti gli host nel dominio richiedono a questo server il servizio di risoluzione
- Quando un host effettua una richiesta DNS, la query viene inviata al server DNS locale che opera da proxy e, tipicamente, invia la query attraverso la gerarchia di server
- Ciascun host deve essere configurato con l'indirizzo del DNS server locale per il dominio. Questa configurazione spesso avviene manualmente, ma in certi casi può avvenire anche in maniera automatica
- L'uso di un server DNS locale consente ai singoli host di fare una sola query DNS verso di essi: sarà poi il local DNS server a fare la sequenza di interrogazioni descritta precedentemente.

Root Name Server (fondamentale)

- Ne esistono 13 in Internet (etichettati da A ad M) e i loro indirizzi sono ben noti alla comunità
- Ad essi si riferiscono i Local Name Server che non possono soddisfare immediatamente una richiesta di risoluzione
- In questo caso il Local Name Server si comporta come client DNS ed invia una richiesta di risoluzione al Root Name Server

I “top-level domain” server (TLD)

- Questi server si occupano dei domini di alto livello (generici e geografici).



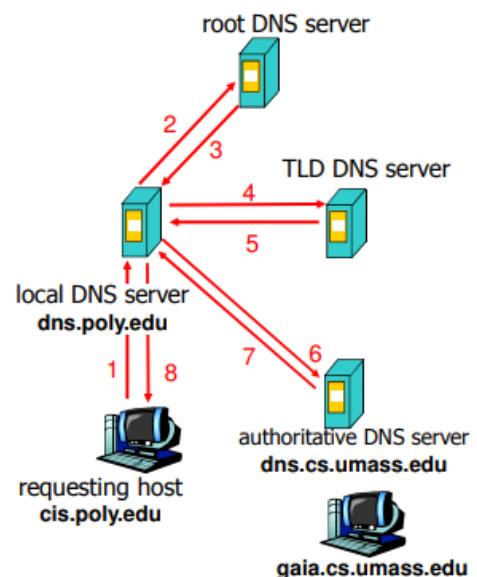
Authoritative Name Server (Assoluto)

- È un server dei nomi capace di risolvere tutti i nomi all'interno di un determinato dominio/organizzazione
 - Per esempio: un server dei nomi assoluto per il dominio unina.it deve essere capace di risolvere tutti i nomi del tipo xyz.unina.it
- Ad essi si riferiscono i Name Server TLD quando, interpellati dai Local Name Server, devono risolvere un indirizzo
- Può essere mantenuto dall'organizzazione o da un provider

Esempio di risoluzione

Un host di cis.poly.edu richiede l'indirizzo IP di gaia.cs.umass.edu

- Query Iterative:
 - Il server contattato risponde con il nome del server da contattare
 - La logica: non conosco questo nome, ma conosco il nome di qualcuno a cui poter chiedere
- Query Ricorsive:
 - Sposta il carico della risoluzione dei nomi sul server contattato, delegando al DNS contattato la responsabilità di risolvere l'indirizzo



Il TLD potrebbe non contattare necessariamente l'Authoritative Name Server finale, ma uno intermedio. Sarà il server intermedio a fornire il nome del servizio di competenza. In questi casi il numero di messaggi DNS aumenta.

Il caching dei nomi

Per esigenze di efficienza un server DNS memorizza localmente un certo numero di corrispondenze.

Per evitare che informazioni non aggiornate restino nella rete, dopo un certo (circa un giorno), le associazioni vengono eliminate dalla cache.

Ad esempio, un server locale può memorizzare associazioni IP/nomi di non sua competenza e/o gli indirizzi dei server TLD in modo da aggirare i server root.

Un DNS memorizza un **Resource Records (RR)**:

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

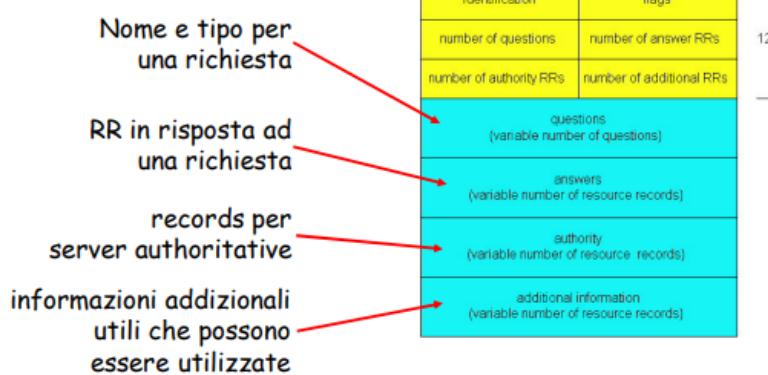
- *TTL*: tempo di vita residuo di un record scaduto il quale viene eliminato dalla cache
- Il significato di *Nome* e *Valore* dipende da *Tipo*:
 - *Tipo = A* → $\begin{cases} \text{Nome} = \text{hostname} \\ \text{Valore} = \text{indirizzo IP} \end{cases}$
 - *Tipo = CNAME* → $\begin{cases} \text{Nome} = \text{alias per il nome canonico (reale)} \\ \text{Valore} = \text{nome canonico} \end{cases}$
 - *Tipo = NS* → $\begin{cases} \text{Nome} = \text{dominio (es.: unina.it)} \\ \text{Valore} = \text{indirizzo IP dell'Authoritative NS} \end{cases}$
 - *Tipo = MX* → $\begin{cases} \text{Nome} = \text{dominio di posta (es.: libero.it)} \\ \text{Valore} = \text{nome dell'host mailserver associato a Nome} \end{cases}$

Il formato dei messaggi

Nel protocollo DNS richieste e risposte hanno lo stesso formato di messaggio.

Header del messaggio:

- **Identification**
 - diverso numero di 16bit per ogni richiesta. Le risposte usano lo stesso identificativo
- **flags**
 - risposta a richiesta
 - ricorsione desiderata
 - ricorsione disponibile
 - risposta autoritativa



Il servizio BIND

BIND (Berkeley Internet Name Domain) è una implementazione dei protocolli DNS ed è liberamente re-distribuibile.

È costituito dai seguenti componenti:

- Un server DNS (named)
- Una libreria per la risoluzione dei nomi di dominio
- Strumenti di diagnostica

Questa implementazione è di gran lunga la più utilizzata su Internet su sistemi Unix-like.

8. Content Delivery Networks (CDN)

Cosa è una CDN

Una Content Delivery Network è un'infrastruttura creata per distribuire efficacemente agli utenti di Internet i contenuti dei siti web più popolari.

Una CDN 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".

Si presenta come un servizio a pagamento del quale usufruiscono i gestori dei siti web commerciali più popolari (esempi: Akamai, Speedera, Inktomi)

Obiettivi di una CDN:

- 1) Alleviare il server web "master" dal carico degli utenti, in particolare proteggerlo da picchi di traffico improvvisi (flash crowds)
- 2) Offrire i contenuti ai singoli utenti tramite server collocati in prossimità degli utenti (alla periferia della rete)
- 3) Rendere il sistema di distribuzione dei contenuti più affidabile e robusto ai guasti

Tramite una infrastruttura, spesso privata, distribuiscono, in maniera capillare i contenuti di uno specifico Content Provider.

Utilizzano forme proprietarie di caching basate su una complessa gestione del DNS, caratterizzata, tra l'altro, dalla conoscenza dell'indirizzo IP del Client. Gestione centralizzata dei contenuti.

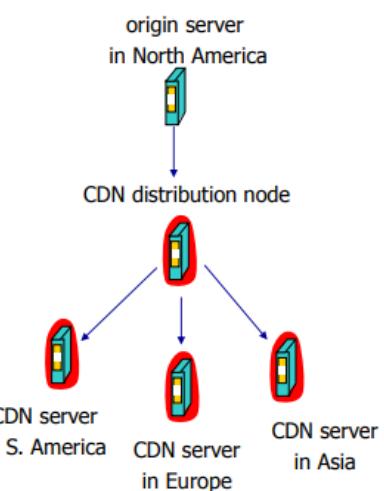
Servizi Offerti

- Le CDN offrono ai Content Provider la possibilità di raggiungere, con una certa QoS, una vasta utenza
- Le CDN, d'altra parte, propongono a ISP di medie e grandi dimensioni, di collaborare, spesso gratuitamente, alla loro struttura.
- Ottimizzano l'uso delle risorse di Internet avvicinando il contenuto all'utente
- Benefici per gli utenti, per gli ISP e per i Content Provider
- Obiettivi: bassa latenza, bassi costi, raggiungibilità, protezione da flash events

Replicazione e aggiornamento

La distribuzione dei contenuti da un singolo provider verso una molteplicità di utenti ha delle problematiche come la perdita di dati o il ritardo. Una soluzione è quella di replicare il contenuto su quanti più nodi possibili su internet e vicino a chi ne fa richiesta (nelle reti di accesso presso la rete ISP).

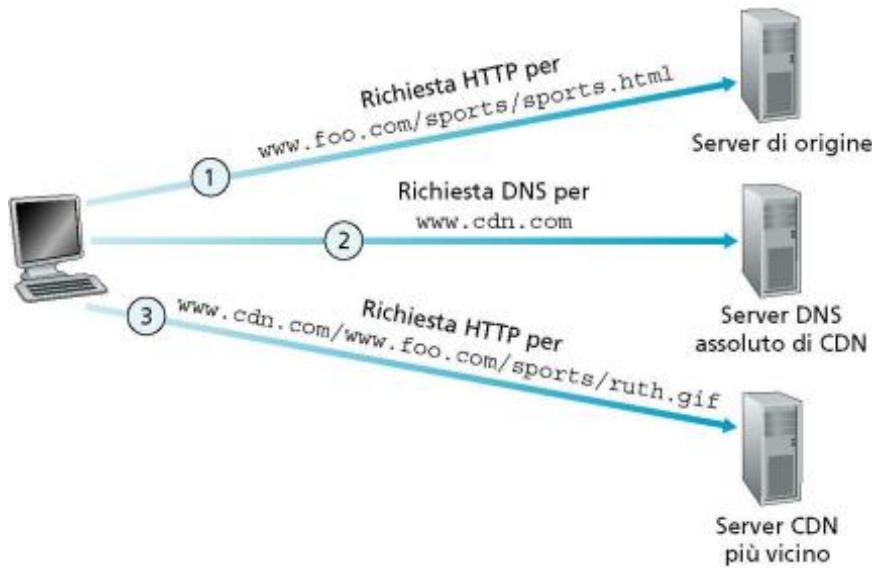
La CDN replica il contenuto del customer sui propri CDN server e quando il provider (CNN) aggiorna il contenuto, la CDN aggiorna i propri server.



Prelievo da parte del client

Dal server origine al nodo CDN: routing delle richieste:

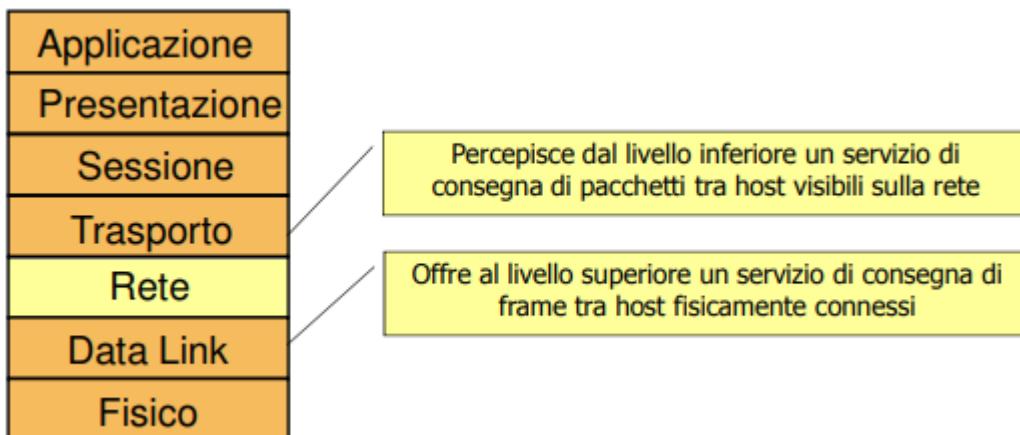
- La CDN crea una mappa che indica le distanze tra i vari ISP e i nodi CDN
- Quando arriva una query al DNS aut.
 - Si determina l'ISP che ha originato la query
 - Si usa la mappa per la scelta del server CDN più vicino



9. Livello di Rete e Protocollo IP

Il livello rete in Internet

Il livello rete è il terzo livello dello stack OSI:

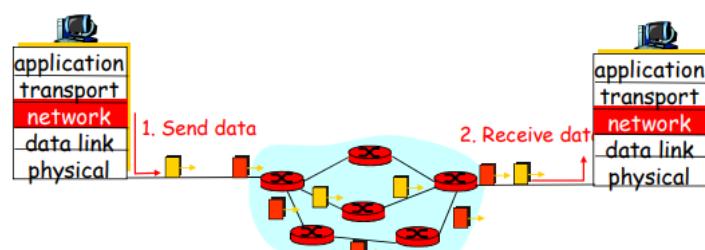


Le funzioni del livello rete sono le seguenti:

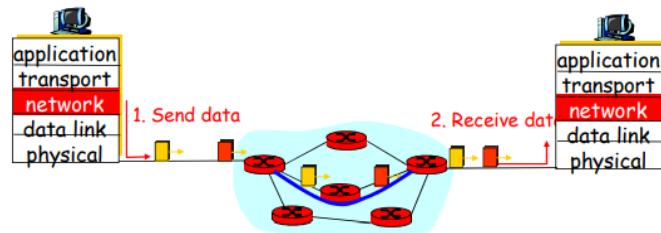
- Trasportare i pacchetti dall'host mittente a quello ricevente
- Implementare protocolli di livello rete in tutti i router e in tutti gli host
- **Forwarding:** spostare i pacchetti dalla coda/interfaccia di ingresso a quella di uscita
- **Routing:** determinare la strada che un pacchetto deve seguire da una sorgente alla destinazione

Le reti possono essere classificate a seconda del metodo utilizzato per trasportare i pacchetti dalla sorgente alla destinazione (parliamo comunque di reti a commutazione di pacchetto):

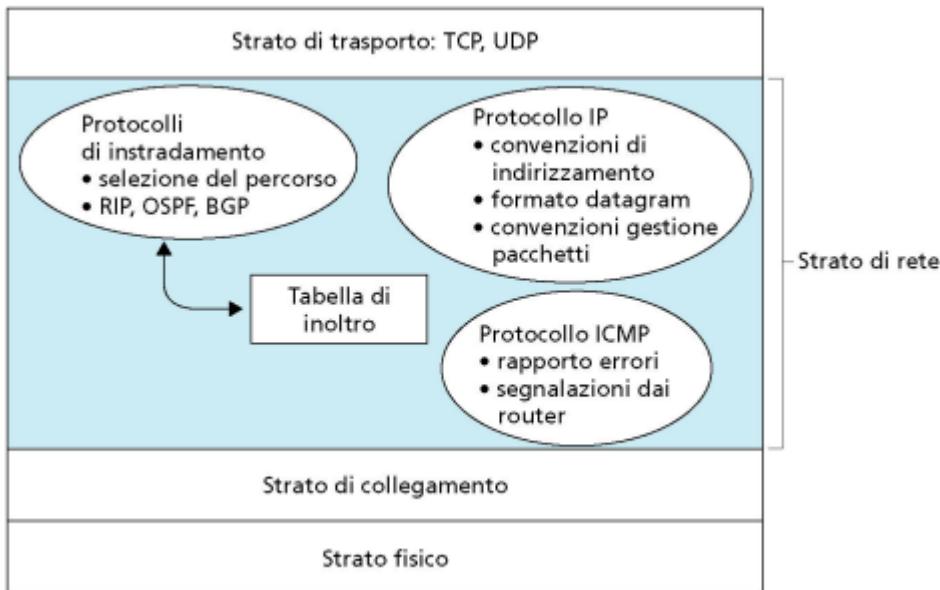
- **Reti a datagrammi:** ogni pacchetto è instradato indipendentemente dagli altri pacchetti dello stesso flusso.
 - Ogni router che riceve un pacchetto decide indipendentemente a chi mandarlo sulla base dell'indirizzo destinazione contenuto nel pacchetto
 - Pacchetti tra la stessa coppia sorgente-destinazione possono seguire percorsi differenti



- **Reti a circuiti virtuali:** viene precalcolato un percorso e tutti i pacchetti del flusso seguono questo percorso
 - Ogni pacchetto contiene il numero del circuito virtuale
 - Il circuito virtuale è stabilito prima della trasmissione dei dati
 - I nodi devono conservare informazioni su tutti i circuiti virtuali che li attraversano



Lo strato di rete di internet



Internet Protocol (IP)

IP è un protocollo di livello rete usato per lo scambio di dati tra reti di calcolatori. I dati sono trasportati con la tecnica dei datagrammi.

IP offre un servizio di comunicazione *connection-less* e gestisce indirizzamento, frammentazione, riassemblaggio e multiplexing dei protocolli.

Costituisce la base sulla quale si basano tutti gli altri protocolli, collettivamente noti come TCP/IP suite (TCP, UDP, ICMP, ARP); inoltre il protocollo IP è responsabile dell'instradamento dei pacchetti.

Un pacchetto IP è anche chiamato **datagramma**:

- È costituito da un *header* e un'area dati
- I datagrammi possono avere dimensioni diverse
- La dimensione dell'header è solitamente fissata (20 byte) a meno che non siano presenti opzioni
- Un datagramma può contenere fino a un massimo di ($2^{16} - 1$) byte

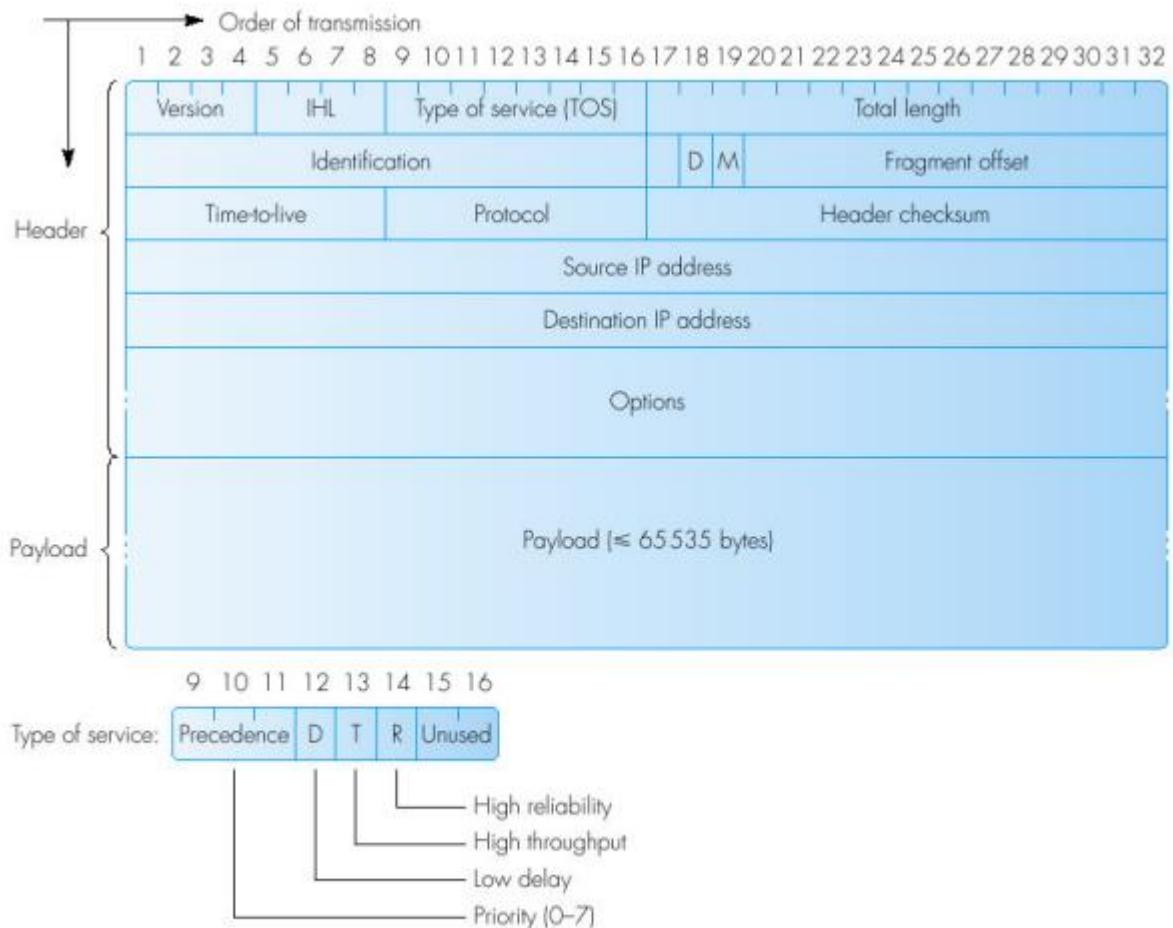
L'**header** contiene tutte le informazioni necessarie per la consegna del datagramma alla destinazione

- Indirizzo destinazione, indirizzo sorgente, identificativo, etc...

I router esaminano l'header di ogni datagramma e inoltrano il pacchetto lungo il percorso verso la destinazione

- Usano tabelle di routing per calcolare il *next hop*
- Aggiornano tali tabelle usando protocolli di routing dinamici

Formato del pacchetto IP

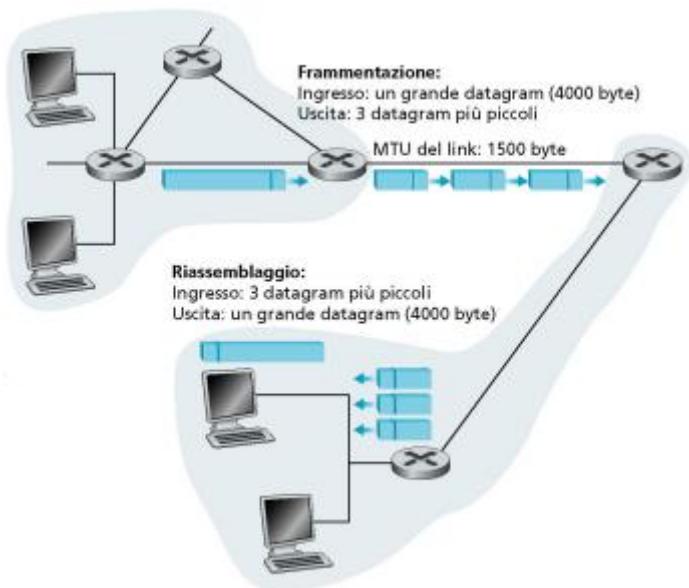


- **Version**
 - 4 bit, versione del protocollo IP cui il pacchetto è conforme
- **IP header length (IHL)**
 - 4 bit, lunghezza dell'header, in multipli di 32 bit (massimo 60 byte)
- **Type-of-Service (ToS)**
 - 8 bit, specifica come un protocollo di livello superiore vorrebbe che il pacchetto fosse trattato
- **Total length**
 - 16 bit, specifica la lunghezza in byte dell'intero pacchetto (header + dati)
 - Massimo 64kB, cioè 65535 byte ($2^{16} - 1$)
- **Time-to-live (TTL)**
 - 8 bit, contatore che viene gradualmente decrementato fino a zero, punto in cui il pacchetto viene scartato
 - Serve ad evitare che un pacchetto resti perennemente in circolo
- **Protocol**
 - 8 bit, indica il protocollo di livello superiore che riceve il pacchetto dopo che l'elaborazione IP è terminata
 - Analogico al numero di porto di livello trasporto: 6 indica TCP, 17 indica UDP
- **Header checksum**
 - 16 bit, aiuta a garantire l'integrità dell'header IP
- **Source Address**
 - 32 bit, specifica il nodo mittente

- **Destination Address**
 - 32 bit, specifica il nodo ricevente
- **Identification**
 - I pacchetti possono essere frammentati lungo il percorso
 - Questo campo (16 bit) è un identificativo del datagramma
- **Flags**
 - Il bit D indica se il pacchetto può essere frammentato
 - Il bit M indica se il pacchetto è l'ultimo frammento
- **Fragment offset**
 - 13 bit, identifica la posizione del frammento all'interno del pacchetto

Frammentazione e riassemblaggio IP

- Tutti i frammenti tranne l'ultimo devono essere multipli di 8 byte
- 8 byte è la dimensione del frammento elementare
- Avendo 13 bit a disposizione, ci possono essere al massimo 8192 frammenti per ogni datagramma
- La dimensione massima di un datagramma è 65535 byte



Opzioni

- È il modo per estendere IP con un numero variabile di opzioni
 - Security
 - Source routing
 - Route recording
 - Stream identification
 - Timestamping
- A causa delle opzioni, l'header può essere di lunghezza variabile
 - Questo è il motivo della presenza del campo IHL
 - Se l'opzione non occupa 4 byte (o un suo multiplo), vengono inseriti dei bit di riempimento (tutti zero)
 - La presenza opzionale di questi campi rende difficile la gestione in implementazioni hw-based

IP è consegna Best effort

- IP non garantisce di prevenire:
 - Datagrammi duplicati
 - Consegnata ritardata o fuori ordine
 - Corruzione di dati
 - Perdita di pacchetti (e di frammenti)
- La consegna affidabile dei pacchetti può avvenire grazie a meccanismi di controllo da parte di protocolli di livello superiore

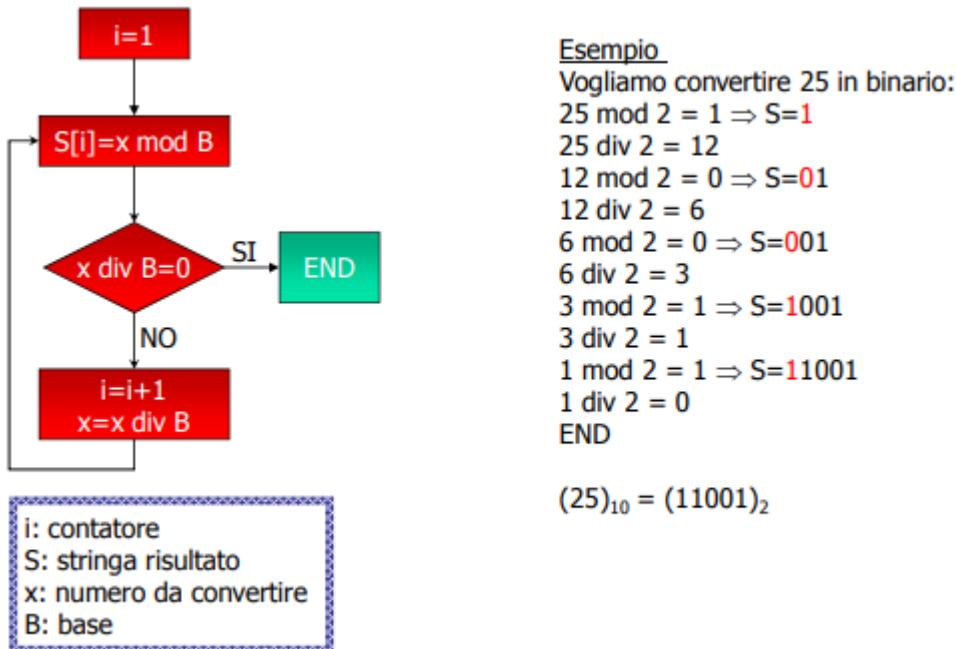
Cenni sui sistemi di numerazione

Riacenniamo ai seguenti sistemi di numerazione:

- Sistema di numerazione decimale
 - Si utilizzano 10 simboli (le cifre da 0 a 9)

- Ogni cifra è pesata secondo una potenza di 10 con esponente crescente da destra verso sinistra
- Es.: $(2536)_{10} = 2 * 10^3 + 5 * 10^2 + 3 * 10^1 + 6 * 10^0$
- Sistema di numerazione binaria
 - Si utilizzano 2 simboli (le cifre 0 e 1)
 - Ogni cifra è pesata secondo una potenza di 2 con esponente crescente da destra verso sinistra
 - Es.: $(11010)_2 = 1 * 2^4 + 1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 0 * 2^0 = (26)_{10}$
 - Abbiamo così visto anche il modo per convertire un numero da base 2 a base 10
- Sistema di numerazione esadecimale
 - Si utilizzano 16 simboli (le cifre da 0 a 9 e le lettere da A a F)
 - Ogni cifra è pesata secondo una potenza di 16 con esponente crescente da destra verso sinistra
 - Es.: $(1E5)_{16} = 1 * 16^2 + 14 * 16^1 + 5 * 16^0 = (485)_{10}$
 - Abbiamo così visto anche il modo per convertire un numero da base 16 a base 10

Conversione da decimale



Relazione tra numeri binari ed esadecimali

- Una stringa di 4 bit può assumere $2^4 = 16$ diversi valori
- Se consideriamo ogni stringa come un numero binario, essa rappresenta un numero compreso tra 0 e 15
- In base 16 abbiamo 16 cifre che assumono un valore compreso tra 0 e 15
- 1 cifra esadecimale rappresenta 4 cifre binarie
- Es.: $(201)_{10} \left\{ \begin{array}{c} (\underline{\underline{1100}} \underline{\underline{1001}})_2 \\ (C \quad 9)_{16} \end{array} \right.$

Indirizzi IP

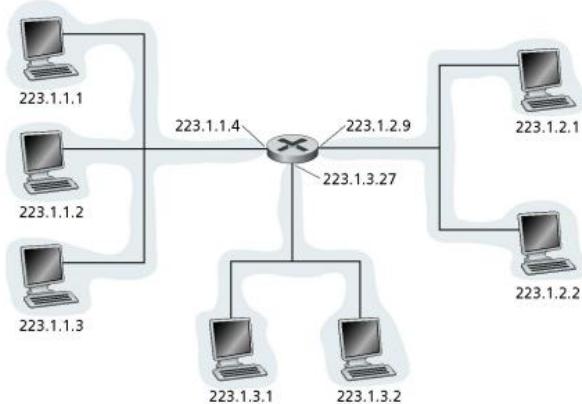
Ad ogni host è assegnato un indirizzo IP o indirizzo Internet, l'IP è un numero di 32 bit (4 byte) ed è unico in tutta Internet.

Ogni indirizzo IP è diviso in un prefisso e un suffisso:

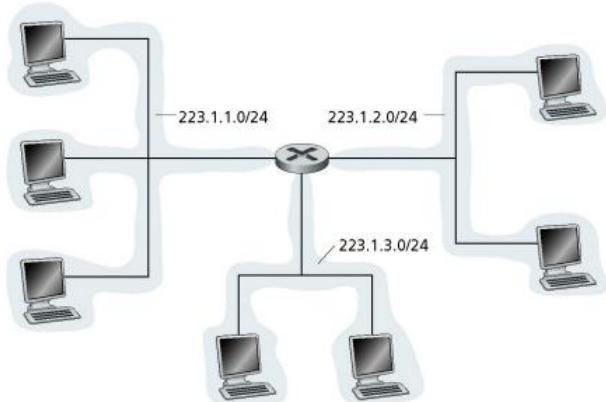
- Il prefisso indica la rete alla quale l'host è collegato
 - Due reti differenti hanno numero di rete differente
- Il suffisso identifica l'host all'interno della rete
 - Due host sulla stessa rete non possono avere lo stesso suffisso, ma host su reti diverse possono avere lo stesso suffisso

L'indirizzo IP è assegnato dall'ICANN (*Internet Corporation for Assigned Names and Numbers*); la quale gestisce il DNS ed assegna i nomi dei domini; inoltre, risolve eventuali dispute (conflitti di nomi e/o indirizzi).

Indirizzi delle interfacce:



Indirizzi delle reti:



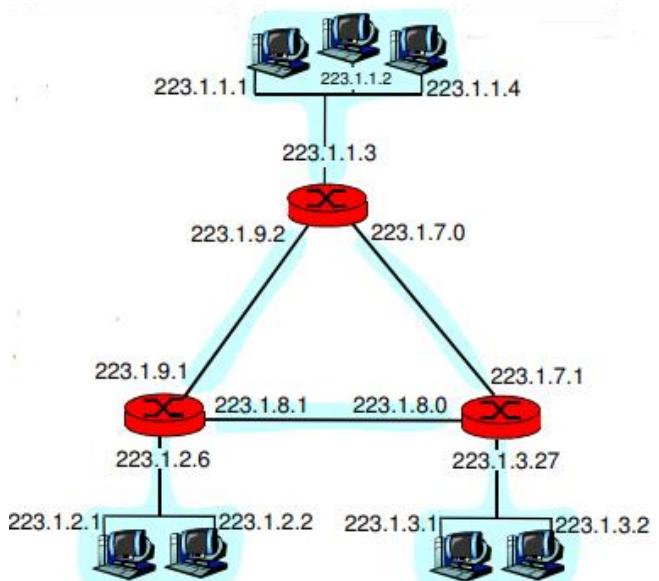
Diamo un esempio di come trovare il numero di reti in un sistema interconnesso costituito da sei reti (vedi immagine a destra):

- “Staccare” ogni interfaccia dal corrispondente router/host
- Creare “isole” costituite da segmenti di rete disgiunti

Notazione dotted decimal

La notazione dotted decimal rappresenta gli indirizzi IP come 4 numeri decimali separati da punto.

Ogni numero decimale, poiché rappresenta un byte, è compreso tra 0 e 255.



