



A.A. 2021-2022



RETI DI CALCOLATORI I

Università degli studi di Napoli

Federico II



Valentino Bocchetti



N86003405



vale.bocchetti@studenti.unina.it

Indice

1	Introduzione al corso	15
1.1	Contatti	15
1.2	Argomenti	15
1.3	Libri di testo	15
1.4	Modalità di esame	15
1.5	Link Utili	15
2	Lezione del 29-09	15
2.1	Tecnologie digitali	15
2.1.1	Comunicazione	16
3	Lezione del 04-10	16
3.1	Mainframe e super computer	16
3.2	L'ecosistema Internet	16
3.3	Comunicazione e commutazione	17
3.4	Commutazione di pacchetto	18
3.5	Ritardo nelle reti a commutazioni di pacchetto	19
3.6	Reti a datagrammi	19
3.7	Reti a circuiti virtuali	20
3.8	Datagrammi vs circuiti virtuali	20
4	Lezione del 11-10	20
4.1	Elementi costitutivi delle reti di calcolatori	20
4.2	Le reti di calcolatori	22

4.2.1	Caratteristiche LAN	22
4.2.2	Caratteristiche MAN	22
4.2.3	Caratteristiche WAN	22
4.3	Gestire la complessità	23
4.4	Modelli a strati	23
4.4.1	Modelli a strati - interfacce e protocolli	24
4.5	Imbustamento dei messaggi	24
4.6	Protocol Data Unit (PDU)	25
4.7	Frammentazione dei messaggi ad un livello	25
4.8	Il modello OSI (Open System Interconnection)	25
4.9	Dispositivi di rete e livelli	25
4.10	Approccio a livelli per la risoluzione dei problemi	27
4.10.1	Livello 1 - Fisico	28
4.10.2	Livello 2 - Data Link	28
4.10.3	Livello 3 - Rete	29
4.10.4	Livello 4 - Trasporto	29
4.10.5	Livello 5 - Sessione	29
4.10.6	Livello 6 - Presentazione	29
4.10.7	Livello 7 - Applicazione	29
5	Lezione del 12-10	29
5.1	Lo stack TCP/IP	29
5.2	Applicazioni e protocolli dello strato applicativo	30
5.3	Strato dell'applicazione	30

5.4	Comunicazione tra processi	31
5.5	Livello trasporti - TCP e UDP	31
5.6	Requisiti di alcune applicazioni	31
6	Lezione del 18-10	32
6.1	Internet - Origine ed Evoluzione	32
6.2	Internet - Architettura della rete	32
6.3	Struttura di Internet	33
6.4	La rete GAAR	33
6.5	Chi regolamenta Internet	34
7	Lezione del 20-10	34
7.1	Strato dell'applicazione	34
7.2	HTTP	34
7.3	Uniform Resource Locator (URL)	35
7.4	Connessioni persistenti e non persistenti	35
7.5	Messaggio HTTP request	35
7.6	Codici di stato - esempi	35
7.7	HTTP per il trasferimento file	36
7.8	Lo scambio di messaggi	36
7.8.1	Il metodo GET	36
7.8.2	Il metodo HEAD	37
7.8.3	Il metodo POST	37
7.8.4	Il metodo PUT	37
7.9	Gli header generali	37

7.10 I cookies	38
7.11 Cookies - header specifici	38
7.12 Web caching	38
7.13 Transazioni web con caching	39
7.14 Gestione della coerenza	39
7.15 Protocollo FTP (File Transfer Protocol)	40
8 Lezione del 25-10	40
8.1 Il protocollo SMTP	40
8.1.1 Entità in gioco	40
8.1.2 Caratteristiche	41
8.1.3 Formato del messaggio SMTP	42
8.2 L'estensione MIME (Multipurpose Internet Mail Extensions)	42
8.2.1 Esempi di tipi MIME	42
8.3 POP3 (Post Office Protocol)	43
8.4 La catena dei protocolli per la posta	43
8.5 Domain Name System (DNS)	43
8.5.1 Altre funzionalità offerte	43
8.5.2 DNS distribuito	43
8.6 I top-leve domain server (TLD)	44
8.7 Tipologie di server DNS (locale)	44
8.8 Tipologie di server DNS (Root)	44
8.9 Tipologie di server DNS (Authoritative)	45
8.10 Caching dei nomi	45

8.10.1	Altri livelli di caching	45
8.11	Cosa memorizza un DNS	45
8.12	Il formato dei messaggi	46
8.13	Note di sicurezza	46
8.14	Il servizio BIND (Berkeley Internet Name Domain)	47
8.14.1	Significato di alcuni parametri	47
8.15	Content Delivery Networks (CDN)	47
8.15.1	Distribuzione dei contenuti off-line	47
8.15.2	Replicazione e aggiornamento	47
8.15.3	Prelievo da parte del client	48
8.16	Architetture p2p	48
8.17	BitTorrent	48
8.17.1	Prelievo di chunk	48
8.17.2	Invio di chunk - <i>tit-for-tat</i>	48
9	Lezione del 27-10	49
9.1	P2P con directory centralizzata	49
9.2	P2P con directory decentralizzata	49
9.3	Il protocollo IP	49
9.3.1	Le funzioni del livello rete	49
9.3.2	Le funzioni di forwarding e routing	50
9.3.3	La relazione tra forwarding e routing	50
9.3.4	Le funzioni del livello rete	50
9.3.5	Lo strato di rete in Internet	51

9.3.6	IP	51
9.3.7	Il datagramma IP	51
9.3.8	L'header IP	52
9.3.9	Formato del pacchetto IP	53
10	Lezione del 03-11	54
10.1	Frammentazione e riassemblaggio IP	54
10.2	Opzioni	54
10.3	IP è consegna Best Effort	55
10.4	Indirizzi IP	55
10.5	ICANN (Internet Corporation for Assigned Names and Numbers)	55
10.6	Indirizzi delle interfacce e delle reti	56
10.7	Classi di indirizzi	56
10.7.1	Indirizzi di classe A	56
10.7.2	Indirizzi di classe B	57
10.7.3	Indirizzi di classe C	57
10.7.4	Indirizzi di classe D e E	58
10.8	Indirizzi IP <i>speciali</i>	58
10.9	Suddivisione degli indirizzi IP	59
10.10	Sottoreti	60
10.10.1	Come viene utilizzata una subnet mask	60
10.10.2	Netmask	61
10.10.3	Subnet e reti fisiche	61
10.10.4	Subnet - instradamento	61

10.11 Default Route	61
10.12 Tabelle di instradamento	62
11 Lezione del 08-11	62
11.1 ICMP (Internet Control Message Protocol)	62
12 Lezione del 10-11	63
12.1 Indirizzi IP ed Indirizzi di Livello 2	63
12.2 Reti Locali	63
12.3 Problema della risoluzione dell'indirizzo	63
12.4 Protocollo ARP	64
12.5 Formato del pacchetto ARP	64
12.6 ARP - scenari tipici	64
12.6.1 Primo Caso	65
12.6.2 Secondo Caso	66
12.7 BOOTP (BOOTstrap Protocol) e DHCP (Dynamic Host Configuration Protocol)	66
12.7.1 DHCP - scenario tipico	67
12.7.2 Interazione client-server via DHCP	67
12.8 NAT - Network Address Translation	67
12.9 Esempio	68
13 Lezione del 15-11	68
13.1 IPv4	68
13.2 IPv6	69
13.2.1 Features	69
13.2.2 Problemi affrontati in fase di progetto	69

13.2.3 Scrivere un IPv6	69
13.2.4 Caratteristiche generali	70
13.2.5 Unicast, Multicast e Anycast	70
13.2.6 Formato del pacchetto IPv6	71
13.2.7 Compatibilità	72
13.2.8 MTU in IPv6	72
13.2.9 Autoconfigurazione IPv	72
13.3 IP Multicasting	72
13.4 Trasmissione multicast	73
13.5 Il gruppo multicast	73
13.6 Session Announcement Protocol (SAP)	74
13.7 Gestione dei gruppi	75
13.8 Il multicast router	75
13.8.1 Funzionamento	75
14 Lezione del 17-11	75
14.1 Protocolli per il multicast in Internet	75
14.1.1 Il protocollo IGMP	75
14.1.2 IGMP - funzionalità	76
14.1.3 IGMP - implementazione	77
14.2 IP multicast - distribuzione sul LAN ethernet	77
14.2.1 Mapping Indirizzi IP multicast e Indirizzi MAC multicast	78
14.3 La rete MBone - Multicast BackBone	78
14.4 Reti di calcolatori e grafi	78

14.4.1 Parametri del processo decisionale	79
14.5 Il processo di routing	79
14.6 Il routing e la funzione di un router	79
14.7 Tabelle di routing - Esempio	80
14.8 Tecniche di routing	80
14.8.1 Routing by Network Address	80
14.8.2 Label Swapping	80
14.9 Routing - reti a circuiti virtuali	80
14.10 Routing - reti a datagramma	81
14.11 Tipologie di routing	81
15 Lezione del 22-11	81
15.1 Routing distribuito	81
15.2 Problematiche associate al routing	81
15.3 Scambio delle informazioni di update	82
15.4 Scelta dell'algoritmo di routing	82
15.5 Distance Vector	82
15.6 Link state	83
15.7 Distance Vector vs Link state	83
16 Lezione del 24-11	84
16.1 Algoritmo Distance Vector	84
16.1.1 Elaborazione	84
16.1.2 Caratteristiche	84
16.1.3 Esempio	85

16.1.4	Modifica dei costi dei collegamenti	86
16.1.5	Poisoned reverse	86
16.2	Algoritmo Link State	86
16.2.1	Il processo di update	86
16.2.2	LSP flooding	86
16.2.3	Trasmissione di un LSP e grafo della rete e LSP-DB - Esempio	87
16.2.4	Gestione degli LSP	87
16.2.5	Routing - decisioni	87
16.2.6	Architettura di un router Link State	88
16.2.7	Caratteristiche	88
16.3	Algoritmo di Dijkstra	88
16.3.1	Interpretazione	90
16.3.2	Esempio	90
16.3.3	Discussione	90
17	Lezione del 29-11	90
17.1	Internet e il routing gerarchico	90
17.2	Routing Interno ed Esterno	91
17.3	Tipi di AS	92
17.4	I gateway router	92
17.5	Il routing multicast e l'albero di copertura	92
17.6	Approcci	93
17.6.1	Group-shared tree	93
17.6.2	Source-based tree	93

17.6.3 Differenze tra gli approcci al routing multicast	93
18 Lezione del 01-12	94
18.1 Livello Trasporto	94
18.2 Livello trasporto e livello rete - differenze	94
18.3 Servizi e Protocolli del livello Trasporto	94
18.4 Multiplexing e Demultiplexing	94
18.4.1 Esempio	95
18.4.2 Connectionless demux	96
18.5 UDP (User Datagram Protocol)	96
18.5.1 Checksum UDP	97
18.6 Realizzare una trasmissione affidabile	97
18.6.1 Esempio	99
18.6.2 Un protocollo senza notifica negativa	99
18.7 Aumentare l'efficienza	99
18.7.1 Go Back N	100
18.7.2 Selective repeat	100
18.7.3 A confronto	100
19 Lezione del 06-12	101
19.1 TCP - Transmission Control Protocol	101
19.2 Struttura del segmento TCP	102
19.3 TCP - Protocol Data Unit	102
19.4 TCP - Principali opzioni	103
19.5 TCP - Caratteristiche	104

19.6 Numeri di sequenza e numeri di riscontro	104
19.7 Numeri di sequenza e ACK di TCP	104
19.8 Caratteristiche del riscontro cumulativo	104
19.9 TCP - Trasferimento dati affidabile	105
19.10 TCP - Eventi del mittente	105
20 Lezione del 13-12	105
20.1 Tecnologie per le LAN	105
20.2 Indirizzi IP e indirizzi LAN	105
20.3 Gli indirizzi MAC (Medium Acces Control)	106
20.4 Ethernet	106
20.4.1 Struttura della Frame Ethernet	107
20.4.2 CSMA/CD	107
20.4.3 Ethernet - 10Base2	107
20.4.4 Ethernet - 100BaseT	108
20.4.5 Gbit Ethernet	108
20.5 Interconnettere più LAN	108
20.6 Hub	108
20.6.1 Caratteristiche	108
20.6.2 Vantaggi	109
20.7 Bridge	109
20.7.1 Vantaggi	109
20.7.2 Bridge Spanning Tree	109
21 Lezione del 15-12	109

21.1 Socket	109
21.1.1 Iterazione tra applicazione e SO	110
21.1.2 Comunicazione locale	110
21.1.3 Comunicazione remota via TCP/IP	111
21.2 Il problema della connessione	111
21.2.1 Paradigma Client-Server	111
21.3 Il concetto di indirizzo	111
21.4 Server concorrente e iterativo	112
21.5 Paradigma di comunicazione Connection-Oriented	112
21.6 Paradigma di comunicazione Datagram	112
21.7 Paradigma di comunicazione Datagram e Connection-Oriented a confronto	113
21.8 Naming/Binding	113
21.9 Byte Stream e Datagram	113
21.10 Progettazione di un Server e Client TCP	113
21.11 Progettazione di un Server e Client UDP	114
21.12 Marshalling dei parametri	114
21.13 Gestione delle opzioni sulle socket	115
21.14 Raw socket	115
21.14.1 Pacchetti inviati	116
21.14.2 Accesso all'header IP	116

1 Introduzione al corso

1.1 Contatti

- ▶ Mail: giorgio@unina.it

1.2 Argomenti

Sostanzialmente si parlerà di Internet e internet (reti fisiche e le tecnologie SW che lo circondano).

1.3 Libri di testo

Computer Networking: A Top-Down Approach (consigliata la versione inglese) by James F. Kurose (7° edizione).

1.4 Modalità di esame

- ▶ Prova scritta selettiva al calcolatore;
- ▶ Prova Orale.

Schema di valutazione:

- ▶ Frequenza;
- ▶ Partecipazione al corso;
- ▶ Risultati prove.

Elaborato

- ▶ Su base volontaria;
- ▶ Di tipo pratico;
- ▶ In autonomia;
- ▶ Su argomenti indicati dal (concordato col) docente.

1.5 Link Utili

[Internet Society](#) (azienda no-profit che mostra l'intero ecosistema di internet)

2 Lezione del 29-09

2.1 Tecnologie digitali

- ▶ Programmabilità → capacità di cambiare il comportamento di una macchina non cambiando la macchina ma attraverso programmi;

- Comunicazione → capacità di inviare informazioni attraverso lo spazio.

In passato queste 2 realtà erano molto diverse, ma ad oggi sono entrambe fortemente connesse.

2.1.1 Comunicazione

Le comunicazioni più facili rendono possibile modelli organizzativi e di esercizio del potere completamente differenti dai classici modelli gerarchici, ad albero.

La comunicazione è oggi *multimediale, ubiqua, economica*.

3 Lezione del 04-10

3.1 Mainframe e super computer

Macchine molto costose che richiedevano un controllo in loco (i cosiddetti CED). Non presentavano inizialmente specifiche di interconnessione.

C'era la presenza di macchine remote, i cosiddetti terminali.

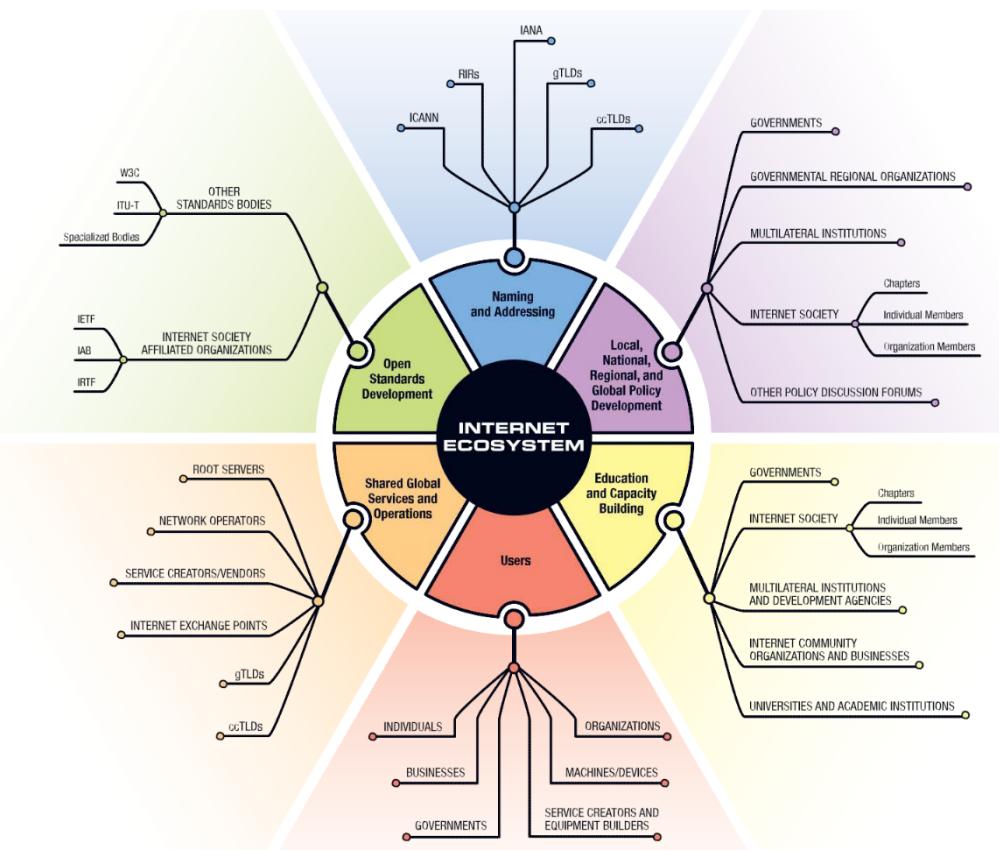
Si passa successivamente ai micro-processori che produce:

- I **mini computer**, che non erano altro che mainframe in scala minore (ricordiamo Digital);
 - ❖ Nasce in questo momento UNIX;

In questo momento si sviluppa dell'idea della connessione tra Host → Nascono quindi le Reti di calcolatori (prima locali).

3.2 L'ecosistema Internet

Internet → mix di infrastrutture; rete di reti (reti locali, geografiche).



Internet è un'incrocio di:

- ▶ Reti globali (WAN Area);
- ▶ Strutture Nazionali;
- ▶ infrastrutture Locali, campus, cittadine.

3.3 Comunicazione e commutazione

Il più semplice tipo di comunicazione è quello diretto, via cavo.

Occorre solo avere un mezzo fisico che consenta la propagazione di un segnale tra tx e rx.

Quando ciò non è possibile occorre avere degli intermediari.

% Commutazione di Circuito

In una rete a *commutazione di circuito*, gli host sono tutti direttamente connessi a uno dei commutatori.

Quando 2 host desiderano comunicare, la rete stabilisce una connessione **end-to-end** dedicata esclusivamente a loro

Nelle reti a commutazione di circuito, la capacità trasmissiva

Nella telefonia tradizionale, 2 coppie di conduttori venivano impegnate per ciascuna conversazione. Successivamente, le coppie fisiche sono state sostituiti da:

- ▶ TDM (Multiplazione a divisione di tempo) → porzioni di tempo;
- ▶ FDM (multiplazione a divisione di frequenza) → porzioni di banda.

3.4 Commutazione di pacchetto

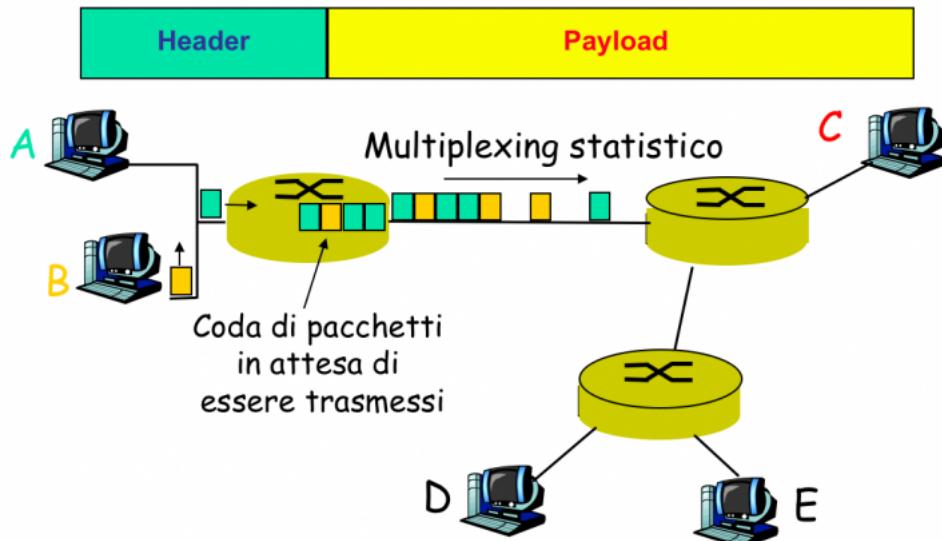
La natura discontinua della trasmissione di dati digitali può essere sfruttata per far sì che flussi di dati differenti possano condividere la stessa connessione, a patto di poterli distinguere.

Questo principio è alla base della tecnica detta *comunicazione di pacchetto* (packet switching)

% Packet Switching

Nel Packet switching ciascun flusso di dati è diviso in pacchetti, cioè 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*. Tra i più importanti ricordiamo:

- ▶ **Delay** → ritardo nella consegna dei pacchetti;
- ▶ **Throughput** → quantità di bit al secondo che la rete è in grado di trasferire tra 2 terminali;
- ▶ **Loss-Rate** → probabilità che un pacchetto non venga consegnato a destinazione;
- ▶ **Jitter** → Variazione temporale del ritardo.

Internet è best effort:

Accetto tutti, Accetto tutto, ma sono caazzi vostri

Giorgio Ventre - 10-04

3.5 Ritardo nelle reti a commutazioni di pacchetto

Determinato da:

- ▶ Tempo di elaborazione nel nodo;
- ▶ Tempo di trasmissione su ciascun link;
- ▶ Tempo di attesa nelle code del router;
- ▶ Tempo di propagazione sulle linee (**lunghezza linea/velocità del segnale**).

3.6 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.

3.7 Reti a circuiti virtuali

Ogni pacchetto appartenente ad una **conversazione/flusso dati** contiene il numero del circuito virtuale corrispondente.

Il circuito virtuale è stabilito prima della trasmissione dei dati.

I nodi devono conservare informazioni sui circuiti virtuali che li attraversano.

3.8 Datagrammi vs circuiti virtuali

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

4 Lezione del 11-10

4.1 Elementi costitutivi delle reti di calcolatori

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 3 modelli:

- ▶ Client-Server → il client invia una richiesta e il server risponde;
- ▶ Peer-to-Peer → le 2 entità comunicanti si scambiano informazioni in modo paritetico;
- ▶ Multicasting → Set di comunicazioni dove i dati trasmessi girano su un gruppo di target allo stesso tempo.

L'infrastruttura della rete è fatta di:

- ▶ Apparati tra loro interconnessi
 - ◊ hub, switch, bridge;
 - ◊ modem;
 - ◊ access point;
 - ◊ router;
 - ◊ ...
- ▶ Supporti trasmissivi che realizzano le interconnessioni
 - ◊ doppini in rame;
 - ◊ cavi coassiali;
 - ◊ Fibre ottiche;
 - ◊ Collegamenti radio punto-punto;
 - ◊ Collegamenti satellitari;
 - ◊ ...

L'infrastruttura della rete si può suddividere grossolanamente in:

- ▶ Reti di accesso → forniscono la connettività agli end-System;
 - ◊ Utilizzano svariate tecnologie
 - Rete telefonica tradizionale;
 - xDSL;
 - Ethernet;
 - FTTH/C;
 - WDM;
 - CATV;
 - WLAN;
 - FWA;
 - Bluetooth;
 - 4G/5G.
- ▶ Reti Backbone → forniscono la connettività alle reti di accesso;
 - ◊ Utilizzano svariate tecnologie
 - SONET/SDH;
 - WDM;
 - MPLS;
 - Metro Ethernet;
 - ATM;
 - Satellite;
 - Ponti Radio a RF.

4.2 Le reti di calcolatori

Suddivisione:

- ▶ LAN → reti di accesso;
- ▶ MAN → reti di accesso;
- ▶ WAN → reti che interconnettono le infrastrutture (rete Backbone).

4.2.1 Caratteristiche LAN

- ▶ Velocità trasmissiva elevata;
- ▶ In teoria distanze ridotte;
- ▶ Non attraversano suolo pubblico;
- ▶ Conformità →
 - ◊ Conformi a standard emessi da ISO/IEEE/ANSI;
 - ◊ Conformi a standard di cablaggio ed a normativa di sicurezza;
 - ◊ Non conformi a standard CCITT/ITU;
- ▶ Scopo → rendere virtualmente locali le risorse distribuite in rete;
- ▶ Tecnologie eterogenee, ma semplici.

4.2.2 Caratteristiche MAN

- ▶ Velocità trasmissiva elevata;
- ▶ Installazioni in ambito urbano e regionale;
- ▶ Vendibilità, capillarità;
- ▶ Conformità →
 - ◊ Conformi sia a standard CCITT/ITU e ISO/IEEE sia anche mondo LAN;
 - ◊ Problematiche analoghe a WAN;
 - ◊ Sviluppate spesso con accordi con enti locali;
- ▶ Fibra come mezzo trasmissivo tipico;
- ▶ Forte eterogeneità.

4.2.3 Caratteristiche WAN

- ▶ Velocità trasmissiva (praticamente uguale a LAN);
- ▶ Installazioni in ambito regionale ed oltre;
- ▶ Affidabilità, Vendibilità, capillarità;

- ▶ Conformità →
 - ◊ Conformi standard CCITT;
 - ◊ Conformi ad agreement di servizio (concessioni);
 - ◊ Conformi ad agreement di interconnessione;
 - ◊ Sottoposti ad azioni di auditing (agenzie e autorità);
- ▶ Mezzi trasmissivi →
 - ◊ Una volta gli stessi per la telefonia convenzionale;
 - ◊ Oggi reti integrate e complesse (problema del management).