32-bit Binary Number	Equivalent Dotted Decimal
10000001 00110100 00000110 00000000	129 . 52 . 6 . 0
11000000 00000101 00110000 00000011	192 . 5 . 48 . 3
00001010 00000010 00000000 00100101	10 . 2 . 0 . 37
10000000 00001010 00000010 00000011	128 . 10 . 2 . 3
10000000 10000000 11111111 00000000	128 . 128 . 255 . 0

Classi di indirizzi

La parte di indirizzo che specifica la rete e quella che specifica l'host non hanno lunghezza fissa, ma variano a seconda della classe a cui appartiene l'indirizzo.

Sono state definite 5 classi:

- 3 (A, B, C) sono usate per gli indirizzi degli host e si differenziano per la lunghezza della parte rete/host
- 1 (D) è usata per il multicast
- 1 (E) è riservata per usi futuri

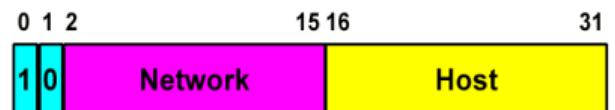
Indirizzi di classe A

- Campo rete
 - 7 bit
 - Massimo 128 reti
 - Il primo byte è compreso tra 0 e 127
- Campo host
 - 24 bit
 - Massimo 2^{24} (circa 16 milioni) host



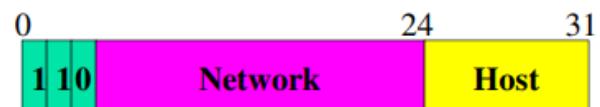
Indirizzi di classe B

- Campo rete
 - 14 bit
 - Massimo 16 mila reti
 - Il primo byte è compreso tra 128 e 191
- Campo host
 - 16 bit
 - Massimo 2^{16} (circa 64 mila) host

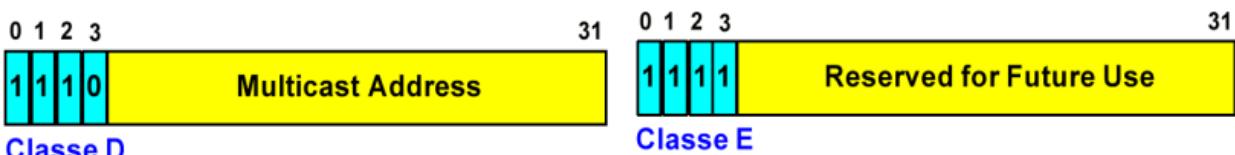


Indirizzi di classe C

- Campo rete
 - 21 bit
 - Massimo 2 milioni di reti
 - Il primo byte è compreso tra 192 e 223
- Campo host
 - 8 bit
 - Massimo 256 host



Indirizzi di classe D ed E



Indirizzi IP “speciali”

- Network address
 - La rete stessa ha un indirizzo, il cui suffisso è costituito da tutti “0”
 - Nessun host può quindi avere tutti “0” nel suffisso
- Directed broadcast address
 - Per mandare un messaggio in broadcast ad una rete il suffisso è costituito da tutti “1”
 - Il pacchetto è inviato a tutti gli host di una specifica rete
- Limited broadcast address
 - L'intero indirizzo (sia la parte rete che la parte host) è costituito da tutti “1” (255.255.255.255)
 - Broadcast sulla LAN locale
- This computer address
 - L'intero indirizzo è costituito da tutti “0”

- Per ottenere un indirizzo automaticamente all'avvio, si potrebbe usare IP per comunicare ma non abbiamo ancora un indirizzo
- Vedremo che per l'assegnazione di tale indirizzo si utilizzano protocolli quali DHCP e BOOTP

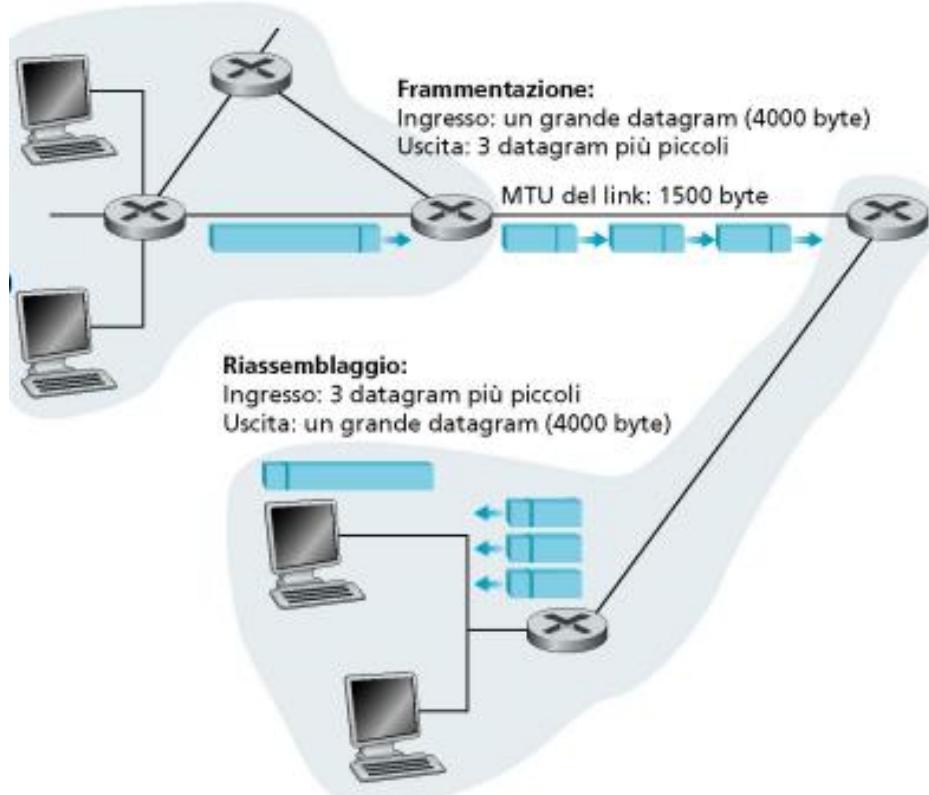
Indirizzi IP privati

La suddivisione degli indirizzi IP in classi non è efficiente perché comporta lo spreco di indirizzi (non tutti i 4 miliardi di indirizzi vengono usati); gli indirizzi privati, insieme alle sottoreti, sono una soluzione a questo problema:

- L'RFC 1597 riserva i seguenti blocchi di indirizzi per uso privato:
 - 10.0.0.0 - 10.255.255.255 Classe A
 - 172.16.0.0 - 172.31.255.255 Classe B
 - 192.168.0.0 - 192.168.255.255 Classe C
- I router di Internet non inoltrano pacchetti aventi indirizzo sorgente o destinazione compreso in uno di questi blocchi
- Vedremo dopo il modo per tradurre tali indirizzi in indirizzi Internet pubblici (NAT)

Il protocollo IP: frammentazione Subnetting

Frammentazione e riassemblaggio IP

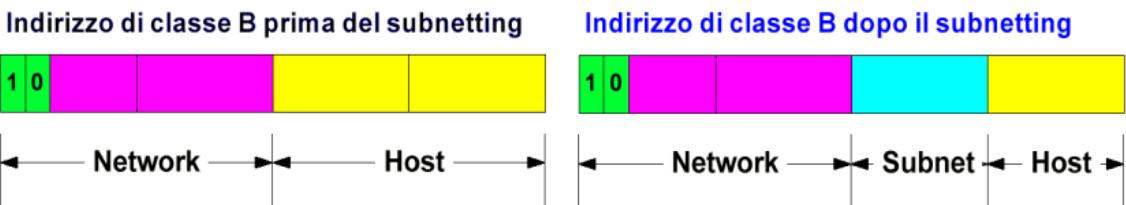


- Se un pacchetto di dimensione N arriva ad un router e deve essere trasmesso su un link di uscita con MTU M < N, il pacchetto è frammentato
- Ogni frammento è trasmesso come singolo pacchetto IP
- La dimensione di ogni frammento sarà un multiplo di 8 byte
- Tutti i frammenti hanno lo stesso ID number

Sottoreti

- Gli indirizzi IP sono assegnati in modo che tutti gli host sulla stessa rete locale appartengono alla stessa sottorete

- Una sottorete è individuata dai bit del prefisso più alcuni bit presi in prestito dal suffisso, come specificato dalla **subnet mask**
- Una subnet mask è una stringa di 32 bit associata ad ogni host:
 - Gli “1” definiscono la porzione di indirizzo che identifica la sottorete
 - Gli “0” definiscono la porzione di indirizzo che identifica l’host
- L’indirizzo della sottorete si ottiene mediante un **AND bit a bit tra l’indirizzo dell’host e la netmask**
 - Esempio: vogliamo utilizzare un unico indirizzo di classe B avendo diverse reti fisiche
Se prendiamo in prestito 8 bit dal suffisso avremo a disposizione 256 sottoreti, ognuna delle quali potrà avere 254 host
 - L’host 128.192.56.50 con netmask 255.255.255.0 appartiene alla sottorete 128.192.56.0



- Suddividere una rete in sottoreti ci consente di allocare in maniera efficiente gli indirizzi, migliorando al tempo stesso le prestazioni (il traffico relativo ad una sottorete non viene introdotto nelle altre)
- Una subnet mask utilizzata da un host che deve trasmettere un pacchetto:
 - Confronta la destinazione con la propria subnet mask
 - Se la destinazione è sulla stessa sottorete, invia sulla LAN
 - Altrimenti, invia al gateway
- Una subnet mask utilizzata da un router all’interno della rete suddivisa in sottoreti:
 - Utilizza la subnet mask con l’indirizzo di rete delle reti collegate per determinare la giusta destinazione

Netmask

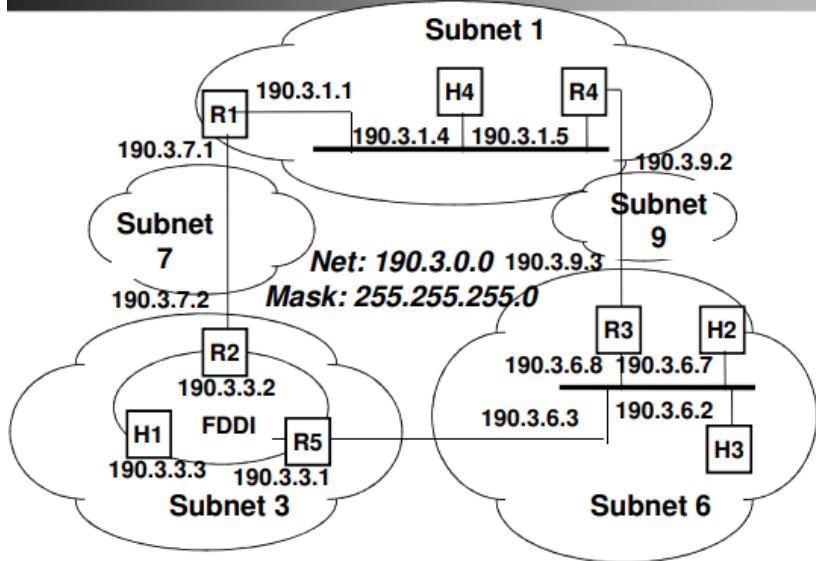
- Parametro che specifica il subnetting
 - bit a 1 in corrispondenza dei campi network e subnetwork
 - bit a 0 in corrispondenza del campo host
- Esempio: si supponga di voler partizionare una rete di classe B in 16 subnet da 4096 host
 - Netmask 11111111 11111111 11100000 00000000
 - Netmask esadecimale ff ff f0 00
 - Netmask decimale 255.255.240.0
 - /20

Subnet

- Subnet e reti fisiche
 - IP assume una corrispondenza biunivoca tra reti fisiche e subnet: routing implicito all’interno di una subnet
 - Il routing tra subnet diverse è esplicito e gestito dai router tramite tabelle di instradamento
- Instradamento
 - All’interno della subnet l’instradamento deve essere fornito dalla rete fisica
 - Corrispondenza tra gli indirizzi di subnet (indirizzi IP) e gli indirizzi di livello 2 gestita da ARP (Address Resolution Protocol)
 - Indirizzi di livello 2: indirizzi MAC sulle LAN, indirizzi di DTE in X.25, identificatori di LCI in frame relay, etc...

- Default Route:
 - Gli host devono conoscere almeno un router presente sulla loro rete fisica
 - Il protocollo ICMP permette di ottimizzare dinamicamente il routing
 - Ad esempio sull'host H4


```
route add default gw 190.3.1.5
```
- Tabelle di Instradamento:
 - L'instradamento tra subnet diverse viene gestito da tabelle di instradamento presenti sui router
 - Esempio: tabelle di instradamento del router R5 (3 subnet non raggiungibili direttamente)



Subnet di Destinazione	Indirizzo del router
190.3.1.0	190.3.3.2
190.3.7.0	190.3.3.2
190.3.9.0	190.3.6.8

10. ARP – RARP – DHCP – ICMP: ping e traceroute

Address Resolution Protocol (ARP)

Due host possono comunicare direttamente solo se sono collegati alla stessa rete fisica; quindi, per potersi scambiare informazioni devono conoscere i rispettivi indirizzi fisici.

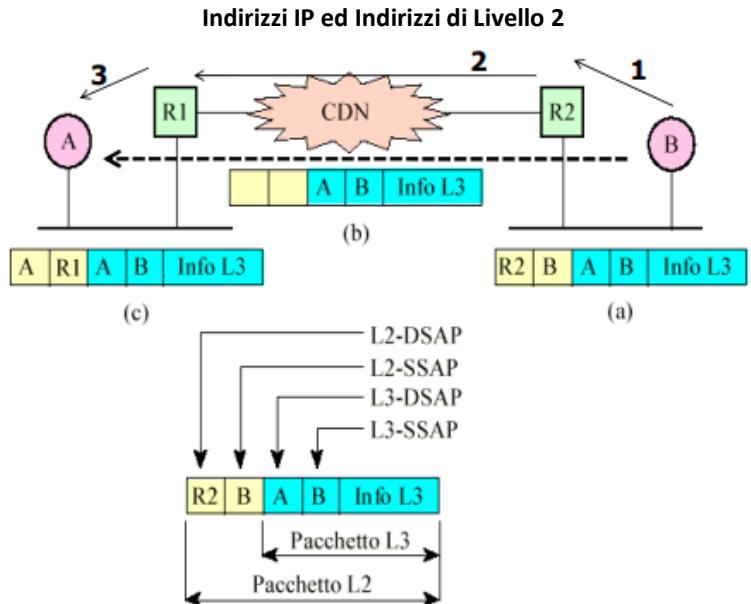
Il protocollo IP consente di individuare univocamente un host tramite un indirizzo logico (indirizzo IP) e tutte le applicazioni usano gli indirizzi logici ed ignorano la rete fisica. Ma per inviare un messaggio occorre necessariamente conoscere anche l'indirizzo fisico; pertanto, serve un meccanismo di corrispondenza tra gli indirizzi logici e gli indirizzi fisici. Tale meccanismo è offerto dal protocollo ARP.

Uno scenario tipico:

- 1) A deve spedire un datagram a B, host appartenente alla medesima rete logica (cioè, alla medesima rete IP)
- 2) A conosce l'indirizzo IP di B, ma non il suo indirizzo fisico

Soluzione tramite ARP:

- A manda in broadcast a tutti gli host della rete un pacchetto contenente l'indirizzo di rete di B, allo scopo di conoscere l'indirizzo fisico di B



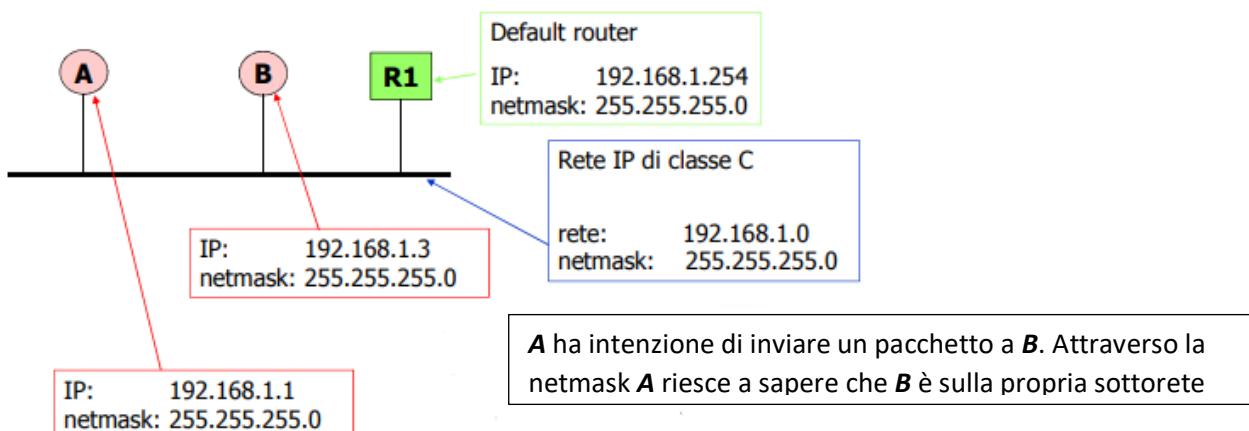
- *B* riconosce il suo indirizzo di rete e risponde ad *A*
- Finalmente *A* conosce l'indirizzo fisico di *B*, quindi può spedire il datagram a *B*

Il protocollo ARP interagisce direttamente con il livello data link.
Il pacchetto ARP viene **incapsulato** in un frame e spedito in broadcast sulla rete; l'header del frame di livello 2 specifica che il frame contiene un pacchetto ARP

ARP: scenario 1

L'host destinazione è sulla stessa LAN (stessa subnet IP)

Formato del pacchetto ARP	
Hardware Type	Protocol Type
HLEN	PLEN
Operation	
Sender Hardware Address	
Sender HW Address	Sender IP Address
Sender IP Address	Target HW Address
Target Hardware Address	
Target IP Address	

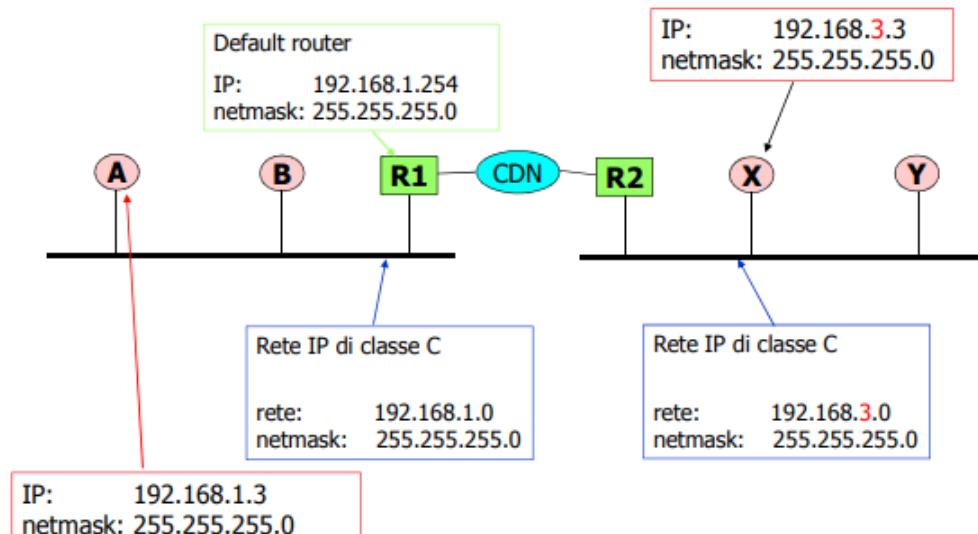


- Ogni computer ha un indirizzo IP ed una netmask. La netmask serve ad individuare la propria sottorete IP:
 - Digitare da una shell win2000 il comando: ipconfig/all
- Il computer **A** esegue una AND tra l'indirizzo IP destinazione e la propria netmask.
 - Nel caso precedente:

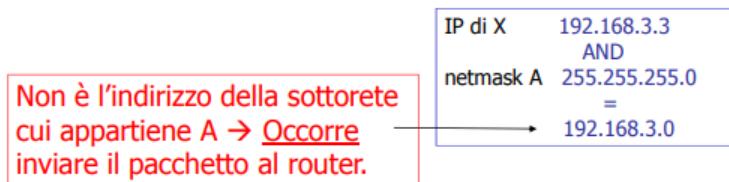
E' proprio l'indirizzo della sottorete IP cui appartiene A	IP di B 192.168.1.2 AND netmask A 255.255.255.0 = 192.168.1.0
--	---
- Se il computer **B** è sulla stessa sottorete IP allora mando un pacchetto ARP *request* in broadcast
 - Tale pacchetto contiene, nel campo DEST IP, l'indirizzo IP di **B**

ARP: scenario 2

L'host destinazione non è sulla stessa LAN (subnet IP)



- Se **A** intende mandare un pacchetto a **X**, l'operazione di AND tra la netmask e l'indirizzo IP DEST fornisce un risultato differente



- In questo caso, pertanto, si prepara un pacchetto ARP in cui si specifica come indirizzo IP DEST proprio l'indirizzo IP del router

ARP: ricapitolando

Operazione di AND logico tra l'indirizzo IP della destinazione e la propria netmask:

- Se il risultato fornisce l'indirizzo della propria subnet IP invia una richiesta ARP per risolvere l'indirizzo della destinazione
- Altrimenti il pacchetto deve essere inviato al router di default
 - Nel caso in cui l'indirizzo MAC del router non sia noto invia una richiesta ARP per risolvere l'indirizzo IP del router

Monitoraggio di ARP

Per ridurre il traffico sulla rete, ogni host mantiene una cache con le corrispondenze tra indirizzi logici e fisici. Prima di spedire una richiesta ARP controlla nella cache.

Il pacchetto ARP contiene indirizzo fisico e logico del mittente. Gli host che leggono il pacchetto possono aggiornare le loro ARP cache.

Con il comando *arp* è possibile leggere e modificare il contenuto della arp cache

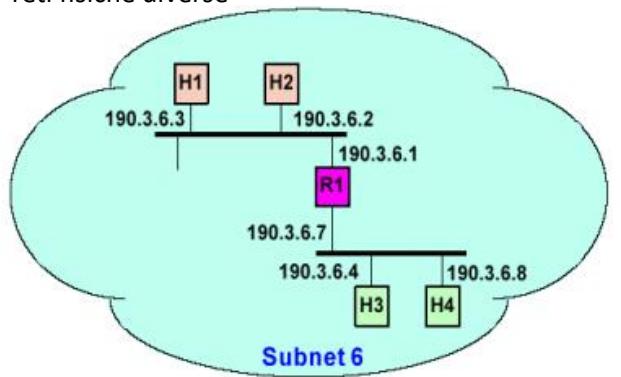
- arp -a*: legge il contenuto di tutta la cache

Con il comando *tcpdump* (o con un software tipo Ethereal) è possibile monitorare tutto il traffico che viaggia sulla rete

- È possibile filtrare solo i pacchetti spediti da un dato protocollo su una data interfaccia
- tcpdump arp*: legge solo i pacchetti arp

Proxy ARP

Permette di usare la stessa subnet su due o più reti fisiche diverse



Reverse ARP (RARP)

Scenario RARP:

- A conosce il proprio indirizzo MAC, ma non conosce il proprio indirizzo IP
- L'host B (server RARP) conosce l'indirizzo IP di A

Soluzione:

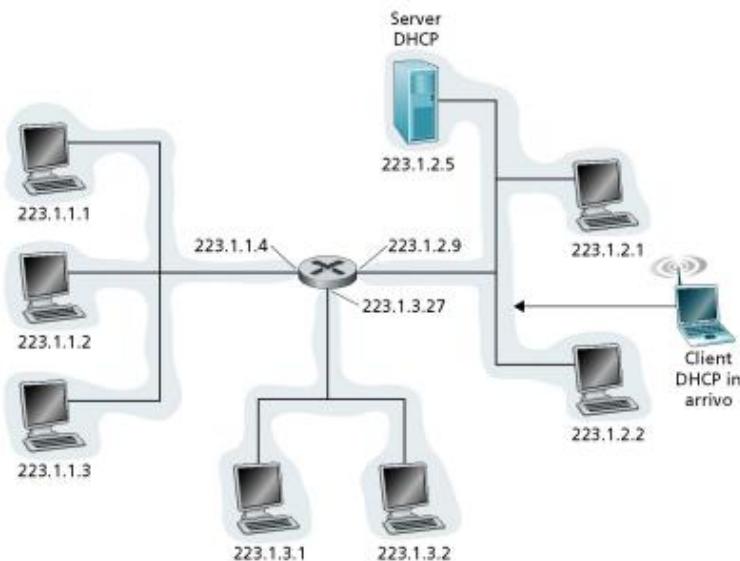
- RARP request sulla rete (in broadcast)
- B risponde con un messaggio RARP reply contenente l'indirizzo IP di A

Il protocollo RARP è stato sostituito da altri protocolli più flessibili e potenti:

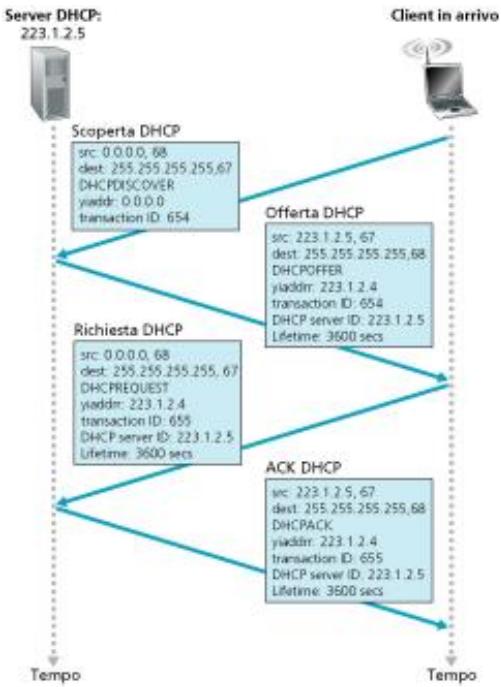
- BOOTP: BOOTstrap Protocol
- DHCP: Dynamic Host Configuration Protocol
- Utilizzati per assegnare dinamicamente gli indirizzi agli host di una rete IP

Dynamic Host Configuration Protocol (DHCP)

Scenario tipico:



Interazione client-server via DHCP:



Internet Control Message Protocol (ICMP)

I messaggi sono individuati da un tipo e da un codice:

Funzionalità

- Verificare lo stato della rete
 - echo request, echo reply
 - Riportare anomalie
 - destination unreachable
 - time exceeded
 - parameter problem
 - Scoprire la netmask
 - mask request
 - address mask reply
 - Migliorare il routing
 - redirect
- | Type | Code | description |
|------|------|---|
| 0 | 0 | echo reply (ping) |
| 3 | 0 | dest. network unreachable |
| 3 | 1 | dest host unreachable |
| 3 | 2 | dest protocol unreachable |
| 3 | 3 | dest port unreachable |
| 3 | 6 | dest network unknown |
| 3 | 7 | dest host unknown |
| 4 | 0 | source quench (congestion control - not used) |
| 8 | 0 | echo request (ping) |
| 9 | 0 | route advertisement |
| 10 | 0 | router discovery |
| 11 | 0 | TTL expired |
| 12 | 0 | bad IP header |

Applicazioni: Ping

- Utilizzato per verificare la connettività a livello rete tra due host, A e B
 - L'host A invia un pacchetto "echo request"
 - Alla ricezione di tale messaggio, l'host B risponde con un pacchetto "echo reply"

Applicazioni: Traceroute

- Utilizzato per scoprire il percorso seguito per raggiungere una certa destinazione
- Viene inviata una serie di pacchetti con TTL via via crescente, a partire da 1:
 - Il router che, decrementando il TTL, lo azzerà invierà indietro un messaggio "time exceeded"
 - In questo modo si riesce a determinare il percorso fino alla destinazione

11. Router NAT

Network Address Translation (NAT)

NAT consente ad un dispositivo 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.

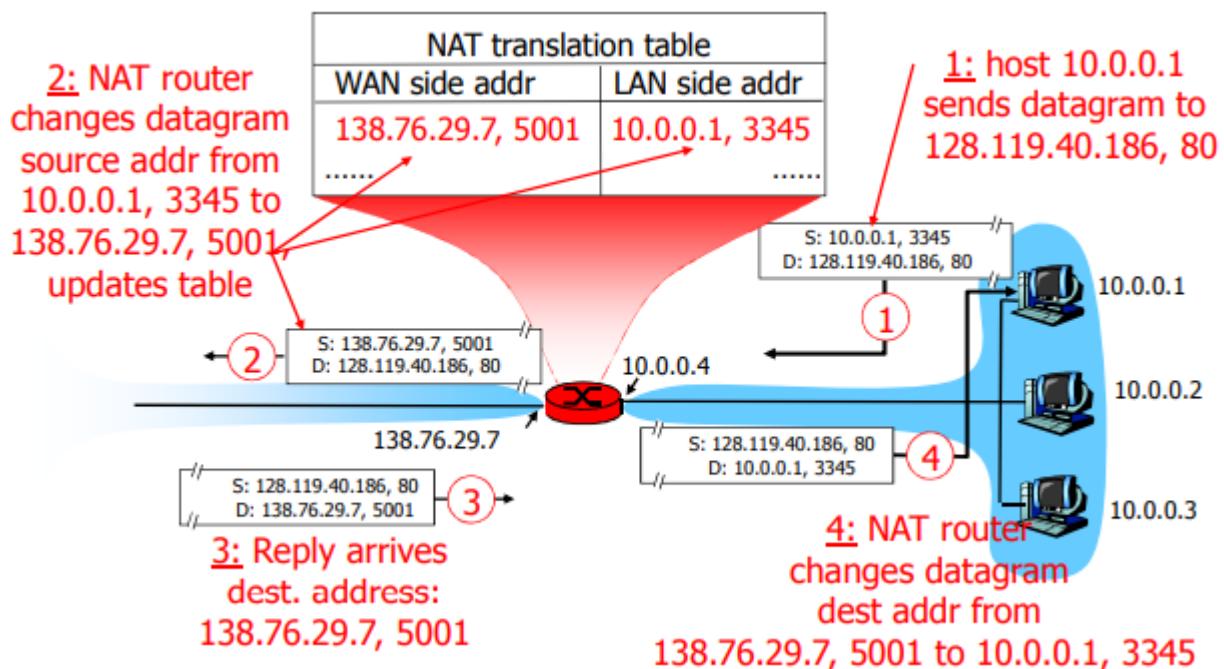
L'uso più comune del NAT è quello di mappare un insieme di indirizzi privati su di un unico indirizzo pubblico, utilizzando differenti porti per mantenere traccia delle diverse connessioni.

Quando il router riceve un pacchetto inviato da un computer della rete privata ad un computer esterno, salva in una tabella l'indirizzo e il porto del mittente, oltre ai nuovi valori che esso assegna.

Tale tabella viene consultata anche quando il router riceve un pacchetto dal computer destinazione

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

Un esempio



Implementazione

Il router NAT deve:

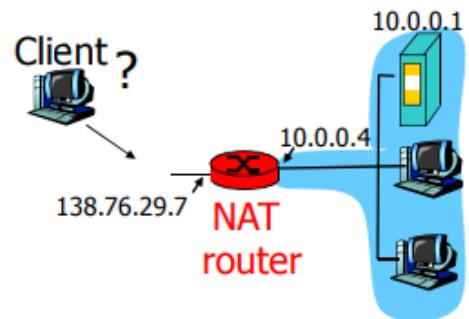
- **Sostituire i datagrammi in uscita**
 - (Indirizzo IP sorgente, numero porta) di ogni datogramma in uscita con (indirizzo NAT IP, nuovo numero porta)
 - client/server remoti risponderanno utilizzando (indirizzo NAT IP, nuovo numero porta) come indirizzo di destinazione
- **Ricordare nella tabella di traduzione NAT**
 - Ogni coppia di traduzioni (Indirizzo IP sorgente, numero porta) a (indirizzo NAT IP, nuovo numero porta)

- **Sostituire i datagrammi in entrata**
 - (indirizzo NAT IP, nuovo numero porta) nei campi destinazione di ogni datagramma di ingresso con il corrispettivo (Indirizzo IP sorgente, numero porta) memorizzato nella tabella NAT

Problemi di attraversamento

Il client desidera connettersi al server con indirizzo 10.0.0.1

- Il server address 10.0.0.1 locale alla LAN non può essere usato come indirizzo di destinazione
- Solo un indirizzo NAT è visibile esternamente: 138.76.29.7



Soluzione 1

Configurare staticamente NAT per inoltrare richieste di connessione in entrata al dato server port

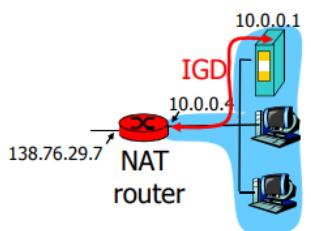
- Esempio: (123.76.29.7, porto 2500) sarà sempre inoltrato a (10.0.0.1, porto 2500)

Soluzione 2

Universal Plug and Play (UPnP) protocollo Internet Gateway Device (IGD).

Consente a host NAT di:

- Imparare l'indirizzo IP pubblico: 138.76.29.7
- Aggiungere/rimuovere la mappatura del porto (con tempo residuo)



12. Il protocollo IPv6

IP Next Generation (IPng o IPv6)

IPv6 è la nuova versione del protocollo Internet e mira a risolvere parte dei problemi che Internet sta incontrando a causa della sua crescita vertiginosa.

Le principali questioni affrontate da questo protocollo sono: indirizzamento e routing, sicurezza, configurazione automatica e servizi di tipo real-time.