4.3 Gestire la complessità

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

- ▶ Ricezione e Trasmissione fisica;
- ▶ Controllo degli errori;
- ▶ Controllo di flussi e di congestione;
- ▶ Instradamento;
- ▶ Conversione dei dati;
- ▶ Crittografia e sicurezza;
- ▶ Sincronizzazione.

Un approccio logico è quello di analizzare tali problematiche singolarmente attraverso il *Divide et Impera*.

Nelle reti di calcolatori questo ha condotto a modelli *a strati*.

4.4 Modelli a strati

La suddivisione delle funzionalità secondo un modello a strati agevola la gestione della complessità. Ciascun livello:

- ▶ È responsabile di un sottoinsieme definito e limitato di compiti;
- ▶ Funziona in maniera lascamente accoppiata con gli altri;
- ▶ Interagisce solo con gli strati immediatamente superiori ed inferiori;
- ▶ Fa affidamento sui servizi forniti dallo strato immediatamente inferiore;
- ▶ Fornisce servizi allo strato immediatamente superiore.

Alcuni strati sono realizzati in SW, altri in HW.

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.

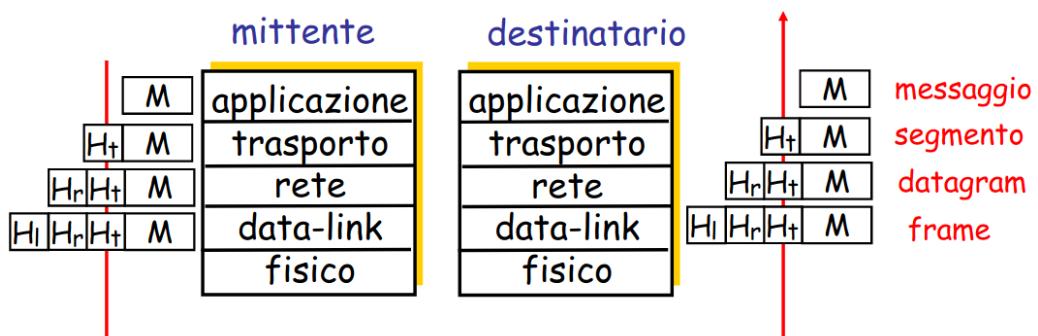
Svantaggi → l'eccessivo numero di strati può portare ad inefficienza

4.4.1 Modelli a strati - interfacce e protocolli

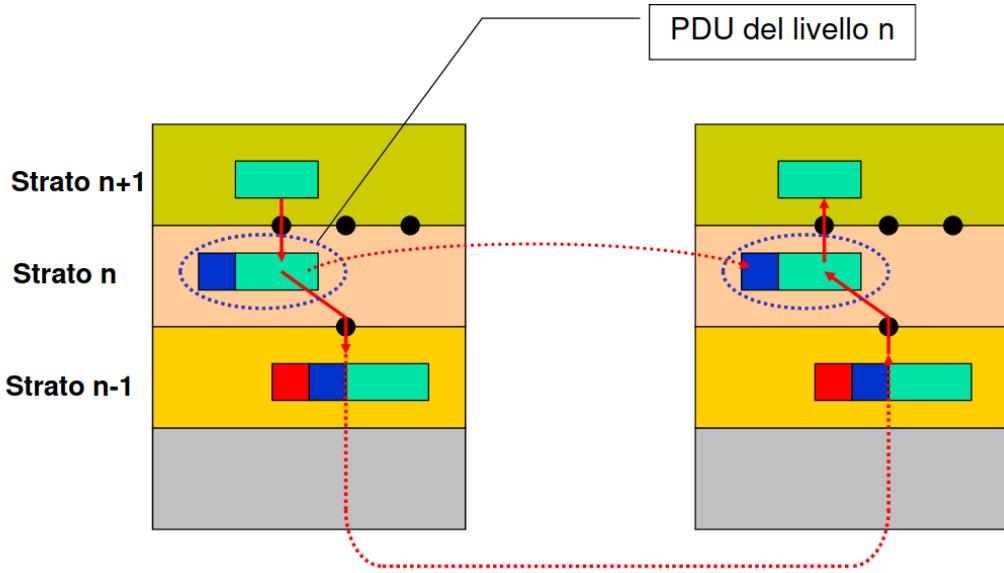
All'interno di ciascun dispositivo di rete, lo scambio di informazioni tra 2 strati adiacenti avviene attraverso una interfaccia, che definisce i servizi dallo strato inferiore allo strato superiore → lo strato **n-esimo** di un dispositivo comunica con lo strato **n-esimo** di un'altra entità secondo un *protocollo* assegnato.

4.5 Imbustamento dei messaggi

- ▶ In trasmissione, ogni strato antepone una intestazione (header) al messaggio ricevuto dallo strato soprastante.
- ▶ L'insieme **messaggio+header** viene passato allo strato sottostante.
- ▶ A destinazione il messaggio risale la pila.
- ▶ In ricezione, ad ogni strato l'header viene rimosso



4.6 Protocol Data Unit (PDU)



4.7 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.

4.8 Il modello OSI (Open System Interconnection)

È un modello a strati su 7 livelli:

- ▶ Applicazione;
- ▶ Presentazione;
- ▶ Sessione;
- ▶ Trasporto;
- ▶ Rete;
- ▶ Data link;
- ▶ Fisico.

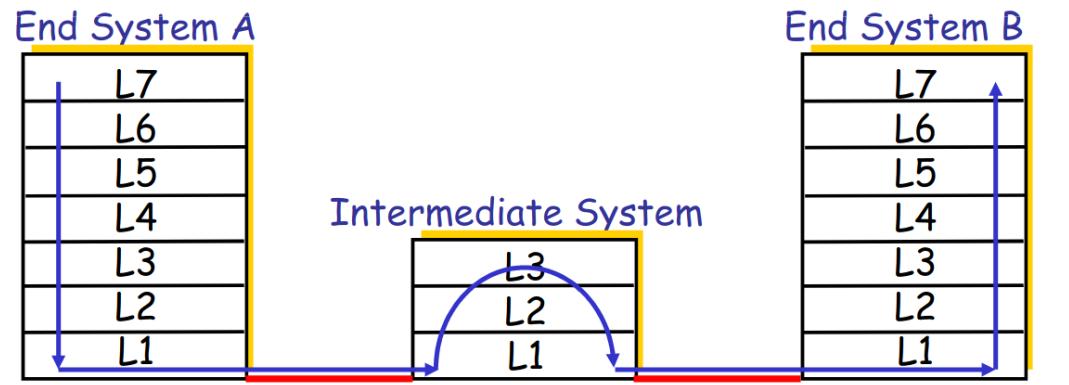
%

Il modello OSI non è risultato vincente, a causa della sua eccessiva complessità

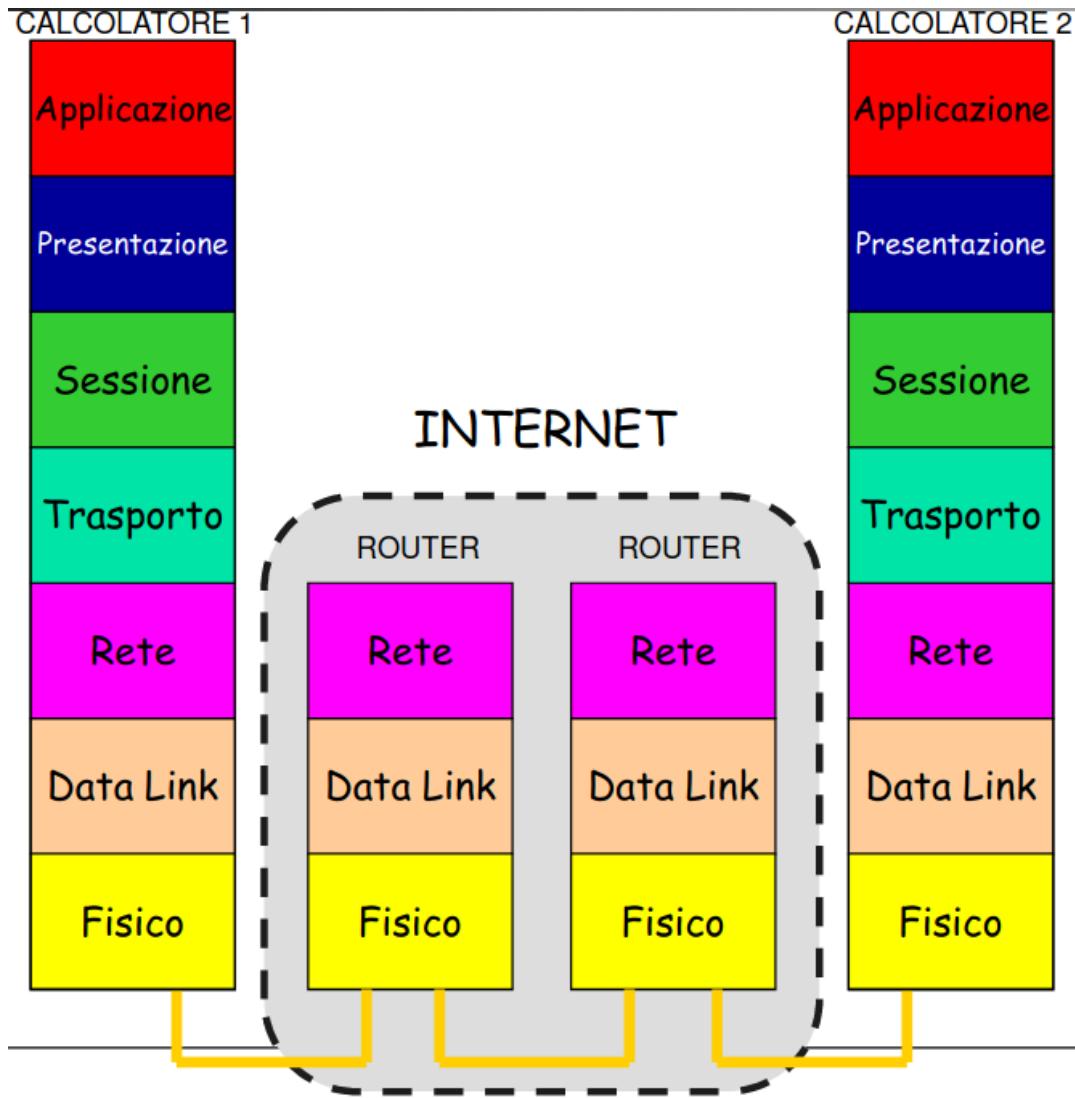
4.9 Dispositivi di rete e livelli

L'intera pila di livelli è realizzata negli end system. 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**



4.10 Approccio a livelli per la risoluzione dei problemi



(NB: I 2 calcolatori presentano una struttura OSI, per Internet vanno esclusi Presentazione e Sessione)

- ▶ Applicazione
 - ◊ Fornitura dei servizi e generica elaborazione;
- ▶ Presentazione
 - ◊ Codifica uniforme dei dati per il colloquio tra entità eterogenee;
- ▶ Sessione
 - ◊ Mantenimento informazioni in un dialogo persistente;

- ▶ Trasporto
 - ◊ Controllo di errore end-to-end;
 - ◊ Controllo di flusso;
 - ◊ Multiplexing e demultiplexing dei flussi;
- ▶ Rete
 - ◊ Instradamento dei pacchetti;
 - ◊ Trattamento di eterogeneità nella rete;
 - ◊ Frammentazione;
- ▶ Data Link
 - ◊ Composizione dei bit in frames;
 - ◊ Controllo dell'errore point-to-point;
 - ◊ Controllo di flusso;
 - ◊ Accesso al mezzo fisico;
- ▶ Fisico
 - ◊ Trasmissione e ricezione di singoli bit;
 - ◊ Conversione digitale/analogico

4.10.1 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 trasmissivi;
- ▶ Tipi di connettori

Il livello fisico è nel dominio dell'ingegneria elettronica → descrizione **elettrico/meccanica** dell'interfaccia.

4.10.2 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* e li trasmette sequenzialmente. Verifica la presenza di errori di trasmissione aggiungendo delle informazioni aggiuntive di controllo → *Frame Control Sequence* (FCS).

Può gestire meccanismi di correzione di errori tramite ritrasmissione.

4.10.3 Livello 3 - Rete

Gestisce l'instradamento dei messaggi. Determina quali sistemi intermedi devono essere attraversati da un messaggio per giungere a destinazione.

Gestisce quindi delle tabelle di instradamento per ottimizzare il traffico sulla rete.

4.10.4 Livello 4 - Trasporto

Fornisce servizi per il trasferimento dei dati end-to-end, indipendentemente dalla rete sottostante.

4.10.5 Livello 5 - Sessione

Responsabile dell'organizzazione del dialogo e della sincronizzazione tra 2 programmi applicativi e del conseguente scambio di dati

4.10.6 Livello 6 - Presentazione

Gestisce la sintassi dell'informazione da trasferire.

4.10.7 Livello 7 - Applicazione

Livello dei programmi applicativi, cioè di quei programmi appartenenti al SO o scritti dagli utenti, attraverso i quali l'utente finale utilizza la rete.

5 Lezione del 12-10

5.1 Lo stack TCP/IP

NB per ING → lezione importante (Botta presiede).

% Protocollo

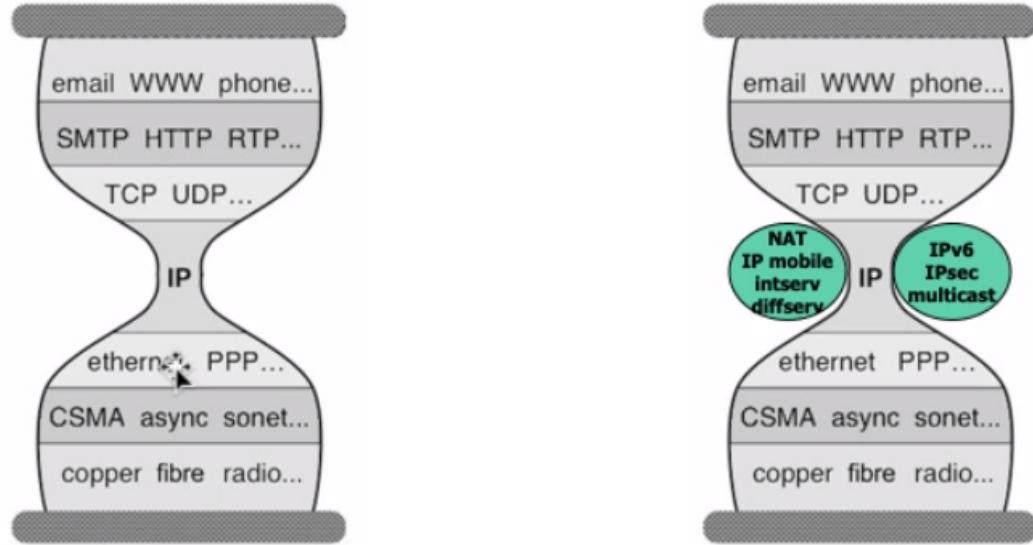
Insieme di regole formalmente descritte che definiscono le modalità di comunicazione tra due o più entità.

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

Il modello prende il nome da 2 protocolli fondamentali:

- ▶ TCP → transmission Control Protocol, di livello Trasporto;
- ▶ IP → Internet Protocol, di livello Rete

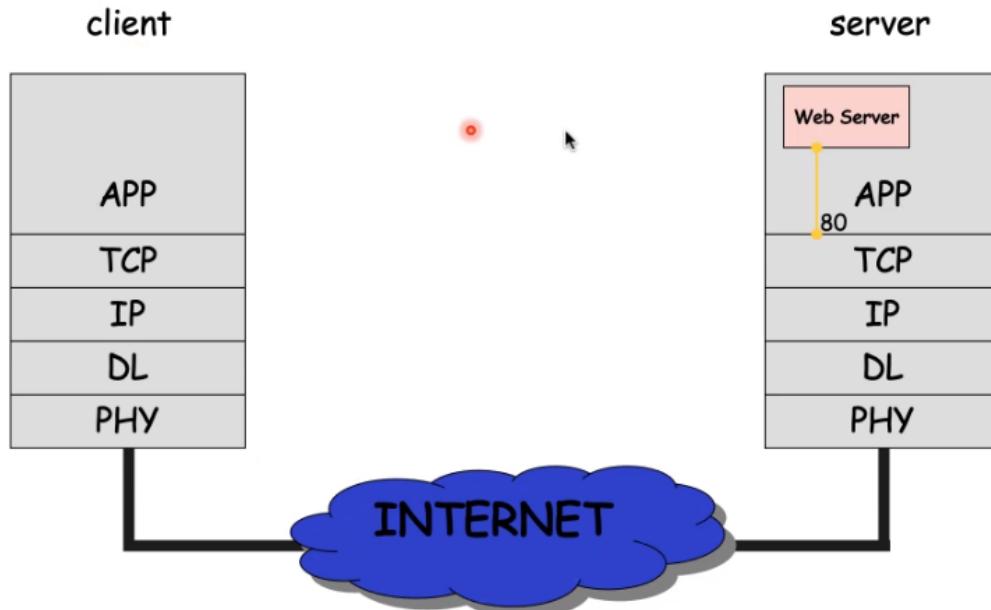
Lo stack di protocolli di Internet come una clessidra.



5.2 Applicazioni e protocolli dello strato applicativo

Quando parliamo di livello applicazione parliamo di comunicazione tra processi

esempio di Interazione Client-Server



5.3 Strato dell'applicazione

- ▶ Comunicazione tra processi;

- ▶ Cosa definisce un protocollo dello strato di applicazione
 - ◊ Tipo di messaggi scambiati;
 - ◊ Sintassi dei messaggi;
 - ◊ Semantica dei campi
- ▶ Applicazioni lato client e lato server

5.4 Comunicazione tra processi

API → Application Programming Interface.

Definisce l'interfaccia tra il livello applicativo e il livello di trasporto.

Socket → Internet API (2 processi comunicano mandando dati in un socket, e leggendo dati dal socket)

Differenza tra libreria e API:

- ▶ La libreria diventa staticamente o dinamicamente legata al programma;
- ▶ Le API sono chiamate esterne che non diventano legate con il programma (potenzialmente possono trovarsi in qualsiasi parte del mondo)

Due processi per identificarsi tra loro possono usare:

- ▶ Indirizzo IP dell'host che esegue l'altro processo;
- ▶ *Numeri di porta* che permette all'host ricevente di sapere a quale processo locale il messaggio è destinato.

5.5 Livello trasporti - TCP e UDP

Protocolli molto diversi, dovuti a diverse richieste da parte delle applicazioni.

5.6 Requisiti di alcune applicazioni

Application	Data Loss	Bandwidth	Time Sensitive
file transfer	no loss	elastic	no
e-mail	no loss	elastic	no
Web document	no loss	elastic	no
real-time audio/video	loss-tolerant	audio : 5Kb-1Mb video : =10Kb-5Mb	yes, 100's msec
stored audio/video	loss-tolerant	same as above	yes, few secs
Interactive games	loss-tolerant	few Kbps up	yes, 100's msec
Instant messaging	no loss	elastic	yes and no

6 Lezione del 18-10

6.1 Internet - Origine ed Evoluzione

Internet nasce come interconnessione di diverse reti di carattere sperimentale.

[Kleinrock](#), attraverso la teoria delle code mostra la fattibilità delle reti a commutazione di pacchetto (1961).

ARPANET → Rete sperimentale basata su IMP (Interface Message Processors) con meccanismi **Store-and-Forward** (1967/1969).

Approccio Datagram per garantire la sopravvivenza in caso di eliminazione di nodi e linee

- ▶ 1974 → V. Cerf e R. Kahn progettano la suite TCP/IP.
- ▶ 1984 → La parte militare di ARPANET si separa;
- ▶ 1986 → La NFS finanzia lo sviluppo di una rete basata su TCP/IP (NSFNET);
- ▶ 1989 → ARPANET viene smantellata;
- ▶ 1990 → NSF smette di finanziare la rete e cede la struttura ad una organizzazione non-profit
 - ◊ Nasce ANS (Advanced Network and Services);
 - ◊ Collegamenti da T1 a T3 (~45 Mbps).
- ▶ 1995 → NSFNET viene smantellata
 - ◊ Nascono i NAP (Network Acces Point) con lo scopo di interconnettere varie reti commerciali.

6.2 Internet - Architettura della rete

- ▶ Architettura completa, dai servizi alle applicazioni;
- ▶ Di dominio pubblico;
- ▶ Gestita da specifici organismi;
- ▶ I protocolli in uso sono standardizzati mediante documenti approvati dall'IETF (le RFC → Request for Comments; scritte in ASCII);
- ▶ Opera su tecnologie di rete standard e non
 - ◊ SLIP, PPP, Dialup;
 - ◊ LAN 802.X, FDDI;
 - ◊ X.25, FR, ATM, SMDS.

La rete è progettata secondo un modello a **datagram**; l'informazione viaggia in pacchetti 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.

NB: 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.

6.3 Struttura di Internet

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

Gli ISP sono collegati tra loro secondo una struttura gerarchica:

- ▶ ISP locali;
- ▶ ISP nazionali

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

Gli NBP sono tra loro collegati in punti di interscambio, detti NAP.

Tra gli ISP globali (Tier One) ricordiamo:

- ▶ UUNet;
- ▶ Sparkle (Telecom Italia);
- ▶ Sprint;
- ▶ BBN/GTE.

Il livello successivo (Tier Two):

- ▶ Non è critico come One;
- ▶ Richiede una connessione con i Tier One;
- ▶ Richiede una connessione con i Tier Two (per uno scambio più logico di pacchetti → si diminuisce il viaggio che ogni pacchetto deve compiere).

Dalla sorgente alla destinazione, un pacchetto attraversa diverse reti (Tier 1, Tier2 e local ISP).

6.4 La rete GAAR

L'accesso a Internet per le Università è gestito dal GARR (Gruppo Armonizzazione Reti di Ricerca). È un Tier 2.

6.5 Chi regolamenta Internet

La standardizzazione dei protocolli in uso su Internet è fatta dall'IETF.

L'assegnazione degli indirizzi e dei nomi di dominio è oggi supervisionata dall'ICANN, che ha preso il posto della IANA.

7 Lezione del 20-10

7.1 Strato dell'applicazione

- ▶ Comunicazione tra processi

Cosa definisce un protocollo dello strato di applicazione:

- ▶ Tipo di messaggi scambiati;
- ▶ Sintassi dei messaggi;
- ▶ Semantica dei campi;
- ▶ Regole di trasmissione.

7.2 HTTP

Per effettuare richieste, specificare cosa si richiede, rispondere alle richieste, si rende necessario un opportuno protocollo → Su internet si utilizza il protocollo **HTTP**.

È un protocollo testuale → messaggi come sequenze di byte (che identifica un carattere secondo la tabella ASCII).

In certi casi, il payload dei messaggi può essere comunque anche in formato binario

Si basa su TCP.

1. Il client apre un socket verso il porto 80 (se non diversamente specificato) dal server;
2. Il server accetta la connessione;
3. Il client manda una richiesta;
4. Il server risponde e chiude la connessione.

Il protocollo è **definito/descritto** nelle RFC:

- ▶ http 1.0 → RFC 1945;
- ▶ http 0.0 → RFC 2068, RFC2616;

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

7.3 Uniform Resource Locator (URL)

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 80);
- ▶ Path identifica la risorsa sul server;
- ▶ #fragment identifica un punto preciso all'interno di un oggetto;
- ▶ ?query è usato per passare informazioni dal client al server.

7.4 Connessioni persistenti e non persistenti

Non persistente	Persistente
HTTP/1.0 Il server analizza una richiesta, la server e chiude la connessione	HTTP/1.1 Sulla stessa connessione il server analizza tutte le richieste e le serve
2 Round Trip Time (RTT) per ciascuna richiesta	Il client riceve la pagina iniziale e invia subito tutte le altre richieste
Ogni richiesta subisce lo slow-start TCP	Si hanno meno RTT ed un solo slow-start
	Esiste anche una versione con parallelismo (with pipeling)

7.5 Messaggio HTTP request



7.6 Codici di stato - esempi

- ▶ 200 (OK)
 - ◊ Successo → l'oggetto richiesto si trova più avanti nel messaggio;

- ▶ 301 (Moved Permanently)
 - ◊ L'oggetto richiesto è stato spostato;
 - ◊ Il nuovo indirizzo è specificato più avanti nel messaggio;
- ▶ 400 (Bad Request)
 - ◊ Richiesta incomprensibile al server;
- ▶ 404 (Not Found)
 - ◊ Il documento non è presente sul server;
- ▶ 505 (HTTP Version Not Supported).

7.7 HTTP per il trasferimento file

Tipicamente descritta da un file testuale in formato HTML.

La pagina è identificata mediante un indirizzo, detto URL.

Un file HTML può contenere riferimenti ad altri oggetti che arricchiscono la pagina con elementi grafici.

Ciascun oggetto è identificato dal proprio URL (possono trovarsi anche su web server diversi).

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

7.8 Lo scambio di messaggi

Sono sostanzialmente:

- ▶ Il metodo GET;
- ▶ Il metodo HEAD;
- ▶ Il metodo POST;
- ▶ Il metodo PUT;
- ▶ La response

7.8.1 Il metodo GET

Viene usato per richiedere una risorsa ad un server → viene attivato cliccando su di 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;
- ▶ Parziale → si richiede una sottoparte di una risorsa memorizzata.

7.8.2 Il metodo HEAD

Simile al GET, ma il server deve rispondere soltanto con gli HEADER relativi, senza il corpo.

Viene utilizzato per:

- ▶ Validità di un URI;
- ▶ L'accessibilità di un URI;
- ▶ Coerenza di cache di un URI.

7.8.3 Il metodo POST

Serve a trasmettere delle informazioni dal client al server, ma senza la creazione di una nuova risorsa.

Il server può rispondere in 3 modi diversi:

- ▶ 200 (OK);
- ▶ 201 (Created);
- ▶ 204 (No content).

7.8.4 Il metodo PUT

Serve per trasmettere delle informazioni dal client al server, creando o sostituendo la risorsa specifica.

7.9 Gli header generali

Si applicano solo al messaggio trasmesso e si applicano sia ad una richiesta che ad una risposta:

- ▶ Date;
- ▶ MIME-Version → la versione MIME usata per la trasmissione;
- ▶ Transfer-Encoding → tipo di formato di codifica usato per la trasmissione;
- ▶ Cache-Control → il tipo di meccanismo di caching richiesto o suggerito per la risorsa;
- ▶ Connection → tipo di connessione da usare
 - ◊ Keep-Alive;
 - ◊ Close.
- ▶ Via → usato da proxy e gateway.

7.10 I cookies

È una breve informazione scambiata tra il server ed il client.

Tramite un cookie il client mantiene lo stato di precedenti connessioni, e lo manda al server di pertinenza ogni volta che richiede un documento.

7.11 Cookies - header specifici

Il metodo dei cookies definisce 2 nuovi header:

- ▶ Set-Cookie → header della risposta
- ▶ Cookie → header della richiesta.

7.12 Web caching

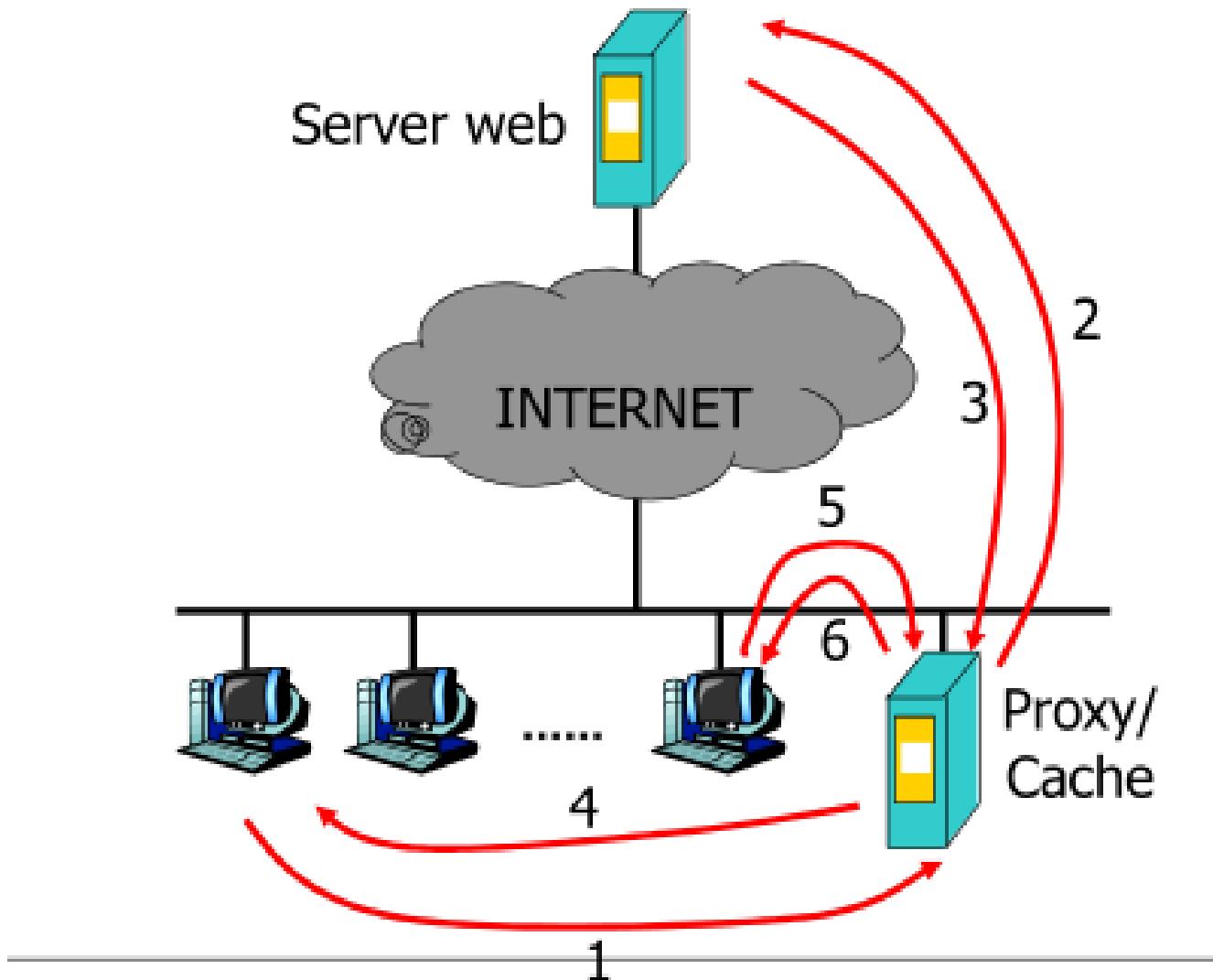
Si parla genericamente di Web caching quando le richieste di un determinato client non raggiungono il Web Server, ma vengono intercettate da una cache.

Tipicamente un certo numero di client di una stessa rete condivide una stessa cache web, posta nelle loro prossimità.

Se l'oggetto richiesto non è presente nella cache, questa lo richiede in vece del client conservandone una copia per eventuali richieste rapidamente.

2 tipi di interazioni HTTP → *client-cache* e *cache-server*.

7.13 Transazioni web con caching



7.14 Gestione della coerenza

La copia in cache deve essere aggiornata per mantenersi uguale all'originale.

HTTP fornisce 2 meccanismi per la gestione della coerenza:

- ▶ TTL (Time To live) → il server quando fornisce un oggetto dice anche quando scade (header *Expires*)
 - ❖ Quando TTL diventa < 0, 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**).

7.15 Protocollo FTP (File Transfer Protocol)

Attraverso questo protocollo è possibile trasferire uno o più file di qualsiasi tipo tra 2 macchine.

Lavora utilizzando 2 connessioni:

- ▶ Una dati
 - ◊ File che fluiscono dal client al server o viceversa;
- ▶ Una controllo (*out of band*)
 - ◊ Scambio di comandi, messaggi di risposta tra il client e il server.

Usa il modello **client/server**:

- ▶ Client → entità che da luogo al trasferimento (sia in un senso che nell'altro);
- ▶ Server → entità remota che è in continua attesa di connessioni FTP da parte di altre entità.

Di default usa la porta 21.

Un server ftp mantiene uno stato:

- ▶ La directory corrente;
- ▶ I dati dell'autenticazione.

8 Lezione del 25-10

8.1 Il protocollo SMTP

La posta elettronica sfrutta degli intermediari per il trasferimento delle e-mail tra le parti.

Per trasferire messaggi di posta elettronica si utilizza un apposito protocollo → *Simple Mail Transfer Protocol*.

8.1.1 Entità in gioco

Le entità in gioco sono 3:

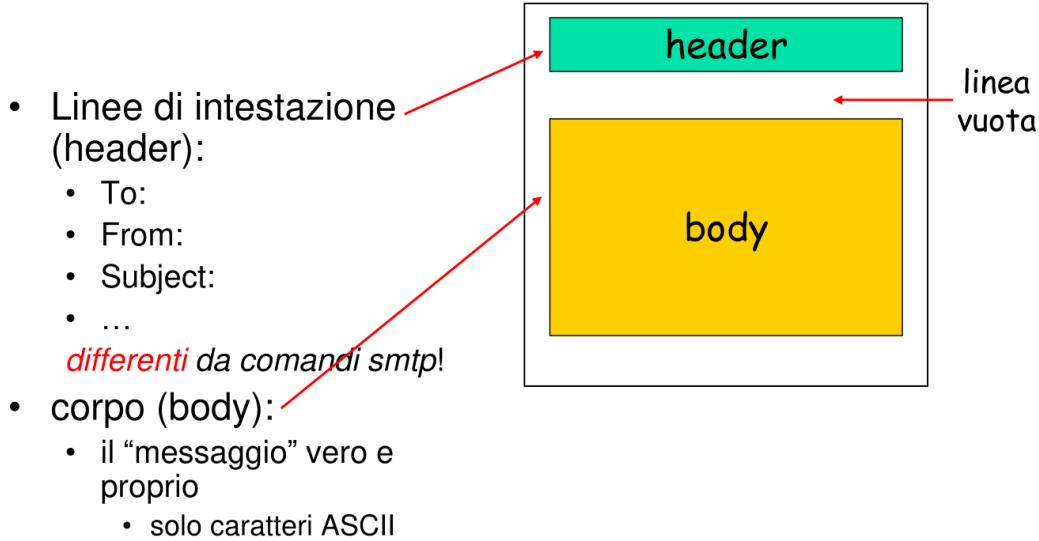
- ▶ User agent
 - ◊ Detto anche mail reader;
 - ◊ Composizione, modifica, lettura di messaggi;

- ◊ 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 server
 - ◊ Mailbox contenente messaggi in entrata (non letti) per l'utente;
 - ◊ Coda dei messaggi in uscita contenente i messaggi non ancora recapitati;
 - ◊ Protocollo SMTP per l'interazione tra 2 mail server
 - *Client* → mail server mittente;
 - *Server* → mail server destinatario;
 - ◊ Funge in momenti diversi da client o da server a seconda del ruolo che ricopre nello scambio del messaggio.
- ▶ Protocollo SMTP.

8.1.2 Caratteristiche

- ▶ Usa il protocollo TCP (porta 25) per consegnare in modo affidabile messaggi dal client al server.
- ▶ Trasferimento diretto dal server mittente al server destinatario;
- ▶ 3 fasi durante il trasferimento via SMTP
 - ◊ Handshaking;
 - ◊ Trasferimento del messaggio;
 - ◊ Chiusura della connessione.
- ▶ Interazione comando/risposta
 - ◊ 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 di contenuto, sia codificato in caratteri ASCII a 7 bit;
- ▶ Alcune combinazioni di caratteri non sono ammesse. Qualora siano presenti il messaggio deve essere opportunamente codificato.
- ▶ SMTP usa CRLF . CRLF per determinare la fine di un messaggio.

8.1.3 Formato del messaggio SMTP



8.2 L'estensione MIME (Multipurpose Internet Mail Extensions)

Righe aggiuntive nell'intestazione informano della presenza di un body MIME.

8.2.1 Esempi di tipi MIME

- ▶ Text
 - ◊ Sottotipi → plain, html;
- ▶ Video
 - ◊ Sottotipi → mpeg, quicktime;
- ▶ Image
 - ◊ Sottotipi → jpeg, gif;
- ▶ Audio
 - ◊ Sottotipi → basic, 32kadpcm;
- ▶ Application
 - ◊ Altri dati che devono essere processati da specifiche applicazioni;
 - ◊ Sottotipi → msword, octet-stream.

8.3 POP3 (Post Office Protocol)

Per accedere alla propria casella di posta elettronica per leggere i propri messaggi è necessario un ulteriore protocollo → POP.

Si tratta sempre di un protocollo **client/server**:

- ▶ Lo user agent ancora una volta gioca il ruolo di client POP;
- ▶ Il mail server gioca il ruolo di server POP.

8.4 La catena dei protocolli per la posta

User agent (*SMTP*) Mail server mittente (*SMTP*) Mail server destinatario (*SMTP*) User agent

8.5 Domain Name System (DNS)

In internet le macchine sono identificati con indirizzi numerici. Questi sono molto più gestibili dalle macchine rispetto a nomi simbolici (più semplici per l'uomo).

Per non 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 (<https://www.grid.unina.it>) un indirizzo IP, permettendo così di raggiungere la macchina.

Esso si basa sullo scambio di messaggi UDP sulla porta 53 (anche TCP può essere usato, a richiesta).

8.5.1 Altre funzionalità offerte

- ▶ Alias degli hostname;
- ▶ Alias dei server di posta;
- ▶ 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 all'intero pool, senza che gli utenti si accorgano di nulla.

8.5.2 DNS distribuito

È chiaro che un unico DNS centralizzato sarebbe fortemente sconveniente per una serie di fattori (Single Point Of Failure, Volume di traffico, Database distante, Manutenzione).

Pertanto si distribuisce le informazioni sul territorio → si distribuisce anche la responsabilità di raccogliere, gestire, aggiornare e divulgare le informazioni che lo riguardano.

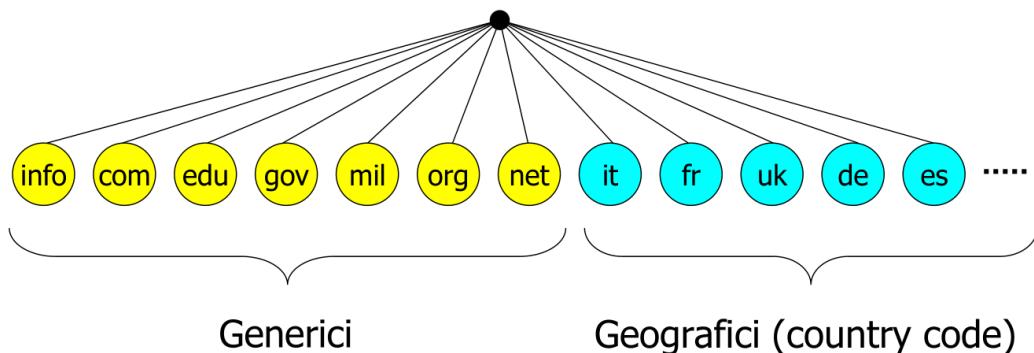
In particolare l'approccio è di tipo gerarchico:

- ▶ Gli elementi più alti nella gerarchia contengono molte informazioni non dettagliate;
- ▶ Gli elementi più bassi nella gerarchia contengono poche informazioni dettagliate.

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

8.6 I top-leve domain server (TLD)

Questi server si occupano dei domini di alto livello (generici e geografici). Domini storici:



8.7 Tipologie di server DNS (locale)

Local Name Server:

- ▶ Non appartengono strettamente alla gerarchia di server;
- ▶ Ciascun ente 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);
- ▶ 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 in precedenza.

8.8 Tipologie di server DNS (Root)

Root Name Server:

- ▶ 13 root server logici in internet (da A a M) i cui indirizzi IP sono ben noti alla comunità
 - ◊ In realtà si tratta di 375 diversi server fisici (vedi [qui](#));
- ▶ Ad essi si riferiscono i Local Name Server che non possono soddisfare immediatamente una richiesta di risoluzione;
- ▶ Il Local Name Server si comporta come client DNS ed invia una richiesta di risoluzione al Root Name Server.

8.9 Tipologie di server DNS (Authoritative)

Authoritative Name Server:

- ▶ È un server dei nomi capace di risolvere tutti i nomi all'interno di un determinato **dominio/organizzazione**;
- ▶ 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.

8.10 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 tempo (circa un giorno), le associazioni vengono eliminate dalla cache.

8.10.1 Altri livelli di caching

- ▶ Applicazione (es web browser);
- ▶ SO (Linux → dnsmasq).

8.11 Cosa memorizza un DNS

Resource records (RR) → Formato composto da *Nome, Valore, Tipo, TTL*.

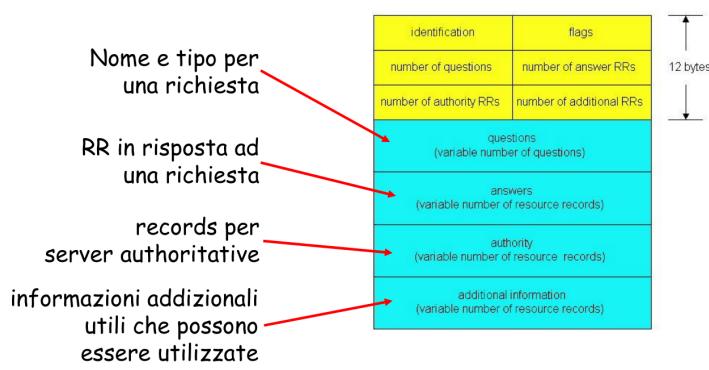
TTL → tempo di vita residuo di un record scaduto il quale viene eliminato dalla cache.

Il significato di *Nome* e *Valore* dipende da *Tipo*:

- ▶ Tipo A
 - ◊ nome → **hostname**;
 - ◊ valore → **indirizzo IP**;
- ▶ Tipo NS
 - ◊ nome → **dominio**;

- ◊ valore → indirizzo IP dell'Authoritative NS;
- Tipo CNAME
- ◊ nome → alias per il nome canonico (reale);
 - ◊ valore → nome canonico;
- Tipo MX
- ◊ nome → dominio di posta;
 - ◊ valore → nome dell'host mailserver associato a nome.

8.12 Il formato dei messaggi



Header del messaggio:

8.13 Note di sicurezza

Il protocollo DNS non offre autenticazione, né integrità, né riservatezza:

- Chiunque può leggere le richieste;
- Chiunque può rispondere al posto del vero DNS resolver;
- Chiunque può alterare in volo le **richieste/risposte**;

Soluzioni:

- Per l'autenticazione ed integrità → **DNSSec** (Nuovi RR, vedi [qui](#));
- Per la riservatezza → al 2016 in discussione **DNS-PRIve WG** (vedi [qui](#))

8.14 Il servizio BIND (Berkeley Internet Name Domain)

È una implementazione dei protocolli Domain Name System (DNS).

Liberamente re-distribuibile.

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 sistema Unix-lire.

8.14.1 Significato di alcuni parametri

- ▶ **Serial** → numero seriale progressivo utilizzato per rilevare aggiornamenti del file. Di solito usa il formato aaaammggxx;
- ▶ **Refresh** → intervallo in secondi tra 2 successivi prelievi del file di zone da parte di un DNS server;
- ▶ **Retry** → intervallo in secondi tra tentativi successivi di recuperare una zona in caso di fallimento;
- ▶ **Expire** → tempo in secondi che deve trascorrere per ritenere scadute le informazioni di una zona che non si riesce ad aggiornare;
- ▶ **Minimum TTL** → tempo di durata di default delle singole entry del file di zona.

8.15 Content Delivery Networks (CDN)

8.15.1 Distribuzione dei contenuti off-line

Contenuti da un singolo provider verso una molteplicità di utente → Problematiche legate al loss, delay...

Soluzione → Replicare il contenuto

- ▶ Su quanti più nodi possibile su internet;
- ▶ Vicino a chi ne fa richiesta;
- ▶ Tipicamente nelle reti di accesso, presso la rete dell'ISP.

8.15.2 Replicazione e aggiornamento

Il customer della CDN è il content provider.

La CDN replica il contenuto del customer sui propri CDN server.

Quando il provider aggiorna il contenuto, la CDN aggiorna i propri server.

8.15.3 Prelievo da parte del client

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 generato la query;
- ▶ Si usa la mappa per la scelta del server CDN più vicino.

8.16 Architetture p2p

Non esistono server sempre connessi (*always-on server*).

End-system (*peer*) comunicano direttamente. Questi sono connessi ad intermittenza e cambiano il proprio IP.

8.17 BitTorrent

Composto da:

- ▶ Tracker → tiene traccia dei peer che compongono un *torrent*;
- ▶ *Torrent* → gruppo di peer che si scambiano porzioni di uno stesso file.

Il file è diviso in *chunk* di 256KB.

Quando un peer si aggiunge ad un torrent:

1. Si registra presso il tracker per avere la lista dei peer e si connette ad un sottoinsieme di tali peer;
2. Non possiede chunk, ma ne accumulerà nel tempo;
3. Durante il download, il peer esegue l'upload di chunk verso altri peer;
4. I peer possono attivarsi e disattivarsi dinamicamente;
5. Una volta scaricato l'intero file, il peer può (egoisticamente) abbandonare, o (altruisticamente) restare nel *torrent*.

8.17.1 Prelievo di chunk

- ▶ 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 → tecnica del *rarest first*.

8.17.2 Invio di chunk - ***tit-for-tat***

L'idea di fondo è quella di dare priorità a chi fornisce dati al rate più alto.

Un peer invia chunk ai 4 neighbors attualmente più veloci (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;
- ▶ *optimistically unchoke*.

9 Lezione del 27-10

9.1 P2P con directory centralizzata

Quando un peer si connette alla rete si collega ad un server centralizzato fornendo:

- ▶ Il proprio indirizzo IP;
- ▶ Il nome degli oggetti resi disponibili per la condivisione.

Il server in questo modo raccoglie le info sui peer attivi e le aggiorna dinamicamente

9.2 P2P con directory decentralizzata

Architettura completamente distribuita (senza server centralizzati)

Si realizza un'architettura di rete sovrapposta (overlay network, fatta con 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 max a 10 altri peer nella overlay.

Due problematiche:

1. Come viene costruita e gestita la rete di peer;
2. Come un peer localizza un contenuto

9.3 Il protocollo IP

9.3.1 Le funzioni del livello rete

- ▶ Identificare i nodi della rete e le sottoreti;
- ▶ Trasportare i pacchetti dall'host mittente a quello ricevente;
- ▶ Implementare protocolli di livello rete in tutti i router e in tutti gli host.

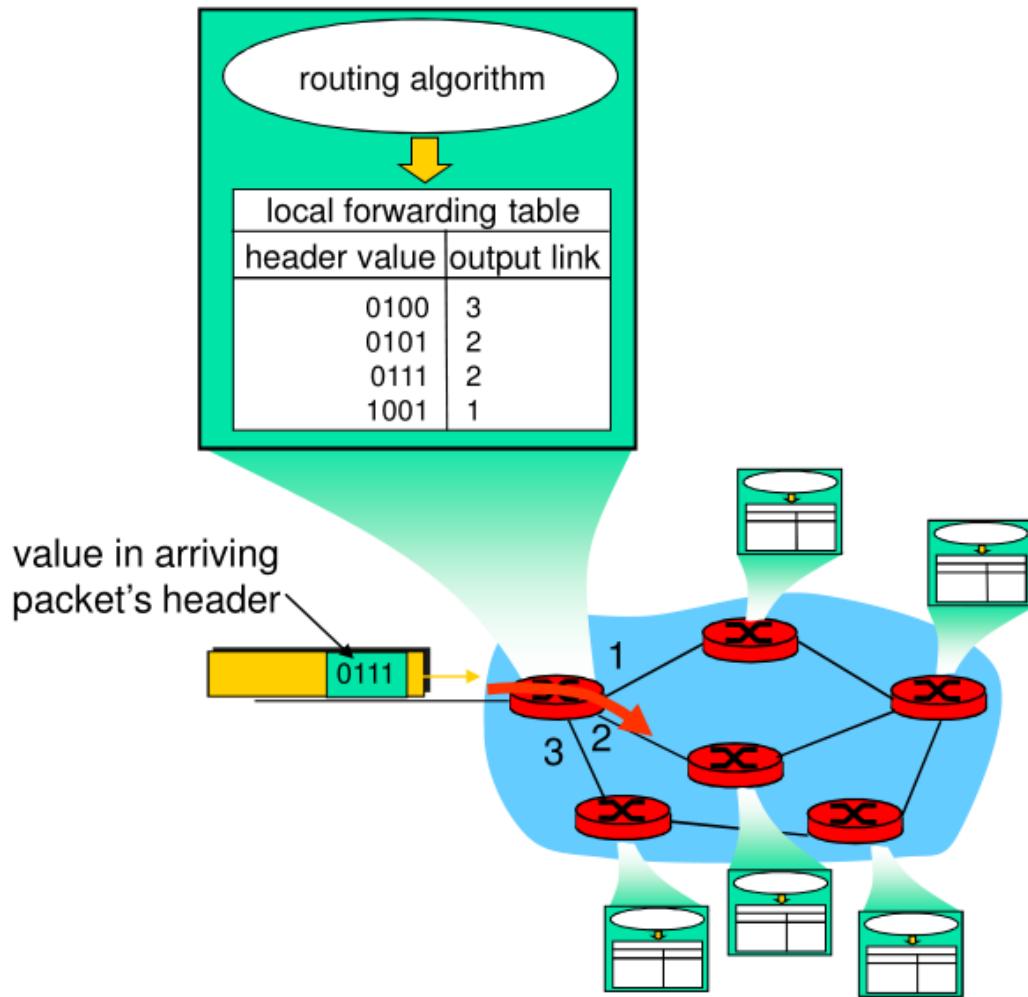
Questo livello non è ovviamente solo **end-to-end**

9.3.2 Le funzioni di forwarding e routing

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 ad una destinazione.

9.3.3 La relazione tra forwarding e routing



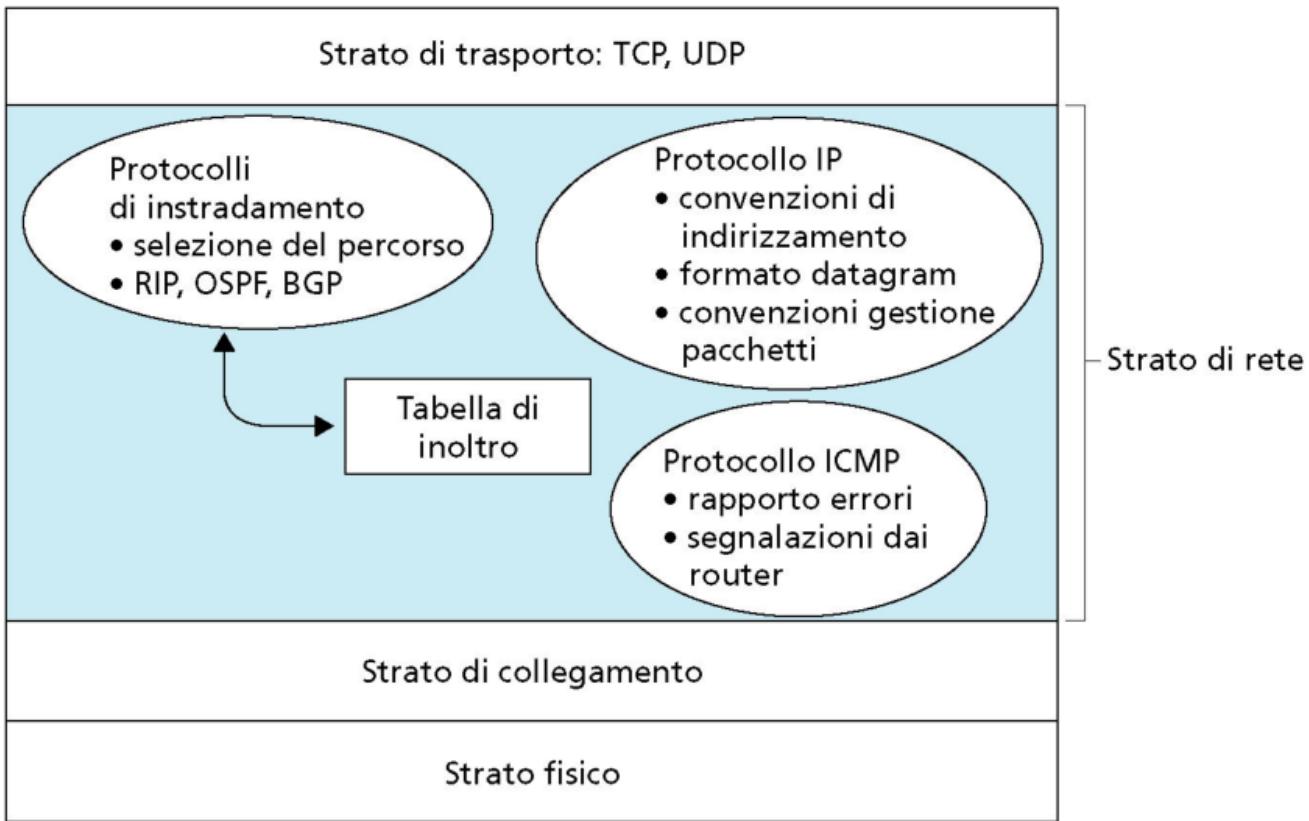
9.3.4 Le funzioni del livello rete

Le reti possono essere classificate a seconda del metodo utilizzato per trasportare i pacchetti dalla sorgente alla destinazione:

- ▶ Reti a datagrammi → ogni pacchetto è instradato indipendentemente dagli altri pacchetti dello stesso flusso;
- ▶ Reti a circuiti virtuali → viene precalcolato un percorso e tutti i pacchetti del flusso seguono questo percorso.