Problemi affrontati in fase di progetto

- Supportare reti interconnesse di tipo globale
- Garantire una transizione chiara e diretta dell'immensa base di sistemi utilizzanti IPv4
- Sostenere l'elevato tasso di crescita delle reti
- Far fronte ai possibili scenari futuri nel mondo dell'internetworking:
 - mobile computing
 - networked entertainment
 - etc...

La sfida per IPng è quella di completare la transizione prima che i problemi di IPv4 legati all'indirizzamento ed al routing emergano in tutta la loro tragicità. Cioè prima che gli indirizzi di Internet perdano la loro unicità a livello globale.

Ciò richiede una strategia di sviluppo flessibile ed incrementale, oltre all'interoperabilità tra le due versioni di IP.

IPv6: modifiche principali rispetto ad IPv4

Espansione capacità di indirizzamento e di routing

- la dimensione degli indirizzi passa da 32 (4 byte) a 128 bit (16 byte), per supportare una gerarchia su più livelli ed un numero di nodi molto più elevato

- la scalabilità del routing multicast è migliorata grazie all'aggiunta di un campo scope agli indirizzi di classe D
- viene definito un nuovo tipo di indirizzo (anycast address) che si riferisce ad un insieme di interfacce
 - un pacchetto inviato ad un indirizzo anycast viene recapitato ad una delle interfacce che fanno parte dell'insieme da esso individuato (tipicamente da quella più "vicina", secondo la misura di "distanza" utilizzata dal protocollo di routing)

IPv6: caratteristiche generali

Semplificazione del formato dell'header:

- alcuni campi dell'header (quelli che vengono sfruttati solo in casi particolari) sono stati eliminati o resi opzionali
 - ciò ha consentito di ottenere che, malgrado gli indirizzi di IPv6 siano 4 volte più lunghi di quelli di IPv4, l'header del primo è soltanto il doppio di quello del secondo

Supporto per le opzioni migliorato:

- alcuni cambiamenti nel modo di codificare le opzioni permettono uno smistamento più efficiente ed una maggiore flessibilità per introdurre, in futuro, nuove funzionalità

Supporto della Quality of Service (QoS):

- viene introdotta una nuova funzionalità per permettere di etichettare (flow label) i pacchetti appartenenti a flussi di dati particolari per i quali si richiede un trattamento di tipo non-default

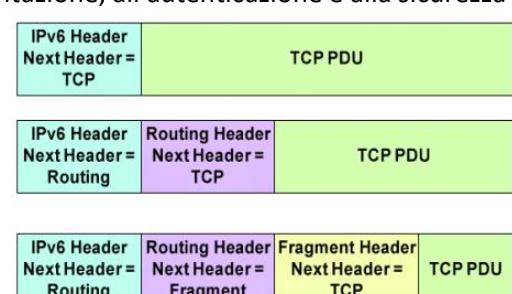
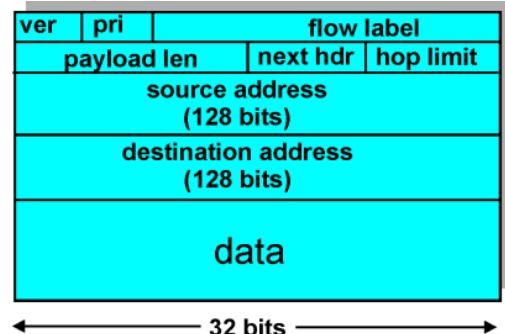
Autenticazione e salvaguardia della privacy:

- definizione di estensioni che forniscono il supporto per:
 - l'autenticazione
 - l'integrità dei dati
 - la sicurezza, considerata elemento fondamentale del nuovo protocollo

L'header IPv6

L'header IPv6 consiste di due parti:

- **header principale**
 - *Vers*: numero della versione
 - *Prio*: livello di priorità del datagramma
 - *Flow Label*: associato alla QoS richiesta
 - *Payload Length*: lunghezza del payload
 - *Next Hdr*: tipo di header che segue quello IPv6
 - *Hop Limit*: contatore del numero di hops
 - *Source Address*: indirizzo del mittente
 - *Destination Address*: indirizzo del destinatario
- **extension headers**: introdotti per ospitare le eventuali opzioni e situati, all'interno del pacchetto, in una posizione intermedia tra l'header principale e l'header del protocollo di trasporto; inoltre, forniscono informazioni relative: al routing, alla frammentazione, all'autenticazione e alla sicurezza
 - **Hop by hop** option header (es.: jumbograms)
 - **Routing** header: strict source routing, loose source routing
 - **Fragment** header: gestisce la frammentazione
 - **Authentication** header
 - **Encrypted security payload** header
 - **Destination option** header



Differenze con l'header IPv4

- **Checksum:** rimossa completamente per ridurre il tempo di processamento nei router ad ogni hop
- **Options:** sono previste, ma non nell'header. È possibile prevederle fuori dall'header utilizzando il campo "Next Header"
- **ICMPv6:** nuova versione di ICMP
 - Messaggi addizionali, e.g. "Packet Too Big"
 - Funzioni per il management dei gruppi multicast

Modalità di scrittura degli indirizzi in IPv6

Si scrivono in esadecimale come 8 numeri naturali separati da ":"

- FEDC:BA98:0876:45FA:0562:CDAF:3DAF:BB01
- 1080:0000:0000:0007:0200:A00C:3423

Esistono delle semplificazioni:

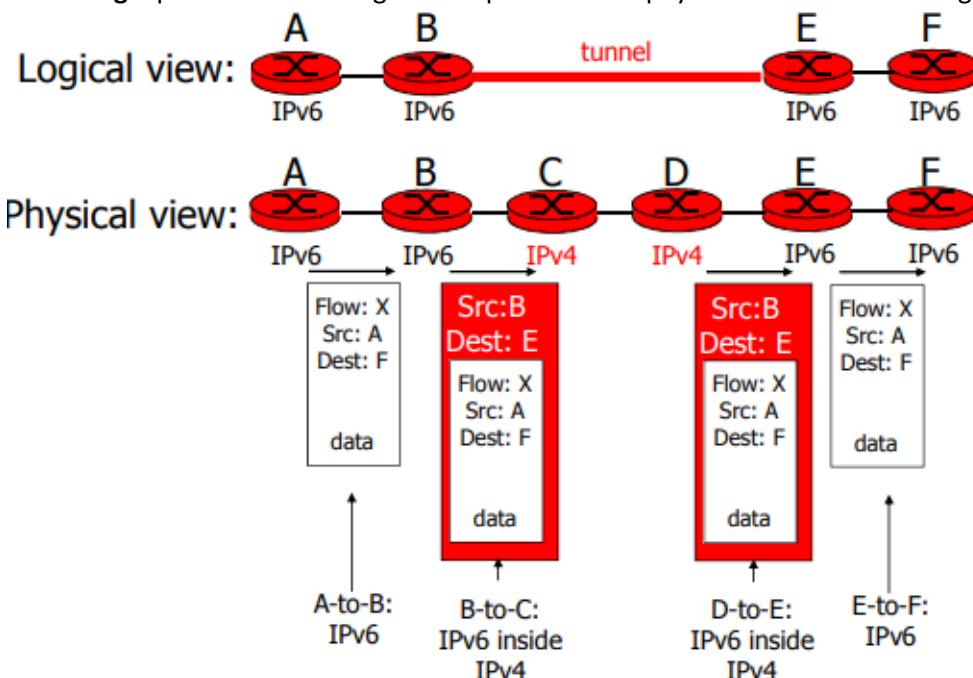
- Si possono omettere gli zero iniziali
 - 1080:0:0:7:200:A00C:3423
- Si possono sostituire gruppi di zero con "::"
 - 1080::7:200:A00C:3423

Indirizzo compatibile con IPv4

- L'indirizzo compatibile con IPv4, 0:0:0:0:0:w.x.y.z oppure ::w.x.y.z, dove w.x.y.z è la rappresentazione decimale separata da punti di un indirizzo IPv4 pubblico.
- Viene utilizzato dai nodi a doppio stack che comunicano con IPv6 su un'infrastruttura IPv4.
- I nodi a doppio stack utilizzano entrambi i protocolli IPv4 e IPv6.
- Quando un indirizzo compatibile con IPv4 viene utilizzato come destinazione IPv6, il traffico IPv6 viene incapsulato automaticamente con un'intestazione IPv4 e inviato alla destinazione utilizzando l'infrastruttura IPv4.
- Usati nella transizione da IPv4 a IPv6, quando un nodo che supporta sia IPv6 che IPv4 non ha un router IPv6 a cui inviare i pacchetti.

Tunneling: transizione da IPv4 a IPv6

Tunneling: I pacchetti IPv6 vengono trasportati come payload all'interno di datagrammi IPv4 tra router IPv4

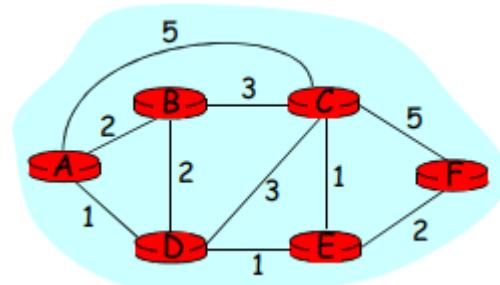


13. Routing: Introduzione

Reti di calcolatori e grafi

Una rete di calcolatori è modellata come grafo dove i router rappresentano i nodi e i link fisici gli archi. Di conseguenza, gli algoritmi per la gestione di una rete sono basati sulla teoria di grafi.

La scelta del cammino non si basa solo su quella a costo minimo ma vengono considerate anche altre possibilità (un cammino calcolato in base a specifici vincoli...)



I parametri del processo decisionale sono i seguenti:

- **Bandwidth:** capacità di un link, tipicamente definita in bit per secondo (bps)
- **Delay:** il tempo necessario per spedire un pacchetto da una sorgente ad una destinazione
- **Load:** una misura del carico di un link
- **Reliability:** riferita, ad esempio, all'error rate di un link
- **Hop count:** il numero di router da attraversare nel percorso dalla sorgente alla destinazione
- **Cost:** un valore arbitrario che definisce il costo di un link (ad esempio, costruito come funzione di diversi parametri, tra cui: bandwidth, delay, packet loss, MTU, etc...)

Il processo di routing

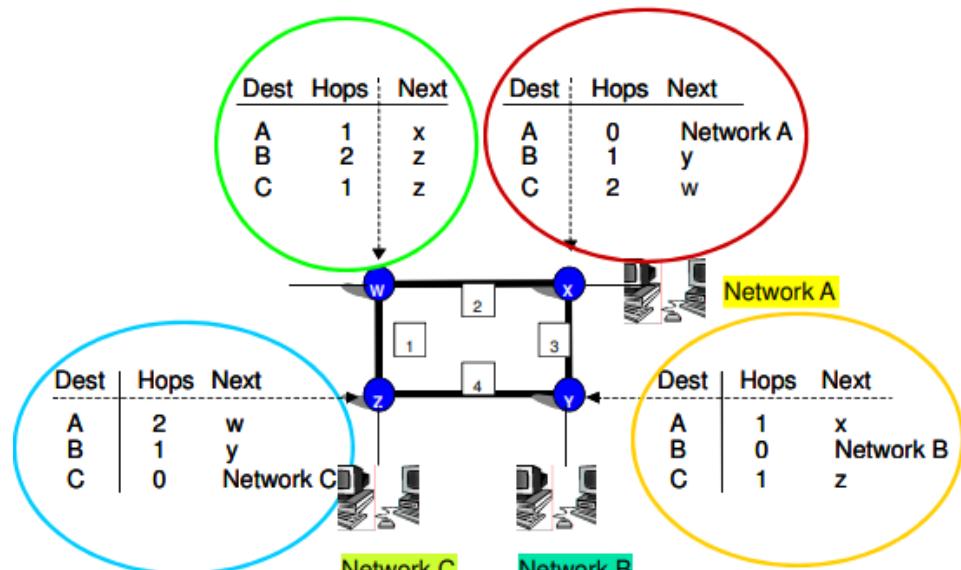
Il processo di routing è un processo decisionale. Ogni entità che partecipa a questo processo:

- mantiene delle informazioni
- in base ad uno specifico algoritmo ed in funzione di determinate metriche:
 - definisce il procedimento di instradamento verso le possibili destinazioni
- può spedire informazioni di aggiornamento alle altre entità coinvolte, secondo diversi paradigmi

La funzione principale di un router è quella di determinare i percorsi che i pacchetti devono seguire per arrivare a destinazione, partendo da una data sorgente: ogni router si occupa, quindi, del processo di ricerca di un percorso per l'instradamento di pacchetti tra due nodi qualunque di una rete.

Problemi da risolvere:

- Quale sequenza di router deve essere attraversata?
- Esiste un percorso migliore (più breve, meno carico, ...)?
- Cosa fare se un link si guasta?
- Trovare una soluzione robusta e scalabile...



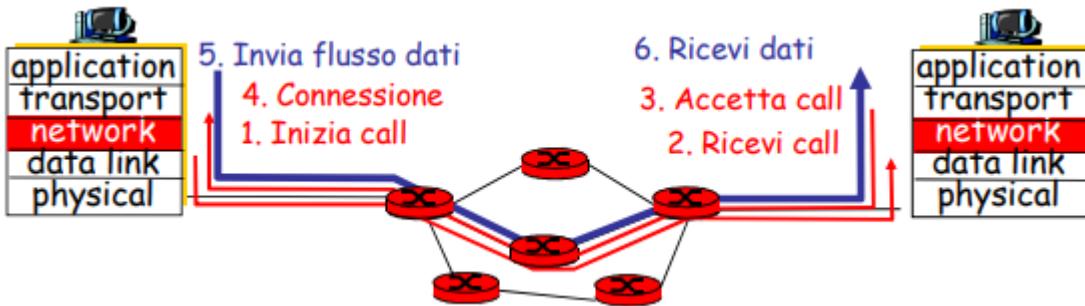
Tecniche di routing

- Routing by Network Address
 - ogni pacchetto contiene l'indirizzo del nodo destinatario, che viene usato come chiave di accesso alle tabelle di instradamento

- usato tipicamente nei protocolli non orientati alla connessione: IPv4 e IPv6, bridge trasparenti, OSI CLNP, ...
- Label Swapping
 - ogni pacchetto è marcato con una label (etichetta) che: identifica la connessione e viene usata come chiave per determinare l'instradamento
 - generalmente usato nei protocolli orientati alla connessione: X.25, ATM, MPLS, ...

Routing: reti a circuiti virtuali

- Viene aperta una connessione prima di inviare i dati



- Non esiste la fase di call setup a livello rete
- Nei router non esiste il concetto di connessione
- I pacchetti sono indirizzati usando un ID di destinazione:
 - pacchetti fra la stessa coppia sorgente-destinazione possono seguire strade diverse

Tipologie di routing

La scelta del percorso di instradamento può essere realizzata mediante due approcci:

- **centralizzato**: più semplice, ma non scalabile
- **distribuito**: più complesso, ma scalabile e robusto

Lo scopo ultimo di un protocollo di routing consiste nel creare una **tavella di instradamento** in ciascun nodo della rete:

- ciascun nodo deve prendere una decisione locale sulla base della conoscenza dello stato dell'intera rete
- Questa è, probabilmente, la difficoltà principale del routing

Routing centralizzato

- Esiste un nodo centrale che calcola e distribuisce le tabelle
 - tale nodo riceve informazioni sullo stato della rete da tutti gli altri e calcola le nuove tabelle
- Ottimizza le prestazioni, ma è poco robusto
 - aggiornamenti parziali delle tabelle dovuti a guasti possono generare loop
 - induce un notevole carico sulla rete, specialmente in prossimità del nodo centrale

Routing distribuito

- Ogni router calcola le sue tabelle dialogando con gli altri router:
 - Ogni router informa gli altri riguardo le "rotte" che conosce
- Il dialogo tra router avviene tramite dei protocolli ausiliari di livello rete
- Comprende due approcci principali:
 - Algoritmi **distance vector**
 - Algoritmi **link state**
- Utilizzato in varie reti proprietarie, in OSI, ed in Internet

Problematiche associate al routing

Un router deve opportunamente sintetizzare le informazioni rilevanti utili alle proprie decisioni:

- Per prendere correttamente decisioni locali bisogna avere almeno una conoscenza parziale dello stato globale della rete
- Lo stato globale della rete è difficile da conoscere in quanto si può riferire ad un dominio molto esteso e che cambia in maniera estremamente dinamica

Le tabelle di routing devono essere memorizzate all'interno dei router:

- Bisogna minimizzare l'occupazione di spazio e rendere rapida la ricerca
- Bisogna minimizzare il numero di messaggi che i router si scambiano

Si deve garantire la robustezza dell'algoritmo.

Scambio delle informazioni di update

- Broadcast periodico
 - I router possono trasmettere agli altri router informazioni circa la raggiungibilità delle reti (destinazioni) di propria competenza ad intervalli regolari di tempo
 - Questa tecnica risulta inefficiente, in quanto si spediscono informazioni anche quando non è cambiato nulla rispetto all'update precedente
- Event-driven
 - In questo caso gli update sono inviati solo quando è cambiato qualcosa nella topologia oppure nello stato della rete
 - Questa tecnica garantisce un uso più efficiente della banda disponibile

Scelta dell'algoritmo di routing

- **Problematiche:**
 - Possono esistere più criteri di ottimalità contrastanti (esempio: "minimizzare il ritardo medio di ogni pacchetto" vs "massimizzare l'utilizzo dei link della rete")
 - Il numero di nodi può essere elevato
 - La topologia può essere complessa
 - Algoritmi troppo complessi, operanti su reti molto grandi, potrebbero richiedere tempi di calcolo inaccettabili
 - Vincoli di tipo amministrativo
- **Parametri**
 - **Semplicità:** i router hanno CPU e memoria finite
 - **Robustezza:** adattabilità alle variazioni (di topologia, di carico, ...)
 - **Stabilità:** l'algoritmo deve convergere in tempo utile
 - **Equità:** stesso trattamento a tutti i nodi
 - **Metrica da adottare:** numero di salti effettuati, somma dei costi di tutte le linee attraversate, etc...

14. Routing Distance Vector e Routing Link State

Distance Vector

Ogni router mantiene una tabella di tutti gli instradamenti a lui noti, inizialmente, solo le reti a cui è connesso direttamente.

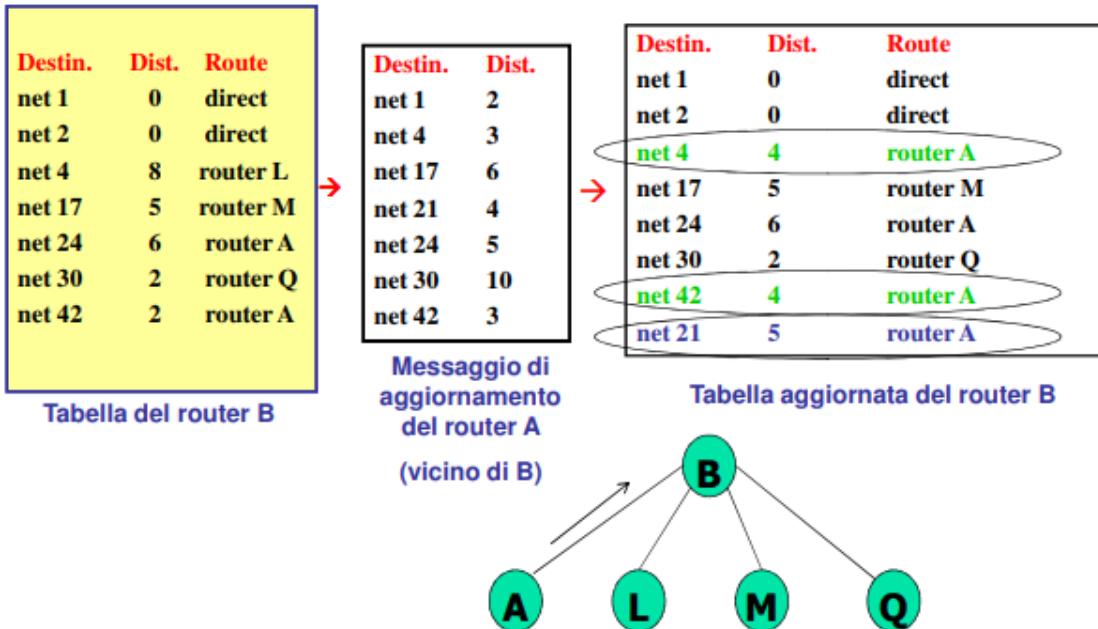
Ogni entry della tabella indica:

- una rete raggiungibile
- il next hop
- il numero di hop necessari per raggiungere la destinazione

Periodicamente, ogni router invia a tutti i vicini (due router sono vicini se sono collegati alla stessa rete fisica) un messaggio di aggiornamento contenente tutte le informazioni della propria tabella (vettore delle distanze o distance vector)

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

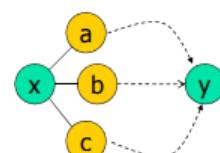


Il distance vector risulta facile da implementare ma ha i seguenti svantaggi:

- ogni messaggio contiene un'intera tabella di routing
- lenta propagazione delle informazioni sui cammini:
 - se lo stato della rete cambia velocemente, le rotte possono risultare inconsistenti

Equazione di Bellman-Fort

Definito $d_x(y) :=$ costo del percorso a costo minore tra x ed y allora $d_x(y) = \min_v \{c(x, v) + d_v(y)\}$ dove il minimo è calcolato tra tutti i nodi v adiacenti ad x .



Ogni nodo:

- Invia ai nodi adiacenti un distance vector, costituito da insieme di coppie (indirizzo, distanza), dove la distanza è espressa tramite metriche classiche, quali numero di hop e costo
- Memorizza per ogni linea l'ultimo distance vector ricevuto.
- Calcola le proprie tabelle di instradamento.
- Se le tabelle risultano diverse da quelle precedenti invia ai nodi adiacenti un nuovo distance vector

Elaborazione

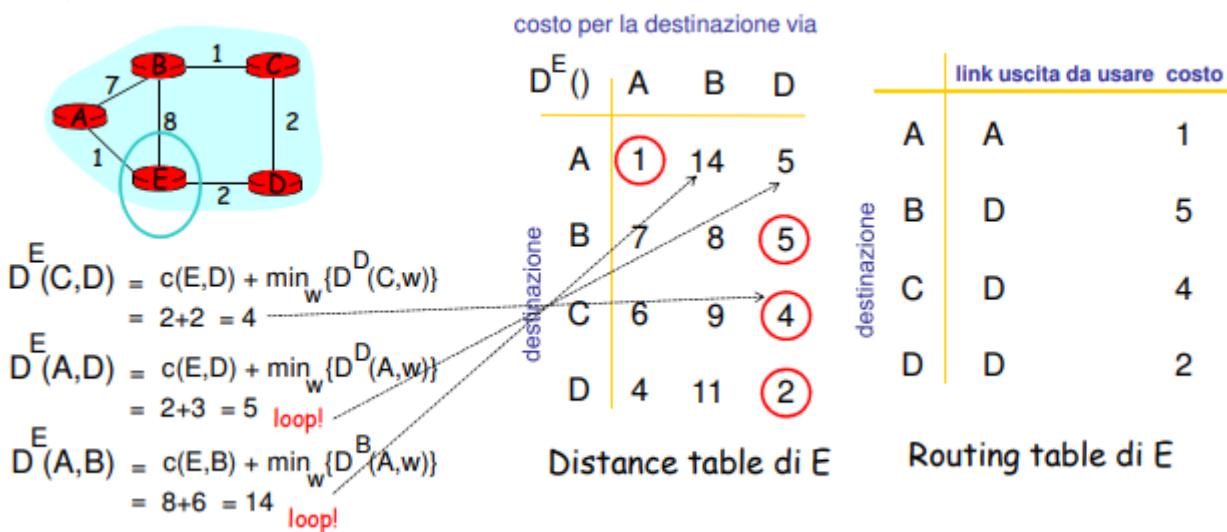
- Il calcolo consiste nella fusione di tutti i distance vector delle linee attive
- Un router ricalcola le sue tabelle se:
 - cade una linea attiva
 - riceve un distance vector, da un nodo adiacente, diverso da quello memorizzato
- Se le tabelle risultano diverse da quelle precedenti:
 - invia ai nodi adiacenti un nuovo distance vector

- Vantaggi:
 - Molto semplice da implementare
- Svantaggi:
 - Possono innescarsi dei loop a causa di particolari variazioni della topologia
 - Converge alla velocità del link più lento e del router più lento
 - Difficile capirne e prevederne il comportamento su reti grandi (nessun nodo ha una mappa della rete)

Caratteristiche

- **Iterativo:** continua fino a quando non c'è più scambio di informazioni
 - Self-terminating: non c'è un esplicito segnale di stop
- **Asincrono**
- **Distribuito:** ogni nodo comunica con i diretti vicini
- **Struttura Distance Table:** ogni nodo ha la sua tabella delle distanze:
 - Una riga per ogni destinazione
 - Una colonna per ogni nodo adiacente
- Notazione: $D^X(Y, Z) = \text{distanza da } X \text{ a } Y, \text{ via } Z (\text{prossimo hop}) = c(X, Z) + \min_w \{D^Z(Y, w)\}$

Esempio



Distance Vector: ricapitolando...

- **Iterativo, asincrono:** ogni iterazione locale è causata da:
 - Cambiamento di costo di un collegamento
 - Messaggi dai vicini
- **Distribuito:** ogni nodo contatta i vicini solo quando un suo cammino di costo minimo cambia
 - I vicini, a loro volta, contattano i propri vicini se necessario



L'algoritmo

Ad ogni nodo, x :

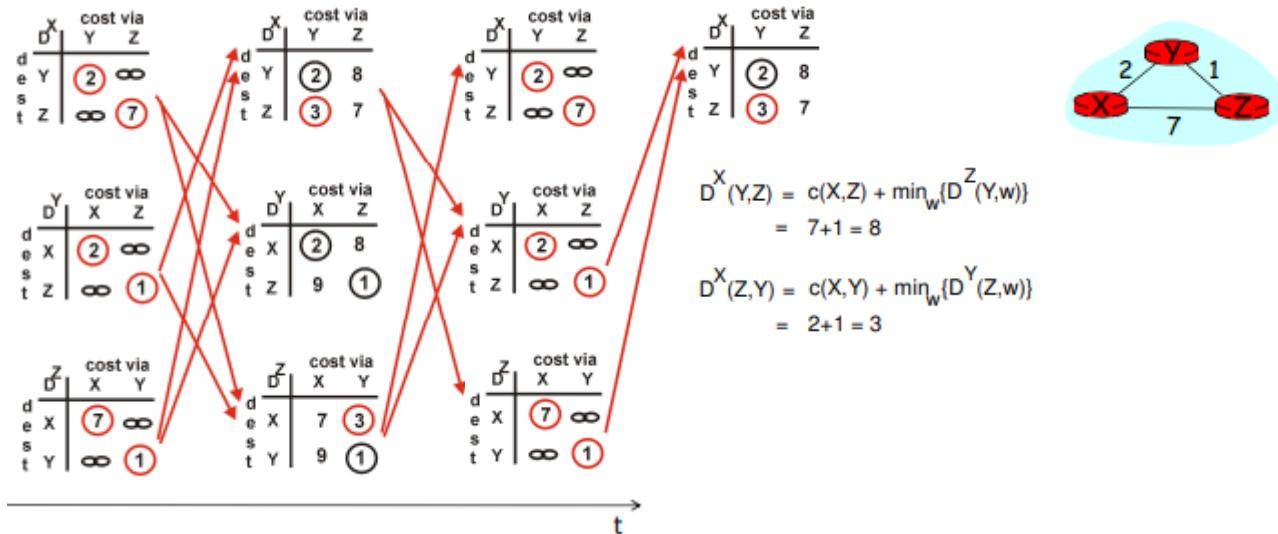
Inizializzazione:

- per tutti i nodi adiacenti v :
 - $D^X(*, v) = \infty$ {il simbolo * significa “per ogni riga”}
 - $D^X(v, v) = c(x, v)$
- Per tutte le destinazioni, y
 - manda $\min_w D(y, w)$ a ogni vicino

→ Loop:

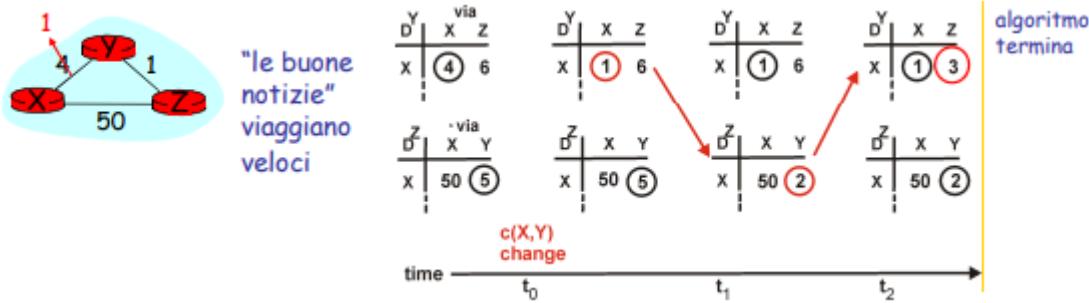
- aspetta
 - fino a quando vedo una modifica nel costo di un collegamento
 - oppure ricevo un messaggio da un vicino v
- if ($c(x, v)$ cambia di d)
 - cambia il costo di tutte le destinazioni via vicino v di d (d può essere positivo o negativo)
 - per tutte le destinazioni y : $D^X(y, v) = D^X(y, v) + d$
- else if (ricevo messaggio aggiornamento da v verso destinazione y)
 - cammino minimo da v a y è cambiato
 - V ha mandato un nuovo valore per il suo $\min_w D^V(y, w)$
 - chiama questo valore newval
 - per la singola destinazione y : $D^X(y, v) = c(x, v) + \text{newval}$
- if (hai un nuovo $\min_w D^X(y, w)$ per una qualunque destinazione y)
 - manda il nuovo valore $\min_w D^X(y, w)$ a tutti i vicini

Esempio completo

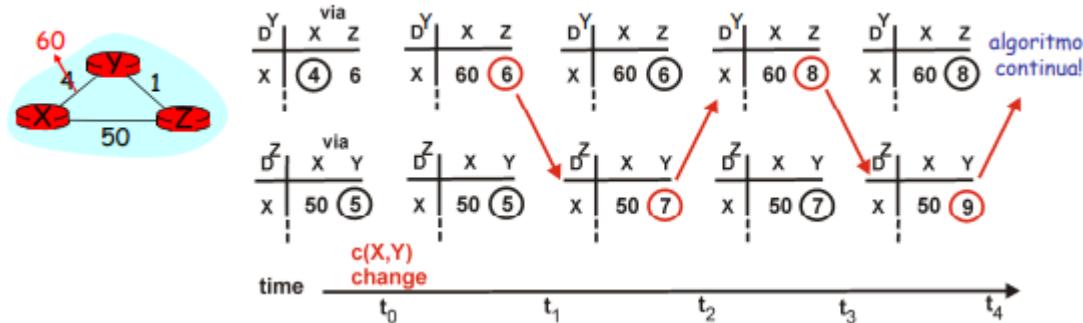


Modifica dei costi dei collegamenti

- Un nodo si accorge di una modifica locale al costo di un link ad esso connesso
- Aggiorna la sua distance table (primo if dell'algoritmo)
- Se cambia il costo di qualche path allora lo notifica ai vicini (ultimo if dell'algoritmo)

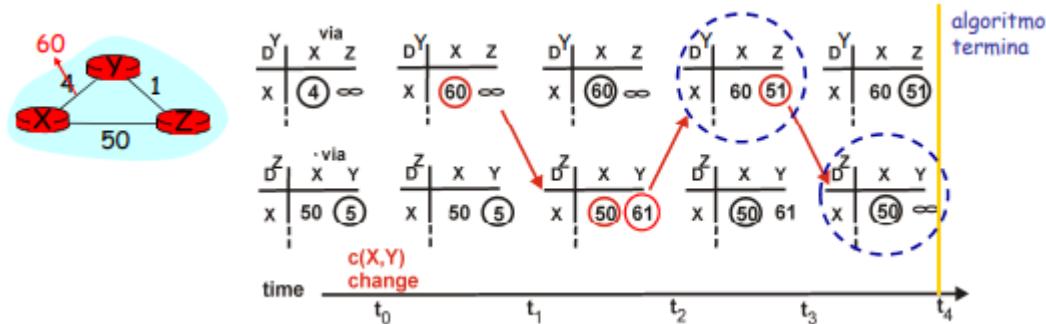


Cambiamento nei costi: le “buone notizie” viaggiano in fretta, le cattive lentamente... Conteggio all’infinito!