NB: Parliamo comunque di reti a commutazione di pacchetto.

9.3.5 Lo strato di rete in Internet



9.3.6 IP

- ▶ È un protocollo di livello di rete usato per lo scambio di dati tra reti di calcolatori;
- ▶ I dati sono trasportati con la tecnica dei datagrammi;
- ▶ Offre un servizio di comunicazione *connection-less*;
- ▶ 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;
- ▶ È responsabile dell'instradamento dei pacchetti.

9.3.7 Il datagramma IP

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.

In linea teoria un datagramma può contenere fino a un massimo di 65535 byte ($2^{16} - 1$)

9.3.8 L'header IP

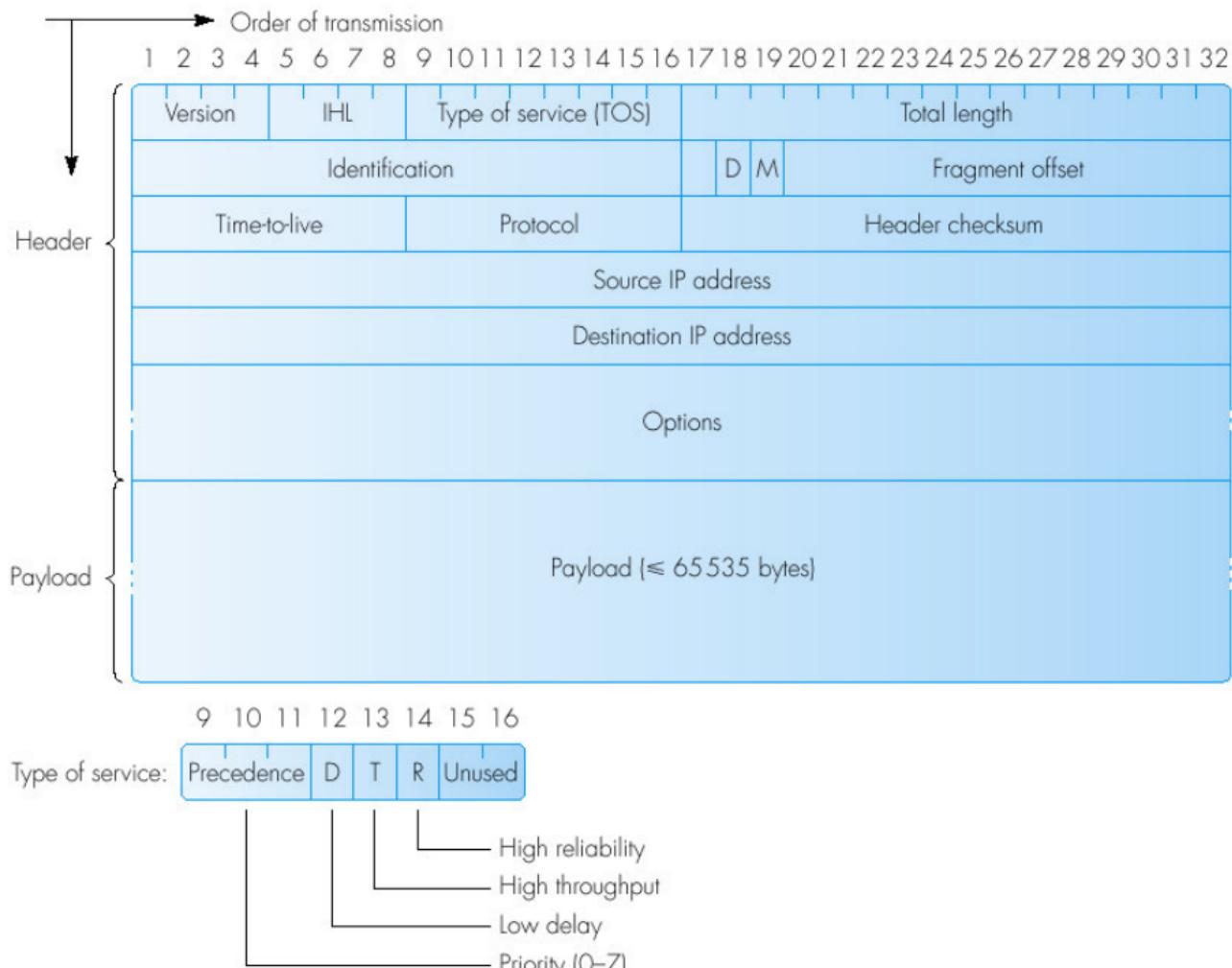
Contiene tutte le informazioni necessarie per la consegna del datagramma alla destinazione:

- ▶ Indirizzo destinazione;
- ▶ Indirizzo sorgente;
- ▶ Identificativo;
- ▶ Ed altro ancora...

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

9.3.9 Formato del pacchetto IP



IHL = intermediate header length
D = don't fragment

M = more fragments

► Time-to-live (TTL)

- ◊ 8 bit, contatore che viene gradualmente decrementato fino a 0, punto in cui il pacchetto viene scaricato. 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;

► 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.

10 Lezione del 03-11

10.1 Frammentazione e riassemblaggio IP

Tutti i frammenti tranne l'ultimo devono essere multipli di 8 byte (che è la dimensione del frammento elementare).

Avendo 13 bit a disposizione, ci possono essere al massimo 8192 frammenti per ogni diagramma.

La dimensione massima di un datagramma è 65535 byte.

10.2 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 0);
- ▶ La presenza opzionale di questi campi rende difficile la gestione in implementazioni HW-based.

10.3 IP è consegna Best Effort

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 controllo da parte di protocolli di livello superiore.

10.4 Indirizzi IP

Ad ogni host è assegnato un indirizzo IP o indirizzo internet:

- ▶ Numero di 32 bit (4 byte);
- ▶ Unico in tutta internet

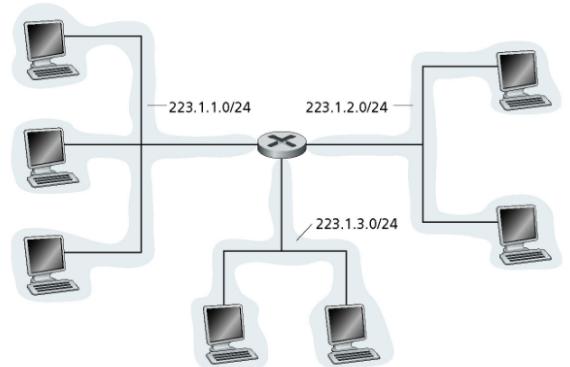
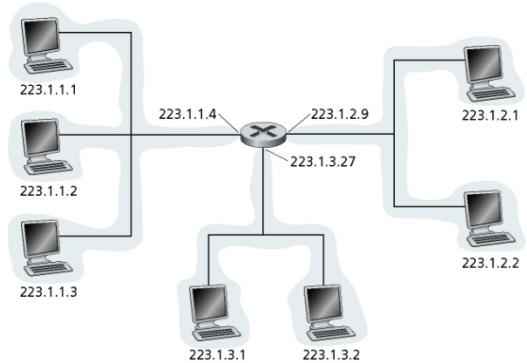
È diviso in prefisso e suffisso:

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

10.5 ICANN (Internet Corporation for Assigned Names and Numbers)

- ▶ Assegna gli indirizzi;
- ▶ Gestisce il DNS;
- ▶ Assegna i nomi dei domini;
- ▶ Risolve eventuali dispute (conflitti di nomi e/o indirizzi).

10.6 Indirizzi delle interfacce e delle reti



10.7 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:

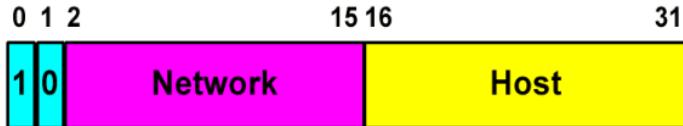
- ▶ 3 (A, B, C) sono usate per gli indirizzi degli host e si differenziano per la lunghezza della parte **rete/host**.
- ▶ 1 (D) è usata per il *multicast*
- ▶ 1 (E) è riservata per usi futuri.

10.7.1 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 $2^{24} \approx 16M$ host

10.7.2 Indirizzi di classe B



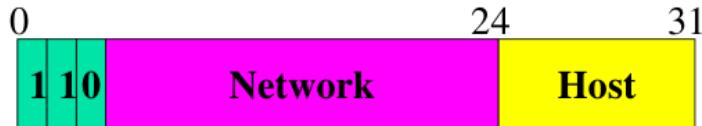
► Campo rete

- ◊ 14 bit;
- ◊ Massimo 16k reti;
- ◊ Il primo byte è compreso tra 128 e 191;

► Campo host

- ◊ 16 bit;
- ◊ Massimo $2^{16} \approx 64M$ host

10.7.3 Indirizzi di classe C



► Campo rete

- ◊ 21 bit;
- ◊ Massimo 2M reti;
- ◊ Il primo byte è compreso tra 192 e 223;

► Campo host

- ◊ 8 bit;
- ◊ Massimo 256 host.

10.7.4 Indirizzi di classe D e E

0 1 2 3

31

1 1 1 0

Multicast Address

Classe D

0 1 2 3

31

1 1 1 1

Reserved for Future Use

Classe E

10.8 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;
 - ◊ Lo utilizzo per inviare un pacchetto a tutte le interfacce sulla mia stessa rete locale.
- ▶ Indirizzo del PC
 - ◊ L'intero indirizzo è costituito da tutti 0;
 - ◊ Per ottenere un indirizzo automaticamente all'avvio, si potrebbe usare IP per comunicare → non abbiamo ancora un indirizzo (per la sua assegnazione si utilizzano protocolli quali **DHCP** e **BOOTP**).

- ▶ Loopback address
 - ◊ Ogni indirizzo che comincia con 127 indica il computer locale;
 - ◊ 127.0.0.1 è quello più comune;
 - ◊ Usato per test, nessun pacchetto esce dalla rete;
 - ◊ Utile quando il computer non possiede una scheda di rete.

10.9 Suddivisione degli indirizzi IP

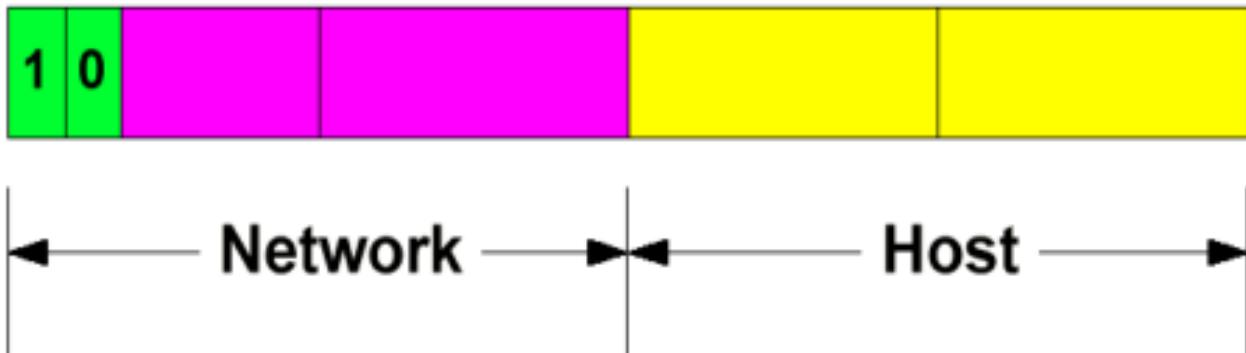
La suddivisione degli indirizzi IP in classi non è efficiente perché comporta lo spreco di indirizzi:

- ▶ 32 bit $\rightarrow 2^{32} \approx 4$ miliardi di indirizzi diversi, ma non tutti vengono usati.

Rimedi:

- ▶ Indirizzi privati
 - ◊ 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 (la traduzione di tali indirizzi verrà eseguita poi con NAT);
- ▶ Sotto reti
 - ◊ Gli indirizzi IP sono assegnati in modo che tutti gli host sulla stessa rete locale appartengono alla stessa 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.

Indirizzo di classe B prima del subnetting



Indirizzo di classe B dopo il subnetting



10.10 Sottoreti

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)

10.10.1 Come viene utilizzata una subnet mask

Da un host che deve trasmettere un pacchetto:

1. Confronta la destinazione con la propria subnet mask;
2. Se la destinazione è sulla stessa sottorete, invia sulla LAN;
3. Altrimenti, invia al gateway.

Da un router all'intero della rete suddivisa in sottorete:

1. Utilizza la subnet mask con l'indirizzo di rete delle reti collegate per determinare la giusta destinazione.

10.10.2 Netmask

Parametro che specifica il subnetting:

- ▶ Bit a 1 in corrispondenza dei campi network e subnetwork
- ▶ Bit a 0 in corrispondenza del campo host.

10.10.3 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:

- ▶ Gestito dai router tramite tabelle di instradamento.

10.10.4 Subnet - 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;
- ▶ ...;

10.11 Default Route

Gli host devono conoscere almeno un router presente sulla loro rete fisica.

Il protocollo ICMP permette di ottimizzare dinamicamente il routing:

- ▶ Ad es sull'host H4 → `route add default gw 190.3.1.5`

10.12 Tabelle di instradamento

L'instradamento tra subnet diverse viene gestito da tabelle di instradamento presenti sui router.

11 Lezione del 08-11

Recap della lezione del 03-11.

11.1 ICMP (Internet Control Message Protocol)

Funzionalità:

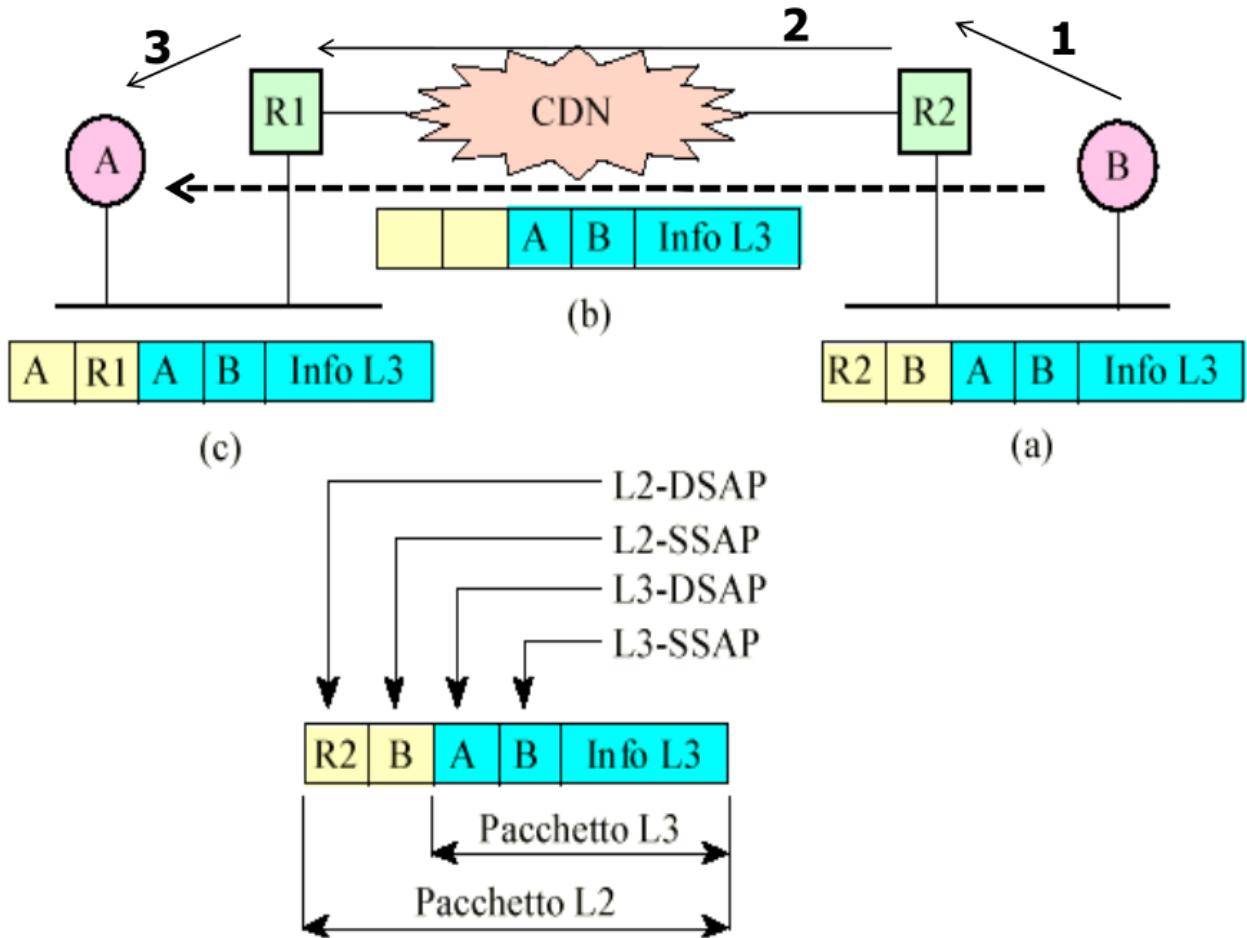
- ▶ Verificare lo stato della rete
 - ◊ `echo request`, `echo reply`;
- ▶ Riportare anomalie
 - ◊ Destination unreachable;
 - ◊ Time exceeded;
 - ◊ Parameter problem;
- ▶ Scoprire la netmask
 - ◊ Mask request;
 - ◊ Address mask reply;
- ▶ Migliorare il routing → redirect.

Applicazioni:

- ▶ Ping
 - ◊ Utilizzato per verificare la connettività a livello rete tra 2 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`;
- ▶ 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 azzera invierà indietro un messaggio *time exceeded* (in questo modo si riesce a determinare il percorso fino alla destinazione).

12 Lezione del 10-11

12.1 Indirizzi IP ed Indirizzi di Livello 2



12.2 Reti Locali

Lo scopo di una rete locale è quello di condividere.

Una rete locale (tipo ethernet) è un mezzo broadcast.

Da un [cavo coassiale](#) si passa ad un contenitore, costituito da interfacce collegate dalle [prese RJ45](#).

12.3 Problema della risoluzione dell'indirizzo

2 host possono comunicare direttamente solo se sono collegati alla stessa rete fisica → Per potersi scambiare informazioni devono conoscere i rispettivi indirizzi fisici.

Il protocollo IP consente di individuare univocamente un host tramite un indirizzo logico:

- ▶ Tutte le applicazioni usano gli indirizzi logici e ignorano la rete fisica. Per inviare un messaggio occorre necessariamente conoscere anche l'indirizzo fisico;

Serve un meccanismo di corrispondenza tra gli indirizzi logici e gli indirizzi fisici.

Tale meccanismo è offerto dal protocollo ARP.

12.4 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).

12.5 Formato del pacchetto ARP

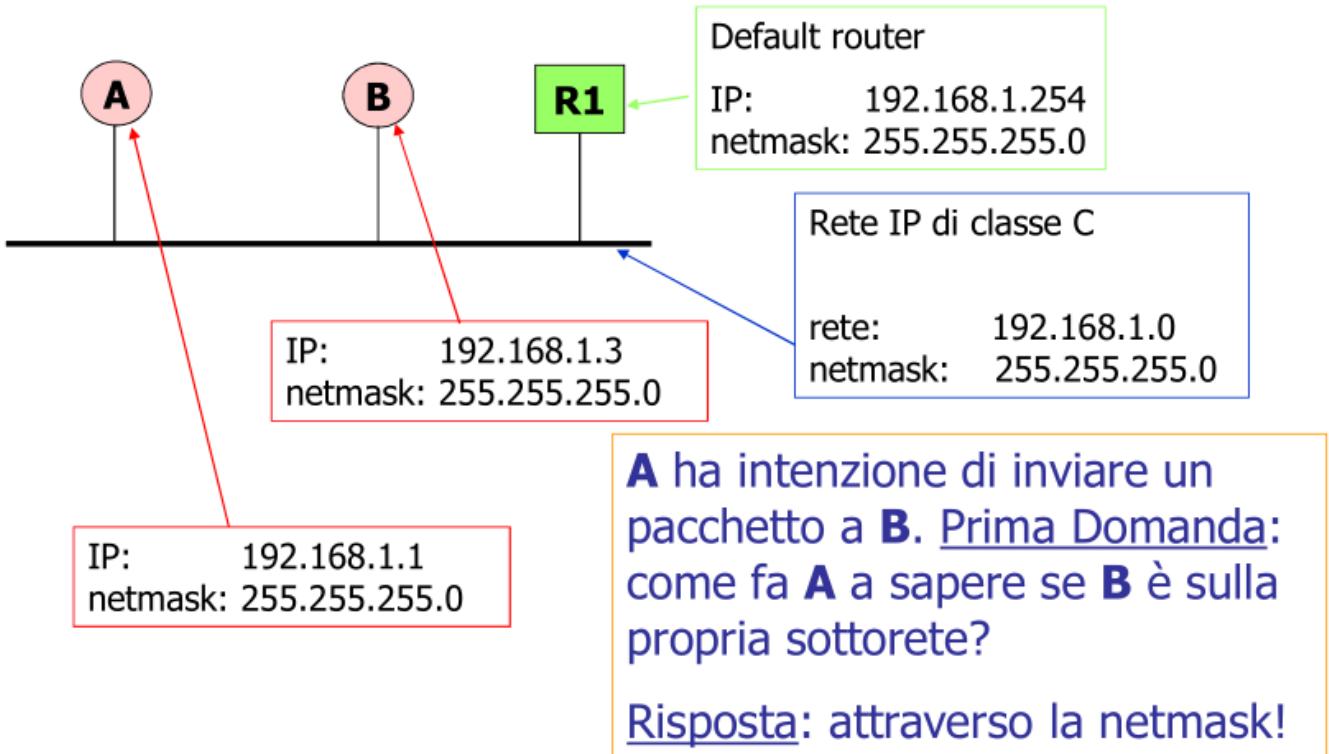
Hardware Type		Protocol Type
HLEN	PLEN	Operation
Sender Hardware Address		
Sender HW Address		Sender IP Address
Sender IP Address		Target HW Address
Target Hardware Address		
Target IP Address		

12.6 ARP - scenari tipici

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

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

12.6.1 Primo Caso

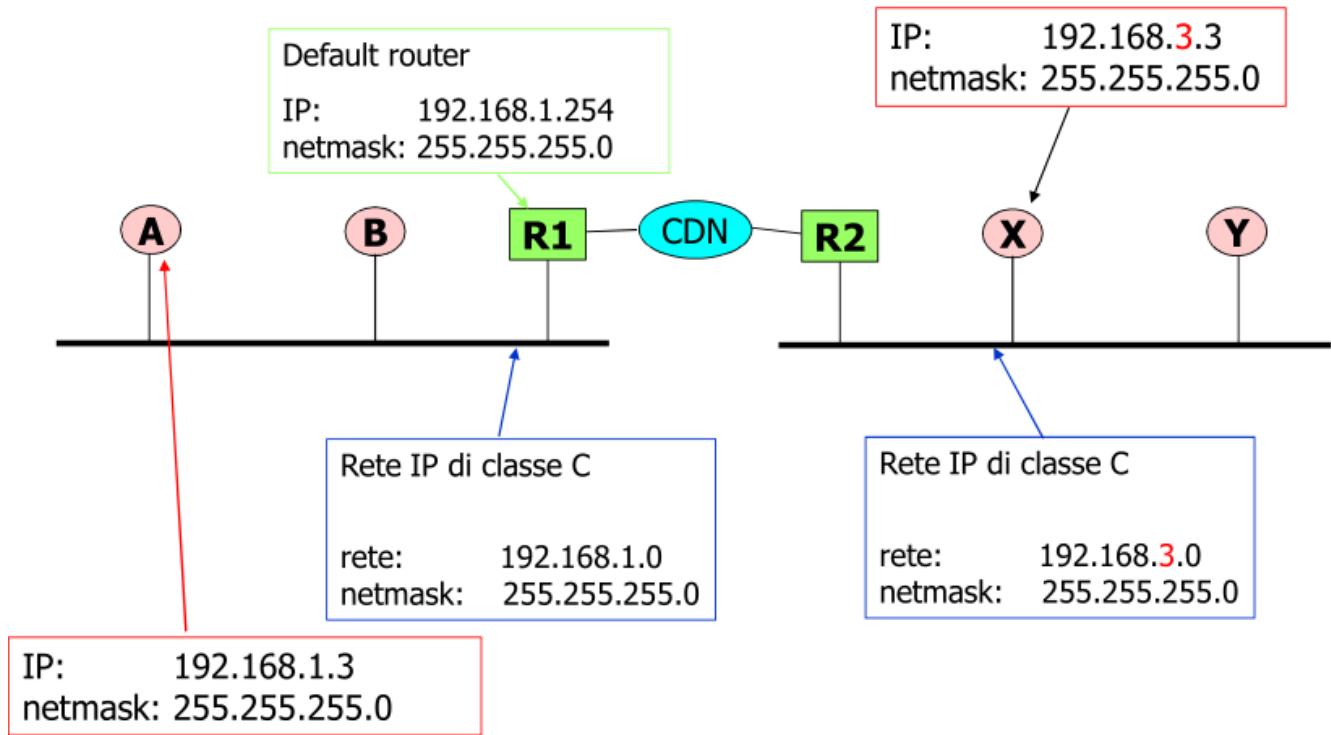


- ▶ Ogni PC ha un indirizzo IP e una netmask. La netmask permette di individuare la propria sottorete IP;
- ▶ Il PC A esegue una AND tra l'indirizzo IP destinazione e la propria netmask.

Se il PC B è sulla stessa sottorete IP:

- ▶ Allora invia un pacchetto ARP request in broadcast → tale pacchetto contiene, nel campo DEST IP, l'indirizzo IP di B.

12.6.2 Secondo Caso



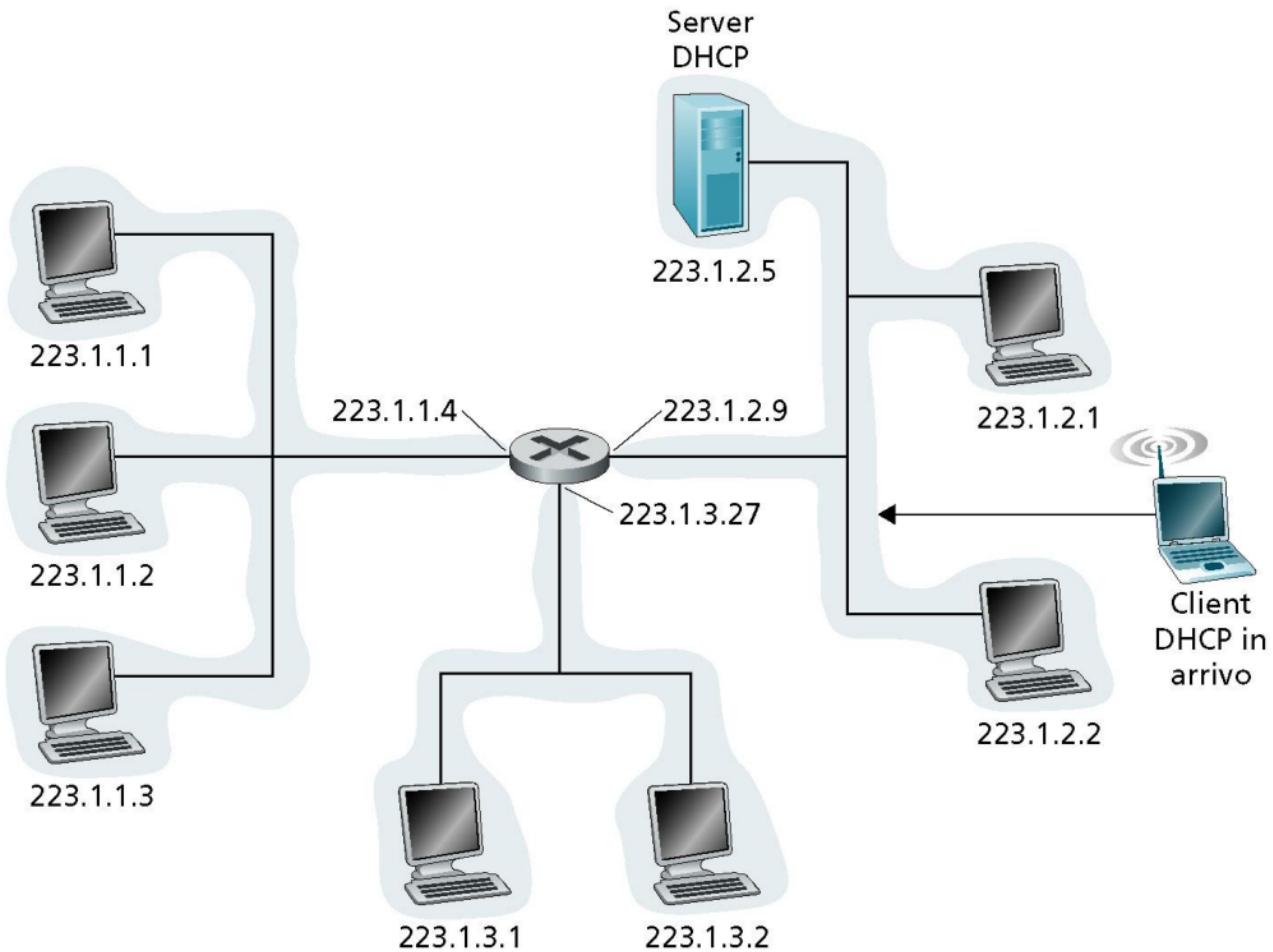
Se A intende mandare un pacchetto a X, l'operazione di AND tra netmask e l'indirizzo IP DEST fornisce un risultato differente.

Pertanto, si prepara un pacchetto ARP in cui si specifica come indirizzo IP DEST proprio l'indirizzo IP del router.

12.7 BOOTP (BOOTstrap Protocol) e DHCP (Dynamic Host Configuration Protocol)

2 protocolli più flessibili e potenti rispetto ad ARP che permettono di assegnare dinamicamente gli indirizzi agli host di una rete IP.

12.7.1 DHCP - scenario tipico



12.7.2 Interazione client-server via DHCP

1. Scoperta DHCP;
2. Offerta DHCP;
3. Richiesta DHCP;
4. ACK DHCP.

12.8 NAT - Network Address Translation

Consente ad un dispositivo di agire come intermediario tra Internet e una rete privata. In questo modo un unico indirizzo IP può rappresentare un intero gruppo di PC.

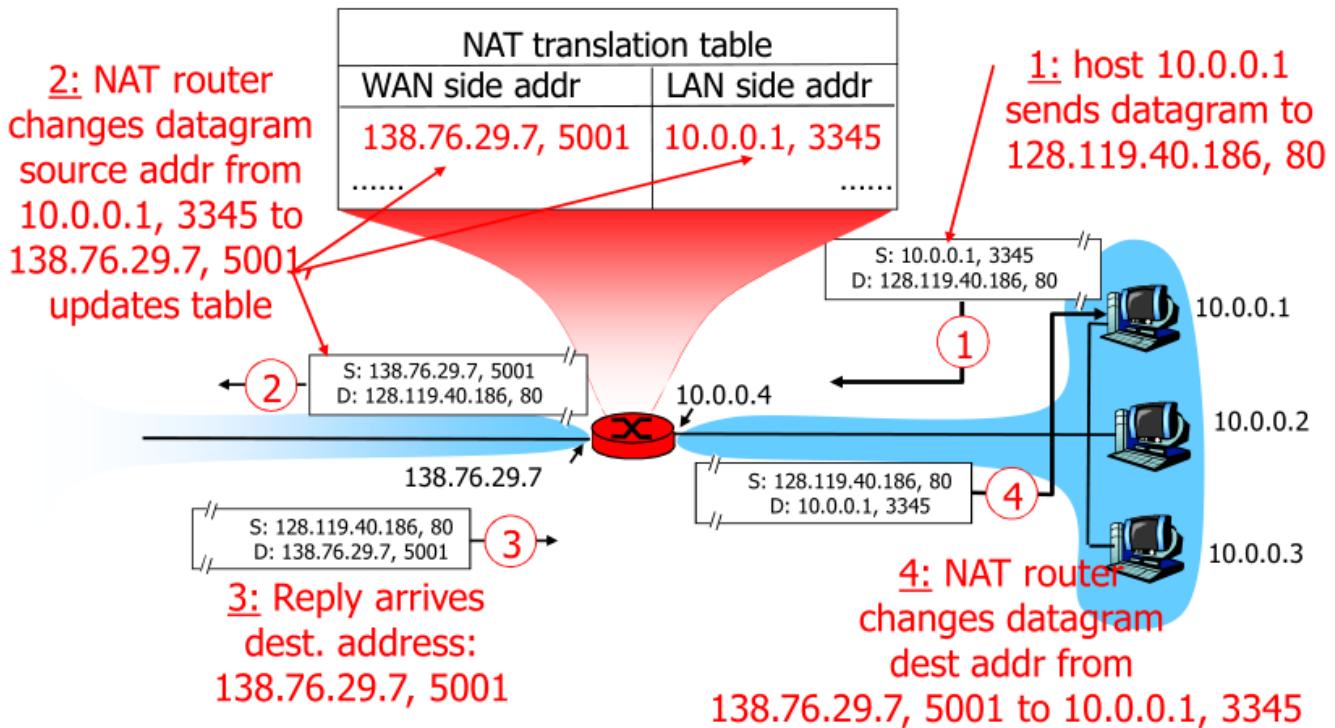
L'utilizzo è quello di mappare un insieme di indirizzi privati su di un unico indirizzo pubblico, utilizzando differenti porti per mantenere traccia delle diverse connessioni.

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

Questa tabella viene consultata anche quando il router riceve un pacchetto dal PC 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

12.9 Esempio



13 Lezione del 15-11

13.1 IPv4

IPv4 nasce come protocollo sperimentale. Lo spazio di indirizzamento in origine era sufficientemente ampio, ma come si può intuire con il passare del tempo questo spazio si è andato via via a ridurre (basta pensare alla quantità in aumento di IoT e altre aree IP). I 32bit ormai non sono più sufficienti

13.2 IPv6

È la nuova versione del protocollo di Internet.

Mira a risolvere parte del problema che Internet sta incontrando a causa della sua crescita vertiginosa.

Le principali questioni affrontate:

- ▶ Indirizzamento e routing;
- ▶ Sicurezza;
- ▶ Configurazione automatica;
- ▶ Servizi di tipo real-time.

13.2.1 Features

- ▶ Spazio di indirizzamento a 128bit
 - ◊ Nessun bisogno di NATTING;
- ▶ Autoconfigurazione dell'host;
- ▶ Eliminazione del layer dovuto al DHCP → autogenerazione dell'IP automatico;
- ▶ Miglioramento di QoS (Qualità del servizio);
- ▶ Miglioramento della sicurezza.

13.2.2 Problemi affrontati in fase di progetto

- ▶ Supportare reti interconnesse di tipo globale;
- ▶ Garantire una transazione chiara e diretta dell'immensa base di sistemi utilizzanti IPv4;
- ▶ Sostenere l'elevato tasso di crescita;
- ▶ Far fronte ai possibili scenari futuri nel mondo dell'internetworking
 - ◊ Mobile computing;
 - ◊ Network entertainment;
 - ◊ ...

13.2.3 Scrivere un IPv6

Utilizzare la stessa notazione (quella decimale) degli IPv4 risulta molto sconveniente.

Utilizziamo una forma abbreviata, costituita da 8 gruppi, separati da :, di 4 cifre esadecimali in cui le lettere vengono scritte in forma minuscola.

Se uno dei gruppi è composto da una sequenza di quattro zeri può essere contratto ad un solo zero. Inoltre, una sequenza di zeri contigui (e una soltanto) composta da 2 o più gruppi può essere contratta con la semplice sequenza ::.

13.2.4 Caratteristiche generali

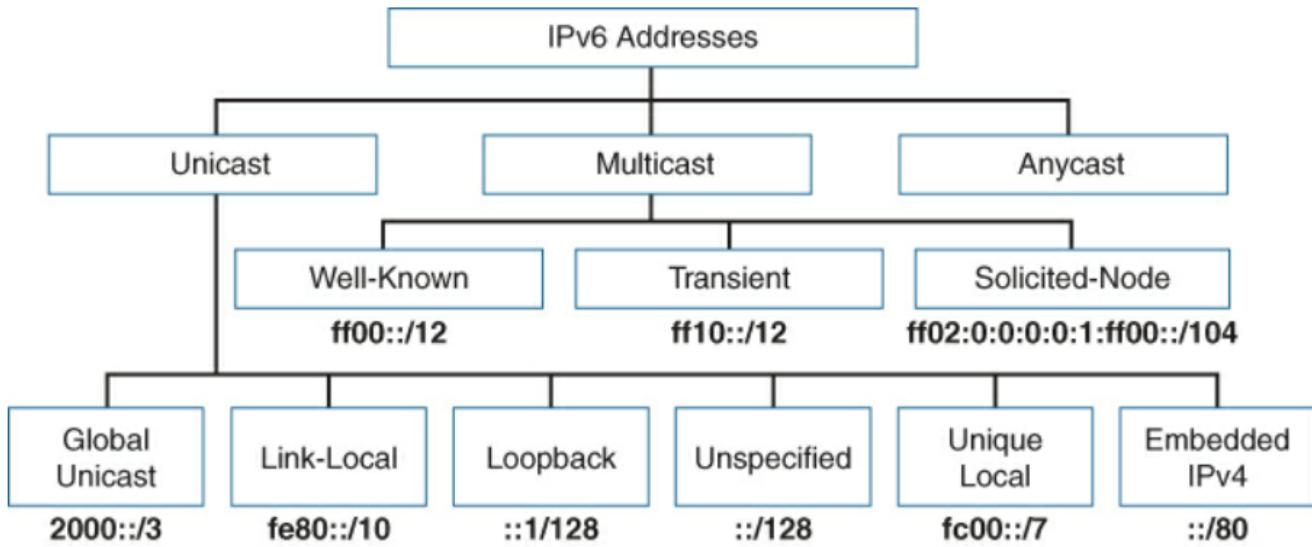
Semplificazione del formato dell'header:

- ▶ Alcuni campi dell'header sono stati semplificati o resi opzionali;
- ▶ Supporto per le opzioni migliorato;
- ▶ Autenticazione e salvaguardia della privacy.

13.2.5 Unicast, Multicast e Anycast

In IPv6, abbiamo 3 tipi di indirizzi:

- ▶ Unicast;
- ▶ Multicast;
- ▶ Anycast.



Un indirizzo unicast identifica in modo univoco un'interfaccia su un dispositivo IPv6.

Un pacchetto inviato a un indirizzo unicast viene ricevuto dall'interfaccia assegnata a tale indirizzo. Simile a IPv4, un indirizzo IPv6 di origine deve essere un indirizzo unicast.

Il multicast è una tecnica in cui un dispositivo invia un singolo pacchetto a più destinazioni contemporaneamente (trasmissione uno a molti). Più destinazioni possono effettivamente essere più interfacce sullo stesso dispositivo, ma in genere sono dispositivi diversi.

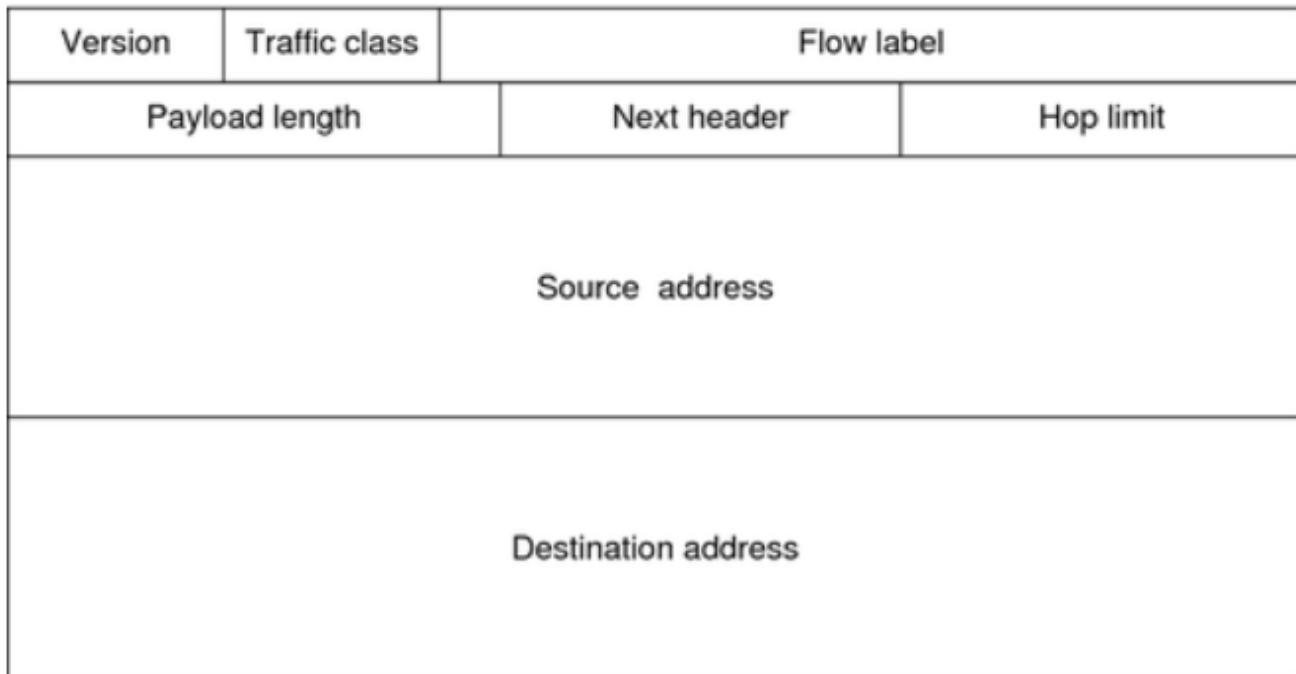
Un indirizzo IPv6 multicast definisce un gruppo di dispositivi noto come gruppo multicast. Gli indirizzi IPv6 utilizzano il prefisso **ff00::/8**, che è equivalente all'indirizzo multicast IPv4 **224.0.0.0/4**. Un pacchetto inviato a un gruppo multicast ha sempre un indirizzo sorgente unicast; un indirizzo multicast non può mai essere l'indirizzo di origine.

A differenza di IPv4, non esiste un indirizzo di trasmissione in IPv6. Invece, IPv6 utilizza il multicast.

13.2.6 Formato del pacchetto IPv6

- ▶ L'header ha meno campi di quello IPv4;
- ▶ Presenta una dimensione variabile;
- ▶ L'header è comunque abbastanza lungo, proprio per gli indirizzi a 128bit.

Per **funzioni/opzioni**, IPv6 usa estensioni di Header, inserite da un sender tra l'header main e il payload → soluzione alla frammentazione e l'IPSec.



(NB → Hop limit corrisponde al TTL)

In IPv6 è assente il campo protocol → inseriamo il campo Next header, che permette:

- ▶ Di definire nuove estensioni Header;
- ▶ Di aggiungere sicurezza.

Il Next Header identifica:

- ▶ Hop-by-Hop header;
- ▶ Opzioni header destinazione;
- ▶ Header di routing;
- ▶ Header frammentati;
- ▶ Autenticazione e ESP header.

13.2.7 Compatibilità

Avendo 2 header differenti non è presente una compatibilità diretta tra i 2 protocolli.

Tuttavia uno stesso dispositivo può eseguire entrambi i protocolli.

13.2.8 MTU in IPv6

Devono essere minimo 1280 (devo arrivare a 1500 bytes per Ethernet).

13.2.9 Autoconfigurazione IPv

I nodi IPv6 possono usare SLAAC per determinare il loro:

- ▶ Indirizzo IP;
- ▶ Gateway di default;
- ▶ (Opzionalmente) DNS resolver.

SLAAC funziona inviando ai router un RAs (Router Advertisement).

Un messaggio RAs può contenere:

- ▶ Un prefisso, e preferibilmente una sua lifetime;
- ▶ MTU;
- ▶ Disponibilità di un DHCPv6;

Essendo lo spazio di indirizzamento enorme, sfrutto il MAC address della macchina. Ad es:

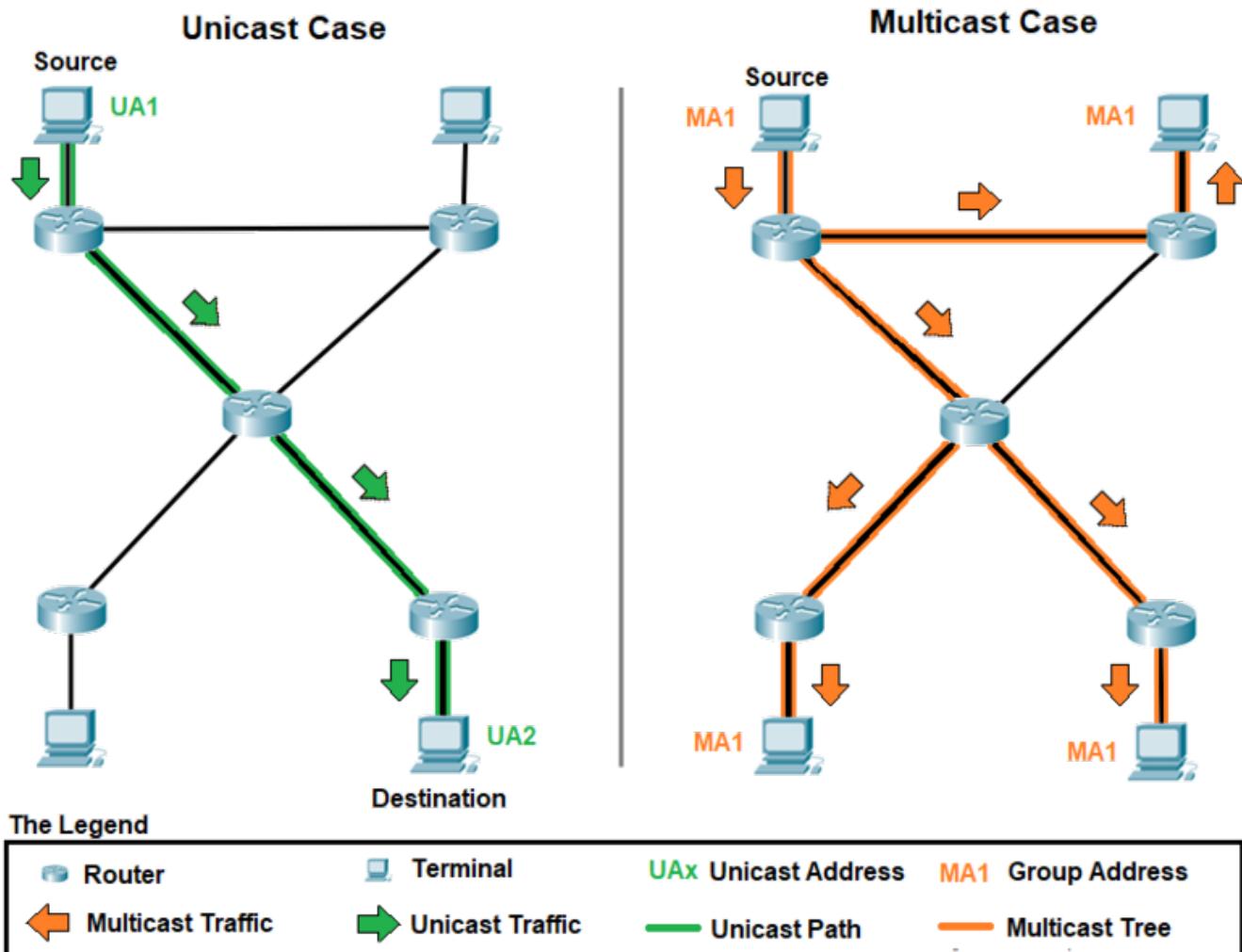
- ▶ Host MAC address → 08:00:20:9c:14:66;
- ▶ Network prefix → 2001:630:80:2::/64;
- ▶ Address → 2001:630:80:2:0a00:20ff:fe9c:1466

13.3 IP Multicasting

% Multicasting

L'invio di un pacchetto da un sender a molti receiver con una singola operazione di spedizione.

13.4 Trasmissione multicast

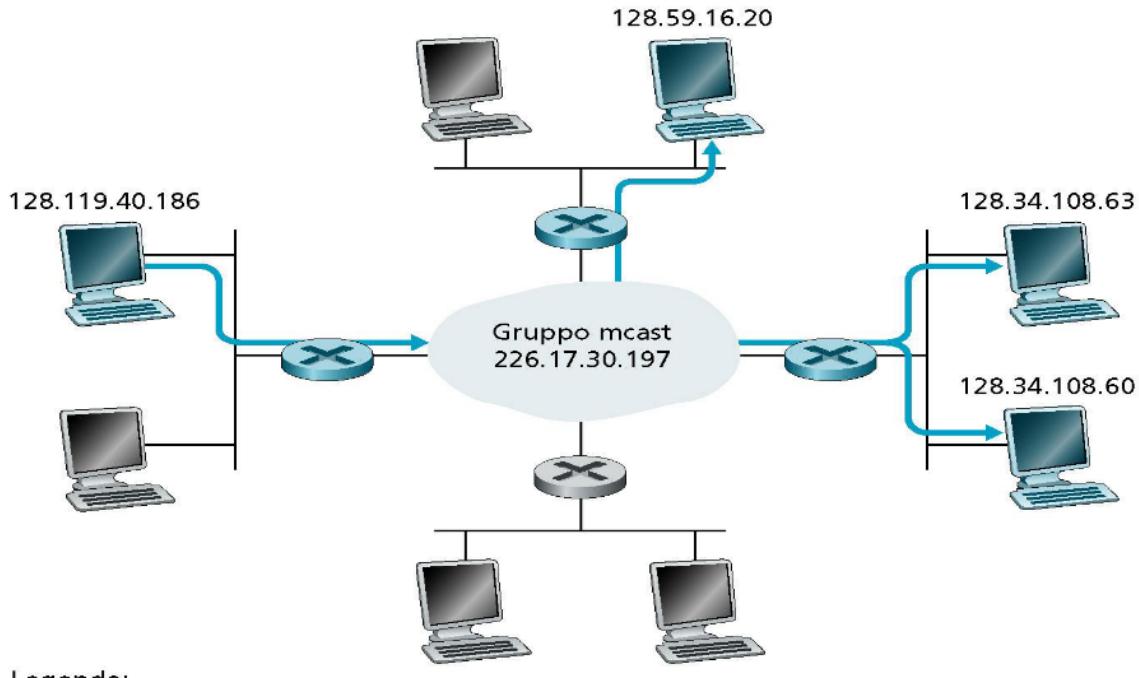


Da *indirizzo per destinazione* ad *indirizzo per evento* → Si utilizza un identificativo unico per il gruppo di ricevitori e una copia del datagramma è inviata, utilizzando tale identificativo, a tutti i membri del gruppo.