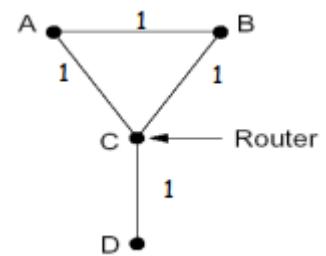
Poisoned reverse: Se z raggiunge x tramite y:

- z dice a y che la sua distanza per x è infinita (così y non andrà a x attraverso z)



Esempio in cui lo split horizon fallisce

- Quando il link tra C e D si interrompe, C “setterà” la sua distanza da D ad ∞
- Però, A userà B per andare a D e B userà A per andare a D.
- Dopo questi update, sia A che B riporteranno un nuovo percorso da C a D (diverso da ∞)



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, noto come Shortest Path First (SPF)

Tale approccio è utilizzato nello standard ISO e nel protocollo OSPF (adottato in reti TCP/IP)

Il processo di update

Ogni router genera un Link State Packet (LSP) contenente:

- 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
- checksum
- Lifetime: la validità di ogni LSP è limitata nel tempo
 - Ad esempio: un errore sul numero di sequenza potrebbe rendere un LSP valido per anni

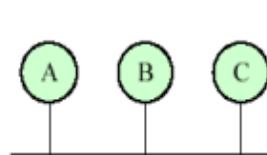
LSP flooding

- Un LSP viene generato periodicamente, oppure quando viene rilevata una variazione nella topologia locale (adiacenze), ossia:
 - Viene riconosciuto un nuovo vicino
 - Il costo verso un vicino è cambiato
 - Si è persa la connettività verso un vicino precedentemente raggiungibile
- Un LSP è trasmesso in flooding su tutti i link del router
- I pacchetti LSP memorizzati nei router formano una mappa completa e aggiornata della rete:
 - Link State Database

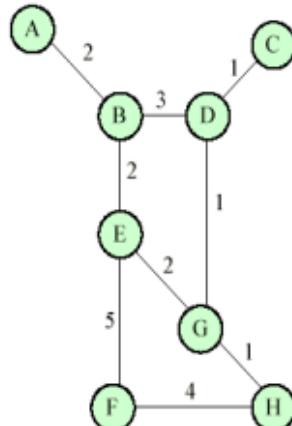
Esempio

Trasmissione di un LSP

Adiacente	Costo
A	4
B	4
C	4
R1	0
R2	3
R3	5



grafo della rete



LSP Database

A	B/2	D/3	E/2
B	A/2		
C	D/1		
D	B/3	C/1	G/1
E	B/2	F/5	G/2
F	E/5	H/4	
G	D/1	E/2	H/1
H	F/4	G/1	

(replicato su ogni IS)

Gestione degli LSP

All'atto della ricezione di un LSP, il router compie le seguenti azioni:

1. Se non ha mai ricevuto LSP da quel router o se l'LSP è più recente di quello precedentemente memorizzato:
 - memorizza il pacchetto
 - lo ritrasmette in flooding su tutte le linee eccetto quella da cui l'ha ricevuto
2. se l'LSP ha lo stesso numero di sequenza di quello posseduto:
 - non fa nulla
3. Se l'LSP è più vecchio di quello posseduto:
 - Trasmette al mittente il pacchetto più recente

Decisioni

- Il router elabora il Link State Database per produrre il Forwarding Database:
 - si pone come radice dello shortest-path tree

- cerca lo shortest path per ogni nodo destinazione
- memorizza il vicino (i vicini) che sono sullo shortest path verso ogni nodo destinazione
- Il Forwarding Database contiene, per ogni nodo destinazione:
 - l'insieme delle coppie {path, vicino}
 - la dimensione di tale insieme

Caratteristiche

Vantaggi:

- può gestire reti di grandi dimensioni
- ha una convergenza rapida
- difficilmente genera loop, e comunque è in grado di identificarli ed interromperli facilmente
- facile da capire: ogni nodo ha la mappa della rete

Svantaggi:

- Molto complesso da realizzare
 - Es.: la prima implementazione ha richiesto alla Digital 5 anni

Esempio: tabelle di instradamento

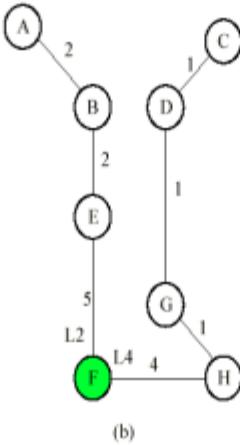
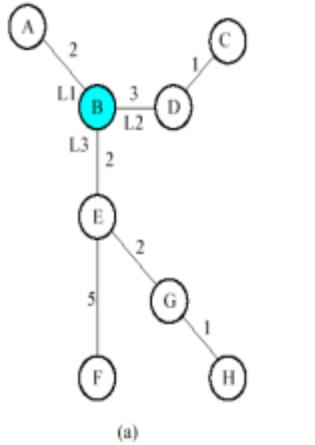


Tabella di B

A	L1
C	L2
D	L2
E	L3
F	L3
G	L3
H	L3

Tabella di F

A	L2
B	L2
C	L4
D	L4
E	L2
G	L4
H	L4

L'algoritmo di Dijkstra

Ogni nodo ha a disposizione il grafo della rete:

- I nodi sono i router
- Gli archi sono le linee di collegamento tra router:
 - Agli archi è associato un costo

Ogni nodo usa l'algoritmo di Dijkstra per costruire lo *Spanning Tree* del grafo, ovvero l'albero dei cammini di costo minimo. Ad ogni nodo si assegna un'etichetta che rappresenta il costo massimo per raggiungere quel nodo. L'algoritmo modifica le etichette cercando di minimizzare il valore e di renderle permanenti.

Formalizzazione

La topologia della rete è nota a tutti i nodi:

- La diffusione è realizzata via link state broadcast
- Tutti i nodi hanno la stessa informazione

Si calcola il percorso minimo da un nodo a tutti gli altri:

- L'algoritmo fornisce la tavola di routing per quel nodo

Iterativo: un nodo, dopo k iterazioni, conosce i cammini meno costosi verso k destinazioni

Notazione:

- $c(i,j)$: costo collegamento da i a j
 - Infinito se non c'è collegamento
 - Per semplicità, $c(i,j) = c(j,i)$
- $D(v)$: costo corrente del percorso, dalla sorgente al nodo v
- $p(v)$: predecessore (collegato a v) lungo il cammino dalla sorgente a v
- N : insieme di nodi per cui la distanza è stata trovata

Inizializzazione:

- $N = \{A\}$ per tutti i nodi v
- if (v è adiacente a A)
- then $D(v) = c(A,v)$
- else $D(v) = \infty$

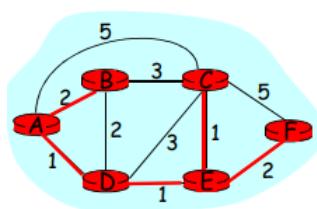
► **Loop:**

- sia w non in N tale che $D(w)$ è minimo
- aggiungi w a N
- aggiorna $D(v)$ per ogni v adiacente a w e non in N : $D(v) = \min(D(v), D(w) + c(w,v))$
- il nuovo costo fino a v è o il vecchio costo, oppure il costo del cammino più breve fino a w più il costo da w a v

fino a quando tutti i nodi sono in N

Interpretazione

- L'algoritmo consiste in un passo di inizializzazione, più un ciclo di durata pari al numero di nodi della rete. Al termine avremo i percorsi di costo minimo da A a tutte le possibili destinazioni.
- **Esempio:** Calcoliamo sulla rete data i percorsi di costo minimo da A a tutte le possibili destinazioni. Ciascuna riga della tabella seguente fornisce i valori delle variabili dell'algoritmo alla fine di ciascuna iterazione



Step	start N	$D(B), p(B)$	$D(C), p(C)$	$D(D), p(D)$	$D(E), p(E)$	$D(F), p(F)$
0	A	2,A	5,A	1,A	infinity	infinity
1	AD	2,A	4,D	2,D	infinity	
2	ADE	2,A	3,E	4,E		
3	ADEB	3,E		4,E		
4	ADEBC			4,E		
5	ADEBCF					

15. Le socket di Berkeley

Le socket

Le socket rappresentano un'astrazione di canale di comunicazione tra processi (locali o remoti). Attraverso di esse un'applicazione può ricevere o trasmettere dati.

I meccanismi restano (quasi) indipendenti dal supporto fisico su cui le informazioni viaggiano.

Inizialmente nascono in ambiente UNIX:

- Negli anni 80 la Advanced Research Project Agency finanziò l'università di Berkeley per implementare la suite TCP/IP nel sistema operativo Unix
- I ricercatori di Berkeley svilupparono il set originario di funzioni che fu chiamato interfaccia socket

Le socket si presentano sotto forma di un'API (Application Programming Interface), cioè un insieme di funzioni scritte in C, che le applicazioni possono invocare per ricevere il servizio desiderato.

Rappresentano una estensione delle API di UNIX per la gestione dell'I/O su periferica standard (files su disco, stampanti, etc...).

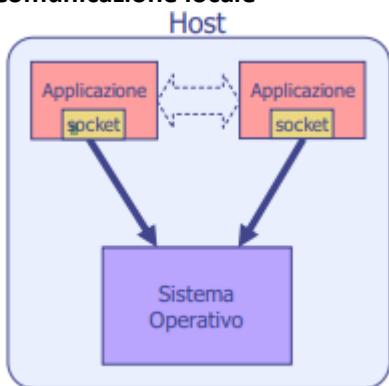
Questa API è poi divenuta uno standard de facto, ed oggi è diffusa nell'ambito di tutti i maggiori sistemi operativi (Linux, FreeBSD, Solaris, Windows, etc...)

Interazione tra Applicazione e SO

- L'applicazione chiede al sistema operativo di utilizzare i servizi di rete
- Il sistema operativo crea una socket e la restituisce all'applicazione
 - Restituito un socket descriptor
- L'applicazione utilizza la socket
 - Open, Read, Write, Close.
- L'applicazione chiude la socket e la restituisce al sistema operativo

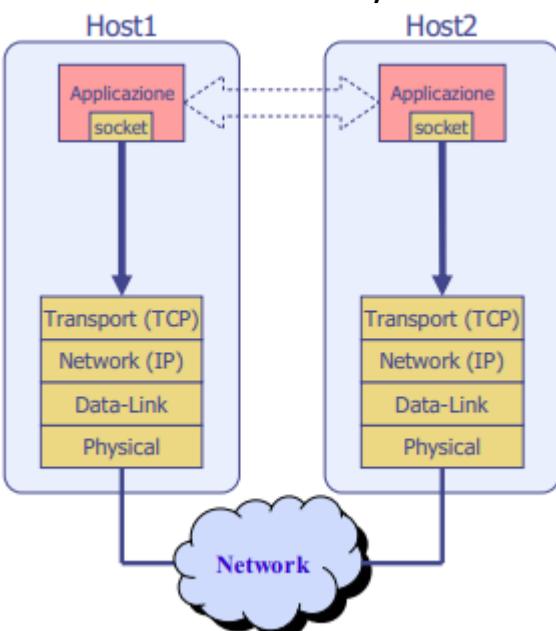
Caso d'uso:

- **Comunicazione locale**



- Due applicazioni, localizzate sulla stessa macchina, scambiano dati tra di loro utilizzando l'interfaccia delle socket.
- Le socket utilizzate a questo scopo vengono comunemente definite Unix-domain socket

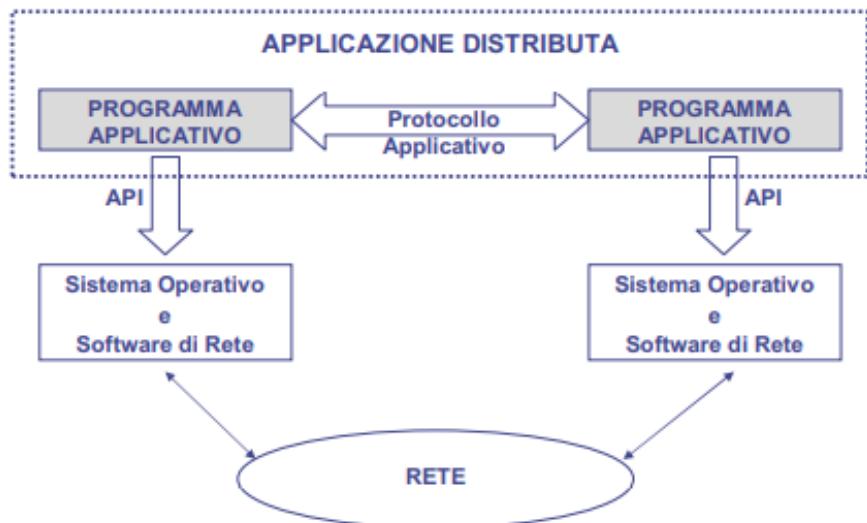
- **Comunicazione remota via TCP/IP**



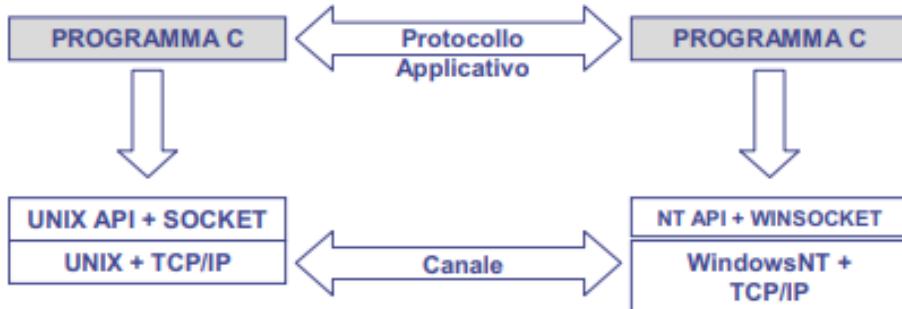
- Anche due applicazioni situate su macchine distinte possono scambiare informazioni secondo gli stessi meccanismi.
- Così funzionano telnet, ftp, ICQ, Napster.

Interfacce e protocolli

Esempio di applicazione distribuita:



La configurazione di riferimento di una applicazione distribuita su TCP/IP e socket è il seguente:



Il paradigma Client-Server (C/S)

Il problema della connessione:

- Nel momento in cui una entità decide di instaurare una comunicazione con un'entità paritaria, come assicurarsi che quest'ultima sia disponibile?
 - La chiamata telefonica: chi desidera instaurare la comunicazione compone il **numero** del destinatario e attende durante il segnale di chiamata. Dall'altro lato **uno squillo** avverte di una chiamata in arrivo. Se si è disponibili alla comunicazione (si è in casa, si può ascoltare lo squillo e non si è sotto la doccia) **si alza la cornetta**. Lo squillo dal lato del chiamante **termina**. Da questo momento in poi la chiamata è instaurata e diviene simmetrica: chiunque può parlare quando vuole.
- È necessario che il chiamante conosca l'indirizzo del chiamato e che il chiamato sia in attesa di eventuali comunicazioni

Il chiamato è il **server**:

- Deve aver divulgato il proprio indirizzo
- Resta in attesa di chiamate
- In genere viene contattato per fornire un servizio

Il chiamante è il **client**:

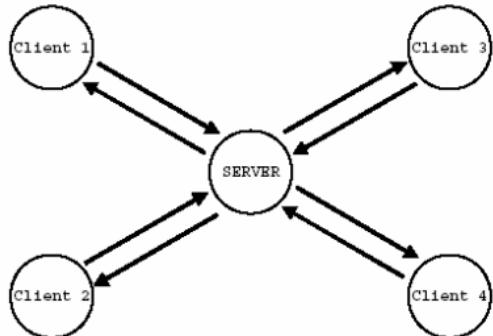
- Conosce l'indirizzo del server
- Prende l'iniziativa di comunicare
- Usufruisce dei servizi messi a disposizione dal server

Il concetto di **indirizzo**:

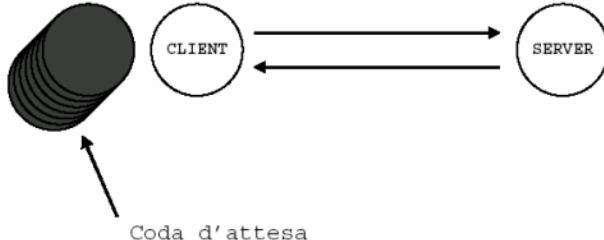
- Una comunicazione può essere identificata attraverso la quintupla:
(protocol, local-addr, local-process, foreign-addr, foreign-process)
- Una coppia (addr, process) identifica univocamente un terminale di comunicazione (end-point)
- Nel mondo IP, ad esempio:
 - local-addr e foreign-addr rappresentano indirizzi IP
 - local-process e foreign-process rappresentano numeri di porto

Server concorrente e iterativo

- Un server può ricevere chiamate anche da più client diversi.
- Ogni comunicazione richiederà un certo tempo prima di potersi considerare conclusa.
E se una chiamata arriva mentre il server è già impegnato in una comunicazione?
- Un server che accetti più comunicazioni contemporaneamente si definisce **concorrente**



- Un server che accetti una sola comunicazione alla volta è detto **iterativo**



- In questo ultimo caso una richiesta può essere servita solo quando la precedente si è già conclusa
- Questo è il paradigma applicato nel modello di comunicazione telefonica di base
- E l'avviso di chiamata?

I paradigmi di comunicazione

Il paradigma “Connection-Oriented”

- In una comunicazione dati Connection-Oriented, i due end-point dispongono di un canale di comunicazione che:
 - trasporta flussi
 - è affidabile
 - è dedicato
 - preserva l'ordine delle informazioni
- Il canale si comporta cioè come una sorta di “tubo”: tutto quello che viene inserito al suo interno, arriverà inalterato dall'altro lato e nello stesso ordine con cui è stato immesso
- Non è detto che vengano però mantenuti i limiti dei messaggi
- La comunicazione telefonica è più simile ad una comunicazione connection-oriented

Il paradigma "Datagram"

- In una comunicazione Datagram (anche detta connectionless), il canale
 - trasporta messaggi
 - non è affidabile
 - è condiviso
 - non preserva l'ordine delle informazioni
- Se si inviano dieci messaggi dall'altro lato essi possono anche arrivare mescolati tra di loro e tra i messaggi appartenenti ad altre comunicazioni. I limiti dei messaggi vengono comunque preservati.
- La posta ordinaria è un esempio di comunicazione a datagramma.

Connection-Oriented vs Datagram

	Connection-Oriented	Datagram
Principali vantaggi	<ul style="list-style-type: none"> • Affidabilità • Controllo di flusso 	<ul style="list-style-type: none"> • Basso overhead
Principali svantaggi	<ul style="list-style-type: none"> • Overhead per instaurare la connessione 	<ul style="list-style-type: none"> • Nessun controllo sulla consegna

Le socket che utilizzano i protocolli Internet sfruttano rispettivamente TCP (Transmission Control Protocol) e UDP (User Datagram Protocol) per implementare le due tipologie di comunicazione.

In entrambi i casi il protocollo di livello inferiore è IP (che è un protocollo datagram)

Progettazione di Client e Server

Naming/Binding

È l'operazione in cui associamo un indirizzo transport ad una socket già creata. In questo modo l'indirizzo diventa noto al sistema operativo ed altre socket sono in grado di stabilire una connessione.

Byte Stream e Datagram

- È possibile impostare il protocollo che verrà utilizzato per il trasferimento dei dati
- Nel caso delle socket abbiamo due opzioni:
 - Byte-stream: I dati vengono trasferiti come una sequenza ordinata ed affidabile di byte
 - Datagram: I dati vengono inviati come messaggi indipendenti ed inaffidabili

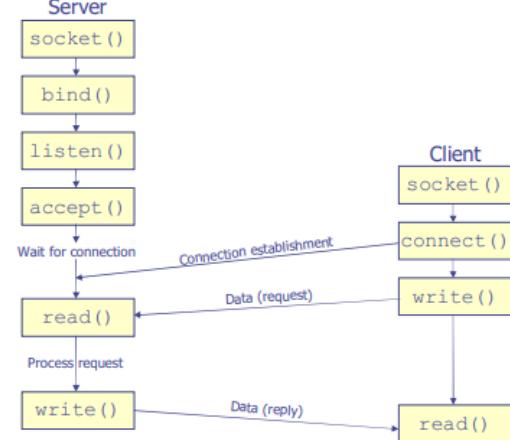
Progettazione di un Server TCP

- Creazione di un endpoint
 - Richiesta al sistema operativo
- Collegamento dell'endpoint ad una porta
 - Ascolto sulla porta (processo sospeso in attesa)
- Accettazione della richiesta di un client
- Letture e scritture sulla connessione
- Chiusura della connessione

Progettazione di un Client TCP

- Creazione di un endpoint
 - Richiesta al sistema operativo
- Creazione della connessione
 - Implementa open di TCP (3-way handshake)
- Lettura e scrittura sulla connessione
 - Analogo a operazioni su file in Unix
- Chiusura della connessione
 - Implementa close di TCP (4-way handshake)

Le chiamate per una comunicazione
Connection-Oriented



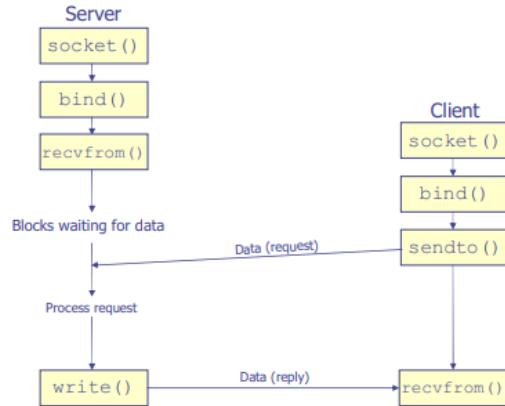
Progettazione di un Server UDP

- Creazione di un endpoint
 - Richiesta al sistema operativo
- Collegamento dell'endpoint ad una porta
 - open passiva in attesa di ricevere datagram
- Ricezione e invio di datagram
- Chiusura dell'endpoint

Progettazione di un Client UDP

- Creazione di un endpoint
 - Richiesta al sistema operativo
- Invio e ricezione di datagram
- Chiusura dell'endpoint

Le chiamate per una comunicazione Datagram



16. Internet e il routing gerarchico

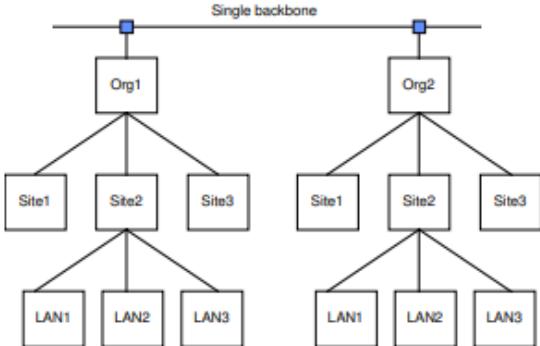
Il routing in Internet

Negli anni 80 l'architettura di Internet era molto semplice:

- c'era un'unica rete backbone
- ogni rete fisica era collegata alla backbone da un core router:
 - ogni core router conosceva le rotte per tutte le reti fisiche

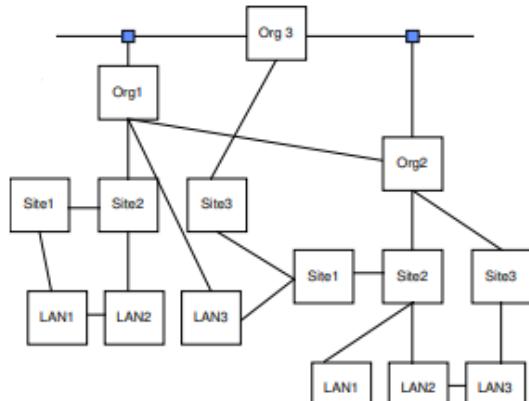
Non è accettabile che ci sia un unico proprietario per la backbone di tutta la rete poiché non tutte le reti fisiche possono essere collegate direttamente alla backbone.

Questa soluzione (non scalabile), infatti, rende impossibile mantenere tutti i router aggiornati al crescere del numero di core router.



Attualmente il routing Internet è un **routing gerarchico** composto da reti con *Perr Backbone*:

- prevedono l'esistenza di diverse dorsali:
 - gli amministratori delle reti backbone devono concordare una politica di routing per evitare la creazione di cicli
 - i core router delle diverse reti devono scambiarsi informazioni sulle rotte
- è strutturata come un insieme di **Autonomous System** (AS):
 - un AS è una collezione di reti amministrate da un'unica autorità
- ogni AS contiene un numero limitato di reti:
 - la gestione delle informazioni di routing all'interno dell'AS è più semplice



Il routing in presenza di Autonomous System

- Ogni AS è responsabile del routing all'interno delle sue reti: **routing interno**
- Gli AS devono scambiarsi informazioni di raggiungibilità: **routing esterno**
 - Garantisce la correttezza e la consistenza delle informazioni memorizzate nelle tabelle dei router
- Ogni AS deve essere identificato da un nome: AS number (16 bit)

Routing interno e routing esterno

Le tabelle di routing interne di un AS sono mantenute dall'**Interior Gateway Protocol (IGP)**:

- I messaggi IGP sono scambiati tra router appartenenti al medesimo AS
- Contengono solo informazioni sulle reti dell'AS
 - [RIP](#) (distance vector)
 - [OSPF](#) (link state)
 - [IGRP](#) (Interior Gateway Routing Protocol – Cisco)

Le tabelle di routing esterne di un AS sono mantenute dall'**Exterior Gateway Protocol (EGP)**:

- I messaggi EGP sono scambiati tra router designati dai rispettivi AS (border router)
- Contengono informazioni sulle rotte conosciute dai due AS
 - EGP (Exterior Gateway Protocol): ormai obsoleto
 - [BGP](#) (Border Gateway Protocol): approccio *path vector*

Inter-AS vs Intra-AS routing

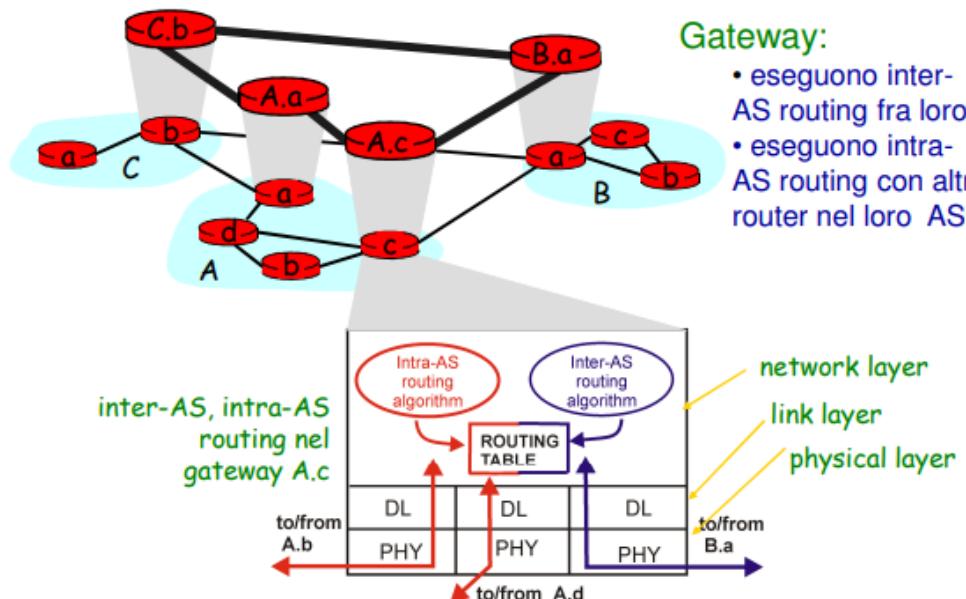
	Inter-AS	Intra-AS
Politica	Si concentra su aspetti politici (es.: quale provider scegliere o evitare)	Si applica in una singola organizzazione: all'interno dell'organizzazione, la politica di routing applicata è coerente
Dimensioni	<ul style="list-style-type: none"> • Si realizza un routing gerarchico • Si diminuisce il traffico per aggiornare le tabelle di routing 	
Prestazioni	Gli aspetti politico-amministrativi sono prevalenti	Si concentra sull'ottimizzazione delle prestazioni

I gateway router

I gateway router sono speciali router dell'AS, che:

- eseguono protocolli di routing intra-AS con altri router appartenenti all'AS
- sono, inoltre, responsabili del routing verso destinazioni esterne al proprio AS:
 - a tal fine, eseguono un protocollo di routing inter-AS con altri gateway router
- Su questi router sono pertanto attivi contemporaneamente sia protocolli di routing IGP (ad es. OSPF) e protocolli di routing EGP (ad es. BGP)

Instradamento gerarchico in Internet

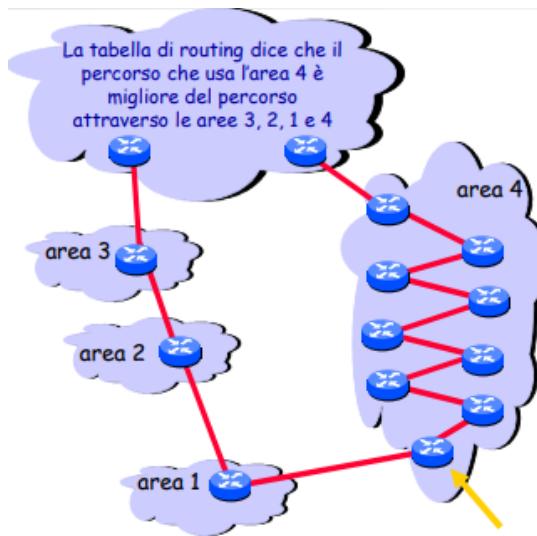


Routing Gerarchico vs Routing Piatto

Il routing gerarchico è usato per migliorare la scalabilità:

- Con 150 milioni di destinazioni:
 - Non è possibile memorizzare tutte le destinazioni nelle routing table
 - Lo scambio di tabelle di routing così grandi diminuisce notevolmente la banda utilizzata

... ma può condurre a scelte sub-ottime.



17. Protocolli IGP

Routing Information Protocol (RIP)

RIP è il più diffuso protocollo di IGP (non necessariamente il migliore), implementato su tutti i sistemi UNIX dal programma **routed**.

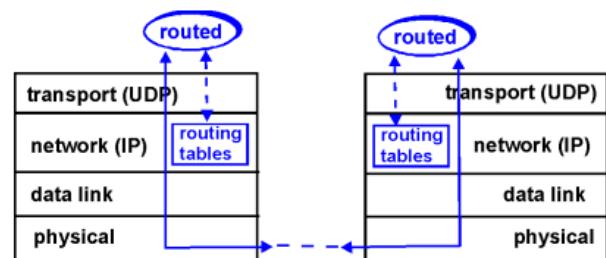
RIP è basato sulla trasmissione broadcast, e ciò lo rende adatto a reti broadcast (Ethernet) ma non a reti WAN. Implementa l'algoritmo **distance vector**.

Implementazione

- RIP non fa distinzione formale tra reti ed host singoli.
 - Le routing entry possono puntare ad un singolo host, anche se è conveniente usare reti che aggregano insiemi di indirizzi
- Divide le entità in attive e passive
 - Le entità passive possono solo ricevere messaggi (es. host)
 - Le entità attive possono anche spedire messaggi (es. i router)
- Le entità attive mandano un messaggio in broadcast ogni 30 secondi (messaggi *RIP response*)
 - Contiene la tabella di routing
 - L'unica metrica utilizzata è il numero di hop
- Ogni RIP response contiene fino a 25 reti destinazione
- Un host aggiorna una rotta solo se ne apprende una strettamente migliore
 - Ogni informazione ha un timeout di 180 secondi
- Se non c'è messaggio di advertisement (annuncio) dopo 180 secondi, il link è considerato morto
- I percorsi che attraversano quel vicino sono resi non validi; un nuovo annuncio è mandato ai vicini
- Vicini propagano l'informazione (se le loro tavole cambiano)
- La notizia dell'interruzione si propaga velocemente a tutta la rete

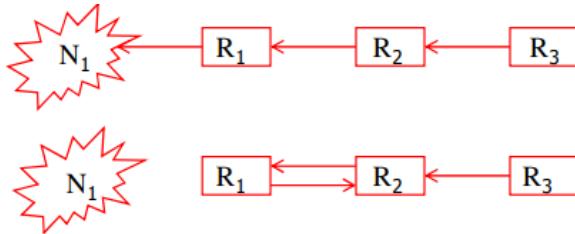
RIP e lo stack TCP/IP

- RIP è un protocollo di livello applicativo: le tavole di routing RIP sono elaborate da un processo a livello applicazione detto **routed**
- Piccoli messaggi regolari non necessitano del meccanismo windowing, di un meccanismo di handshaking o di ritrasmissioni.
- RIP usa il protocollo UDP. I pacchetti sono ricevuti e inviati usando il porto UDP 520



Analisi

- Il protocollo non individua esplicitamente i cicli
 - Assume che tutte le rotte pubblicate siano corrette
- Per prevenire inconsistenze fissa una distanza massima di routing.
 - Distanza massima: 15
 - 16 significa non connesso
- Gli aggiornamenti delle rotte si propagano lentamente
 - Slow convergence problem
- Il collegamento tra R_1 e N_1 cade



R_2 invia la sua tabella a R_1

- R_1 utilizza una nuova rotta lunga 3, passante per R_2

R_1 invia la sua tabella

- R_2 utilizza una nuova rotta lunga 4, passante per R_1

Si prosegue fino ad arrivare a 16

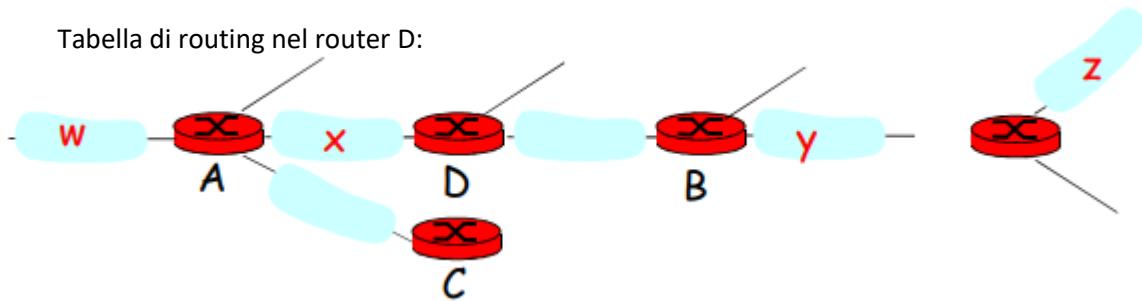
- Utilizza hop count come unica metrica
 - Il routing è indipendente dal traffico sulla rete
 - Non adatto a gestire la congestione
- Crede a tutte le informazioni che gli arrivano
 - Un router malizioso può indurre gli altri router a modificare le loro tabelle a suo vantaggio
 - Accettabile all'interno dello stesso AS
 - Inaccettabile tra AS distinti

Prevenire le instabilità

- Sono state studiate diverse tecniche per combattere la slow convergence
 - Nessuna risolve completamente il problema
- Split horizon (obbligatorio)
 - R_2 non invia ad R_1 le rotte che passano per R_1
 - Previene solo i loop tra due router
- Split horizon con poisoned reverse (opzionale)
 - R_2 dichiara ad R_1 a distanza infinita le reti che R_2 raggiunge attraverso R_1 stesso
 - Produce una più veloce eliminazione dei loop
 - Non elimina del tutto la possibilità dei loop che si creano tra nodi non adiacenti
- Triggered Updates
 - Appena un router aggiorna la propria tabella di routing, invia i distance vector aggiornati ai suoi vicini
- Hold down
 - R_2 dopo aver ricevuto il messaggio di R_1 ignora tutte le rotte per N_1 per un certo periodo di tempo (60 secondi)
 - I loop sono preservati per tutta la durata dell'hold time

Contenuto della routing table

Tabella di routing nel router D:



Destination Network	Next Router	Num. of hops to dest.
W	A	2
Y	B	2
Z	B	7
X	--	1
....

- **Address/Destination:** Indirizzo IP (IPv4) dell'host o della rete destinazione.
- **Router/Gateway:** Primo router lungo la route per la destinazione.
- **Interface:** La rete fisica che deve essere usata per raggiungere il prossimo router.
- **Metric:** Un numero che indica la distanza dalla destinazione. Questo numero è la somma dei costi dei link che bisogna attraversare per raggiungere la destinazione.
- **Timers:** Il tempo tra due update della stessa entry nella tabella.
- **Flags:** Ci sono diversi flag. Per esempio, possono indicare lo stato dei router direttamente collegati

Destination	Gateway	Flags	Ref	Use	Interface
127.0.0.1	127.0.0.1	UH	0	26492	lo0
192.168.2.	192.168.2.5	U	2	13	fa0
193.55.114.	193.55.114.6	U	3	58503	le0
192.168.3.	192.168.3.5	U	2	25	qaa0
224.0.0.0	193.55.114.6	U	3	0	le0
default	193.55.114.129	UG	0	143454	

Interior Gateway Routing Protocol (IGRP)

È un protocollo proprietario CISCO ed è il successore di RIP (come RIP usa distance vector).

Usa diverse metriche di costo (ritardo, banda, affidabilità, carico, etc...).

Usa TCP per scambiare aggiornamenti. Le tavole di routing sono scambiate solo quando si modificano costi.

Algoritmo di routing è DUAL (Distributed Updating Algorithm). È un algoritmo che garantisce dai cicli (free loop): dopo l'incremento di una distanza, la tavola di routing è congelata fino a quando tutti i nodi influenzati sanno del cambiamento.

Esiste anche un E-IGRP (enhanced-IGRP) il protocollo IGP più diffuso nei router CISCO.

Open Shortest Patch First (OSPF)

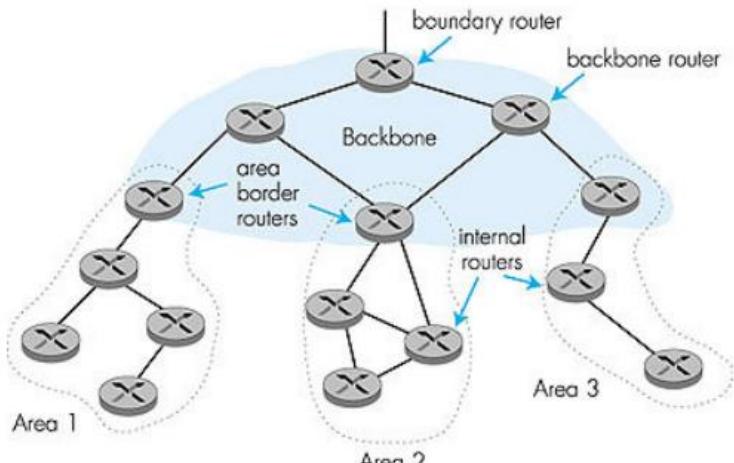
L'OSPF è il più diffuso protocollo IGP basato su tecnica link state:

- È pubblicamente disponibile
- Pacchetti LS sono disseminati ai vicini e ogni nodo conosce la topologia della rete

Supporta molte funzionalità (non sempre presenti nelle varie implementazioni): TOS routing, load balancing, subnet routing, CIDR, etc...

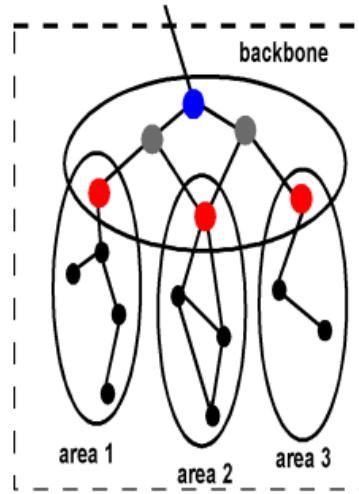
Tipologie di router

- **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



Gerarchie ed aree

- Gerarchia a due livelli: local area e backbone
- Gli Advertisement Link-state non lasciano le rispettive aree
- 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
- Backbone routers eseguono algoritmo OSPF solo per la backbone
- Boundary routers si connettono a altri Sistemi Autonomi (AS)



Analisi

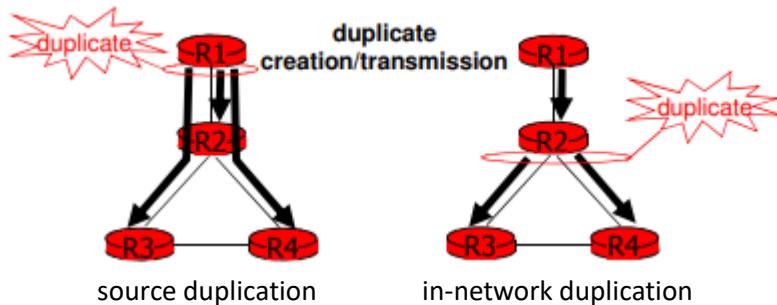
- OSPF non usa un protocollo di trasporto, ma incapsula i suoi messaggi direttamente in datagram IP con numero di protocollo 89
 - Scelta diversa da RIP e BGP
 - Meccanismi di affidabilità della comunicazione sono gestiti direttamente da OSPF
- **OSPF è gerarchico:** divisione di un AS in aree
 - Ogni area non deve conoscere la topologia delle altre aree
- Possibilità di definire una topologia virtuale della rete e di pubblicizzare rotte apprese da altri AS
- Sicurezza:
 - tutti i messaggi OSPF sono autenticati (per prevenire attacchi);
 - Autenticazione semplice (password in chiaro) o con MD5 (trasmesso in ogni pacchetto l'hash del pacchetto a cui è stata aggiunta una chiave segreta non trasmessa ma nota a tutti i router)
- Cammini multipli sono possibili (con lo stesso costo)
 - Nel protocollo RIP ne è possibile uno solo
 - Bilanciamento del carico tra percorsi multipli
- Supporto multicast integrato:
 - Multicast OSPF (MOSPF) usa lo stesso database di OSPF
- Ogni router manda periodicamente un messaggio HELLO ad ogni router direttamente collegato
 - Verifica che sia raggiungibile
- I router si scambiano informazioni sulla topologia della rete
- Ogni router periodicamente pubblica lo stato dei suoi link
 - Trasmissione broadcast

18. Multicasting

Tecniche di trasmissione broadcast

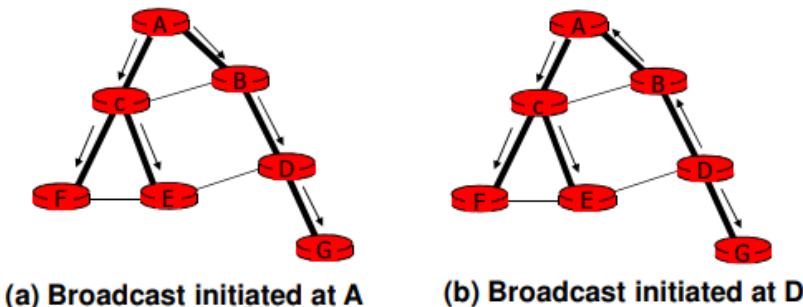
Broadcast Routing

- Consegna i pacchetti dalla sorgente a tutti gli altri nodi
- La duplicazione della fonte è inefficiente



In-network duplication

- 1) **Flooding:** quando un nodo riceve un pacchetto broadcast, manda copie a tutti i vicini (può causare cicli o broadcast storm)
- 2) **Controlled flooding:** un nodo trasmette il pacchetto solo se non lo ha già trasmesso in precedenza
 1. **Sequence number:** ogni nodo tiene traccia dei pacchetti che ha già trasmesso
 2. **Reverse path forwarding (RPF):** il pacchetto viene trasmesso solo se è arrivato dal percorso più breve tra il nodo e la sorgente
- 3) **Spanning tree:**
 - Nessun nodo riceve pacchetti ridondanti
 - Prima si costruisce uno spanning tree (albero di copertura)
 - Ogni nodo inoltra le copie solo lungo lo spanning tree



Trasmissione multicast in reti IP

Multicasting: definizione del problema

- Un numero sempre maggiore di applicazioni di rete richiede la spedizione di pacchetti da uno o più sender a un gruppo di receiver
 - la trasmissione dell'aggiornamento di un software dal suo sviluppatore agli utenti che richiedono l'aggiornamento
 - il trasferimento di audio, video e testi per lettura diretta a un gruppo distribuito di partecipanti alla lettura
 - una riunione aziendale o una teleconferenza condivisa tra molti partecipanti distribuiti
 - quotazioni in borsa, distribuzione di listini e cataloghi in tempo reale
 - training, teledidattica
 - sfruttare Internet per trasmissioni di tipo televisivo
- Per ciascuna di queste applicazioni, un'astrazione molto utile è la nozione di **multicast**: l'invio di un pacchetto da un sender a molti receiver con una singola operazione di spedizione.

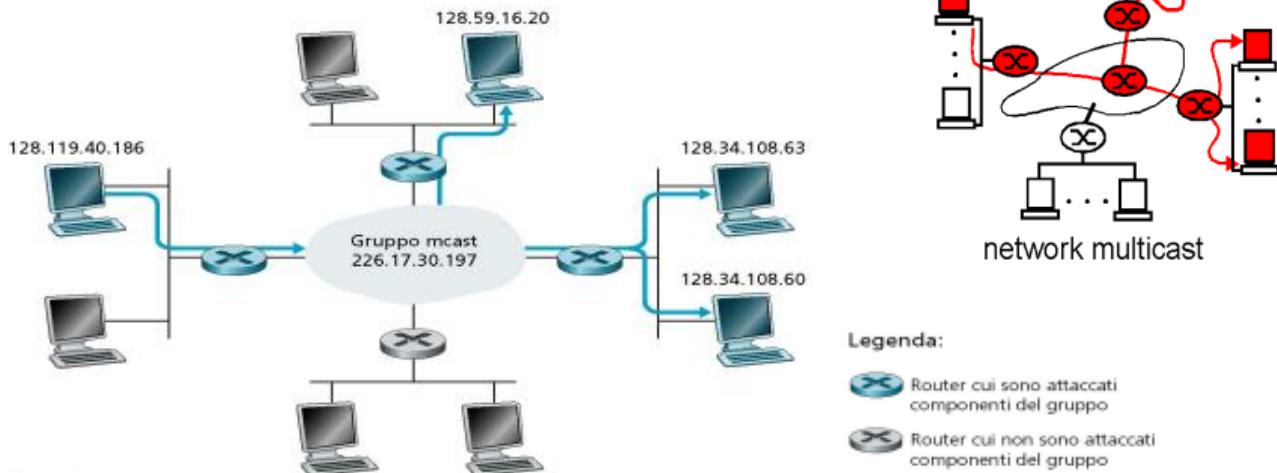
La trasmissione multicast

Come identificare i ricevitori di un datagramma multicast?

Come inviare un datagramma ai ricevitori, una volta identificati?

Address indirection (da “indirizzo per destinazione” ad “indirizzo per evento”): 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

Ad ogni gruppo è associato un **indirizzo multicast**, cioè un indirizzo IP di classe D. All’interno di tale classe esiste un certo numero di indirizzi che sono riservati dall’authority che gestisce Internet a dei gruppi permanenti: tali indirizzi sono detti “well-known”.



Le gestione dei gruppi

La gestione dei gruppi è di tipo dinamico:

- Un host può unirsi o abbandonare un gruppo in qualsiasi momento e può appartenere contemporaneamente a più gruppi
- Non è necessario appartenere ad un gruppo per poter inviare ad esso dei messaggi
- I membri del gruppo possono appartenere alla medesima rete o a reti fisiche differenti

Il multicast router

Si occupa dello smistamento dei datagrammi multicast, in maniera trasparente riguardo agli host interessati ad una determinata sessione di gruppo.

Funzionamento

- Ogni elaboratore 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

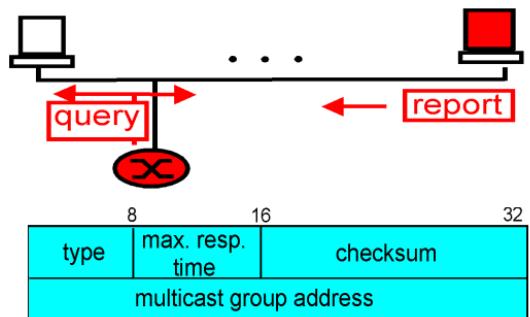
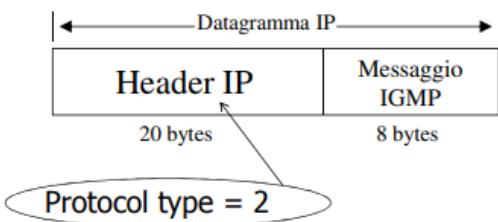
Il protocollo IGMP

Internet Group Management Protocol (IGMP) è un protocollo per il multicast in internet.

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.

- IGMP serve a garantire la trasmissione, tra host e multicast router ad essi direttamente collegati, dei messaggi relativi alla costituzione dei gruppi

- Pacchetti IGMP sono incapsulati in datagrammi IP con numero di protocollo 2



- Il raggio di interazione di tale protocollo è locale
 - cioè i messaggi IGMP sono scambiati tra end-system e router

Tipo di messaggio	Inviato da	Scopo
Membership query: generale	Router	Informarsi sui gruppi multicast cui gli host locali partecipano
Membership query: specifico	Router	Informarsi se uno o più host locali partecipano ad un determinato gruppo multicast
Membership report	Host	Informa il multicast router locale che l'host vuole unirsi ad (o fa parte di) un determinato gruppo multicast
Leave group	Host	Informa il multicast router locale che l'host vuole lasciare un determinato gruppo multicast

Il protocollo IGMP

Funzionalità

Le funzioni di IGMP sono relative a due fasi differenti:

- Fase 1**
 - Quando un host si unisce ad un nuovo gruppo, invia un messaggio IGMP ad un particolare indirizzo multicast
 - I multicast router appartenenti alla rete locale sulla quale tale host è situato, ricevono il messaggio e stabiliscono i meccanismi di routing propagando le informazioni concernenti il gruppo attraverso la rete interconnessa
- Fase 2**
 - Dovendo gestire i gruppi in maniera dinamica, i multicast router interrogano periodicamente (mediante opportune tecniche di "polling") gli host sulle varie reti locali, per aggiornare le informazioni relative alla composizione dei gruppi stessi

Implementazione

IGMP è stato accuratamente progettato per evitare di aggiungere carico eccessivo sulla rete:

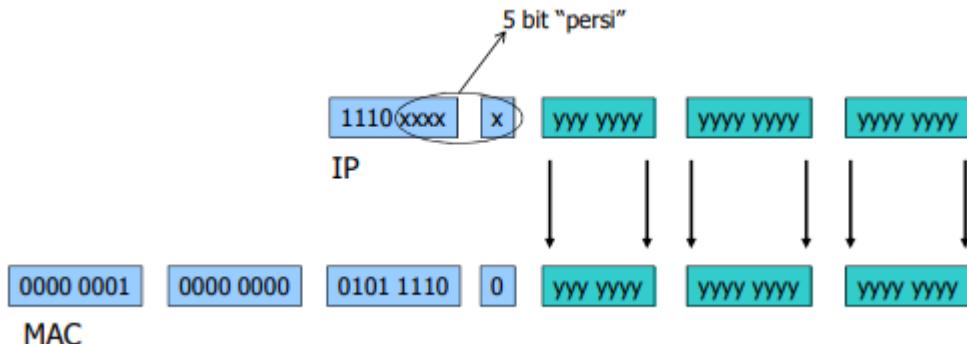
- Esso cerca, laddove possibile, di sfruttare al massimo i meccanismi hardware dei livelli sottostanti
- Il multicast router evita di trasmettere messaggi di richiesta individuali per ciascun gruppo, cercando, piuttosto, di raccogliere informazioni relative alla composizione dei singoli gruppi con una sola richiesta ("poll request")
- Host appartenenti a più di un gruppo non inviano risposte multiple in contemporanea, ma le diluiscono, in maniera random, su di un intervallo di 10 secondi (campo max resp. time)
- Ogni host ascolta le risposte inviate dagli altri e sopprime le proprie nel caso in cui risultino superflue

IP multicast

Distribuzione su LAN ethernet

- Per la trasmissione di datagram IP multicast su reti LAN ethernet, occorre mappare un indirizzo in classe D su un indirizzo MAC multicast
- Non è possibile in maniera univoca, dato il range degli indirizzi MAC riservati al multicast da 01: 00: 5e: 00: 00: 00 a 01: 00: 5e: 07: ff: ff
- L'OUI 01: 00: 5e è riservato al mapping degli indirizzi IP multicast
- L'indirizzo IP multicast è composto da 28 bit liberi, mentre l'indirizzo MAC ethernet ha 23 bit

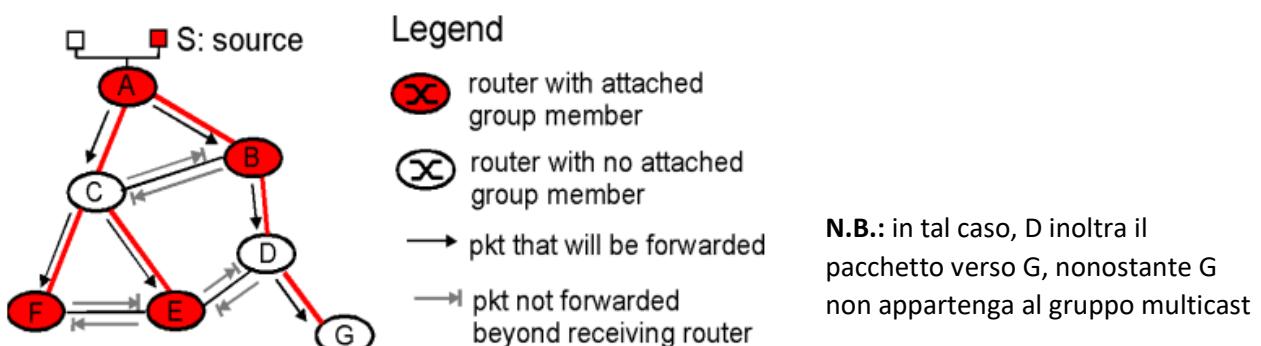
Il mapping tra indirizzi IP multicast ed indirizzi MAC multicast avviene come illustrato sotto:



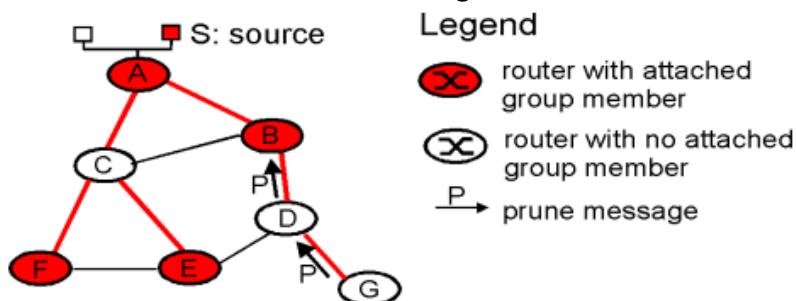
Reverse Path Forwarding (RPF)

Quando un router riceve un pacchetto multicast con un dato indirizzo sorgente, lo trasmette su tutte le interfacce di uscita solo se il pacchetto è giunto da un link appartenente al proprio shortest path verso il sender in questione

- Utilizza le informazioni di routing unicast già note
- Costruisce un nuovo albero ad ogni pacchetto
- Non tiene in considerazione i gruppi
 - questo lo rende più un algoritmo per la costruzione di un "broadcast tree"



Il Truncated Reverse Path Forwarding

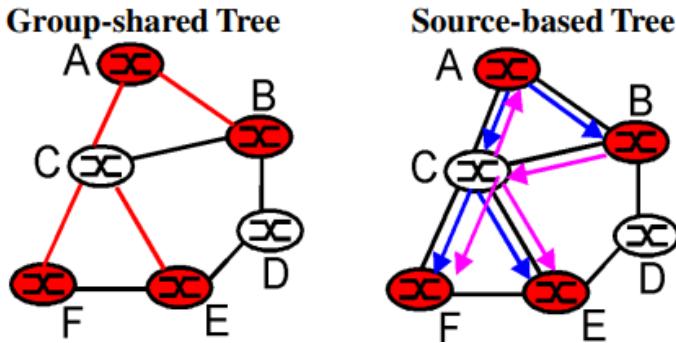


I router che ricevono pacchetti multicast pur non essendo connessi ad host appartenenti al gruppo destinazione, inviano un apposito messaggio di "pruning" verso il router a monte. Un router che riceve tale messaggio da tutti i suoi successori iterà il procedimento

Il routing multicast e l'albero di copertura

L'obiettivo dell'instradamento multicast è di trovare un albero di link che colleghi tutti i router cui sono attaccati gli host che appartengono al gruppo multicast. I pacchetti multicast saranno allora instradati attraverso questo albero dal sender a tutti gli host appartenenti all'albero multicast. Naturalmente, l'albero può contenere router che non hanno collegati host appartenenti al gruppo multicast.

Due possibili approcci:



- **Group-shared Tree:** Singolo albero condiviso dal gruppo per distribuire il traffico di tutti i mittenti
 - Ha il problema di trovare un albero a costo minimo
 - Esistono, tuttavia, algoritmi che approssimano la soluzione ottimale in maniera soddisfacente
- **Source-based Tree:** Specifico albero d'instradamento costruito per ciascun singolo mittente
 - L'unione dei percorsi minimi dalla sorgente a tutte le destinazioni

Protocolli per il multicast in Internet

Distance Vector Multicast Routing Protocol (DVMRP)

- Utilizza un algoritmo Distance Vector che permette ad ogni router di calcolare il link di uscita sul percorso minimo verso ciascuna possibile sorgente
- Implementa un algoritmo source-based tree con RPF, pruning e grafting (innesto)
 - Pruning: Se non ci sono iscrizioni al gruppo indicato viene mandato un messaggio prune ai router limitrofi
 - Tale messaggio non viene mandato sull'interfaccia corrispondente al percorso verso la sorgente, a meno che lo stesso messaggio non venga ricevuto su tutte le altre interfacce

Multicast Open Shortest Path First (MOSPF)

- Estende OSPF facendo sì che i router si scambino anche le informazioni relative all'appartenenza ai gruppi
- In tal modo, i router possono costruire alberi specifici per ogni sorgente, pre-potati, relativi ad ogni gruppo multicast

Core-Based Tree (CBT)

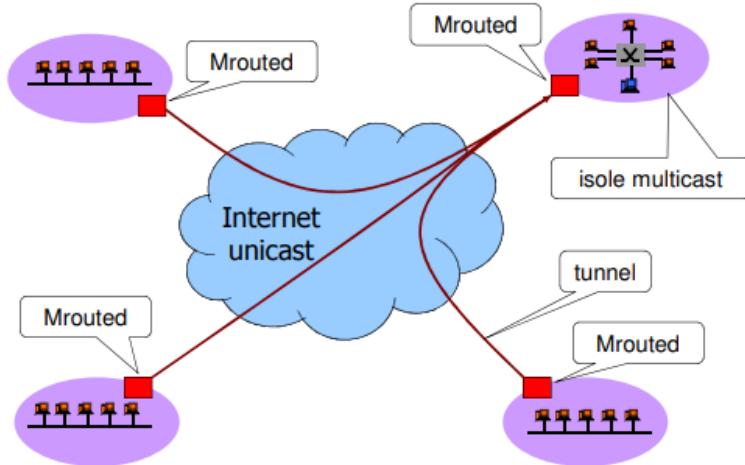
- Costruisce un albero "group-shared" bidirezionale, con un unico centro ("core")
- L'aggiunta di rami avviene mediante appositi messaggi di "join"
- La gestione dell'albero è affidata a meccanismi di refresh (soft-state)

Protocol Independent Multicast (PIM)

- È in grado di funzionare utilizzando un qualunque protocollo di routing unicast come supporto
- Può essere utilizzato per routing sia inter-domain che extra-domain. Se usato come extra-domain allora riesce ad interoperare con qualsiasi altro protocollo usato inter-domain
- Prevede due scenari alternativi:

- **dense mode**: progettato ed ottimizzato per reti densamente popolate da utenti di un certo gruppo (utilizza un approccio simile a quello adottato da DVMRP)
- **sparse mode**: progettato per reti scarsamente popolate (approccio simile a CBT)

La rete MBone

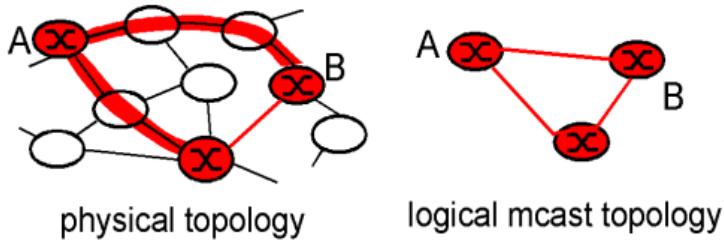


Multicast BackBone

- Un banco di prova semi-permanente per il multicast.
- Una rete virtuale che si appoggia su porzioni dell'Internet fisica.
- Composta da "isole" capaci di supportare il multicast IP (es: reti locali dotate di meccanismi hardware per il multicasting, quali Ethernet), collegate mediante link virtuali di tipo punto-punto chiamati "tunnel"

I tunnel multicast

- I pacchetti IP multicast vengono incapsulati prima di essere trasmessi attraverso i tunnel, in modo da apparire, all'esame dei router e delle sottoreti intermediarie, come normali datagrammi unicast
- Un multicast router 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
- Il router situato all'altro estremo del tunnel deve, alla ricezione del pacchetto, eliminare l'header unicast che fungeva da capsula e smistare il pacchetto multicast nel modo appropriato



Il programma Mrouted

Si occupa del routing multicast sui sistemi UNIX.

Il suo funzionamento è del tutto simile a quello del demone routed nel caso unicast:

- opera in stretta collaborazione col sistema operativo per installare le informazioni relative all'instradamento dei pacchetti multicast
- Può essere utilizzato solo con una versione speciale di UNIX, conosciuta come "multicast kernel", contenente:
 - una tabella apposita per il routing dei pacchetti multicast
 - il codice necessario per il loro smistamento

Propagazione delle informazioni relative al routing:

- mrouted utilizza DVMRP per propagare tali informazioni
- un calcolatore che supporta mrouted è anche in grado di costruire la “multicast routing table”:
 - Impiego di algoritmi quali il Truncated Reverse Path Broadcast (TRPB)

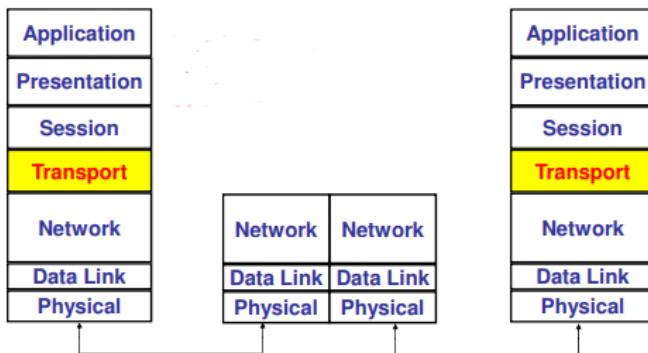
Creazione dei tunnel multicast:

- non tutti i router di Internet sono capaci di smistare i datagrammi di tipo multicast
 - mrouted si occupa, quindi, della configurazione di un tunnel tra due router, attraverso elementi intermediari che non partecipano al multicasting

19. Il Livello Trasporto

Introduzione

I protocolli di livello trasporto sono presenti solo negli end-system

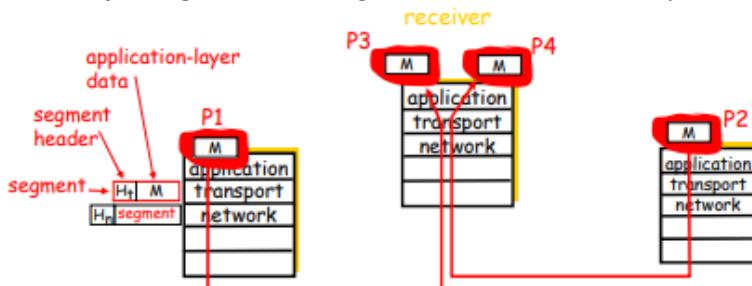


Servizi del livello trasporto

- Offre un canale di **comunicazione logica** tra applicazioni attive su differenti host
- Differenze tra livello trasporto e livello rete:
 - **Network-layer**: trasferimento dati tra end-system
 - **Transport layer**: trasferimento dati tra processi. Naturalmente necessita dei servizi offerti dal livello rete
- Poiché il livello rete offre un servizio inaffidabile, è il livello trasporto che deve rimediare ad aumentare l'efficienza e l'affidabilità.
 - In particolare: controllo degli errori, sequenza ordinata, controllo di flusso, controllo di congestione
- I protocolli di Livello Trasporto sono realizzati al di sopra del Livello Rete, quindi è necessario gestire:
 - apertura della connessione (setup)
 - memorizzazione dei pacchetti all'interno della rete
 - un numero elevato di connessioni. Multiplexing e Demultiplexing

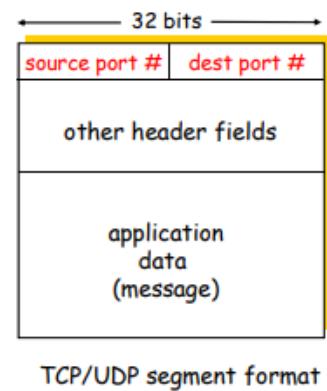
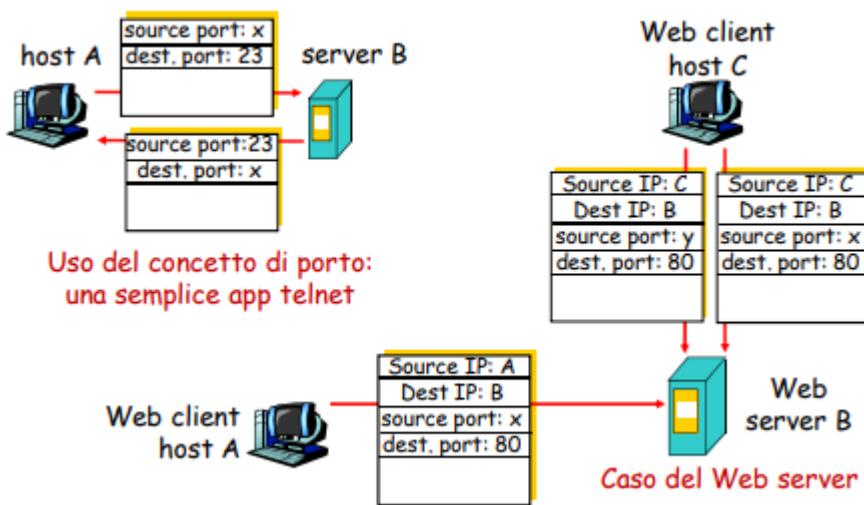
Multiplexing e Demultiplexing

- **Demultiplexing**: inoltrare i segmenti ricevuti al corretto processo cui i dati sono destinati



- Segment: dati che sono scambiati tra processi a livello trasporto

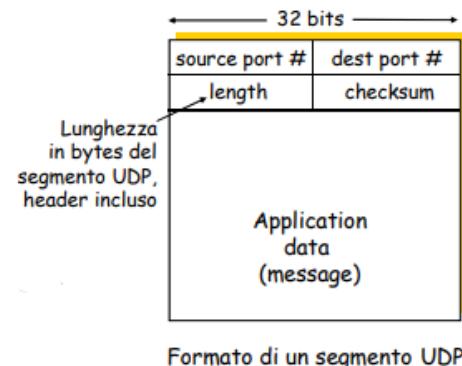
- Multiplexing:** Raccogliere i dati provenienti dalle applicazioni, imbustare i dati con un header appropriato (per il demultiplexing)
- multiplexing/demultiplexing: Realizzato attraverso la coppia (indirizzo IP, numero di porto)
 - source, dest port # è presente in ogni segmento
 - numeri di porto “well-known” per applicazioni particolari