Gli indirizzi multicast coprono un range da 224.0.0.0 a 239.255.255.255 (più altri riservati dallo IANA). Suddivido gli indirizzi in aree.

13.5 Il gruppo multicast

Insieme dei device interessati su un determinato indirizzo multicast.



Legenda:

- Router cui sono attaccati componenti del gruppo
- Router cui non sono attaccati componenti del gruppo

13.6 Session Announcement Protocol (SAP)

Per annunciare una sessione multicast e la sua descrizione si utilizza il protocollo SAP. Ci sono diverse possibilità:

- ▶ Sessioni Global Scope;
- ▶ Sessioni Administrative Scope.

Protocollo UDP:

- ▶ Porta 9875;
- ▶ TTL 255.

Per cancellare una sessione:

- ▶ **Explicit Timeout** → la durata è parte dell'annuncio;
- ▶ **Implicit Timeout** → Se non si riceve nulla per un intervallo prefissato;
- ▶ **Explicit Deletion**.

13.7 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.

13.8 Il multicast router

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

13.8.1 Funzionamento

- ▶ Ogni elaboratore trasmette i datagrammi multicast sfruttando il meccanismo HW 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

14 Lezione del 17-11

14.1 Protocolli per il multicast in Internet

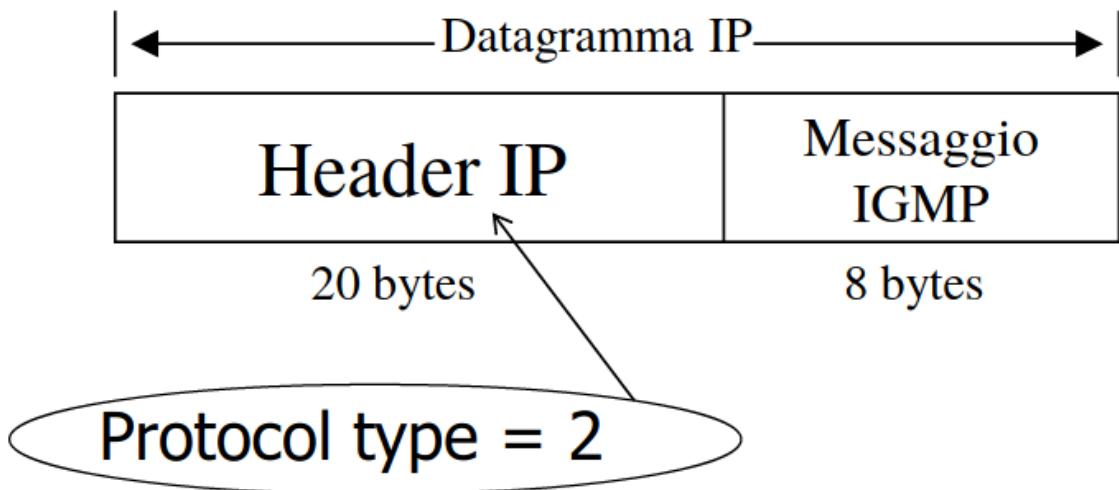
- ▶ IGMP (Internet Group Management Protocol) → Fornisce ad un host i mezzi per informare il multicast router ad esso più vicino che un'applicazione vuole unirsi ad un determinato gruppo multicast
 - ❖ Opera tra un host ed il router ad esso direttamente collegato;
- ▶ Algoritmi per il multicast routing → Coordinano i multicast router all'interno della rete Internet, per permettere l'instradamento dei datagrammi multicast.

14.1.1 Il protocollo IGMP

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

I pacchetti sono incapsulati in datagrammi IP con numero di protocollo 2.

Il raggio di interazione di tale protocollo è locale → 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 (fa parte di) un determinato gruppo multicast
leave group	host	Informa il multicast router locale che l'host vuole lasciare un determinato gruppo multicast

14.1.2 IGMP - funzionalità

Si suddividono in 2 fasi differenti:

1. Fase 1

- ▶ Quando un host si unisce a 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;

2. Fase 2

- Dovendo gestire i gruppi in maniere 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.

14.1.3 IGMP - implementazione

È stato progettato per evitare di aggiungere carico eccessivo sulla rete:

- Esso cerca, laddove possibile, di sfruttare al massimo i meccanismi HW 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 (*pool 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;
- Ogni host ascolta le risposte inviate dagli altri e sopprime le proprie nel caso in cui risultino superflue.

14.2 IP multicast - distribuzione sul LAN ethernet

Per la trasmissione di datagrammi IP multicast su reti LAN ethernet, occorre mappare un indirizzo in classe D su di un indirizzo MAC multicast.

Non è possibile in maniera univoca, dato il range degli indirizzi MAC riservati al multicast

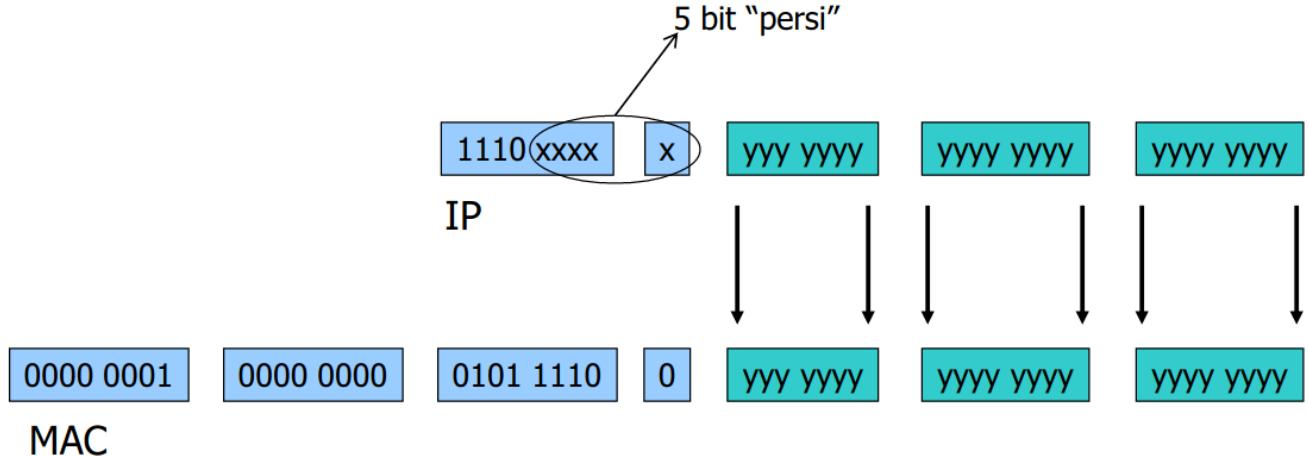
da 01:00:5e:00:00:00 a 01:00:5e:7f:ff:ff.

L'OUI 01:00:5e è riservato al mapping degli indirizzi IP multicast.

Indirizzo IP multicast (28 bit liberi) si mappa ad un indirizzo MAC ethernet (23 bit).

Aliasing → $2^5 = 32$ gruppi multicast IP per ogni MAC multicast. Ne seguono possibili conflitti.

14.2.1 Mapping Indirizzi IP multicast e Indirizzi MAC multicast



14.3 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, collegate mediante link virtuali di tipo punto-punto chiamati *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.

14.4 Reti di calcolatori e grafi

Immaginiamo una rete modellata come grafo in cui:

- ▶ Nodi → router;
- ▶ Archi → link fisici
 - ◊ Costo link (ritardo, costo trasmissione, congestione, . . .).

Scelta del cammino:

- ▶ Cammino a costo minimo;
- ▶ Alternative (cammino calcolato in base a specifici vincoli).

Per questo gli algoritmi di routing fanno uso della teoria dei grafi.

14.4.1 Parametri del processo decisionale

- ▶ **Bandwidth** → Capacità di un link definita in *bps* (bit per secondo);
- ▶ **Delay** → Tempo necessario per spedire un pacchetto da una sorgente ad una destinazione;
- ▶ **Load** → Una misura del carico di un link;
- ▶ **Reliability** → Riferita, all'error rate di un link;
- ▶ **Hop count** → Numero di router da attraversare nel percorso dalla sorgente alla destinazione;
- ▶ **Cost** → Valore arbitrario che definisce il costo di un link
 - ◊ Costituito come funzione di diversi parametri (Bandwidth, delay, packet loss, MTU, . . .).

14.5 Il processo di routing

Processo decisionale → ogni entità che partecipa a questo processo:

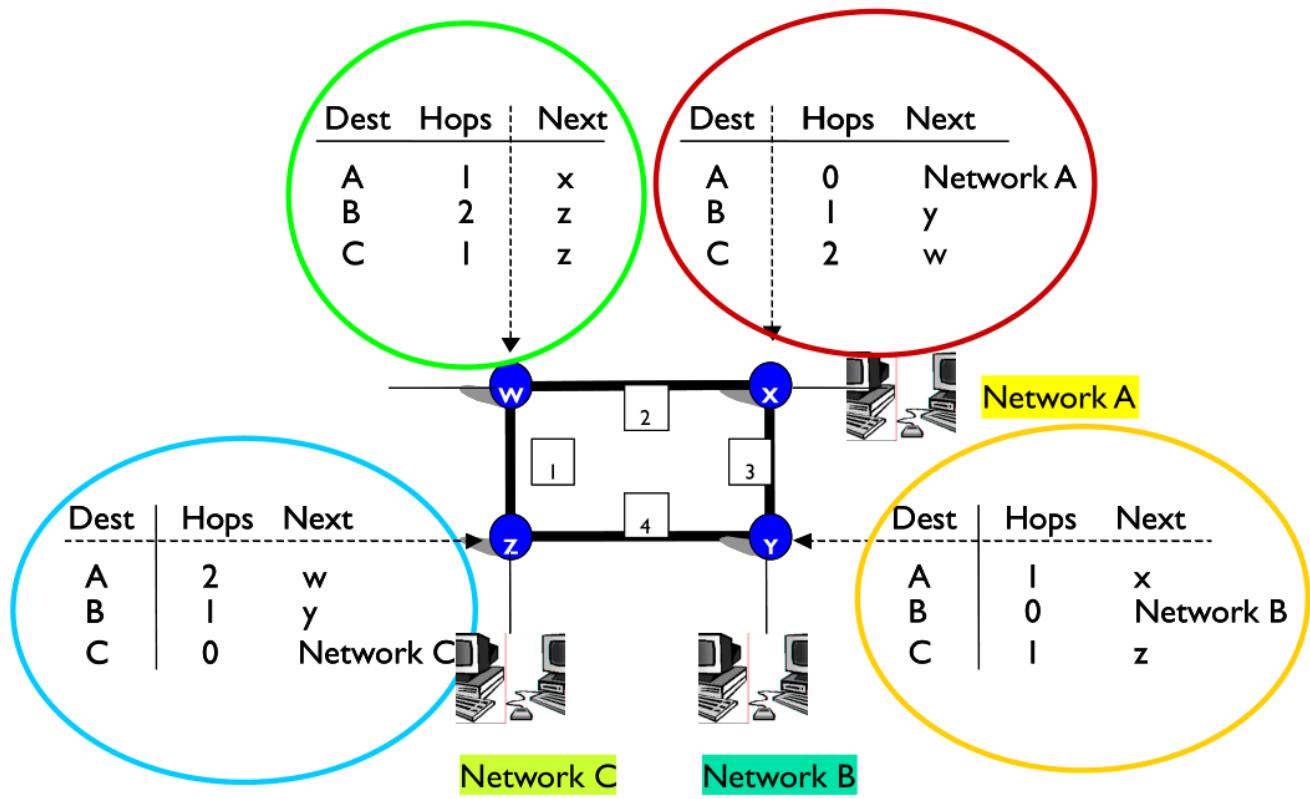
- ▶ Mantiene delle informazioni;
- ▶ In base ad uno specifico algoritmo ed in funzione di determinate metriche definisce il procedimento di instradamento verso le possibili destinazioni;
- ▶ Può spedire informazioni di aggiornamento alle altre entità coinvolte, secondo diversi paradigmi.

14.6 Il routing e la funzione di un router

La funzione principale di un router è quella di determinare i percorsi che i pacchetti devono eseguire per arrivare a destinazione, partendo da una data sorgente.

Ogni router si occupa del processo di ricerca di un percorso per l'instradamento di pacchetti tra 2 nodi qualunque di una rete.

14.7 Tabelle di routing - Esempio



14.8 Tecniche di routing

14.8.1 Routing by Network Address

Ogni pacchetto contiene l'indirizzo del nodo destinatario, che viene usato come chiave di accesso alle tabelle di instradamento.

Viene usato tipicamente nei protocolli non orientati alla connessione → IPv4, IPv6, bridge trasparenti, OSI CLNP, ...

14.8.2 Label Swapping

Ogni pacchetto è marcato con una *label* (etichetta) che:

- ▶ Identifica la connessione;
- ▶ Viene usata come chiave per determinare l'instradamento.

Generalmente usato nei protocolli orientati alla connessione.

14.9 Routing - reti a circuiti virtuali

Viene aperta una connessione prima di inviare dati.

14.10 Routing - reti a datagramma

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.

14.11 Tipologie di routing

La scelta de percorso di instradamento può essere realizzata mediante 2 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 tabella 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 dei routing.

15 Lezione del 22-11

15.1 Routing distribuito

Ogni router calcola le sue tabelle dialogando con 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 2 approcci principali:

- ▶ Algoritmi *distance vector*;
- ▶ Algoritmi *link state*

15.2 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 non si può riferire ad un dominio molto esteso e che cambia in maniera estremamente dinamica.

Le tabelle di routing vanno memorizzate all'interno di router:

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

L'algoritmo deve essere robusto.

15.3 Scambio delle informazioni di update

- Broadcast periodico;
- Event-driven.

15.4 Scelta dell'algoritmo di routing

Possono esistere più criteri di ottimalità contrastanti. Andiamo quindi a tener conto di:

- Un numero di nodi elevato;
- Una topologia complessa;
- Algoritmi troppo complessi, su reti molto grandi, potrebbero richiedere un tempo di calcolo troppo elevato;
- Vincoli di tipo amministrativi.

Parametri fondamentali per questa scelta sono:

- Semplicità;
- Robustezza;
- Stabilità;
- Equità;
- Metrica da adottare.

15.5 Distance Vector

Ogni router mantiene una tabella di tutti gli instradamenti a lui noti (All'inizio 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 un messaggio di aggiornamento contenente tutte le informazioni della propria tabella.

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.

Vantaggi	Svantaggi
Facile da implementare	Ogni messaggio contiene una intera tabella di routing
	Lenta propagazione delle informazioni sui cammini (rischio di inconsistenza)

15.6 Link state

Utilizza l'algoritmo SPF (*Short Path First*):

- ▶ Non basato su tabelle;
- ▶ Tutti i router devono conoscere l'intera topologia della rete.

Ogni router esegue 2 azioni:

1. Controlla lo stato di tutti i router vicini;
2. Periodicamente invia, in broadcast, un messaggio contenente lo stato dei link a cui è collegato.

I router utilizzano i messaggi ricevuti per aggiornare la loro mappa di rete → se la mappa cambia, il router ricalcola i percorsi di instradamento applicando l'algoritmo di Dijkstra.

15.7 Distance Vector vs Link state

Distance Vector:

- ▶ Vengono mandate ai vicini le informazioni su tutti (il vettore delle distanze).

Link state:

- ▶ Vengono mandate a tutti informazioni sui vicini.

16 Lezione del 24-11

16.1 Algoritmo Distance Vector

Formula di [Bellman-Ford](#). Ogni nodo:

- ▶ Invia ai nodi adiacenti un distance vector, costituito da
 - ◊ Insieme di coppie (*indirizzo, distanza*), dove la distanza è espressa tramite metriche classiche (hop e costo);
- ▶ Memorizza per ogni linea l'ultimo Distance vector ricevuto;
- ▶ Calcola le proprie tabelle di instradamento
 - ◊ Se queste risultano diverse dai quelle precedenti → Invia ai nodi adiacenti un nuovo distance vector.

16.1.1 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 nuovo adiacente, diverso da quello memorizzato

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

16.1.2 Caratteristiche

- ▶ Iterativo

- ◊ Continua fin tanto che non sia più presente scambio di informazioni;
- ◊ *Self terminating* → Non esiste un esplicito segnale di stop;
- Asincrono;
- Distribuito → Ogni nodo comunica con i direttivi vicini;

Struttura Distance Vector

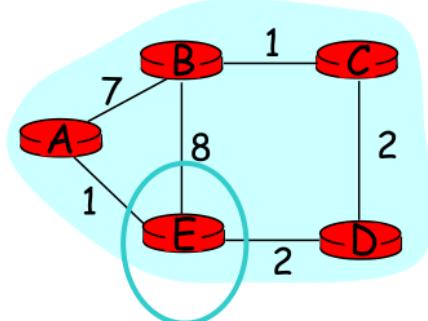
Ogni nodo ha la sua tabella delle distanze:

- Una riga per ogni destinazione;
- Una colonna per ogni nodo adiacente.

Notazione adoperata:

$$D^X(Y, Z) = \text{distanza da } X \text{ a } Y, \text{ via } Z \text{ (prossimo hop)} = c(X, Z) + \min_w \{D^Z(Y, w)\}$$

16.1.3 Esempio



$$D^E(C, D) = c(E, D) + \min_w \{D^D(C, w)\} \\ = 2+2 = 4$$

$$D^E(A, D) = c(E, D) + \min_w \{D^D(A, w)\} \\ = 2+3 = 5 \quad \text{loop!}$$

$$D^E(A, B) = c(E, B) + \min_w \{D^B(A, w)\} \\ = 8+6 = 14 \quad \text{loop!}$$

costo per la destinazione via

$D^E()$	A	B	D
A	1	14	5
B	7	8	5
C	6	9	4
D	4	11	2

16.1.4 Modifica dei costi dei collegamenti

1. Un nodo si accorge di una modifica locale al costo di un link ad esso connesso.
2. Aggiorna la sua distance table;
3. Se cambia il costo di qualche path allora lo notifica ai vicini.

16.1.5 Poisoned reverse

Se Z raggiunge X attraverso Y, Z dice a Y che la sua distanza per X è infinita (così Y non andrà ad X attraverso Z).

16.2 Algoritmo Link State

Ogni router:

- ▶ Impara il suo ambito locale (linee e nodi adiacenti);
- ▶ Trasmette queste informazioni a tutti gli altri router della rete attraverso un *Link State Packet* (LSP);
- ▶ Memorizza gli LSP trasmessi dagli altri router e costituisce una mappa della rete;
- ▶ Calcola, in maniera indipendente, le sue tabelle di instradamento applicando alla mappa della rete l'algoritmo di Dijkstra, noto come SPF.

16.2.1 Il processo di update

Ogni router genera un 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.

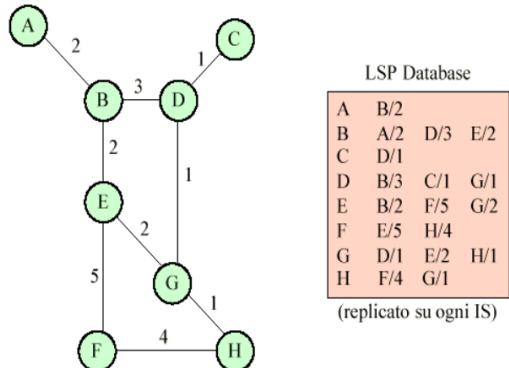
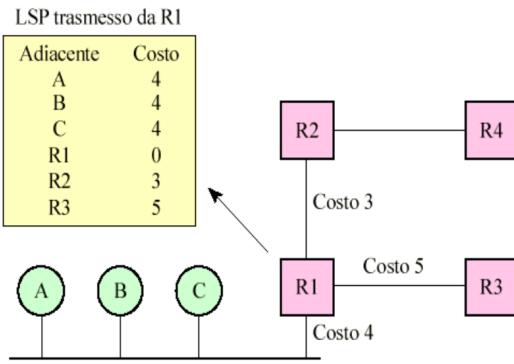
16.2.2 LSP flooding

Un LSP viene generato periodicamente, oppure quando viene rilevata una variazione nella topologia locale (adiacenze), ovvero:

- ▶ Viene riconosciuto un nuovo vicino;
- ▶ Il costo verso il 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**.

16.2.3 Trasmissione di un LSP e grafo della rete e LSP-DB - Esempio



16.2.4 Gestione degli LSP

Nel momento in cui riceve un LSP, il router compie le seguenti azioni:

- 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;
- Se l'LSP ha lo stesso numero di sequenza di quello posseduto → Non fa nulla;
- Se l'LSP è più vecchio di quello posseduto → Trasmette al mittente il pacchetto più recente.

16.2.5 Routing - decisioni

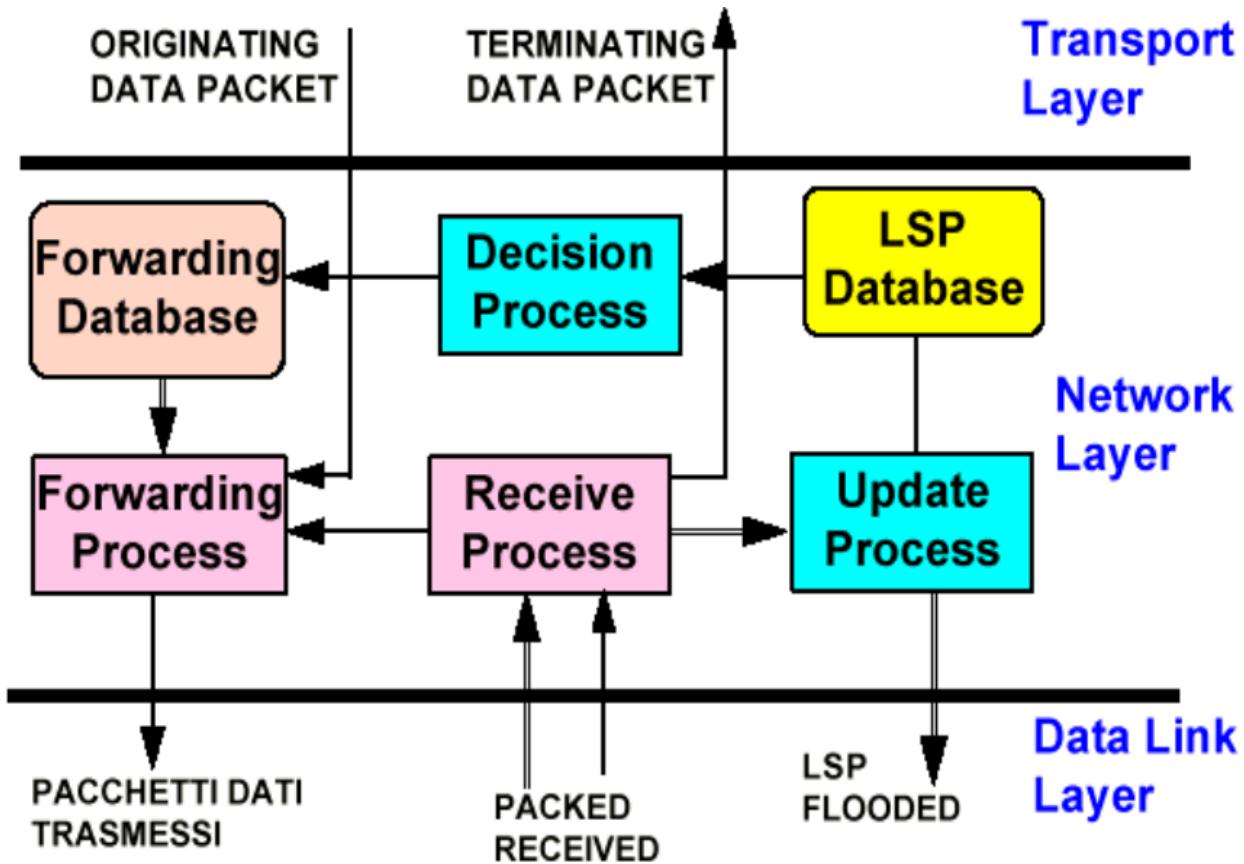
Il router elabora il Link State Database per produrre il Forwarding Database:

- Si pone come radice dello shortest-path tree;
- Cerca lo shortest-path per ogni nodo destinazione;
- Memorizza il vicino che sono sullo stesso shortest-path verso ogni nodo destinazione.

Il Forwarding Database contiene, per ogni nodo destinazione:

- L'insieme delle coppie *path, vicino*;
- Dimensione di tale insieme.

16.2.6 Architettura di un router Link State



16.2.7 Caratteristiche

Vantaggi	Svantaggi
Può gestire reti di grandi dimensioni	Molto complesso da realizzare (La prima implementazione ha richiesto alla Digital 5 anni)
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)	

16.3 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 associa 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.

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 tabella di routing per quel nodo.

Notazione:

- ▶ $c(i,j)$ → Costo collegamento da i a j
 - ◊ Infinito se non esiste 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.

```

1 // Inizializzazione:
2
3 N = {A}
4 Per tutti i nodi v
5 if (v è adiacente a A)
6   then D(v) = c(A,v)
7   else D(v) = infinity
8
9 // Loop
10 Sia w non in N tale che D(w) è minimo
11 aggiungi w a N
12 agiorna D(v) per ogni v adiacente a w e non in N:
13   D(v) = min(D(v), D(w) + c(w,v))
14 // {Il nuovo costo fino a v è o il vecchio costo, oppure il costo del cammino più breve fino a w più
15   il costo da w a v}
16 // Fino a quando tutti i nodi sono in N

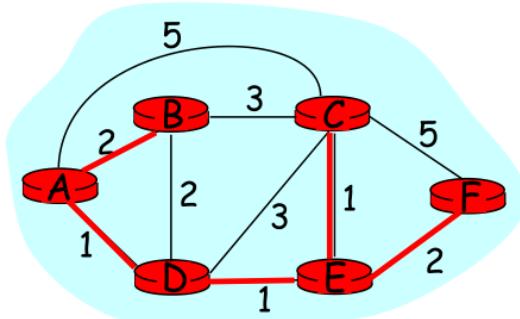
```

16.3.1 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 più brevi dal nodo sorgente a tutti gli altri nodi.