Il protocollo UDP

User Datagram Protocol

- Aggiunge poco ad IP:
 - Servizio **“best effort”**: i pacchetti UDP possono subire perdite; giungere a destinazione in ritardo, o non arrivare affatto; giungere a destinazione non ordinati.
 - Servizio **connectionless**: non è prevista una fase di inizializzazione; ogni segmento UDP è inviato indipendentemente dagli altri
- Perché è stato introdotto UDP?
 - non è necessaria la fase di inizializzazione (setup) che introduce delay
 - semplice: sender e receiver non devono conservare informazioni di stato
 - intestazione di dimensioni contenute: basso overhead
 - controllo della congestione assente: le applicazioni possono inviare alla velocità desiderata (utile per alcune applicazioni ma rischioso per la rete)
- Ampiamente usato per applicazioni multimediali:
 - tolleranti alle perdite di pacchetti
 - sensibili ai ritardi
- Domanda: si può rendere UDP affidabile?
 - Gestione degli errori
 - Conferma di avvenuta ricezione



Checksum UDP

Obiettivo: rilevare “errori” (bit alterati) nel segmento trasmesso

Mittente:

- Tratta il contenuto del segmento come una sequenza di interi espressi su 16 bit
- checksum:** complemento ad 1 della somma (in complemento ad 1) dei numeri da 16 bit che costituiscono il segmento UDP
 - Incluso uno pseudoheader che include gli indirizzi IP sorgente e destinazione

- Il mittente pone il valore della checksum nel campo checksum del segmento UDP

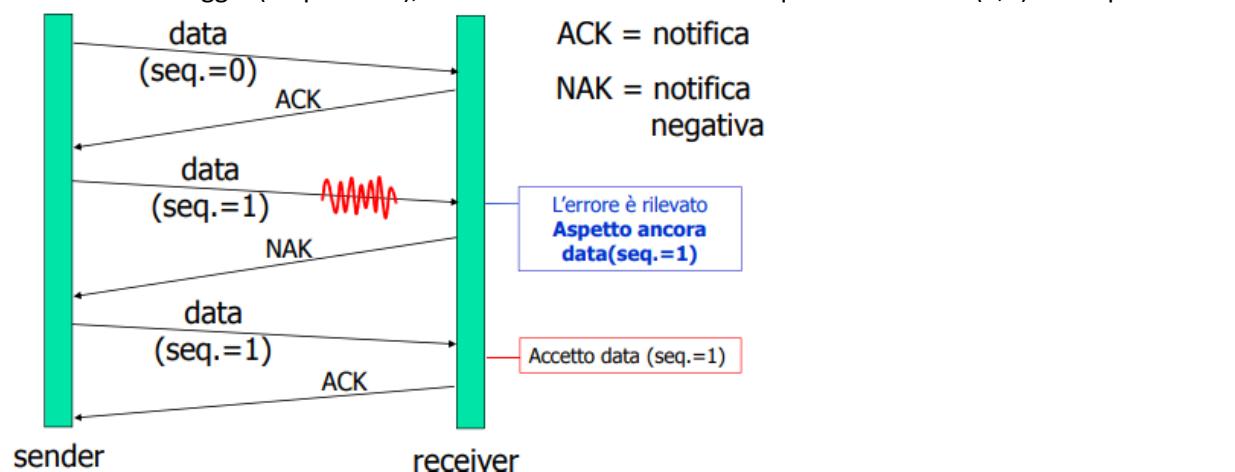
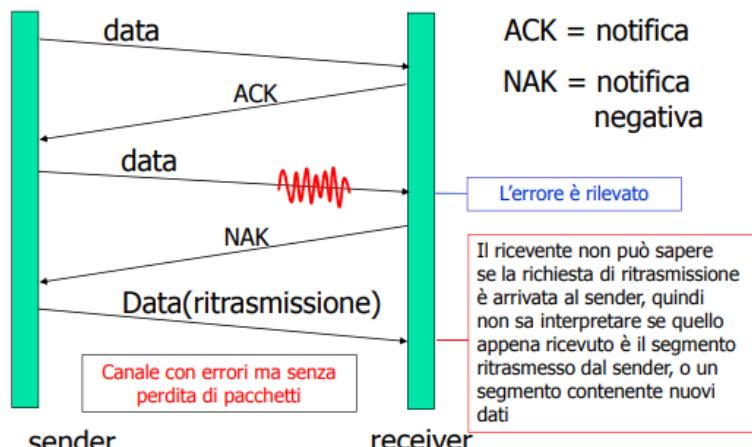
Ricevente:

- calcola la somma in complemento ad 1 dei campi del segmento ricevuto compresa la checksum
- Risultato composta da tutti 1?
 - No: errore
 - Si: nessun errore rilevato (non vuol dire che non vi siano stati errori)

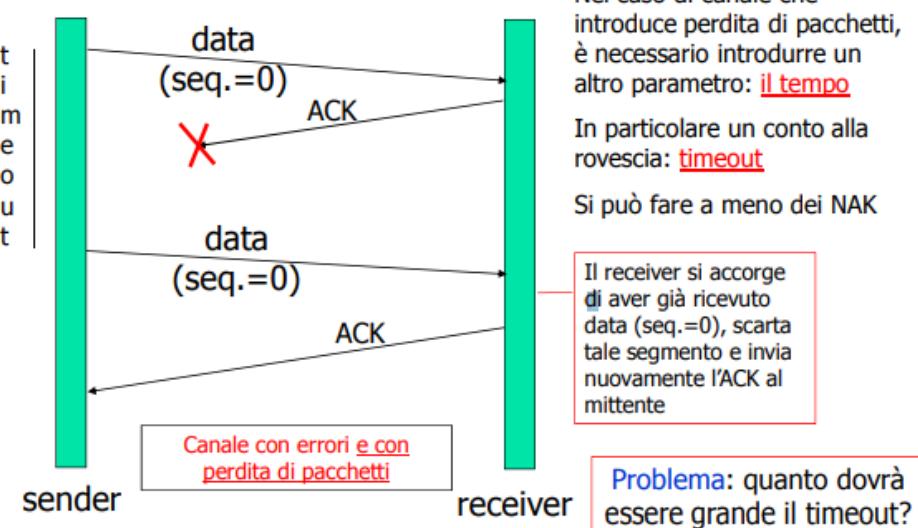
Tecniche di trasmissione affidabile dei dati

Realizzare una trasmissione affidabile

- Se il livello rete è inaffidabile:
 - Presenza di errori
 - Perdita di pacchetti
 - Ordine dei pacchetti non garantito
 - Duplicazione di pacchetti
 - Inoltre, bisogna tenere in considerazione: le risorse del computer ricevente (controllo di flusso) e le risorse della rete (controllo di congestione)
- Soluzioni:
 - Rete che presenta errori di trasmissione. In questo caso il ricevente deve effettuare un rilevamento degli errori e correggerli (prima soluzione) oppure notificare al mittente la richiesta di ritrasmissione (seconda soluzione)
 - 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 che il ricevente non è in grado di interpretare, occorre inserire nell'header del segmento da inviare un' ulteriore informazione: **numero di sequenza**
- Nel caso di protocolli che inviano un messaggio e quindi aspettano un riscontro prima di ritrasmettere un nuovo messaggio (stop & wait), è sufficiente un numero di sequenza su un bit (0,1). Esempio:



- Un protocollo senza NAK:



- Stessa funzionalità del precedente, utilizzando soltanto gli ACK
- Al posto del NAK, il destinatario invia un ACK per l'ultimo pacchetto ricevuto correttamente. Il destinatario deve includere esplicitamente il numero di sequenza del pacchetto con l'ACK
- Un ACK duplicato presso il mittente determina la stessa azione del NAK: ritrasmettere il pacchetto corrente

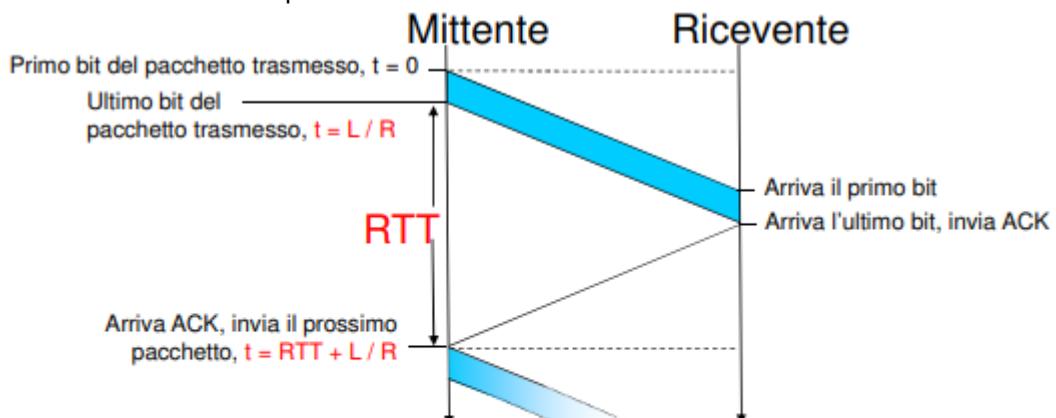
Aumentare l'efficienza

In alternativa al semplice Stop&Wait c'è il **Pipelining**: il mittente invia pacchetti prima di ricevere il riscontro dei precedenti.

- Occorre aumentare l'intervallo dei num. Sequenza
- Aggiungere buffer nel sender e/o receiver
- Due alternative per il pipelining: [go-Back-N](#), [selective repeat](#)
- Performance:

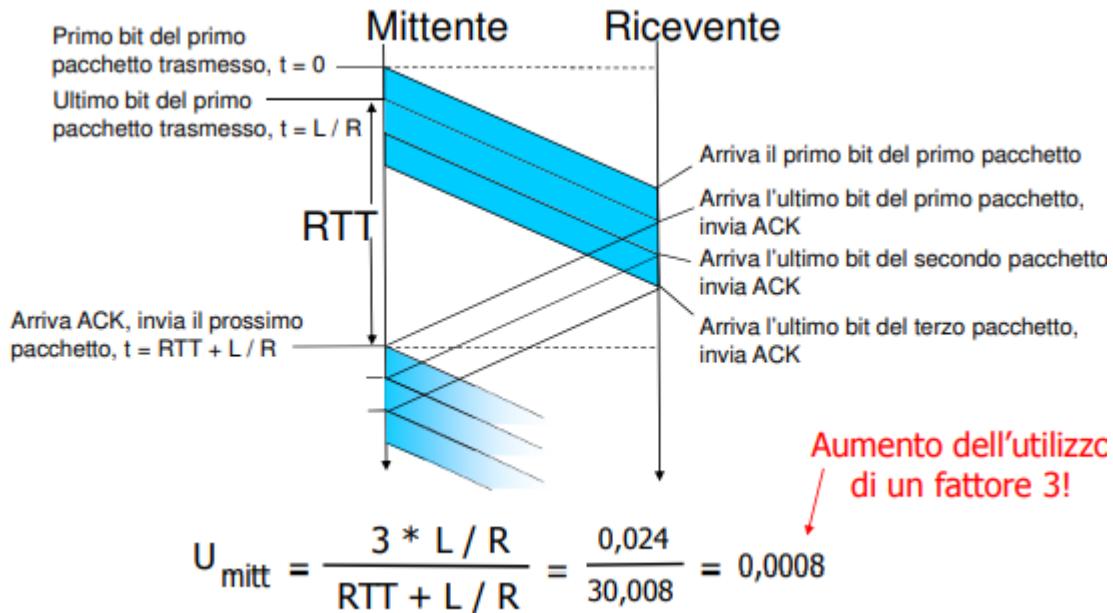
$$T_{\text{transmit}} = \frac{L \text{ (packet length in bits)}}{R \text{ (transmission rate, bps)}} = \frac{8\text{kb/pkt}}{10^{*9} \text{ b/sec}} = 8 \text{ microsec}$$

- Funzionamento con stop-and-wait:



$$U_{\text{mitt}} = \frac{L / R}{RTT + L / R} = \frac{0,008}{30,008} = 0,00027 = 0,027 \%$$

- Pipelining: aumento dell'utilizzo



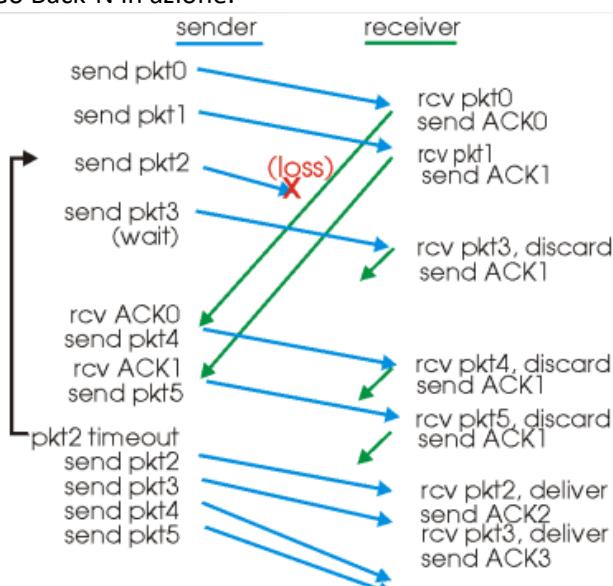
Go Back-N

Sender:

- Nell'header del segmento k-bit per il numero sequenza
- Una finestra di max N pacchetti senza riscontro
- ACK numerati

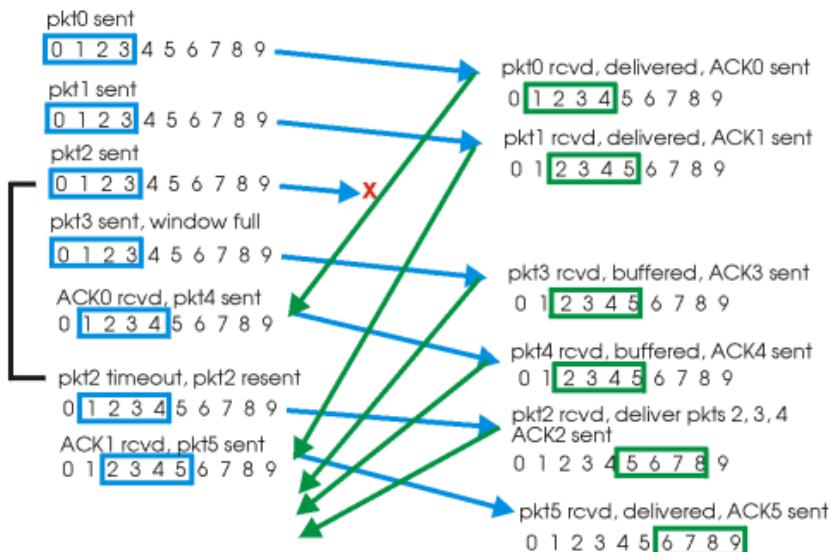


- ACK cumulativo: ricevere $ACK(n)$ significa che tutti i pacchetti precedenti l' n -esimo sono stati ricevuti correttamente
- Un timer per ogni pacchetto trasmesso e non riscontrato
- timeout(n): ritrasmetti pacchetto n e tutti i pacchetti che seguono n
- Go Back-N in azione:



Ripetizione selettiva

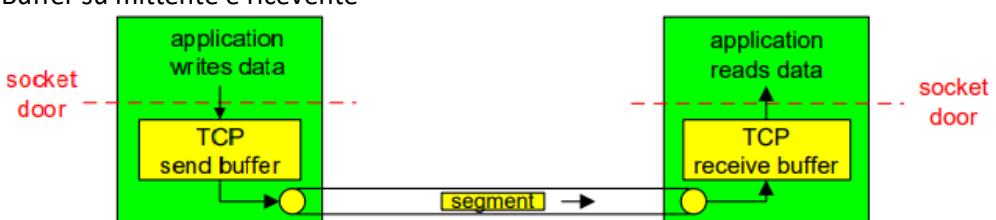
- Il ricevente invia riscontri specifici per tutti i pacchetti ricevuti correttamente
 - buffer dei pacchetti, se necessario, per eventuali consegne in sequenza al livello superiore
- Il mittente ritrasmette soltanto i pacchetti per i quali non ha ricevuto un ACK
 - timer del mittente per ogni pacchetto non riscontrato
- Finestra del mittente
 - N numeri di sequenza consecutivi
 - limita ancora i numeri di sequenza dei pacchetti inviati non riscontrati
- Selective Repeat in azione:



20. Livello Trasporto: Il protocollo TCP

Transmission Control Protocol

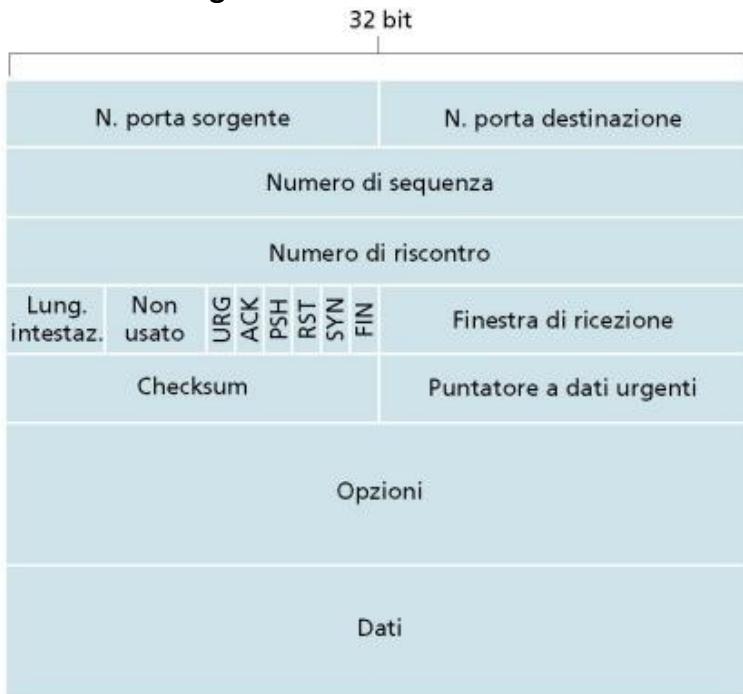
- End-to-end e punto-punto
 - Una connessione unica tra mittente e ricevente
- Senza errori, sequenza ordinata
- Pipelined
 - Controllo di flusso e di congestione impostano la TCP window
- Buffer su mittente e ricevente



- Full duplex data

- Flusso di dati bi-direzionale all'interno della stessa connessione
- MSS: Maximum Segment Size
- Connection-oriented
 - Handshaking (scambio di msg di controllo a tre vie) prepara mittente e ricevente prima della comunicazione
- Controllo di flusso
 - Il mittente non invia più di quanto il ricevente non possa accettare

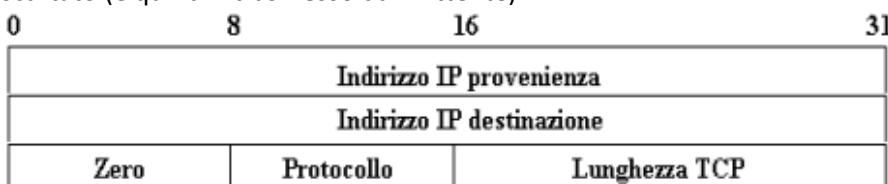
Struttura del segmento TCP



TCP: PDU

- **HLEN**
 - 4 bit, contiene un numero intero che indica la lunghezza dell'intestazione TCP del datagramma in parole da 32 bit. Questa informazione è necessaria perché il campo **opzioni** è di lunghezza variabile. Generalmente il campo opzioni è vuoto e la lunghezza consueta è 20 byte.
- **Porta** (provenienza/destinazione)
 - Contengono i numeri di porta di protocollo TCP che identificano gli applicativi alle estremità della connessione (mux/demux).
- **Numero di Sequenza**
 - Contiene il numero sequenziale del byte successivo a quello correttamente ricevuto dalla destinazione. Tale campo è valido solo nei segmenti di riscontro, o nei segmenti utilizzanti la tecnica trasmissiva **Piggy-backing**, e fa riferimento allo stream di dati che fluisce nella direzione opposta a tale segmento.
 - Nel calcolo del numero di riscontro, oltre a considerare i bytes contenuti nel payload TCP, bisogna considerare anche la presenza di bytes SYN e FIN inviati, che valgono come un singolo byte.
- **Bit di Codice**: per identificare il tipo di informazione contenuta nel segmento vengono impiegati i 6 bit di codice
 - ACK: il campo riscontro è valido
 - RST: Effettua il reset della connessione
 - SYN: Sincronizza i numeri di sequenza

- FIN: Il trasmettitore ha raggiunto la fine del suo stream di byte
 - PSH: Questo segmento richiede una “spinta” (a destinazione)
 - URG: il campo puntatore urgente è valido
- **Finestra**
 - Numero intero senza segno di 16 bit che specifica la dimensione del buffer che il TCP ha a disposizione per immagazzinare dati in arrivo. È utilizzato per la gestione dinamica della dimensione della finestra scorrevole.
- **Puntatore urgente**
 - Il TCP permette la trasmissione di dati informativi ad alta priorità. Questi devono essere trasmessi il prima possibile, indipendentemente dalla loro posizione nello stream. Questo campo, se valido, conterrà un puntatore alla posizione, nello stream, dei dati non urgenti (ultimo byte dei dati urgenti).
- **Checksum:** campo di 16 bit contenente un valore intero utilizzato dal TCP della macchina host di destinazione, per verificare l'integrità dei dati e la correttezza dell'intestazione
 - Questa informazione è di essenziale importanza perché il protocollo IP non prevede nessun controllo di errore sulla parte dati del frame
 - Per il calcolo del valore checksum il TCP ha bisogno di aggiungere una **pseudointestazione** al datagramma, per effettuare così un controllo anche sugli indirizzi IP di destinazione e provenienza
- La Pseudointestazione viene creata e posta in testa al datagramma TCP
- Viene inserito in essa un ulteriore byte di zeri per raggiungere un multiplo di 16 bit
- Successivamente viene calcolata la checksum su tutto il messaggio così formato, viene scartata la pseudointestazione e passato il datagramma al livello IP
- In fase di ricezione, il livello TCP ricrea la pseudointestazione interagendo con l'IP sottostante, calcola la checksum e verifica la correttezza del messaggio ricevuto. In caso di errore il datagramma verrà scartato (e quindi ritrasmesso dal mittente)



- Principali **opzioni** di TCP
 - Maximum TCP payload: durante la fase di connessione, ciascun end-point annuncia la massima dimensione di payload che desidera accettare; la minima tra le due dimensioni annunciate viene selezionata per la trasmissione
 - Window Scale: per negoziare un fattore di scala per la finestra; utile per connessioni a larga banda e/o elevato ritardo di trasmissione
 - Selective Repeat: nel caso in cui un segmento corrotto sia stato seguito da segmenti corretti, introduce i NAK (Not Acknowledge), per permettere al receiver di richiedere la ritrasmissione di quello specifico segmento; è un'alternativa al “go back n”, che prevede la ritrasmissione di tutti i segmenti

TCP: Caratteristiche

Riscontro e ritrasmissione

- Consiste nella ritrasmissione di un segmento se non è giunta conferma entro un tempo massimo (**time-out**)

Time-Out

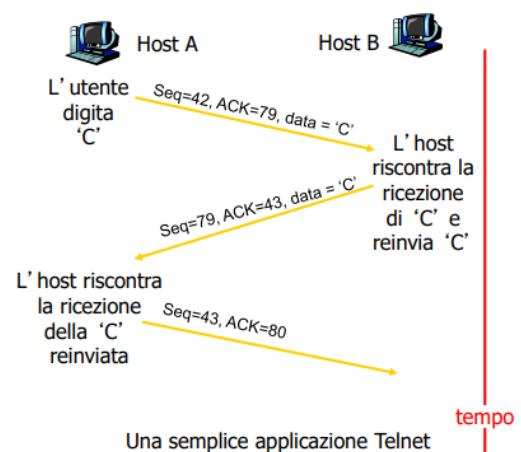
- Al momento della trasmissione di un segmento, il TCP attiva un timer

Numeri di sequenza e numeri di riscontro

- TCP vede i dati come un flusso di byte non strutturati, ma ordinati
- I numeri di sequenza riflettono questa visione: **il numero di sequenza per un segmento** è quindi il numero nel flusso di byte del primo byte nel segmento
 - Flusso lungo 500.000 byte, MSS uguale a 1000 byte, primo byte numerato con 0
 - TCP costruisce 500 segmenti, con numeri di sequenza 0, 1000, 2000...
- Per i numeri di riscontro, vista la natura full-duplex della connessione TCP, si ha che ad esempio
 - A invia e contemporaneamente riceve da B
 - I segmenti B -> A, contengono un numero di sequenza relativo ai dati B -> A
 - **Il numero di riscontro** che A scrive nei propri segmenti è il numero di sequenza del byte successivo che A attende da B (e lo può mandare anche inviando dati a B)
- TCP invia riscontri cumulativi

Numeri di sequenza e ACK di TCP

- Numeri di sequenza
 - “numero” del primo byte del segmento nel flusso di byte
- ACK
 - Numero di sequenza del prossimo byte atteso dall’altro lato
 - ACK cumulativo
- Come gestisce il destinatario i segmenti fuori sequenza?
 - La specifica TCP non lo dice, dipende dall’implementazione



Caratteristiche del riscontro cumulativo

Nel datagramma di riscontro la destinazione comunica quale byte dello stream si aspetta di ricevere successivamente

- I riscontri specificano sempre il numero sequenziale del primo byte non ancora ricevuto
 - Esempio: in uno stream di 1000 byte segmentato in blocchi di 100 byte, partendo da 0, il primo riscontro conterrà il numero sequenziale 100
- Con questo metodo di riscontro cumulativo si ha il vantaggio che la perdita di un riscontro non blocca la trasmissione se confermato dal riscontro successivo

Round Trip Time e Timeout

Il valore del timeout deve essere impostato ad un valore maggiore del RTT (Round Trip Time).

Nota bene: RTT varia nel tempo.

- Se timeout è scelto troppo breve (timeout prematuro)
 - Ritrasmissioni ridondanti
 - Scarsa efficienza
- Se timeout è scelto troppo lungo
 - Scarsa efficienza nella gestione delle ritrasmissioni

Calcolo del timeout

$$\text{EstimatedRTT} = (1 - \alpha) * \text{EstimatedRTT} + \alpha * \text{SampleRTT}$$

- Una media esponenziale pesata dei campioni
 - L’influenza di un singolo campione sul valore della stima decresce in maniera esponenziale, e si dà più importanza a campioni recenti.
 - Valori tipici per α : 0.125

Valore del timeout

EstimatedRTT più un “margine di sicurezza” proporzionale alla variabilità della stima effettuata

- variazione significativa di EstimatedRTT -> margine più ampio
 - Timeout = EstimatedRTT + 4 * DevRTT,
dove $DevRTT = (1 - \beta) * DevRTT + \beta * |SampleRTT - EstimatedRTT|$
 - Valore raccomandato per β : 0.25

Trasmissione affidabile

Per un trasferimento dati affidabile:

- TCP crea un servizio di trasferimento dati affidabile sul servizio inaffidabile di IP
- Pipeline dei segmenti
- ACK cumulativi
- TCP usa un solo timer di ritrasmissione
- Le ritrasmissioni sono avviate da
 - Eventi di timeout
 - ACK duplicati
- Inizialmente consideriamo un mittente TCP semplificato
 - Ignoriamo gli ACK duplicati
 - Ignoriamo il controllo di flusso e il controllo di congestione

Eventi del mittente

Dati ricevuti dall'applicazione:

- Crea un segmento con il numero di sequenza
- Il numero di sequenza è il numero del primo byte del segmento nel flusso di byte
- Avvia il timer, se non è già in funzione (pensate al timer come se fosse associato al più vecchio segmento non riscontrato)
- Intervallo di scadenza TimeOutInterval

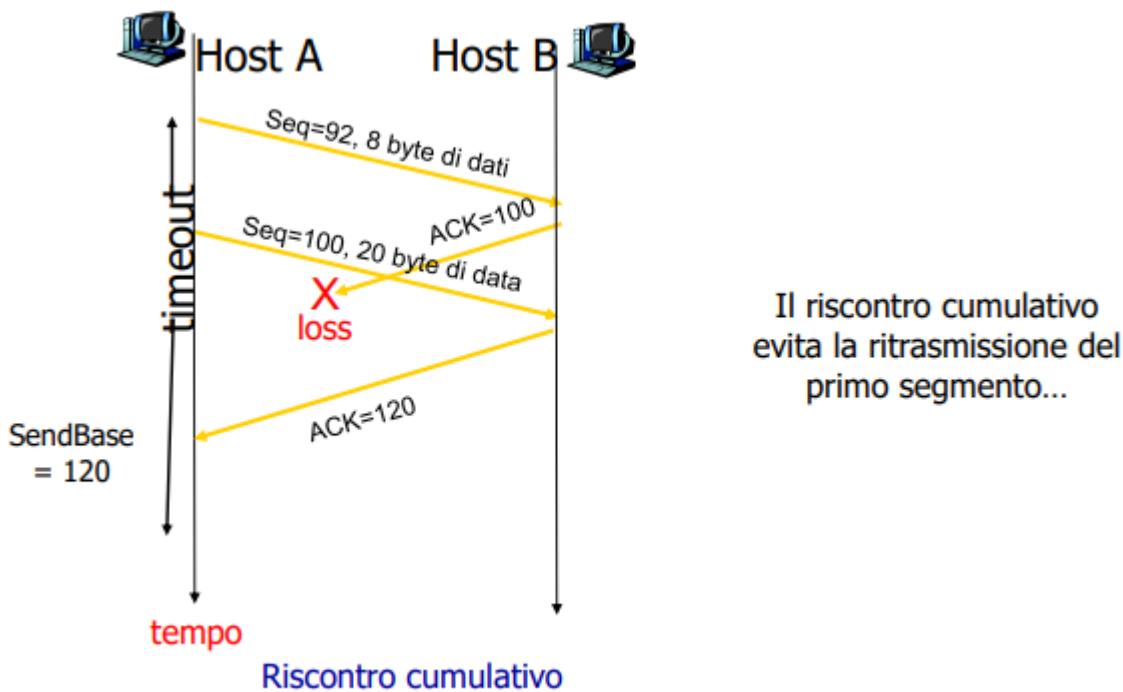
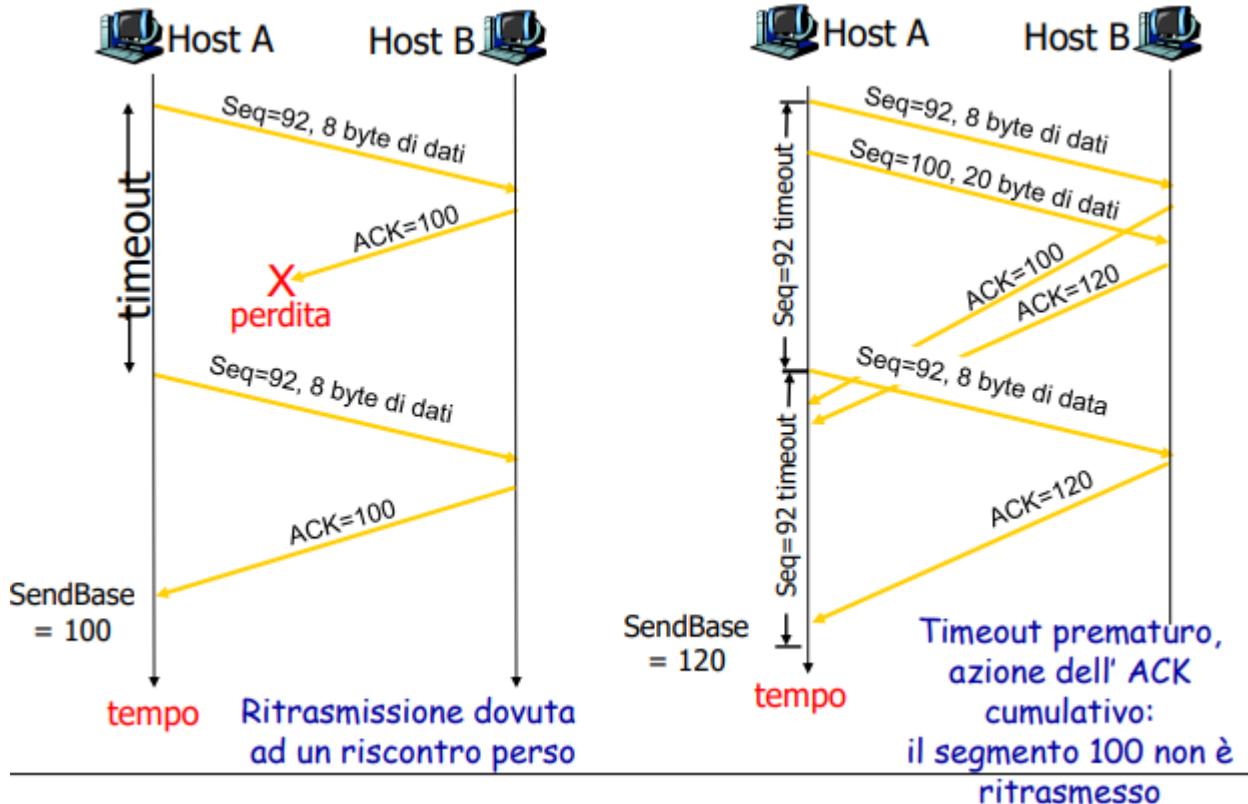
Timeout:

- Ritrasmette il segmento che ha causato il timeout
- Riavvia il timer

ACK ricevuti

- Se riscontra segmenti precedentemente non riscontrati
 - aggiorna ciò che è stato completamente riscontrato
 - avvia il timer se ci sono altri segmenti da completare

Alcuni scenari di rilievo



Modifiche tipiche del TCP

Raddoppio dell'intervallo di timeout

- Allo scadere di un timeout si imposta il prossimo intervallo al doppio del valore precedente (invece di usare la stima di RTT)
 - Crescita esponenziale degli intervalli dopo ogni ritrasmissione
- Quando il timer viene riavviato (ricezione di un ACK o di nuovi dati dall'applicazione)

- L'intervallo di timeout viene nuovamente configurato in funzione dei valori più recenti di EstimatedRTT e DevRTT

Fornisce una forma limitata di controllo della congestione

- Il mittente, nel caso supponga una situazione di congestione (perdita di un segmento), ritrasmette ad intervalli sempre più lunghi

Ritrasmissione veloce: ACK duplicati consentono di rilevare la perdita di un pacchetto prima del timeout

- un receiver che rileva un “buco” nei segmenti ricevuti (ricezione di un segmento con numero di sequenza maggiore di quello atteso):
 - invia un nuovo riscontro per l'ultimo byte di dati che ha ricevuto correttamente
- poiché il mittente spesso manda molti segmenti contigui, se uno di tali segmenti si perde, ci saranno molti ACK duplicati contigui:
 - un sender che riceve tre ACK duplicati per gli stessi dati assume che il segmento successivo a quello riscontrato tre volte è andato perso ed effettua, quindi, una ritrasmissione prima della scadenza del timeout

Generazione di ACK

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

TCP Connection Management

Mittente e Ricevente concordano l'apertura della connessione prima di inviare i dati.

Impostare le variabili del TCP:

- Numeri di sequenza
- Allocare i buffer, impostare un valore iniziale della RcvWindow

Three way handshake

Passo 1: client invia segmento di controllo TCP SYN al server

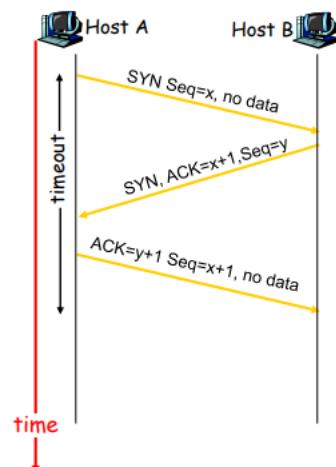
- Specifica il primo seq#

Passo 2: server riceve SYN, risponde con segmento di controllo SYN/ACK

- ACK del SYN ricevuto
- Alloca buffer
- Specifica il primo seq# per la connessione server→client

Passo 3: client riceve SYN/ACK, invia ACK al server

- Connessione instaurata



Chiusura della connessione

Il client decide di chiudere la connessione...

Passo 1: client invia un segmento di controllo TCP con FIN alto al server

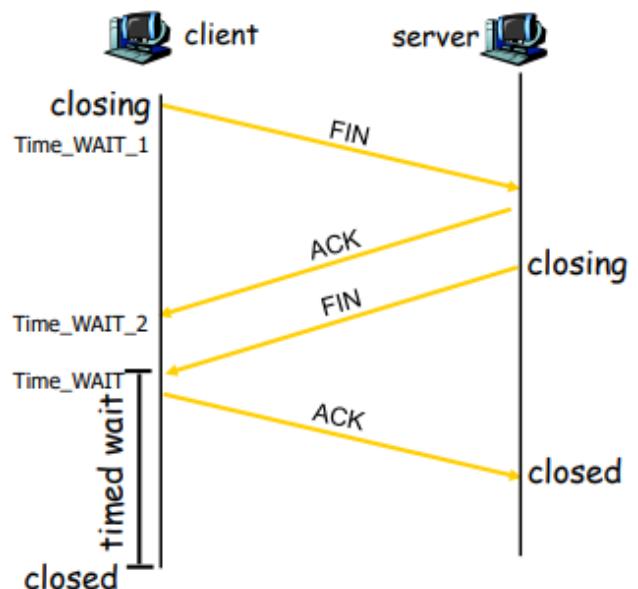
Passo 2: server riceve FIN, risponde con ACK. Quindi chiude la connessione inviando FIN al client

Passo 3: client riceve FIN, risponde con un ACK

- Attende in uno stato TIMED_WAIT (nel caso in cui l'ultimo ACK vada perso, e riceva un ulteriore FIN dal server)

Passo 4: server, riceve ACK. Chiude la connessione

N.B.: una piccola variante al Passo 2: il server invia ACK e FIN contemporaneamente all'interno dello stesso segmento



Sequenza tipica degli stati nel client e nel server

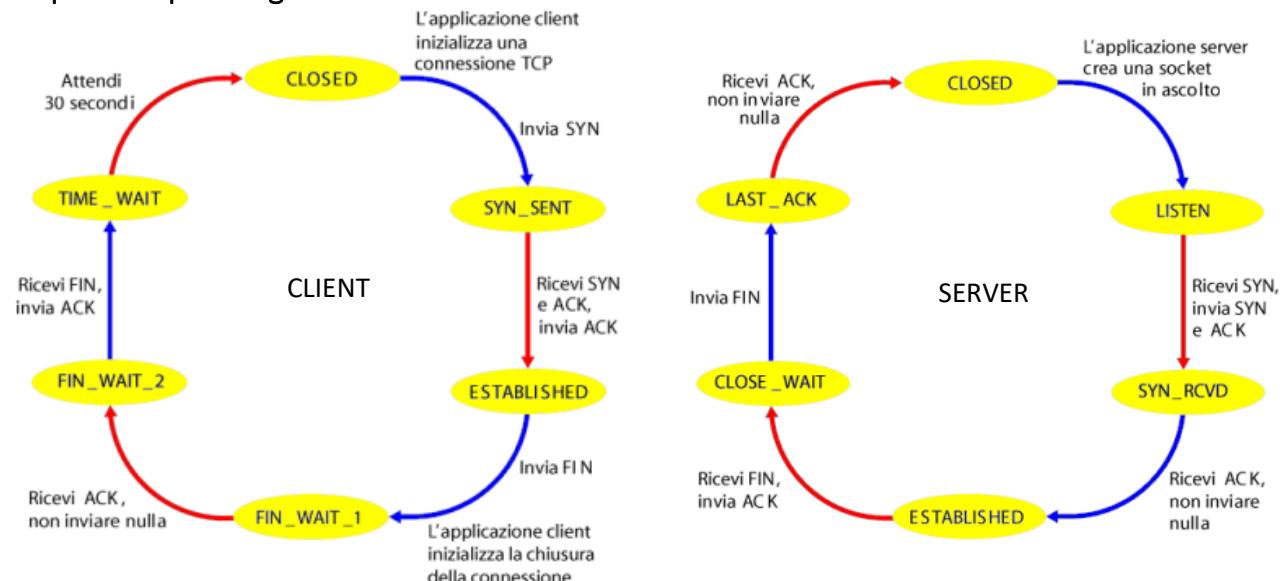
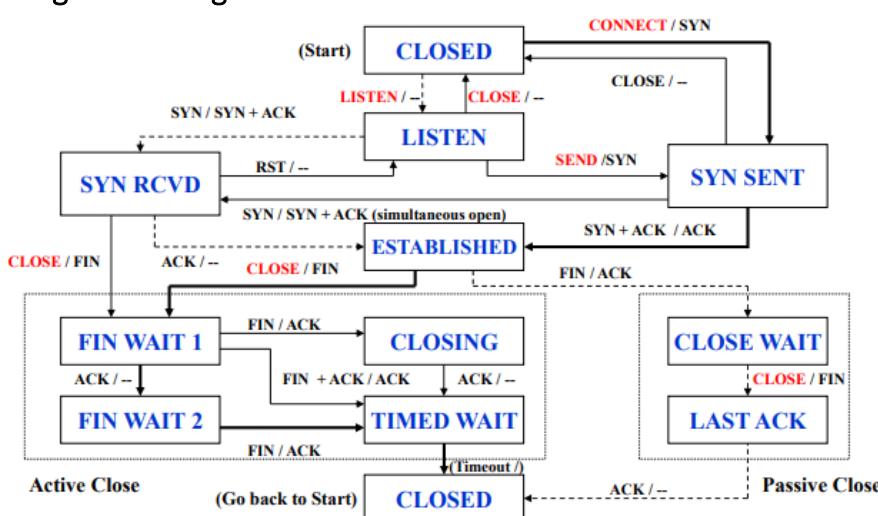
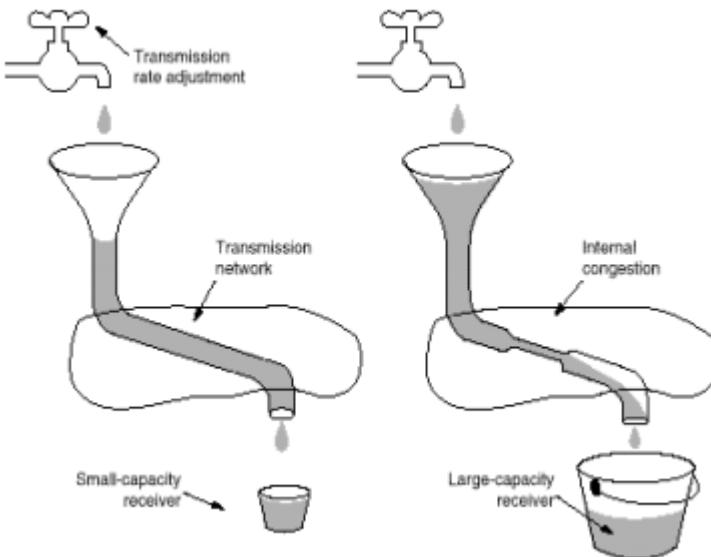


Diagramma degli stati del TCP



Controllo di flusso e controllo della congestione



Come gestire entrambi i tipi di controllo?



Receiver window: dipende dalla dimensione del buffer di ricezione

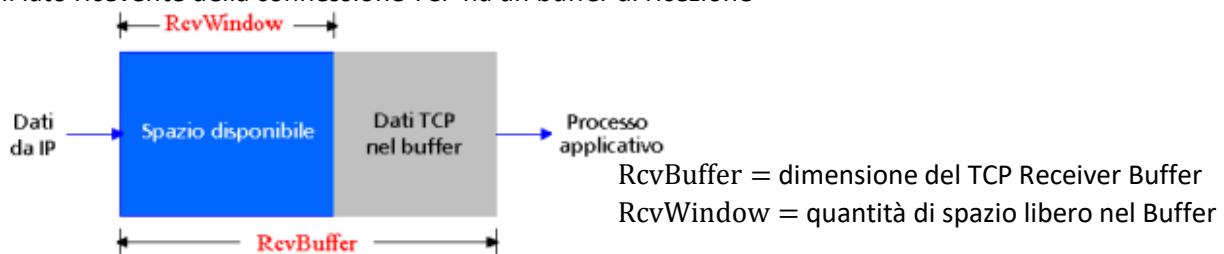
Congestion window: basata su una stima della capacità della rete



I byte trasmessi corrispondono alla dimensione della finestra più piccola

Controllo di flusso

- Il lato ricevente della connessione TCP ha un buffer di ricezione



- Il processo applicativo potrebbe essere rallentato dalla lettura nel buffer
- Servizio di corrispondenza delle velocità: la frequenza d'invio deve corrispondere alla frequenza di lettura dell'applicazione ricevente (Supponiamo che il destinatario TCP scarti i segmenti fuori sequenza)
- Flow control:** Il mittente non dovrà sovraccaricare il ricevente inviando dati ad una velocità troppo elevata.
 - Ricevente:** comunica dinamicamente al mittente la dimensione corrente del buffer; campo RcvWindow nel segmento TCP
$$\text{RcvWindow} = \text{RcvBuffer} - [\text{LastByteRcvd} - \text{LastByteRead}]$$
 - Mittente:** conserva i dati già trasmessi ma non riscontrati e limita tale quantità all'ultima RcvWindow ricevuta.
$$\text{LastByteSent} - \text{LastByteAcked} \leq \text{RcvWindow}$$

L'algoritmo di Nagle

Per applicazioni che inviano dati un byte alla volta (es: TELNET).

- invia il primo byte e bufferizza il resto finché non giunge l'ACK
- in seguito
 - invia, in un unico segmento, tutti i caratteri bufferizzati
 - ricomincia a bufferizzare finché non giunge l'ACK per ognuno di essi
- Elimina la sindrome della Silly Window al trasmettitore

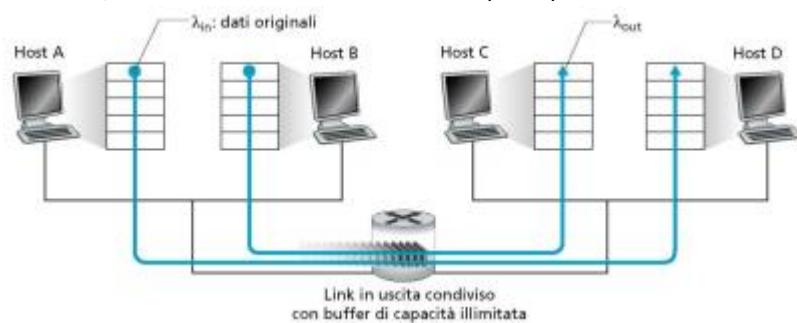
La sindrome della Silly Window

- Sender:** invia blocchi grandi
- Receiver:** legge un byte alla volta
- Soluzione di Clark:** impedisce al receiver di aggiornare la finestra un byte alla volta

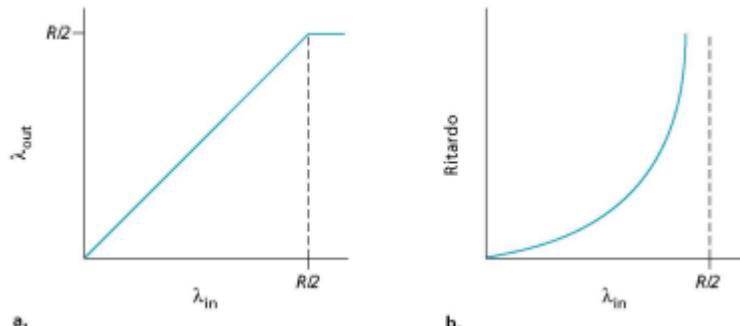
- Il ricevitore indica una finestra nulla finché il buffer di ricezione non si è svuotato per metà o per una porzione uguale a MSS

Controllo della congestione

- La congestione nella rete è tecnicamente dovuta a
 - un numero elevato di sorgenti di traffico
 - sorgenti di traffico che inviano troppi dati
 - traffico inviato a una frequenza troppo elevata
- In presenza di questi fenomeni, singoli o concomitanti, la rete è **sovraffollata**. Causando degli effetti:
 - Perdita di pacchetti: buffer overflow nei router
 - Ritardi nell'inoltro dei pacchetti: accodamenti nei buffer dei router
 - Scarso utilizzo delle risorse di rete
- Esempio sugli effetti della congestione:
 - 2 mittenti, 2 riceventi 1 router con buffer (coda) infinito: non ci sono ritrasmissioni



- I ritardi aumentano all'avvicinarsi del limite di capacità del canale



- Non si può superare il max throughput

Tecniche di controllo della congestione

Approccio end-to-end

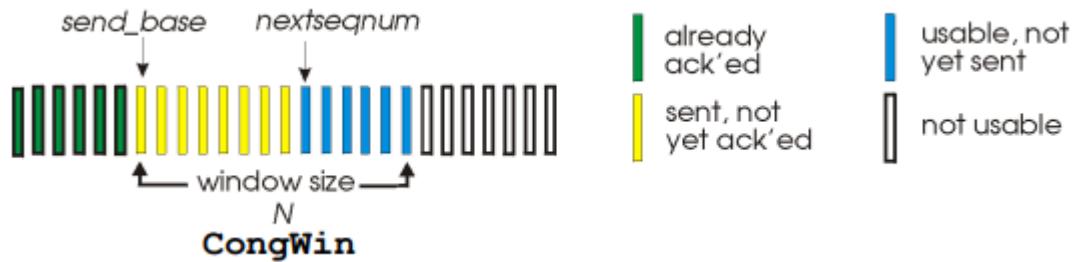
- Nessuna segnalazione esplicita dalla rete
- A partire dall'osservazione di ritardi e perdite di pacchetti gli end-system deducono uno stato di congestione nella rete
- Approccio utilizzato da TCP

Approccio in base a segnalazione della rete

- I router forniscono informazioni circa lo stato della rete agli end-system
 - l'invio di un singolo bit indica lo stato di congestione
 - in alternativa, il sender è informato circa la massima frequenza alla quale può trasmettere

Controllo della congestione in TCP

In TCP si usa un controllo end-to-end: nessun feedback dalla rete. La frequenza di trasmissione variabile dipende dalla cosiddetta finestra di congestione (CongWin):



Considerando controllo di flusso e controllo di congestione insieme, si ha, dunque:

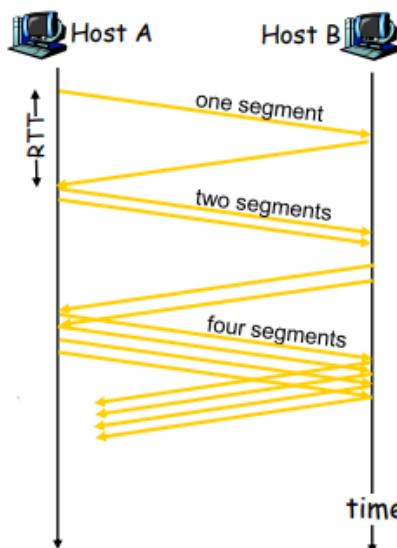
$$\text{LastByteSent} - \text{LastByteAcked} \leq \min\{\text{RcvWindow}, \text{CongWin}\}$$

Idea di base: Si procede per tentativi, per stabilire quanto si può trasmettere

- obiettivo
 - trasmettere alla massima velocità possibile (CongWin quanto più grande possibile) senza perdite
- approccio utilizzato
 - incrementare CongWin finché non si verifica la perdita di un segmento (interpretata come il sopraggiungere dello stato di congestione)
 - In seguito alla perdita di un segmento: decrementare CongWin e ricominciare daccapo

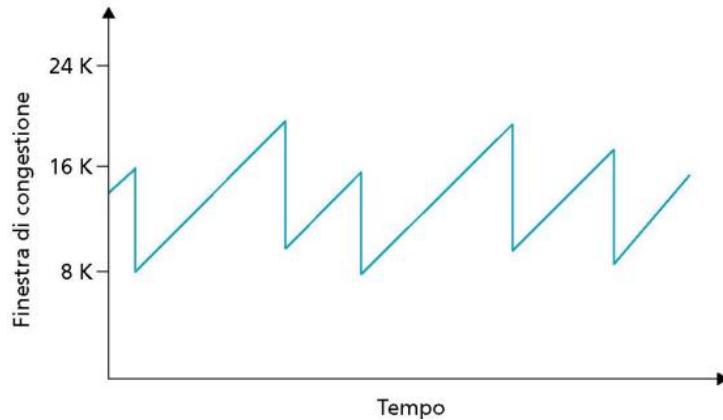
Fasi del controllo della congestione:

- **Slow Start:** partenza lenta



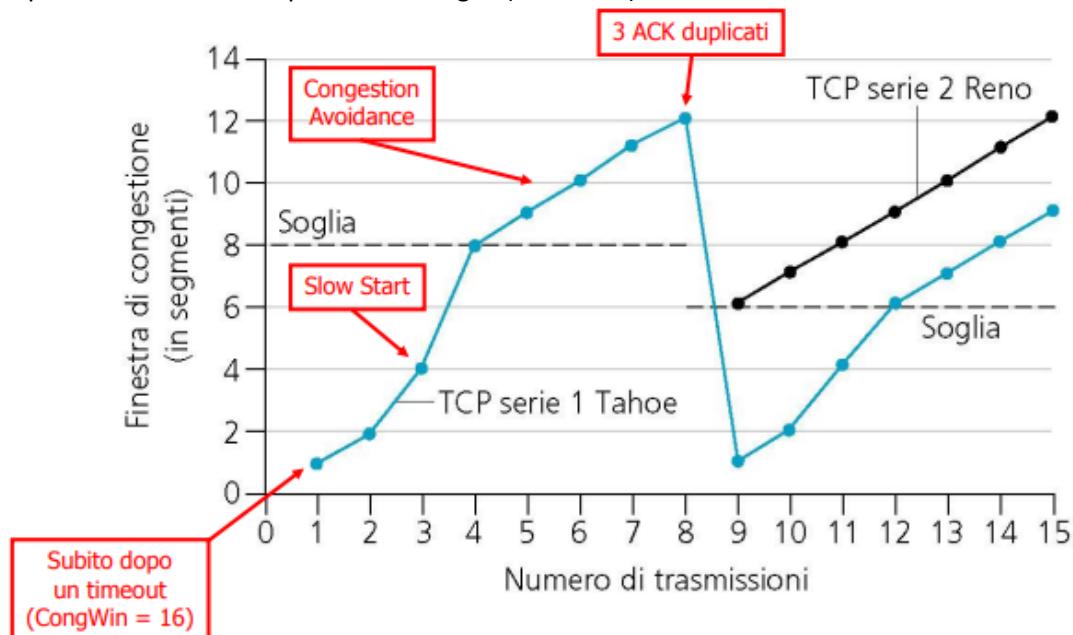
- Crescita esponenziale della dimensione della finestra (ogni RTT)
- Evento di perdita: timeout; tre ACK duplicati consecutivi
- **Congestion Avoidance:** Additive Increase, Multiplicative Decrease (AIMD)
 - **Incremento additivo:** una volta raggiunta la soglia ci si avvicina con cautela al valore della banda disponibile tra le due estremità della connessione. Il meccanismo adottato è di incrementare CongWin di una quantità pari a $\text{MSS} * (\text{MSS}/\text{CongWin})$
 - **Decremento moltiplicativo:** Al sopraggiungere della congestione (scadenza di un timeout o ricezione di tre ACK duplicati consecutivi). La finestra di congestione viene dimezzata

- AIMD ha un andamento a “dente di sega”



Controllo della congestione in Internet

È presente un ulteriore parametro: soglia (threshold)



Ricapitolando...

- Finestra di congestione sotto la soglia
 - Slow start
 - Crescita esponenziale della finestra
- Finestra di congestione sopra la soglia
 - Prevenzione della congestione
 - Crescita lineare della finestra
- Evento di perdita dedotto da ACK duplicato 3 volte
 - Soglia posta alla metà del valore attuale della finestra
 - **TCP Reno:** Finestra posta pari alla soglia
 - **TCP Tahoe:** Finestra posta pari ad un segmento (MSS -- Maximum Segment Size)
- Evento di perdita dedotto da timeout
 - Soglia posta alla metà del valore attuale della finestra
 - Finestra posta pari ad un segmento (MSS -- Maximum Segment Size)

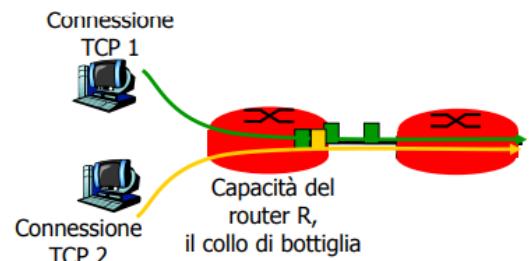
TCP Reno: "fast recovery"

TCP Reno elimina la fase di partenza lenta dopo un evento di perdita dedotto dalla ricezione di tre ACK duplicati

- tale evento indica che, nonostante si sia perso un pacchetto, almeno 3 segmenti successivi sono stati ricevuti dal destinatario
 - a differenza del caso di timeout, la rete mostra di essere in grado di consegnare una certa quantità di dati
 - è possibile, quindi, evitare una nuova partenza lenta, ricominciando direttamente dalla fase di prevenzione della congestione

Equità tra le connessioni TCP

Equità: se K sessioni TCP condividono lo stesso collegamento con ampiezza di banda R, che è un collo di bottiglia per il sistema, ogni sessione dovrà avere una frequenza trasmissiva media pari a R/K



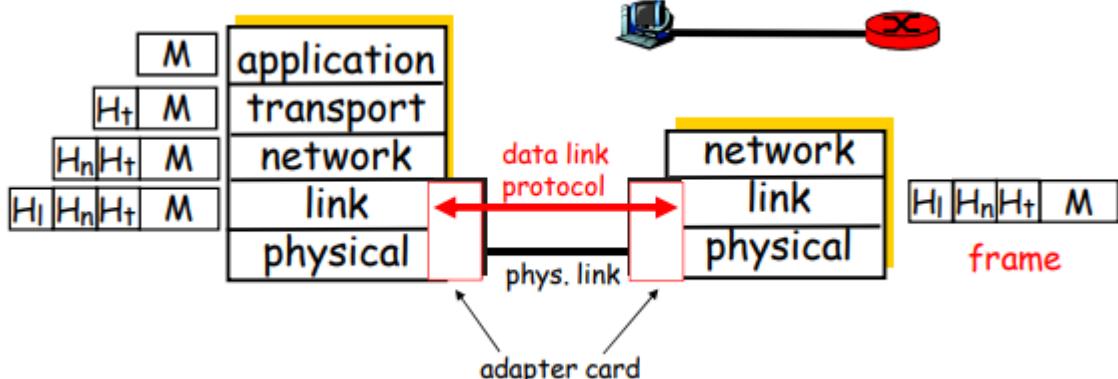
Ipotesi

- MSS e RTT uguali per le due connessioni
 - a parità di dimensioni della finestra quindi il throughput è lo stesso
- Entrambe le connessioni si trovano oltre lo slow start
 - fase di prevenzione della congestione: AIMD

21. Lo Strato di Collegamento

Caratteristiche del livello data link

Si occupa della comunicazione tra due dispositivi **fisicamente connessi**: host-router, router-router, host-host. L'unità di dati è il **frame**.



Servizi del livello data link

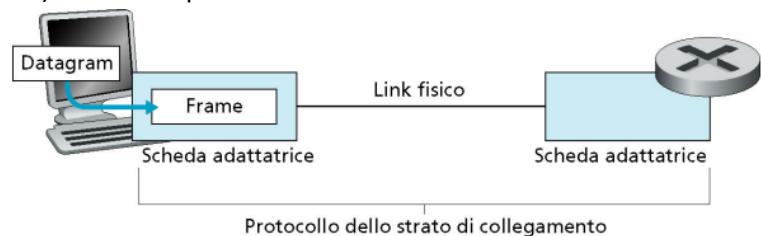
- Framing (incorniciatura) ed accesso al link:
 - Incapsulamento di datagrammi all'interno di frame, aggiunta di campi di intestazione (header e trailer)
 - Gestione dell'accesso al canale, in caso di mezzo condiviso
 - Utilizzo di "indirizzi fisici" all'interno delle frame, per identificare nodo sorgente e destinazione.
N.B.: indirizzi fisici diversi dagli indirizzi di rete
- Trasferimento affidabile dei dati tra due dispositivi fisicamente connessi:
 - Utile soprattutto in caso di collegamenti con alta probabilità di errore, quali i link wireless
- Controllo di flusso:
 - Per regolare la velocità di trasmissione tra mittente e destinatario

- Rilevazione degli errori:
 - Errori causati da attenuazione del segnale o da presenza di rumore (interferenza)
 - Il ricevente rileva la presenza di errori e segnala tale evento al mittente, oppure elimina la frame ricevuta
- Correzione degli errori:
 - Il ricevente identifica e corregge errori su alcuni bit della frame, evitando ritrasmissioni da parte del mittente
- Trasferimento dati di tipo half-duplex o fullduplex

Interfacce di rete

Un adattatore è un circuito (es: scheda PCMCIA) che si occupa di:

- Ricevere datagram dallo strato di rete
- Incapsulare i datagram ricevuti all'interno di frame
- Trasmettere le frame all'interno del link di comunicazione
- In ricezione, effettuare le operazioni inverse...



Ci sono due tipi di "link" di rete: punto-punto e broadcast (mezzo condiviso: Ethernet, Satellite, etc...)

Protocolli di accesso multiplo

Un unico canale di comunicazione condiviso. Due o più trasmissioni simultanee da parte dei nodi della rete: interferenza (solo un nodo può inviare dati con successo).

Protocolli di accesso multiplo:

- Un algoritmo distribuito determina le modalità di condivisione del canale, vale a dire quando una stazione può trasmettere
- Le comunicazioni per regolare l'accesso al canale utilizzano il canale stesso!
- Caratteristiche di un protocollo di accesso multiplo:
 - Sincrono o asincrono
 - Necessità di informazioni riguardanti le altre stazioni
 - Robustezza (ad esempio, in relazione agli errori sul canale)
 - Prestazioni

Tassonomia

- Channel Partitioning
 - Suddivide il canale in "porzioni" più piccole (slot temporali o di frequenza)
 - Ogni nodo gode dell'uso esclusivo di una di tali porzioni
- Random Access
 - Consente collisioni dovute a trasmissioni simultanee
 - Gestisce le collisioni
- Taking turns
 - Coordina opportunamente l'accesso al mezzo, in modo da evitare le collisioni

Obiettivi

Per un canale con velocità di R bit/sec:

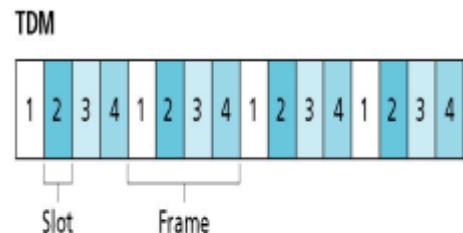
- Se un solo nodo ha dati da inviare:
 - Quel nodo ha un throughput di R bit/sec
- Se M nodi hanno dati da spedire:
 - Ognuno di essi ha un throughput medio di R/M bit/sec

- Il protocollo per la gestione dell'accesso è distribuito:
 - Assenza di "single points of failure"
- Il protocollo è semplice:
 - Implementazione economica

Protocolli di suddivisione del canale

Time Division Multiple Access (TDMA)

- L'accesso al canale avviene a "cicli":
 - Ogni stazione ottiene uno slot di trasmissione di lunghezza fissa in ogni ciclo
 - Gli slot inutilizzati da una stazione vanno deserti
- Vantaggi:
 - Elimina le collisioni
 - È equo
- Svantaggi:
 - Throughput max per un nodo, in una rete con N stazioni: R/N bit/sec anche se il nodo in esame è l'unico ad avere frame da spedire
 - Un nodo deve sempre aspettare il suo turno nella sequenza di trasmissione

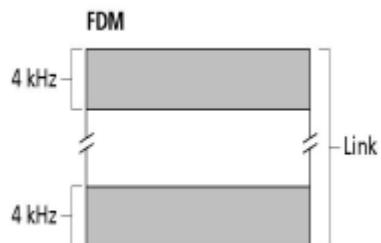


Legenda:

Tutti gli slot etichettati "2" sono dedicati a una specifica coppia sender-receiver

Frequency Division Multiple Access (FDMA)

- Lo spettro di trasmissione è diviso in bande di frequenza
- Ad ogni stazione è assegnata una banda di frequenza fissa
- Il tempo di trasmissione inutilizzato nelle singole bande di frequenza risulta sprecato
- Vantaggi: Come per il TDMA
- Svantaggi: Come per il TDMA



Code Division Multiple Access (CDMA)

- Un codice unico è assegnato ad ogni utente: code set partitioning
- Usato principalmente nei canali wireless di tipo broadcast (reti cellulari, satellitari, ecc.)
- Tutti gli utenti condividono la stessa frequenza di trasmissione, ma ognuno di essi possiede un codice unico (noto come "chipping sequence") per codificare i dati
- segnale codificato = (dati originali) X (chipping sequence)
- decodifica: prodotto scalare del segnale codificato e della chipping sequence
- Consente a diversi nodi di trasmettere simultaneamente, riducendo al minimo l'interferenza nel caso in cui si siano scelti codici "ortogonali"

Protocolli ad accesso casuale

Quando un nodo ha un pacchetto da trasmettere:

- Trasmette alla massima velocità consentita dal canale
- Non esiste nessuna forma di coordinamento a priori tra i nodi

Trasmissione simultanea di due o più nodi: collisione.