16.3.2 Esempio

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	infinity
2	ADE	2,A	3,E		4,E	
3	ADEB		3,E		4,E	
4	ADEBC				4,E	
5	ADEBCF					



Notazione:

- $c(i,j)$: costo collegamento da i a j (infinito se non c'è collegamento e 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

16.3.3 Discussione

Se il costo di un link è proporzionale al traffico di quel link, allora sono possibili oscillazioni → I collegamenti non sono simmetrici

17 Lezione del 29-11

17.1 Internet e il routing gerarchico

Al crescere del numero di core router diventa impossibile mantenerli tutti aggiornati.

Reti con *Peer 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.

Internet oggi è strutturata come un insieme di *Autonomous System* (AS) → collezione di reti amministrate da un'unica autorità.

Ognuno di questi contiene un numero limitato di reti → in questo modo la gestione delle informazioni di routing risulta più semplice.

Tra di loro gli AS sono connessi secondo relazioni:

- ▶ Paritarie (I Tier1);
- ▶ Di acquisto (I Tier3);
- ▶ Ibride (I Tier2).

Ogni AS è responsabile del routing all'interno delle sue reti:

- ▶ Routing interno;

Ogni AS deve essere identificato dal un nome → *AS number* (16bit).

Gli AS devono scambiarsi informazioni di raggiungibilità → Routing esterno (garantisce la correttezza e la consistenza delle informazioni memorizzate nelle tabelle dei router).

17.2 Routing Interno ed 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).

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 2 AS
 - ❖ **EGP** (Exterior Gateway Protocol, ormai obsoleto);
 - ❖ **BGP** (Border Gateway Protocol, con approccio *path vector*).

17.3 Tipi di AS

Un solo border router:

- ▶ Stub o *single-homed*.

Più border router:

- ▶ *multi-homed*
 - ◊ **Transit** (provider) → Accetta di essere attraversato da traffico diretto ad altri AS;
 - ◊ **Non-Transit** → Non accetta di essere attraversato da traffico diretto ad altri AS;

17.4 I gateway router

Sono speciali router dell'AS, che

- ▶ Eseguono protocolli di routing *intra-AS* con altri router appartenenti all'AS;
- ▶ Sono responsabili del routing verso destinazioni esterne al proprio AS
 - ◊ Eseguono un protocollo di routing *inter-AS* con altri gateway router.

Su questi router sono attivi contemporaneamente sia protocolli di routing IGP che di routing EGP.

17.5 Il routing multicast e l'albero di copertura

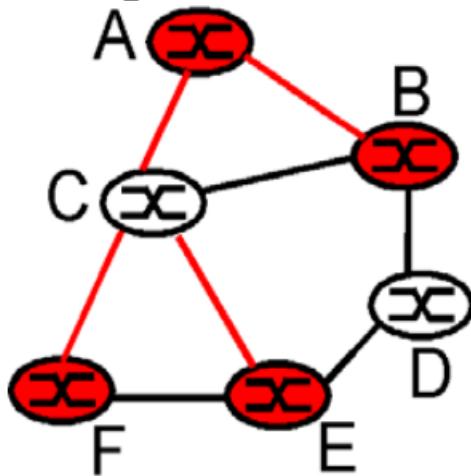
L'obiettivo dell'instradamento multicast è di trovare un albero di link che colleghi tutti i router a cui sono attaccati gli host che appartengono al gruppo multicast.

I pacchetti multicast saranno allora instradati attraverso questo albero dal sender a tutti gli host appartenenti all'albero multicast.

L'albero può contenere router che non hanno collegati host appartenenti al gruppo multicast.

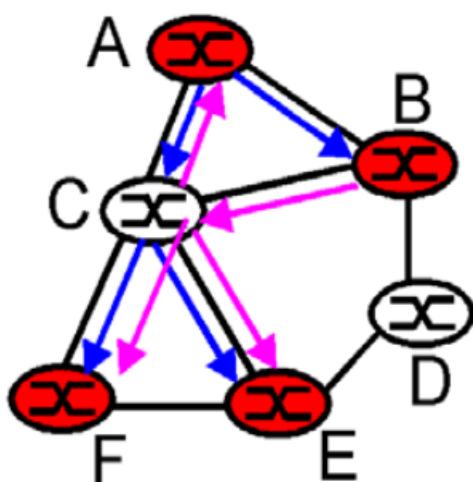
17.6 Approcci

Group-shared Tree



Singolo albero condiviso dal gruppo per distribuire il traffico di tutti i mittenti

Source-based Tree



Specifico albero d'instradamento costruito per ciascun singolo mittente

17.6.1 Group-shared tree

Problema → Il problema di trovare un albero a costo minimo (problema NP-completo). Esistono, algoritmi che approssimano la soluzione ottimale in maniera soddisfacente.

Si utilizza un approccio centralizzato definendo un nodo centrale come punto di rendezvous o nucleo:

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

17.6.2 Source-based tree

L'unione dei percorsi minimi dalla sorgente a tutte le destinazioni.

17.6.3 Differenze tra gli approcci al routing multicast

Least unicast-cost path tree → Minimizza il costo dalla sorgente ad ognuna delle destinazioni. Steiner tree → Minimizza la somma dei costi dei link dell'albero multicast.

18 Lezione del 01-12

18.1 Livello Trasporto

I protocolli di Livello Trasporto sono presenti solo negli end-point.

18.2 Livello trasporto e livello rete - differenze

Network Layer	Transport Layer
Trasferimento dati tra end-system	Trasferimenti dati tra processi (Necessita dei servizi offerti dal livello rete)

18.3 Servizi e Protocolli del livello Trasporto

Offre un canale di comunicazione logica tra applicazioni attive su differenti

Il livello rete offre un servizio inaffidabile, pertanto il livello trasporto deve rimediare:

- ▶ Aumentare l'efficienza;
- ▶ Aumentare l'affidabilità.

Questo viene garantito con:

- ▶ Controllo degli errori;
- ▶ Sequenza ordinata;
- ▶ Controllo di flusso;
- ▶ Controllo di congestione.

I protocolli di Livello Trasporto sono realizzati al di sopra del livello rete, pertanto è necessario gestire:

- ▶ L'apertura della connessione (il setup);
- ▶ La memorizzazione dei pacchetti all'interno della rete;
- ▶ Un numero elevato di connessioni (Multiplexing e Demultiplexing).

18.4 Multiplexing e Demultiplexing

% segment

Dati che sono scambiati tra processi e livello trasporto.

% Demultiplexing

Inoltrare i segmenti ricevuti al corretto processo a cui i dati sono destinati.

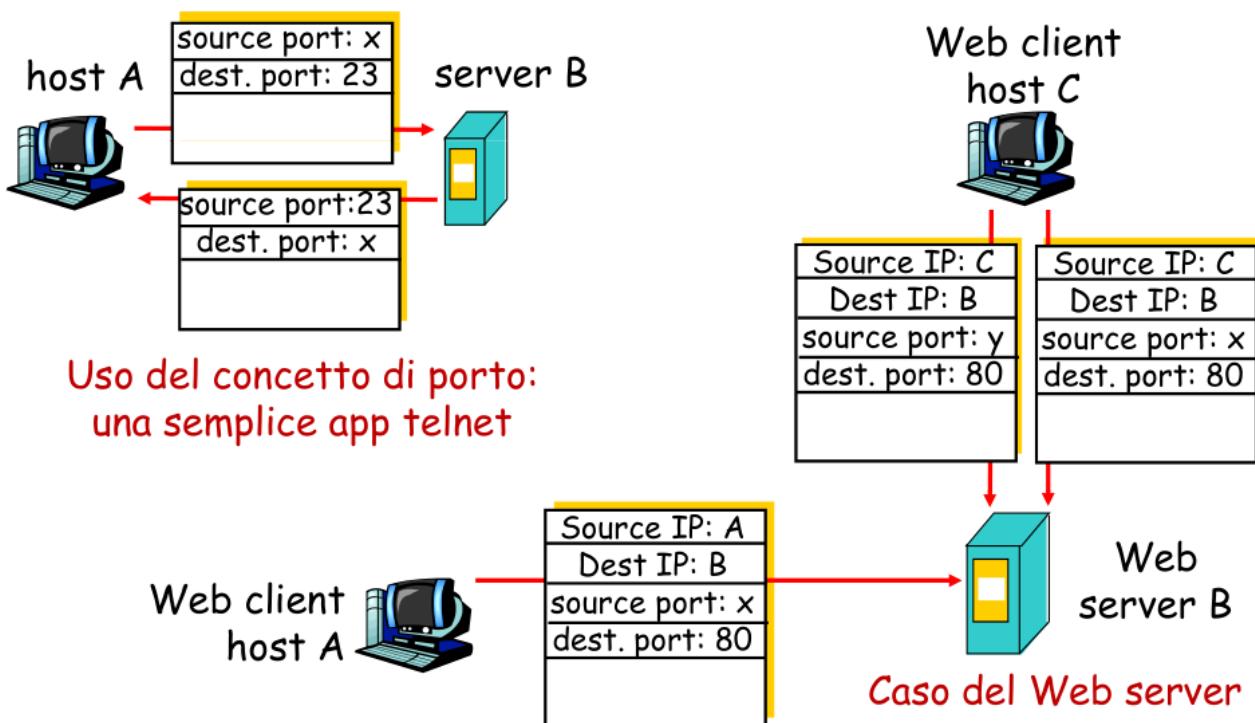
% Multiplexing

Raccogliere i dati provenienti dalle applicazioni, imbustare i dati con un header appropriato (per il demultiplexing).

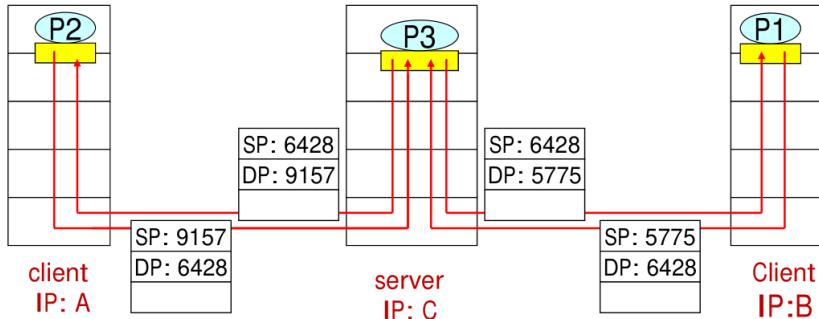
Multiplexing e Demultiplexing:

- ▶ Realizzato attraverso la coppia <indirizzo IP, numero di porto>;
- ▶ `source, dest port #` è presente in ogni segmento;
- ▶ Numero di porto ben definito per applicazioni particolari.

18.4.1 Esempio



18.4.2 Connectionless demux



Su P3 è presente un processo in ascolto sul porto UDP 6428
Il SP fornisce il *return address*.

18.5 UDP (User Datagram Protocol)

Aggiunge poco ad IP → è un servizio best-effort. I pacchetti UDP infatti possono:

- ▶ Subire perdite;
- ▶ Giungere a destinazione in ritardo, o non arrivare affatto;
- ▶ Giungere a destinazione in maniera non ordinata;

È un servizio connectionless:

- ▶ Non è prevista infatti una fase di inizializzazione;
- ▶ Ogni segmento UDP è inviato indipendentemente dagli altri.

UDP è stato introdotto perché:

- ▶ Non è necessaria la fase di inizializzazione, che introduce delay (ricordiamo infatti che DNS è basato su UDP);
- ▶ È semplice → sender e receiver non devono conservare informazioni di stato;
- ▶ Intestazione di dimensioni contenute → Abbiamo in questo modo un basso overhead;
- ▶ È assente il controllo della congestione → Le applicazioni possono inviare alla velocità desiderata (utile per alcune applicazioni, ma rischioso per la rete).

UDP viene ampiamente usato per applicazioni multimediali, dove si ha un'alta tolleranza a perdite di pacchetti e a ritardi.

Per rendere affidabile UDP:

- ▶ Gestione di avvenuta ricezione;
- ▶ Gestione degli errori.

18.5.1 Checksum UDP

L'obiettivo è quello di rilevare gli errori (i bit alterati) nel segmento trasmesso.

Mittente:

- ▶ Tratta il contenuto del segmento come una sequenza di parole da 16 bit;
- ▶ **checksum** → somma (completamento a 1) i contenuti del segmento;
- ▶ Il mittente pone il valore della checksum nel campo checksum del segmento UDP.

Ricevente:

- ▶ Calcola la checksum del segmento ricevuto;
- ▶ Controlla se la checksum calcolata è uguale al valore del campo checksum
 - ◊ No → Errore rivelato;
 - ◊ Si → nessun errore rilevato.

18.6 Realizzare una trasmissione affidabile

Se il livello rete è inaffidabile:

- ▶ Presenza di errori;
- ▶ Perdita di pacchetti;
- ▶ Ordine di pacchetti non garantito;
- ▶ Duplicazione di pacchetti.

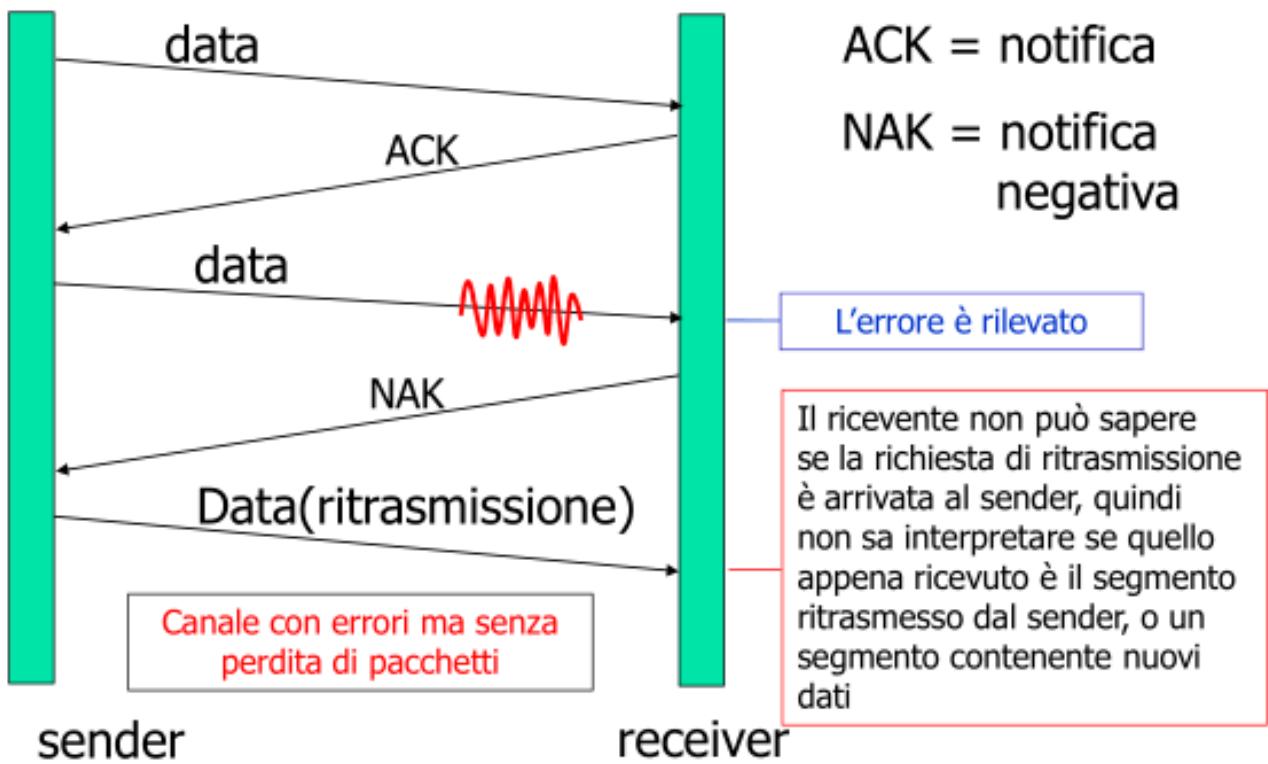
È inoltre necessario tenere in considerazione:

- ▶ Le risorse del PC ricevente → controllo di flusso;
- ▶ 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 compiere una scelta tra:
 - ◊ Correzione degli errori;
 - ◊ Notifica al mittente → Richiesta trasmissione.

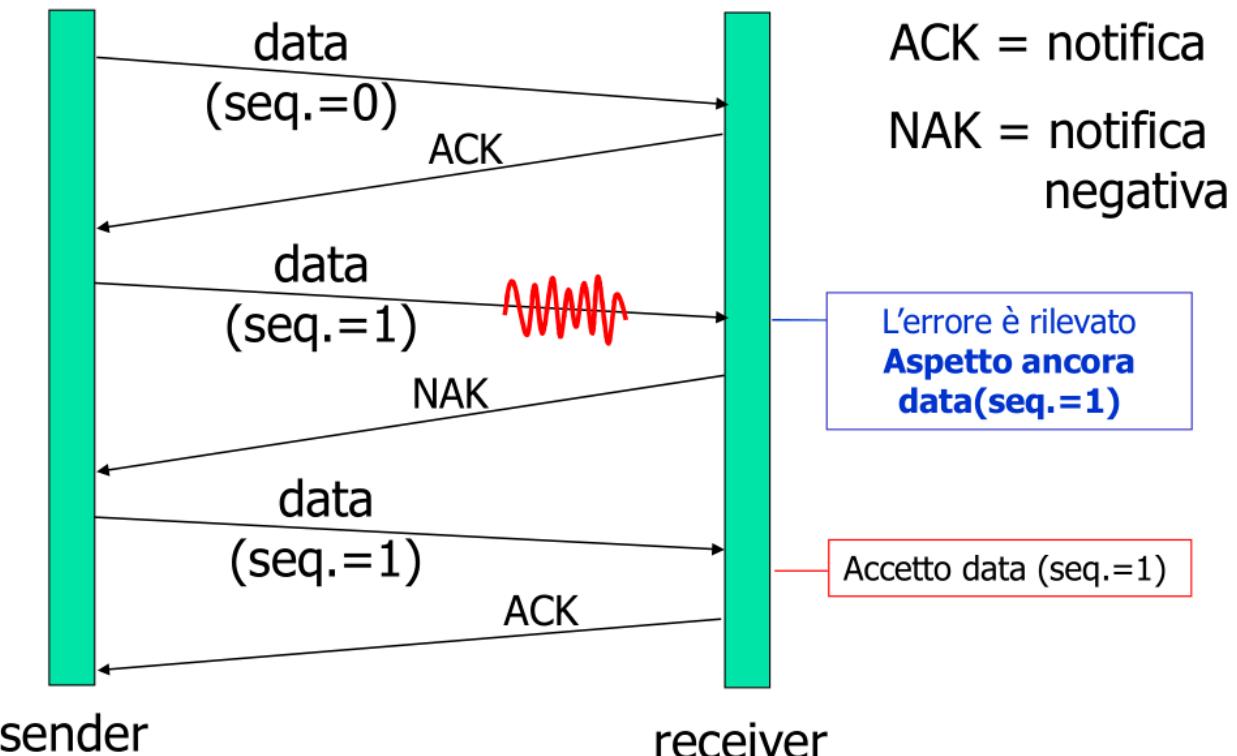
La prima soluzione introduce complicazioni. La seconda introduce possibili duplicazioni sulla rete che il ricevente non è in grado di interpretare



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, il **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.

18.6.1 Esempio



18.6.2 Un protocollo senza notifica negativa

Stessa funzionalità, facendo affidamento solo sugli ACK (notifica).

Il destinatario invia quindi una notifica 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 → ritrasmissione del pacchetto corrente.

Nel caso di canale che introduce perdita di pacchetti è necessario introdurre il parametro temporale → un **timeout**. È possibile fare a meno dei NAK.

18.7 Aumentare l'efficienza

In alternativa allo **stop & wait** possiamo fare affidamento su **pipelining**. Il mittente invia pacchetti prima di ricevere il riscontro dei precedenti:

- ▶ Occorre aumentare l'intervallo dei numeri di sequenza;
- ▶ Occorre aggiungere buffer nel sender e/o receiver.

Altre alternative per il pipeling sono il **go-Back-N** e il **selective repeat**

18.7.1 Go Back N

Sender:

- ▶ Nell'header del segmento k-bit per il numero di sequenza;
- ▶ Una finestra di max N pacchetti senza riscontro;
- ▶ ACK numerati;
- ▶ ACK cumulativo → Ricevere ACK(n) significa che tutti i pkts precedenti l' n -esimo sono stati ricevuti correttamente;
- ▶ Un timer per ogni pacchetto trasmesso e non riscontrato;
- ▶ **timeout (n)** ritrasmetti pkt n e tutti i pacchetti che seguono n .

18.7.2 Selective repeat

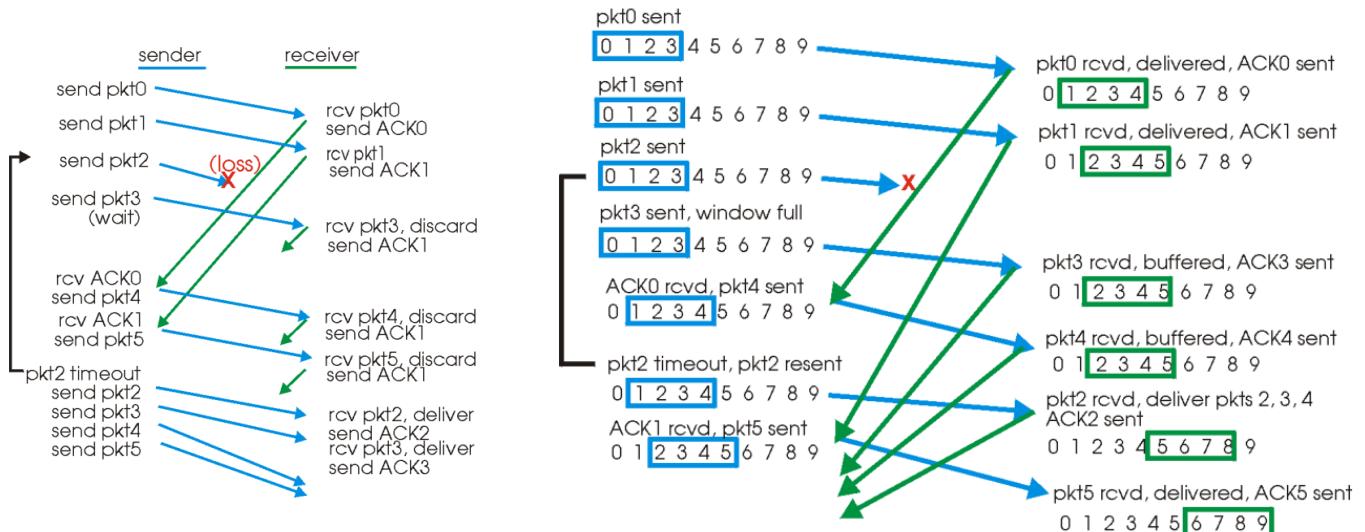
Il ricevente invia riscontri specifici per tutti i pacchetti ricevuti correttamente → buffer dei pacchetti, se necessario, per eventuali consegne in sequenza al livello superiore.

Il mittente ritrasmette soltanto i pacchetti per i quali non ha ricevuto un ACK → il 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.

18.7.3 A confronto

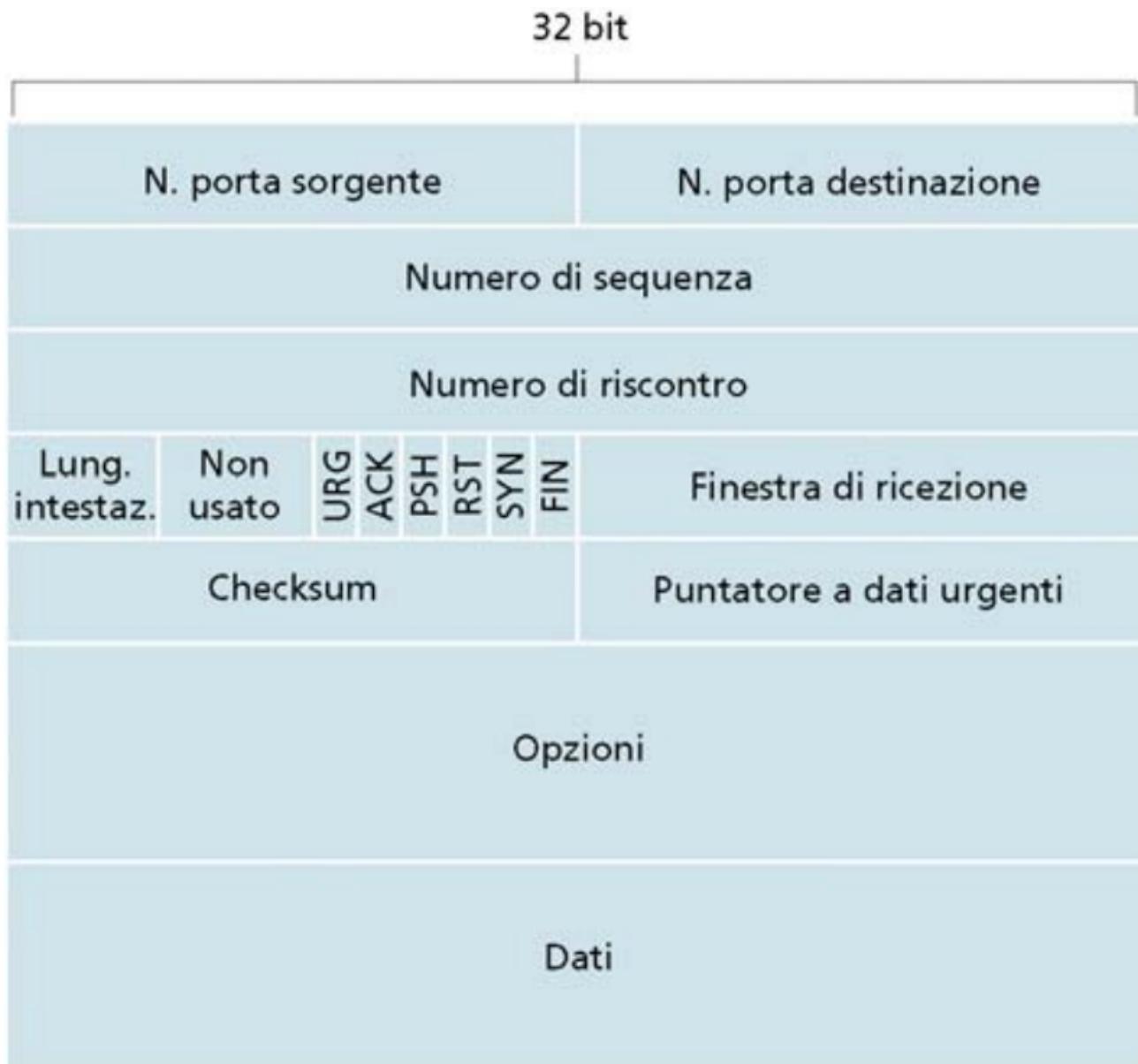


19 Lezione del 06-12

19.1 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 impostando una finestra (TCP window);
- ▶ Buffers su mittente e ricevente;
- ▶ Full duplex data
 - ◊ Flusso di dati bi-direzionale all'interno delle stessa connessione
 - ◊ MSS → Maximum Segment Size;
- ▶ Connection-oriented
 - ◊ **Handshaking** prepara mittente e ricevente prima della comunicazione;
- ▶ Controllo di flusso → Il mittente non invia più quando il ricevente non può accettare

19.2 Struttura del segmento TCP



19.3 TCP - Protocol Data Unit

- ▶ HLEN → 4 bit, che contengono un numero intero che indica la lunghezza dell'intestazione TCP del datagramma in parole da 32 bit. Questa informazioni è necessaria perché il campo opzioni è di lunghezza variabile (In generale il campo opzioni è vuoto e ha una lunghezza di 20byte);
- ▶ Porta → contengono i numeri di porta di protocollo TCP che identificano gli applicativi alle estremità della connessione;
- ▶ 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 contenute 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.
- 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 pseudo-intestazione al datagramma, per effettuare così un controllo anche sugli indirizzi IP di destinazione e provenienza.