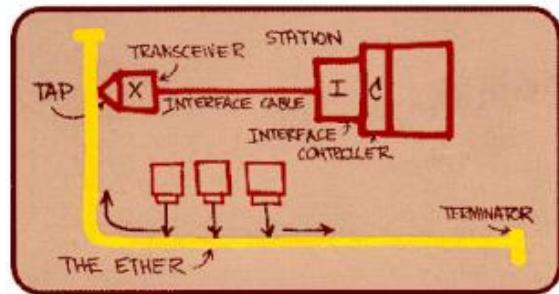
Un protocollo ad accesso casuale specifica:

- Come rilevare le collisioni
- Come risolvere le collisioni:
 - Es: tecniche di ritrasmissione delle frame

Ethernet

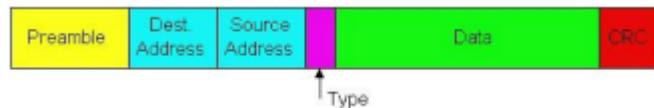
Tecnologia "dominante" per le LAN:

- La prima tecnologia LAN ampiamente diffusa
- Più semplice ed economica rispetto alle LAN "a token" e ad ATM
- Aggiornata nel corso degli anni: 10, 100, 1000, 10000 Mbps



Struttura della Frame Ethernet

L'interfaccia di rete del mittente incapsula i datagrammi IP (o altri pacchetti di livello rete) in **frame Ethernet**



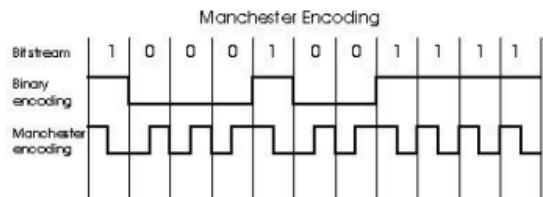
- **Preambolo** (8 byte):
 - 7 byte con una sequenza 10101010 seguiti da un byte con la sequenza 10101011
 - utilizzato per sincronizzare i clock del mittente e del destinatario
- **Indirizzi** (6 byte): La frame è ricevuta da tutti gli adattatori di rete presenti sulla LAN, e scartata se l'indirizzo destinazione non coincide con quello della scheda stessa (indirizzo broadcast: ff:ff:ff:ff:ff:ff)
- **Type** (2 byte): indica il protocollo di livello rete sovrastante, principalmente IP, ma altri protocolli (ad esempio Novell IPX e AppleTalk) sono supportati
- **CRC** (4 byte): controllo effettuato alla destinazione
 - se l'errore è rilevato, la frame viene scartata

CSMA/CD

Jam Signal: consente alle altre stazioni di accorgersi dell'avvenuta collisione (48 bit)

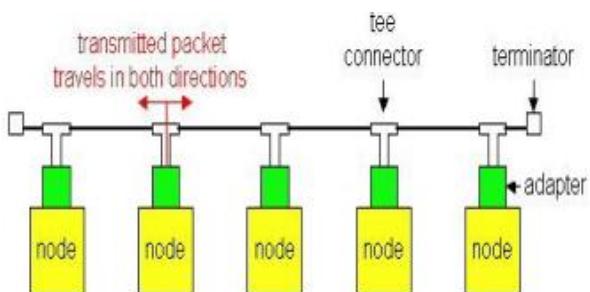
Exponential Backoff

- Obiettivo: algoritmo per adattare i successivi tentativi di ritrasmissione al carico corrente della rete
- In presenza di sovraccarico il tempo d'attesa casuale sarà maggiore
 - prima collisione: scegli K tra {0,1}; il ritardo di trasmissione è pari ad un intervallo $K \times 512$ bit (pari a 51.2 usec in una Ethernet a 10 Mbps)
 - dopo la seconda collisione: scegli K tra {0,1,2,3}...
 - dopo 10 o più collisioni, scegli K tra {0,1,2,3,4,...,1023}
- Segnale: in banda base, **codifica Manchester**
 - Usata in 10BaseT, 10Base2
 - Ogni bit ha una transizione
 - Permette ai clock dei nodi riceventi e trasmittenti di sincronizzarsi. Non è richiesto un clock centralizzato e globale tra tutti i nodi
 - È una problematica di livello fisico



Tecnologia Ethernet: 10Base2

- **10**: 10Mbps; **2**: massima lunghezza del cavo: 200 metri
- Topologia a bus su cavo coassiale sottile (thin)
- Impiego di ripetitori per collegare più segmenti
- I ripetitori ritrasmettono i bit in entrata da un'interfaccia verso le altre interfacce
 - entità di livello fisico



- 10BaseT e 100BaseT
 - 10/100 Mbps
 - La versione a 100Mbps è nota come “fast ethernet”
 - T sta per Twisted Pair (doppino intrecciato)
 - Topologia “a stella”, mediante un concentratore (hub) al quale gli host sono collegati con i doppini intrecciati
- Massima distanza tra nodo e hub pari a 100 metri
- Gli hub possono disconnettere le schede malfunzionanti: “jabbering”
- Gli hub possono fornire informazioni utili al monitoraggio e collezionare statistiche per effettuare previsioni, agevolando il compito degli amministratori della LAN

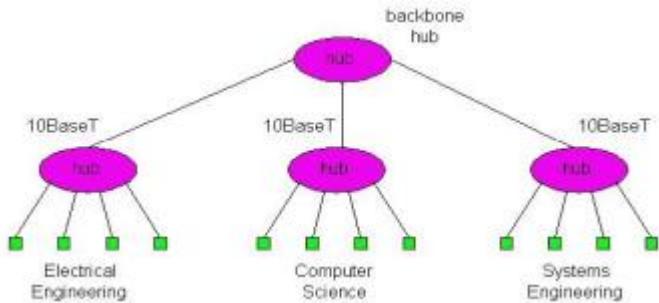
Gbit Ethernet

- Usa il formato delle frame di Ethernet standard
- Funziona in modalità collegamento point-to-point ed a canale broadcast condiviso
- In modalità condivisa, è utilizzato il protocollo CSMA/CD
 - le distanze tra i nodi sono ridotte al minimo per aumentare l'efficienza
- Usa gli hub, che in questa tecnologia prendono il nome di “Buffered Distributors”
- Full-Duplex a 1 Gbps nel caso di collegamento di tipo point-to-point

Hub

Dispositivi di livello fisico; sostanzialmente si tratta di ripetitori di bit che riproducono i bit in ingresso ad un'interfaccia su tutte le altre interfacce.

Gli hub possono essere organizzati in una gerarchia (o architettura multilivello), con un backbone hub al livello più alto.



Caratteristiche

- Ogni LAN collegata è considerata come un segmento di LAN
- Gli hub non isolano i domini di collisione
 - le stazioni possono subire una collisione per una trasmissione simultanea con qualunque stazione presente su qualunque segmento
- Vantaggi degli hub
 - Sono dispositivi semplici e poco costosi
 - L'organizzazione multilivello garantisce una parziale tolleranza ai guasti: porzioni di LAN continuano a funzionare in caso di guasto ad uno o più hub
 - Estende la massima distanza esistente tra i nodi (100m per ogni Hub)

Limiti degli hub

- La creazione di un singolo dominio di collisione non comporta alcun aumento del throughput massimo
 - Il throughput complessivo in una rete multilivello è lo stesso di una rete con un unico segmento
- 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
- Solo una tipologia di Ethernet (per esempio, 10BaseT e 100baseT)

Bridge

Dispositivi di livello 2: in grado di leggere le intestazioni di frame Ethernet, ne esaminano il contenuto, e selezionano il link d'uscita sulla base dell'indirizzo destinazione.

I bridge isolano i domini di collisione, grazie alla loro capacità di porre le frame in un buffer (dispositivi store & forward).

Non appena una frame può essere inoltrata su un link d'uscita, un bridge usa il protocollo CSMA/CD sul segmento LAN d'uscita prima di trasmettere.

Vantaggi dei bridge:

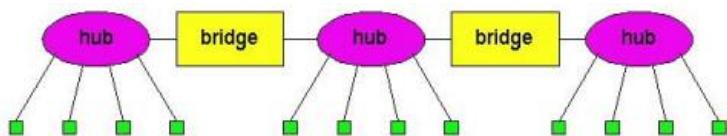
- Isolano i domini di collisione, determinando un aumento complessivo del throughput massimo
- Non introducono limitazioni sul numero massimo delle stazioni, né sull'estensione geografica
- Possono collegare differenti tecnologie, dal momento che sono dispositivi di tipo store & forward
- Trasparenti: non richiedono alcuna modifica negli adattatori dei computer

Frame filtering & forwarding

- Forwarding
 - Come fare a sapere su quale segmento una frame deve essere inoltrata?
- I bridge filtrano i pacchetti
 - Stesso segmento di LAN: le frame non sono inoltrate su altri segmenti di LAN

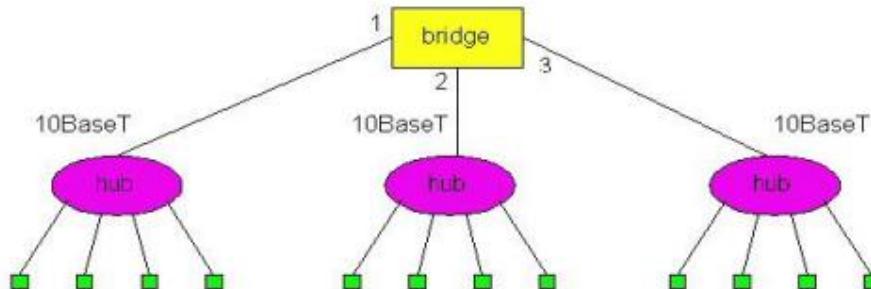
Interconnessione senza backbone

Soluzione non consigliata a causa di due motivi:



- esiste un punto critico presso l'hub di Computer Science, in caso di rottura dello stesso
- il traffico tra EE e SE deve necessariamente attraversare il segmento CS

Backbone Bridge



Bridge Filtering

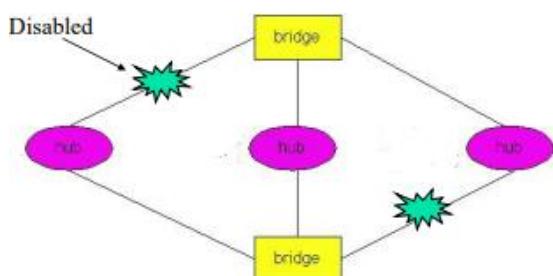
I bridge eseguono un algoritmo di auto apprendimento per scoprire a quali interfacce sono collegati gli host

Tali informazioni sono salvate in delle "filtering tables"

- Quando una frame è ricevuta, il bridge "prende nota" del segmento di LAN di provenienza
- L'interfaccia di provenienza è memorizzata in una filtering table

Bridge Spanning Tree

- Per incrementare l'affidabilità, può essere utile introdurre un certo grado di ridondanza
 - percorsi alternativi
- In presenza di percorsi alternativi simultanei, vengono create copie molteplici delle frame (loop)
- SOLUZIONE: organizzare i bridge mediante uno spanning tree, disabilitando alcune interfacce

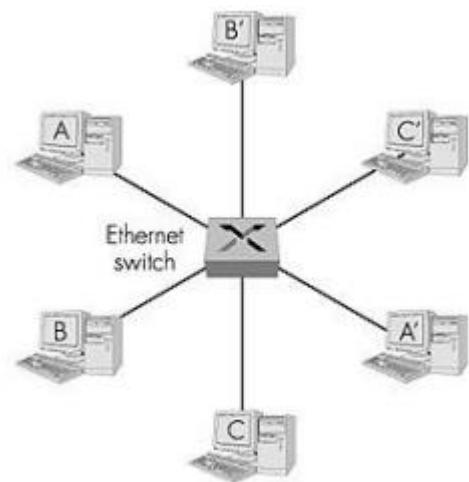


Switch

Effettuano l'inoltro di frame a livello 2: filtraggio mediante l'uso di indirizzi LAN.

Switching: da A a B e da A' a B' simultaneamente: non ci sono collisioni.

Alto numero di interfacce, spesso host singoli con topologia a stella con collegamento ad uno switch (è praticamente Ethernet, ma senza collisioni)



Cut-through switching

- Pro
 - frame inoltrate dall'ingresso all'uscita senza attendere l'assemblamento dell'intera frame
 - Leggera diminuzione della latenza
 - Consentono la combinazione di interfacce condivise/dedicate, a 10/100/1000 Mbps
- Contro: E le frame affette da errore ?

22. Reti Wireless

Wireless LAN: 802.11

Le reti wireless rappresentano una tecnologia in rapida evoluzione per la connessione di computer. In una rete locale wireless, i dispositivi non sono collegati fisicamente, ma, per comunicare, usano onde elettromagnetiche che si propagano nello spazio.

Come altre tecnologie LAN, l'802.11 è progettato per un impiego in aree geografiche limitate ed ha lo scopo principale di "fare da mediatore" nell'accesso ad un mezzo condiviso di comunicazione (in questo caso, una frequenza radio).

Livello fisico

- 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*":
 - L'obiettivo di tali tecniche è quello di diffondere il segnale su di un intervallo di frequenza ampio, in modo tale da minimizzare l'effetto dell'interferenza da parte di altri dispositivi

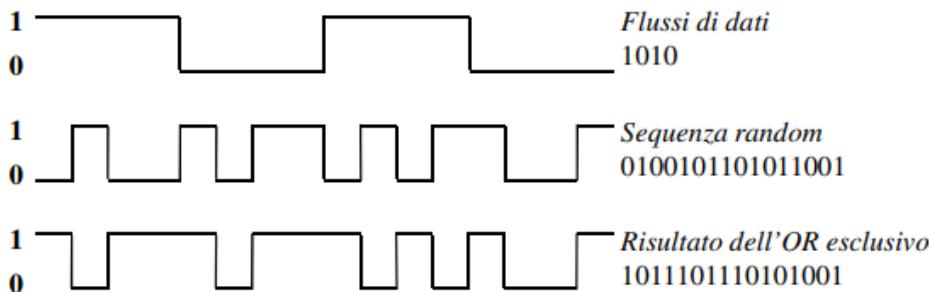
Frequency hopping

- Il segnale è trasmesso su una sequenza "random" di frequenze
- Tale sequenza è in realtà calcolata in maniera algoritmica, tramite un generatore di numeri pseudo-casuali
- Il ricevitore:
 - utilizza il medesimo algoritmo del mittente: inizializzazione con il medesimo seme
 - è dunque in grado di "saltare" le frequenze in maniera sincronizzata con il mittente, per ricevere correttamente le frame

Direct sequence

- Ogni bit di una frame è rappresentato da molteplici bit nel segnale trasmesso

- Il mittente invia, in effetti, il risultato dell'OR esclusivo di tale bit e di n bit scelti in maniera casuale
- Come nel caso del frequency hopping, la sequenza di bit casuali è generata da un generatore di numeri "pseudo-casuali" nota sia al mittente che al ricevitore
- I valori trasmessi sono noti come chipping sequence (come nel caso del CDMA)
- L'802.11 utilizza una chipping sequence a 11 bit
- Un esempio: chipping sequence a 4 bit



Canali ed associazione ad un Access Point

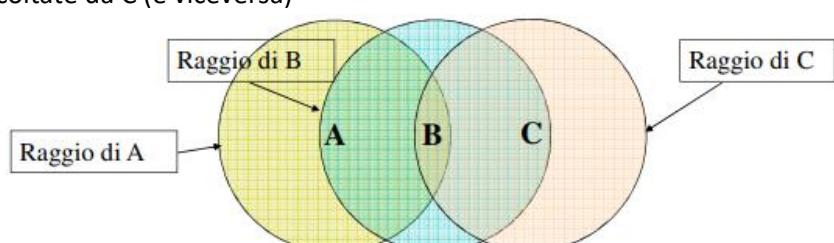
- 802.11b: 2.4GHz-2.485GHz
 - Lo spettro è diviso in 11 canali a differenti frequenze (solo 3 canali risultano non sovrapposti)
 - All'atto dell'installazione di un AP, l'amministratore di rete sceglie il canale da utilizzare per la trasmissione
 - Possibilità di interferenza nel caso in cui due AP vicini utilizzino lo stesso canale
- Un host deve associarsi ad un AP
 - Controlla i vari canali ascoltando le cosiddette beacon frame, contenenti MAC address ed identificativo (SSID – Service Set Identifier) dell'AP
 - Seleziona l'AP cui associarsi ed inizia la procedura di associazione (che può prevedere anche una fase di autenticazione)
 - Al termine di tale procedura, tipicamente effettuerà una richiesta DHCP per ottenere un indirizzo IP nella subnet dell'AP

Medium Access Control

- Il metodo di accesso è simile ad Ethernet:
 - Prima di trasmettere, si attende finché il canale diventa libero
 - In caso di collisione: algoritmo del binary exponential backoff
- Tuttavia, bisogna tenere in considerazione il fatto che non tutti i nodi sono sempre alla portata l'uno dell'altro
 - Ciò determina due tipi di problemi: problema del nodo nascosto (Hidden node problem), problema del nodo esposto (Exposed node problem)

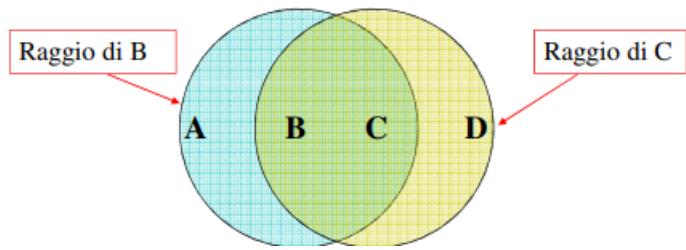
Hidden nodes problem

- Le trasmissioni di A non sono ascoltate da C (e viceversa)
- A e C possono inviare dati simultaneamente verso B causando una collisione in ricezione
- Né A né C sono in grado di rilevare la collisione
- A e C sono detti nodi nascosti (l'uno rispetto all'altro)



Exposed nodes problem

- B invia dati ad A
- C è al corrente di tale comunicazione perché ascolta le trasmissioni di B:
 - È un errore per C concludere di non poter trasmettere a nessuno
 - Ad esempio, C potrebbe inviare frame a D senza interferire con la capacità di A di ricevere dati da B



IEEE 802.11: accesso multiplo

Come Ethernet, usa il CSMA (accesso random e si evitano collisioni con eventuali trasmissioni in corso), ma a differenza di Ethernet non effettua collision detection (tutte le frame sono trasmesse nella loro interezza) e usa gli acknowledgement (conferma di avvenuta ricezione)

Perché non si effettua la collision detection?

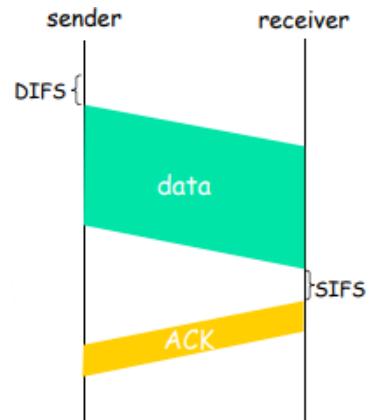
- Difficoltà a ricevere durante la trasmissione, a causa della debolezza dei segnali ricevuti (fading)
- Impossibile in alcuni casi accorgersi delle collisioni:
 - Stazione nascosta (hidden terminal)
 - Fading

Obiettivo: evitare le collisioni: CSMA/C(ollision)A(voidance)

Protocollo CSMA/CA

802.11 sender

- Se il canale è inattivo per un tempo pari a DIFS (Distributed Inter Frame Space) allora
 - Trasmette un'intera frame (senza CD)
- Se il canale è occupato
 - Sceglie un backoff time casuale
 - Il timer viene decrementato mentre il canale è inattivo
 - Allo scadere del timer, trasmette una frame
 - Se non riceve ACK, incrementa l'intervallo di backoff casuale, torna al passo 2



802.11 receiver

- se la frame è ricevuta in maniera corretta
 - restituisce un ACK dopo un tempo SIFS (Short Inter Frame Space)

Collision Avoidance: RTS/CTS

- **Idea:** consentire al mittente di "prenotare" il canale
 - Evitare collisioni per le frame di dati "lunghe"
- Soluzione opzionale
- Il mittente trasmette prima una piccola frame request-to-send (RTS) all'AP, usando il CSMA
 - Le frame RTS possono collidere (ma sono piccole...)
- L'AP invia in broadcast una frame clear-to-send CTS in risposta alla frame RTS
- La frame CTS è ascoltata da tutti i nodi
 - Il mittente trasmette la frame dati
 - Le altre stazioni differiscono le loro trasmissioni

Si evitano completamente le collisioni sui dati, usando piccoli pacchetti di prenotazione!

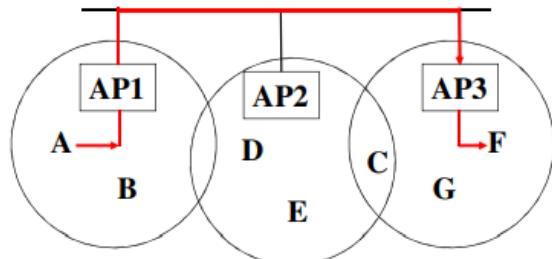
WLAN/802.11

Collision avoidance

- Lo standard 802.11 risolve i due problemi precedenti introducendo l'algoritmo CSMA/CA visto in precedenza
- Prima di inviare i dati, il mittente trasmette una frame di "richiesta di trasmissione":
 - Request to Send (RTS): in tale frame è presente anche un campo che indica la lunghezza della frame dati da trasmettere
- Il ricevitore risponde con una frame di "permesso di trasmissione":
 - Clear to Send (CTS): in tale frame viene replicato il valore relativo alla lunghezza dei dati, annunciato dal mittente
- Un nodo che vede la frame CTS sa di essere vicino al ricevitore:
 - Esso non può trasmettere per tutto il tempo necessario ad inviare la frame dati (la cui lunghezza è stata specificata nella frame RTS)
- Un nodo che vede la frame RTS, ma non quella CTS, non è abbastanza vicino al ricevitore per interferire con esso e può quindi trasmettere senza attendere
- Il ricevitore invia un ACK dopo aver ricevuto una frame
- I nodi non rilevano le collisioni:
 - Se due nodi inviano una frame RTS in contemporanea, queste frame collideranno
 - I nodi assumono che vi sia stata una collisione se non ricevono una frame CTS di risposta

Distribution system

- Per fornire il supporto alla mobilità e la connessione ad altre reti (prima tra tutte, la rete Internet), si utilizzano dei nodi speciali:
 - Access Point (AP): Si tratta di nodi connessi ad un'infrastruttura di rete fissa, chiamata Distribution System
- Ogni nodo si associa ad un particolare access point
- Se A vuole comunicare con F:
 - A invia una frame al suo access point (AP1)
 - AP1 inoltra ad AP3 la frame attraverso il distribution system
 - AP3 trasmette la frame ad F
- La tecnica per selezionare un Access Point è detta scanning e prevede quattro passi:
 1. Il nodo invia una frame di probe
 2. Tutti gli AP alla portata del nodo rispondono con una frame di risposta al probe
 3. Il nodo seleziona uno degli AP (tipicamente quello con la migliore qualità del segnale ricevuto), e gli invia una frame di richiesta di associazione
 4. L'AP selezionato risponde con una frame di conferma di associazione
- Il protocollo descritto è utilizzato:
 - Quando il nodo si unisce alla rete
 - Quando il nodo diventa "scontento" dell'attuale AP utilizzato
Questo avviene, per esempio, perché il segnale ricevuto da tale AP risulta indebolito a causa del fatto che il nodo si sta allontanando da esso
- Durante lo spostamento, un nodo potrebbe preferire un nuovo AP ed inviargli una richiesta di associazione:
 - Il nuovo AP invia una notifica del cambiamento al vecchio AP, attraverso il distribution system



23. Protocolli per la trasmissione di flussi multimediali in Internet

Trasferimento di informazioni multimediali su rete

Problema: trasferire informazioni multimediali (audio, video, ...) da una sorgente ad uno o più ricevitori attraverso una rete.

Per ridurre la quantità di informazioni trasferita sulla rete, il **trasmettitore** effettua una compressione mediante un'opportuna tecnica (MPEG 1-2-4, MJPEG, MP3, ...). Sulla **rete** l'informazione è trasferita a **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 in forma di video (sequenza di fotogrammi).

Nel caso di **informazioni live**, l'informazione è prodotta dalla sorgente mediante un apposito sistema di acquisizione (microfono + scheda audio, telecamera + video capture board), opportunamente compressa (in tempo reale) e trasmessa sulla rete ai ricevitori.

Nel caso di **informazioni preregistrate**, l'informazione è già registrata in formato compresso (MPEG, MJPEG, MP3, ...) in un file memorizzato su memoria di massa (hard-disk, CDROM, DVD, ...).

Informazioni multimediali preregistrate

- Trasferimento dell'intero file da sorgente a ricevitore e successiva riproduzione: file transfer
 - La riproduzione può iniziare solo al termine del trasferimento del file (ritardo proporzionale alla dimensione del file)
 - È necessaria una adeguata capacità di memorizzazione (su memoria di massa) da parte del ricevitore
 - Questa soluzione è idonea solo per documenti di piccole dimensioni (audio-clip e/o video-clip)
- Riproduzione progressiva del contenuto multimediale durante il trasferimento dell'informazione: streaming
 - Il ricevitore memorizza l'informazione ricevuta in un buffer (playout buffer) che viene continuamente alimentato dai dati ricevuti dalla rete e svuotato progressivamente
 - La riproduzione può iniziare non appena il buffer si è "sufficientemente" riempito
 - Il ricevitore non deve memorizzare l'intero file
 - La qualità della riproduzione può degradare se la rete non mantiene la continuità temporale del flusso di informazioni trasmesso dalla sorgente (sensibilità al jitter)

Informazioni multimediali 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: trasmissione in **streaming**

Sensibilità dello streaming alla qualità del servizio

Il ricevitore riceve i pacchetti, recupera l'informazione originaria e la riconverte in forma audio/video.

Il **ricevitore** riesce a recuperare la **continuità del flusso di informazioni** prodotto dalla sorgente se tutti i pacchetti arrivano a destinazione, con la stessa tempificazione relativa.

La **rete** può alterare la continuità temporale del flusso di informazioni in due modi:

- Facendo occasionalmente perdere dei pacchetti
- Consegnando i pacchetti al ricevitore con una tempificazione relativa diversa da quella con cui sono stati trasmessi (jitter)

Perché la rete possa effettivamente supportare la trasmissione di flussi multimediali occorre che alcuni parametri di Qualità del Servizio (QoS) siano soddisfatti (percentuale di perdita di pacchetti, latenza, etc...).

Degradazione introdotta dalla rete

- Gli effetti sono diversi a seconda della natura del media (audio/video), a seconda della tecnica di compressione utilizzata ed a seconda del grado di alterazione introdotto
 - nel caso di flusso audio, vengono percepiti dei "disturbi" (hiccups)
 - nel caso di flusso video, si hanno dei disturbi (glitches) che possono essere più o meno localizzati nel tempo e nello spazio
- Sia audio che video possono in genere tollerare una parziale degradazione, ma quando si oltrepassano dei valori di soglia l'informazione diventa inintelligibile

Contromisure

- Rispetto alla perdita occasionale di pacchetti, ci si difende mediante l'adozione di tecniche di compressione robuste, per le quali l'informazione audio/video ricostruita non è sensibilmente degradata quando occasionalmente si perde un pacchetto
 - In alcuni casi si adottano tecniche di Forward Error Correction (FEC)
 - L'adozione di tecniche basate sulla ritrasmissione (alla TCP) non sono considerate idonee per lo streaming
- Per limitare gli effetti del jitter si adotta una strategia di bufferizzazione: un buffer in ricezione fa da volano e compensa (introducendo un ritardo extra) la variabilità del ritardo di attraversamento della rete
- NOTA: non sarebbe necessario introdurre delle contromisure se la rete fosse in grado di offrire servizi a **qualità garantita**
 - Internet offre un servizio best-effort!

Trasferimento di informazioni multimediali su Internet

Il trasferimento di informazioni multimediali su Internet mediante la tecnica del **file transfer** è tipicamente realizzato adottando il protocollo applicativo HTTP, il quale si appoggia sul protocollo di trasporto TCP.

Per la trasmissione in **streaming** sono adottate due tecniche:

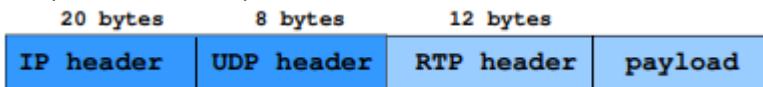
- mediante un protocollo ad-hoc (RTP) su UDP
- mediante HTTP su TCP

Realtime Transport Protocol (RTP)

- RTP offre un servizio di livello trasporto specificamente progettato per i requisiti di flussi multimediali
- I pacchetti RTP sono incapsulati in datagrammi UDP
 - Un protocollo di livello trasporto su un altro di livello trasporto
- RTP è un protocollo concepito per essere implementato direttamente nelle applicazioni, e non come uno strato aggiuntivo dello stack protocollare
- Offre le funzionalità minimali richieste dalla trasmissione di flussi continui tipici delle applicazioni multimediali
- È neutrale rispetto alla codifica utilizzata (MPEG, etc...)
- RTP fornisce informazioni di tempificazione (timestamp) per consentire
 - **sincronizzazione intra-media**: ricostruzione della corretta tempificazione della sequenza di pacchetti ricevuti
 - **sincronizzazione inter-media**: finalizzata a mantenere "al passo" flussi multimediali trasmessi separatamente (es. audio e video: sincronizzazione "lip-sync")
- RTP supporta sia la trasmissione unicast che la trasmissione multicast
- I suoi meccanismi sono scalabili rispetto al numero di appartenenti al gruppo multicast

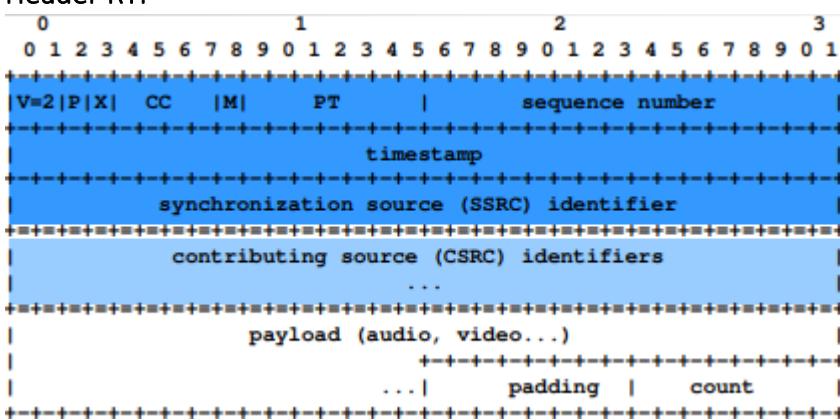
- Separa la trasmissione dei dati dalla trasmissione delle informazioni di controllo
- RTP è definito congiuntamente ad un protocollo di controllo (RTCP) utilizzato per scambiare informazioni di servizio e di controllo sulla qualità della trasmissione
- Fornisce informazioni necessarie a combinare flussi di informazioni differenti mediante appositi mixer software

Incapsulamento di pacchetti RTP



- 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 dati
- Se $2n$ è il numero di porto destinazione usato per uno flusso, il numero successivo $2n+1$ è usato da RTCP per trasmettere le informazioni di controllo relative a quel flusso

Header RTP



- **Payload Type (PT)**: 7 bit, specifica la codifica utilizzata per i dati (PCM, MPEG2 video, etc...)
- **Sequence Number**: 16 bit, serve ad identificare perdite di pacchetti
- **Timestamp**: 32 bit, specifica il tempo di campionamento del primo byte del payload; serve a rimuovere il jitter introdotto dalla rete mediante bufferizzazione
- **Synchronization Source identifier (SSRC)**: 32 bit, identifica la sorgente del flusso, ed è scelto casualmente dalla sorgente stessa; è introdotto per non dover fare affidamento sull'indirizzo IP per identificare la sorgente (sono possibili conflitti)
- **Contributing Source identifier list (CSRC)**: sequenza di n campi da 32 bit ($0 \leq n \leq 15$), ciascuno dei quali identifica la sorgente originaria in un flusso prodotto dalla "fusione" di flussi diversi mediante un mixer software
 - esempio: audio-conferenza a più partecipanti: SSRC identifica il mixer e CSRC indica lo speaker corrente
- **Version (V), Padding (P), eXtension (X), CSRC Count (CC), Marker (M)**

Sessione RTP

- Una associazione tra un gruppo di entità che comunicano mediante RTP
- Alcune applicazioni danno vita a sessioni RTP differenti per media differenti (es. audio e video), a meno che la tecnica di codifica adottata non effettui un multiplexing di flussi differenti in un singolo flusso di dati
- Sessioni RTP differenti (es. audio e video) vengono distinte da un ricevitore mediante il port number di livello trasporto (UDP)

RTP timestamp e numero di sequenza

- Il valore di timestamp inserito in ogni pacchetto riferisce la tempificazione dei dati inseriti nel payload rispetto ad un clock specifico per il media trasportato
- Possono essere generati pacchetti RTP consecutivi con lo stesso timestamp
- Il numero di sequenza identifica un pacchetto rispetto agli altri principalmente per consentire di identificare pacchetti persi
- non possono essere generati due pacchetti con lo stesso numero di sequenza

RTCP

- 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 (percentuale pacchetti persi, ...)
 - identificazione dei partecipanti ad una sessione RTP mediante un identificativo detto CNAME
- Nel caso di trasmissione RTP tra partecipanti ad un gruppo multicast, RTCP consente ad ogni partecipante di conoscere il numero di partecipanti

Messaggi RTCP

- 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, numero di pacchetti ricevuti, percentuale di pacchetti persi, jitter dei tempi di inter-arrivo, ecc. e servono a monitorare la qualità della trasmissione
- I messaggi di tipo description, invece, descrivono la sorgente del flusso (contengono tra l'altro il CNAME)
- BYE serve a notificare l'uscita da una sessione
- APP è un tipo di messaggio le cui funzioni sono definibili dall'applicazione

Banda usata da RTCP

- Nel caso di trasmissione multicast, ciascun ricevitore invia periodicamente (allo stesso gruppo multicast) i report RTCP
- Cosa succede se il numero di membri del gruppo diventa molto elevato?
- Per contenere il traffico di controllo, si inserisce una minima forma di coordinamento:
 - L'intervallo temporale tra due report è proporzionale al numero di partecipanti alla sessione
 - in modo che la banda consumata da RTCP non superi il 5% della banda usata dalla sessione