La pseudo-intestazione viene creata e posta in testa al datagramma TCP. Viene inserito in essa un ulteriore byte di 0 per raggiungere un multiplo di 16 bit.

Successivamente viene calcolata la checksum su tutto il messaggio così formato, viene scartata la pseudo-intestazione e passato il datagramma al livello IP.

In fase di ricezione, il livello TCP ricrea la pseudo-intestazione interagendo con l'IP sottostante, calcola la checksum e verifica la correttezza del messaggio ricevuto. In caso di errore il datagramma verrà scartato (verrà quindi ritrasmesso).

19.4 TCP - Principali opzioni

- **Maximum TCP payload** → Durante la fase di connessione, ciascun end-point annuncia la massima dimensione di payload che desidera accettare; la minima tra le 2 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 (alternativa al **go back n**, che prevede la ritrasmissione di tutti i segmenti).

19.5 TCP - Caratteristiche

Riscontro e ritrasmissione → Consiste nella ritrasmissione di un segmento se non è giunta conferma entro un tempo massimo (timeout).

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

19.6 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.

TCP invia riscontri cumulativi.

19.7 Numeri di sequenza e ACK di TCP

Numeri di sequenza:

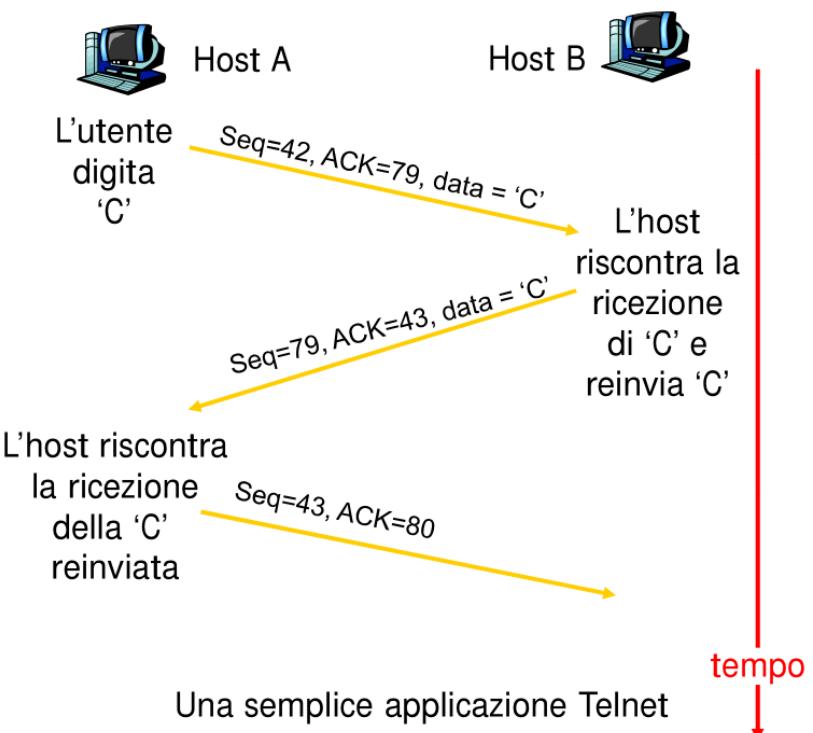
- “numero” del primo byte del segmento nel flusso di byte

ACK:

- numero di sequenza del prossimo byte atteso dall’altro lato
- ACK cumulativo
- ACK duplicato =NACK

D: come gestisce il destinatario i segmenti fuori sequenza?

- R: la specifica TCP non lo dice – dipende dall’implementatore



19.8 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.

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

19.9 TCP - Trasferimento dati affidabile

Le ritrasmissioni sono avviate da:

- ▶ TCP crea un servizio di trasferimento dati affidabile sul servizio inaffidabile IP;
- ▶ Pipeline dei segmenti;
- ▶ ACK cumulativi;
- ▶ TCP utilizza un solo timer di ritrasmissione.

- ▶ 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.

19.10 TCP - 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;
- ▶ 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.

20 Lezione del 13-12

20.1 Tecnologie per le LAN

Compiti del livello Data Link → servizi, **rilevamento/correzione** degli errori, accesso al canale.

20.2 Indirizzi IP e indirizzi LAN

Indirizzi IP a 32 bit:

- ▶ Indirizzi di *livello rete*;
- ▶ Usati per permettere la corretta consegna del pacchetto ad un destinatario collegato alla rete.

Indirizzi LAN (o MAC o fisici):

- ▶ Usati per permettere la trasmissione di una frame da una scheda di rete ad un'altra scheda con cui sussiste un collegamento diretto (stessa rete fisica);
- ▶ Indirizzi MAC di 48 bit (per la maggior parte delle LAN) cablati nelle ROM delle schede di rete.
- ▶ Ogni scheda di rete su una LAN ha un indirizzo LAN univoco;
- ▶ La distribuzione degli indirizzi MAC è gestita da IEEE
 - ◊ I produttori di schede di rete detengono una porzione degli indirizzi MAC (per garantirne l'univocità).

20.3 Gli indirizzi MAC (Medium Access Control)

Specifico per ogni LAN e risolve il problema della condivisione del mezzo trasmissivo.

Ne esistono di diversi tipi:

- ▶ Ad allocazione di canale fissa;
- ▶ Ad allocazione di canale dinamica;
- ▶ Deterministici;
- ▶ Statistici;
- ▶ ...;

Si compongono di 2 parti grandi 3 byte ciascuna:

- ▶ I 3 byte più significativi indicano il lotto di indirizzi acquistato dal costruttore della scheda, detto anche *vendor code* o *OUI* (Organization Unique Identifier);
- ▶ I 3 meno significativi sono una numerazione progressiva decisa dal costruttore

Ogni scheda di rete nel momento in cui riceve un pacchetto lo passa ai livelli superiori nei seguenti casi:

- ▶ Broadcast (Tutte le stazioni) → Sempre;
- ▶ Single (Singola stazione) → Se il DSAP è uguale a quello HW della scheda (scritto in una ROM) o a quello caricato dal SW in un apposito buffer;
- ▶ Multicast (Di un gruppo di stazioni) → Se ne è stata abilitata la ricezione via SW.

20.4 Ethernet

Metcalf inizialmente immagina di gestire la rete Ethernet facendo uso di cavi coassiale rinforzati.

È la tecnologia dominante per le LAN.

20.4.1 Struttura della Frame Ethernet

L'interfaccia di rete del mittente incapsula i datagrammi IP (o altri pacchetti di livello rete) in *frame Ethernet*. Questo è composto da:

- ▶ Preambolo (8 byte) di cui
 - ◊ 7 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;
- ▶ Type (2 byte)
 - ◊ Indica il protocollo di livello rete sovrastante, principalmente IP, ma altri protocolli sono supportati;
- ▶ CRC (4 byte)
 - ◊ Controllo effettuato alla destinazione → Se l'errore è rilevato, la frame viene scartata.

Ethernet attraverso il carrier service gestisce accessi multipli, gestendone le possibili rilevazioni di collisioni (impiego del protocollo CSMA/CD).

20.4.2 CSMA/CD

Jam Signal → Consente alle altre stazioni di accorgersi dell'avvenuta collisione.

Backoff esponenziale. L'obiettivo è quello di utilizzare un algoritmo per adattare i successivi tentativi di ri-trasmissione al carico corrente della rete:

- ▶ In presenza di sovraccarico il tempo di attesa casuale sarà maggiore.

20.4.3 Ethernet - 10Base2

Topologia a bus su cavo coassiale sottile.

Si impiegano ripetitori per collegare più segmenti. Questi ritrasmettono i bit in entrata da un'interfaccia verso le altre (entità di livello fisico).

Lo svantaggio rispetto al primo è una diminuzione dell'area di copertura (dovuta ad una sezione del cavo minore).

20.4.4 Ethernet - 100BaseT

Topologia a stella, mediante un concentratore (hub) al quale gli host sono collegati con i doppini intrecciati.

Permette una distanza massima tra nodo e hub pari a 100 metri.

Gli hub possono disconnettere le schede mal funzionanti.

Possono:

- ▶ Fornire informazioni utili al monitoraggio;
- ▶ Collezionare statistiche per effettuare previsioni, agevolando il compito degli amministratori della LAN.

20.4.5 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*.

20.5 Interconnettere più LAN

L'interconnessione di più LAN comporta:

- ▶ Una limitata quantità di banda disponibile, considerato che su una singola LAN tante stazioni dovrebbero condividere la banda;
- ▶ Estensione limitata;
- ▶ Dominio di collisione troppo ampio (una trasmissione può collidere con molte altre);
- ▶ Numero limitato di stazioni.

20.6 Hub

Dispositivo di livello fisico → Si tratta di ripetitori di bit (riproducono i bit in ingresso ad un'interfaccia su tutte le altre).

Possono essere organizzati in una gerarchia con una backbone hub al livello più alto.

20.6.1 Caratteristiche

- ▶ Ogni LAN collegata è considerato come un segmento di LAN;
- ▶ Gli hub non isolano i domini di collisione;

20.6.2 Vantaggi

- ▶ Dispositivi semplici e poco costosi;
- ▶ L'organizzazione Multi-livello garantisce una parziale tolleranza ai guasti → porzioni di LAN continuano a funzionare in caso di guasto ad uno o più hub;
- ▶ Estendere la massima distanza esistente tra i nodi.

20.7 Bridge

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

Isolano i domini di collisione, grazie alla loro capacità di porre le frame in un buffer.

Nel momento in cui 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

20.7.1 Vantaggi

- ▶ Isolamento dei domini di collisioni;
- ▶ Non introducono limitazioni sul numero massimo delle stazioni;
- ▶ Possono collegare differenti tecnologie;
- ▶ Trasparenti → Non richiedono alcuna modifica negli adattatori del PC.

20.7.2 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).

Si organizzano i bridge secondo uno spanning tree, disabilitando alcune interfacce.

21 Lezione del 15-12

21.1 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.

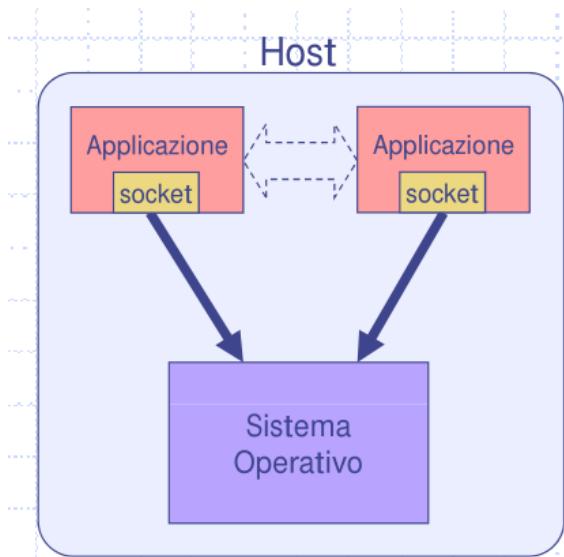
Si presentano sotto-forma di un'API (Application Programming Interface), cioè un insieme di funzioni scritte in C, che le applicazioni possono invocare per ricevere il servizio desiderato.

Rappresentano una estensione delle API di UNIX per la gestione dell' I/O su periferica standard.

21.1.1 Iterazione tra applicazione e SO

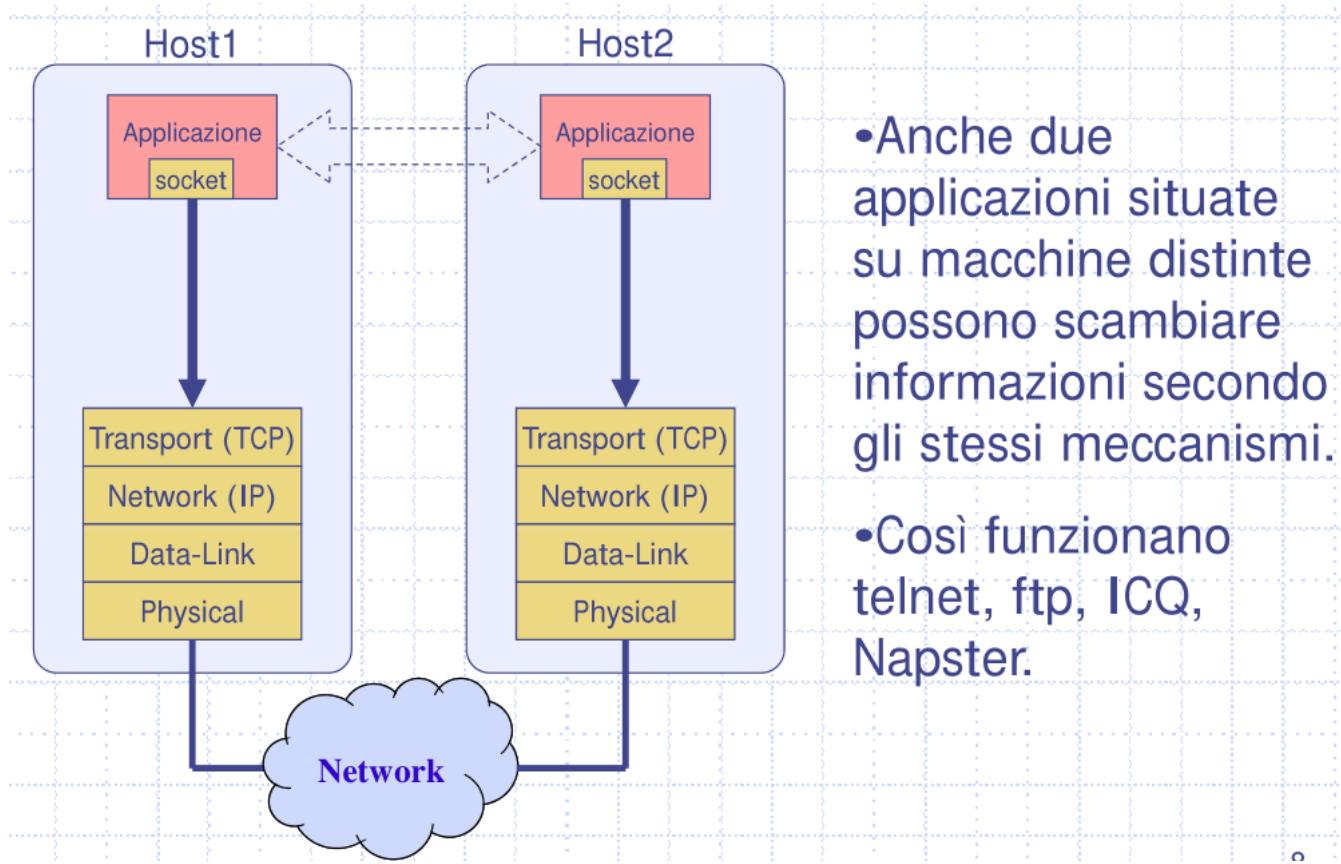
1. L'applicazione chiede al SO di utilizzare i servizi di rete;
2. Il SO crea una socket e la restituisce all'applicazione → restituisce un **socket descriptor**;
3. L'applicazione utilizza la socket;
4. L'applicazione chiude la socket e la restituisce al SO.

21.1.2 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.

21.1.3 Comunicazione remota via TCP/IP



21.2 Il problema della connessione

È necessario che il chiamante conosca l'indirizzo del chiamato e che il chiamato sia a sua volta in attesa di eventuali comunicazioni.

21.2.1 Paradigma Client-Server

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.

21.3 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.

In IP:

- ▶ `local-addr` e `foreign-addr` rappresentano indirizzi IP;
- ▶ `local-process` e `foreign-process` rappresentano numeri di porto.

21.4 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.

Un server che accetti più comunicazione contemporaneamente si definisce **concorrente**.

Un server che accetti una sola comunicazione alla volta è detto **iterativo** → In questo caso una richiesta può essere servita solo quando la precedente si è già conclusa.

21.5 Paradigma di comunicazione 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 come una sorta di *tubo*, in cui ciò che viene inserito arriverà inalterato dall'altro lato e nello stesso ordine con cui è stato immesso.

NB → Non è detto che vengano mantenuti i limiti dei messaggi.

21.6 Paradigma di comunicazione Datagram

In una comunicazione Datagram (detta anche *connectionless*) il canale:

- ▶ Trasporta messaggi;
- ▶ Non è affidabile;
- ▶ È condiviso;
- ▶ Non preserva l'ordine delle informazioni.

21.7 Paradigma di comunicazione Datagram e Connection-Oriented a confronto

Paradigma	Vantaggi	Svantaggi
Datagram	Basso overhead	Nessun controllo sulla consegna
Connection oriented	Affidabilità Controllo di flusso	Overhead per instaurare la connessione

21.8 Naming/Binding

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

21.9 Byte Stream e Datagram

È possibile impostare il protocollo che verrà per il trasferimento dei dati.

Nel caso di socket abbiamo 2 opzioni:

1. **byte-stream** → I dati vengono trasferiti come sequenza ordinata ed affidabile di byte (**SOCK_STREAM**);
2. Datagram → I dati vengono inviati come messaggi indipendenti ed inaffidabili (**SOCK_DGRAM**).

21.10 Progettazione di un Server e Client TCP

Server:

1. Creazione di un end-point
 - ▶ Richiesta al SO;
2. Collegamento dell'end-point ad una porta
 - ▶ Ascolto sulla porta
 - ❖ Processo sospeso in attesa;
3. Accettazione della richiesta di un client;
4. Letture e scritture sulla connessione;
5. Chiusura della connessione.

Client:

1. Creazione di un end-point
 - ▶ Richiesta al SO;
2. Creazione della connessione;
3. Lettura e scrittura sulla connessione;
4. Chiusura della connessione.

21.11 Progettazione di un Server e Client UDP

Server:

1. Creazione di un end-point
 - ▶ Richiesta al SO;
2. Collegamento dell'end-point ad una porta
 - ▶ open passiva in attesa di ricevere datagram;
3. Ricezione ed invio di datagram;
4. Chiusura dell'end-point.

Client:

1. Creazione di un end-point
 - ▶ Richiesta al SO;
2. Invio e ricezione di datagram;
3. Chiusura dell'end-point.

21.12 Marshalling dei parametri

Quando si invia un dato sulla rete, in generale, non si hanno informazioni sull'architettura dell'host ricevente. È necessario pertanto che i dati in transito.

È necessario pertanto che i dati in transito siano codificati secondo una convenzione standard.

Con il termine *marshalling* si intende la traduzione di un'informazione in un formato prefissato e comprensibile universalmente. L'operazione inversa prende il nome di *un-marshalling*.

È un'operazione tipica del sesto livello della pila OSI il cui intento è di assicurare portabilità ad un programma.

Proprio per la grande varietà di architetture, una serie di funzioni è stata prevista.

Vanno sotto il nome di *Byte Ordering Routines* → Da host byte order a TCP/IP byte order (big endian) e viceversa.

Sui sistemi che adottano la stessa convenzione fissata per internet queste routines sono implementate come *null-macros* (funzioni vuote).

```
1 #include <sys/types.h>
2 #include <netinet/in.h>
3
4 u_long htonl (u_long hostlong); // host to net long
5 u_short htons (u_short hostshort); // host to net short
6 u_long ntohl (u_long hostlong); // net to host long
7 u_short ntohs (u_short hostshort); // net to host short
```

È implicito in queste funzioni che uno short occupi 16 bit e un long ne occupi 32.

21.13 Gestione delle opzioni sulle socket

Le opzioni che si possono impostare su una socket ne influenzano il comportamento di default. Sono offerte diverse possibilità per gestire le opzioni:

- ▶ La funzione `setsockopt()` (applicabile esclusivamente alle socket);
- ▶ La funzione `fcntl()` (system call standard UNIX);
- ▶ La funzione `ioctl()` (system call standard UNIX);

21.14 Raw socket

La chiamata:

```
1 sockfd = socket(AF_INET, SOCK_RAW, protocol);
```

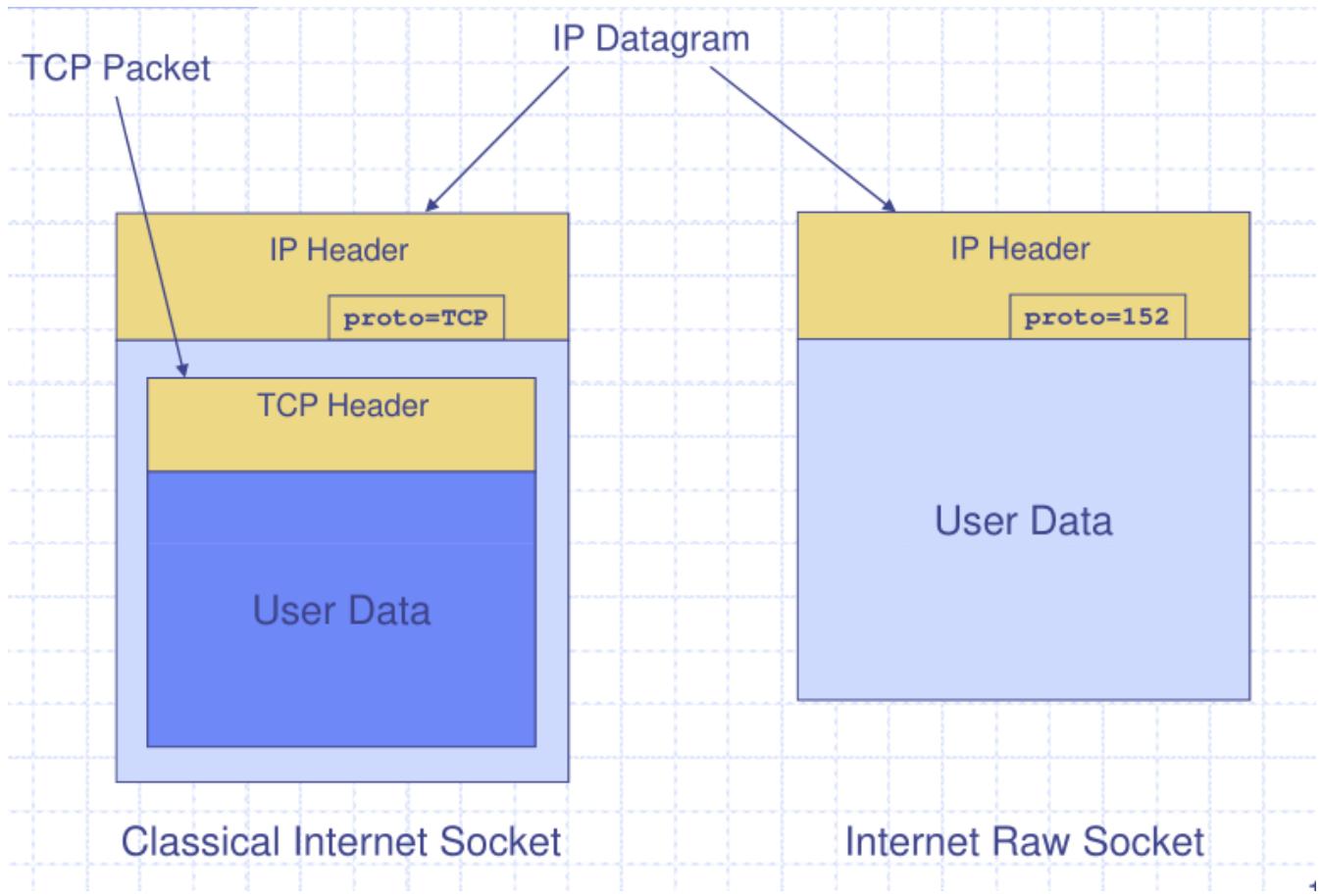
Produce l'istanziazione di una socket definita Raw.

Una tale socket si aggancia direttamente al livello IP con la conseguenza che è possibile di fatto creare un nuovo protocollo di quarto livello.

Ogni messaggio viene quindi incapsulato direttamente in un pacchetto IP.

Il valore di `protocol` identifica il numero che si desidera assegnare al protocollo in questione. Questo numero sarà pertanto il valore dell'omonimo campo nell'intestazione dei datagrammi IP inviati da una tale socket.

21.14.1 Pacchetti inviati



21.14.2 Accesso all'header IP

Una volta allocata una `raw socket`, con una chiamata del tipo:

```
1 int hincl = 1;  
2 setsockopt(sockfd, IPPROTO_IP, IP_HDRINCL, &hincl, sizeof(hincl));
```

È possibile gestire completamente l'accesso all'intestazione del pacchetto IP, cambiando i singoli campi a proprio piacimento.

Questa tecnica è destinata a soddisfare un campo di esigenze molto ristrette.