PAS 2013/14
SISTEMI E RETI DI CALCOLATORI PER L'INSEGNAMENTO
Lezione n.1
Reti: Concetti base

21/05/2014

Laura Ricci

PROGRAMMA E MATERIALE DIDATTICO

- Introduzione: lo stack dei protocolli Internet
- Livello applicazione: applicazioni client server e peer to peer
- Livello di trasporto:
  - Protocolli conlectionless:UDP
  - Protocolli orientati alla comunicazione: TCP
  - Controllo della congestione
- Livello di rete: algoritmi di istradamento
- Livello data link e reti locali
- Esercitazioni: sviluppo di semplici programmi distribuiti

Libri di testo:

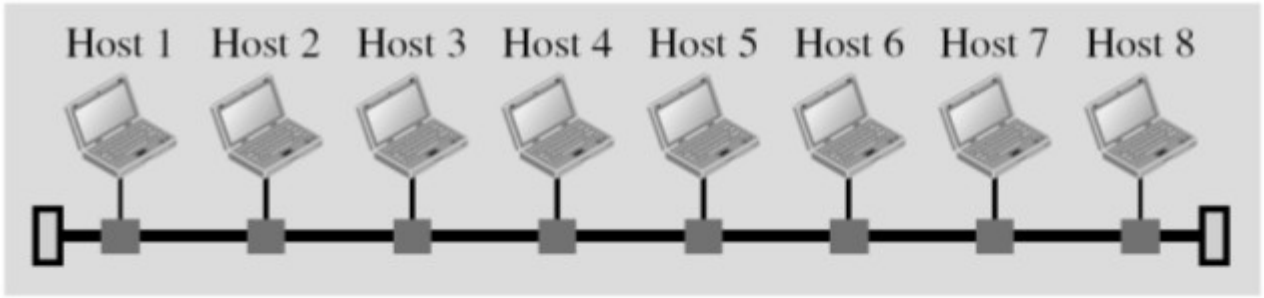
Kurose, Ross: Reti di calcolatori e internet:un approccio top down,

Forouzan, Mosharraf: Reti di calcolatori: un approccio topo-down

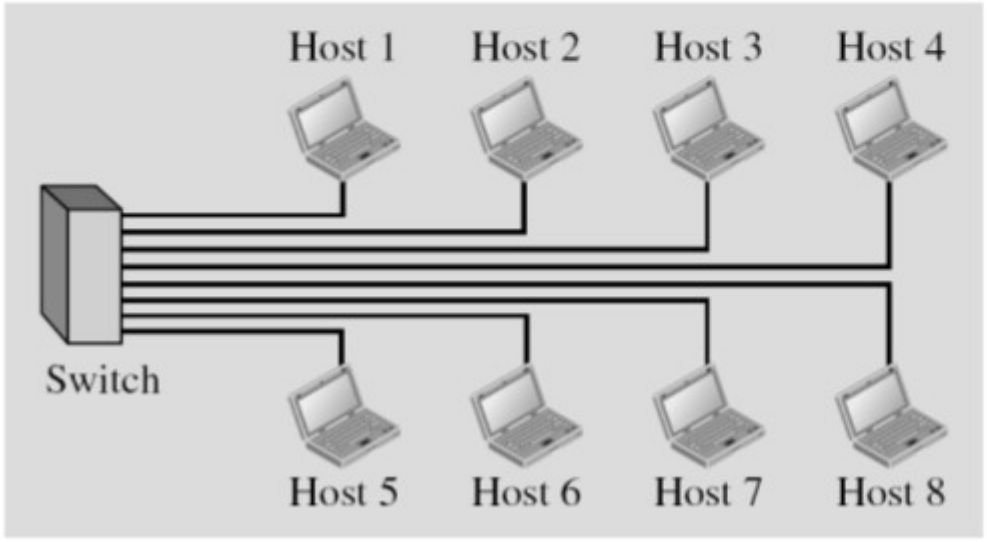
RETI: DEFINIZIONI

- Internet: interconnessione di una grande molteplicità di reti, rete di reti
- Rete: interconnessione di dispositivi in grado di scambiarsi informazioni:
- Dispositivi:
  - end system: host e server
  - router
  - switch
  - modem
- Mezzi di collegamento
  - mezzi cablati
  - wireless

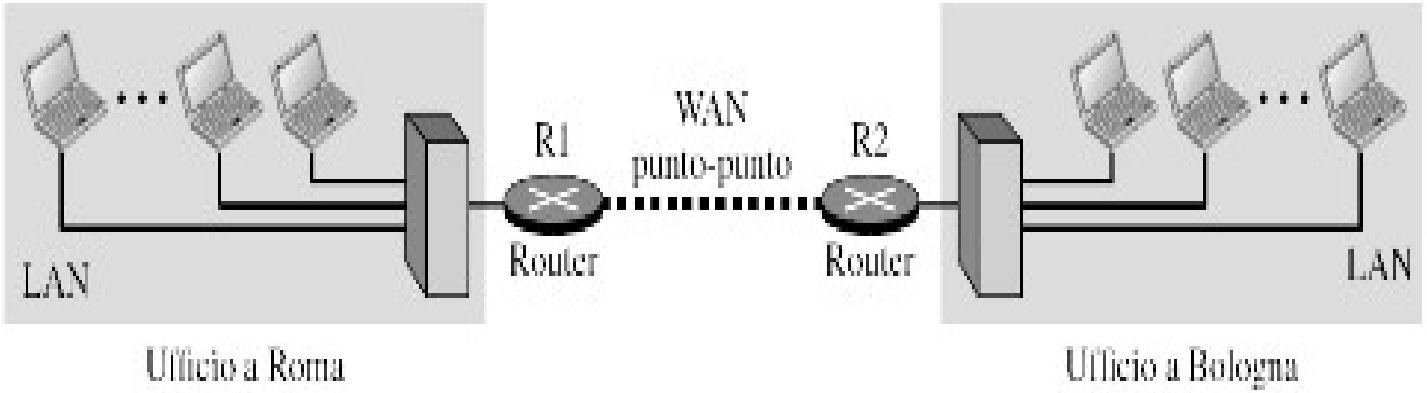
LOCAL AREA NETWORK



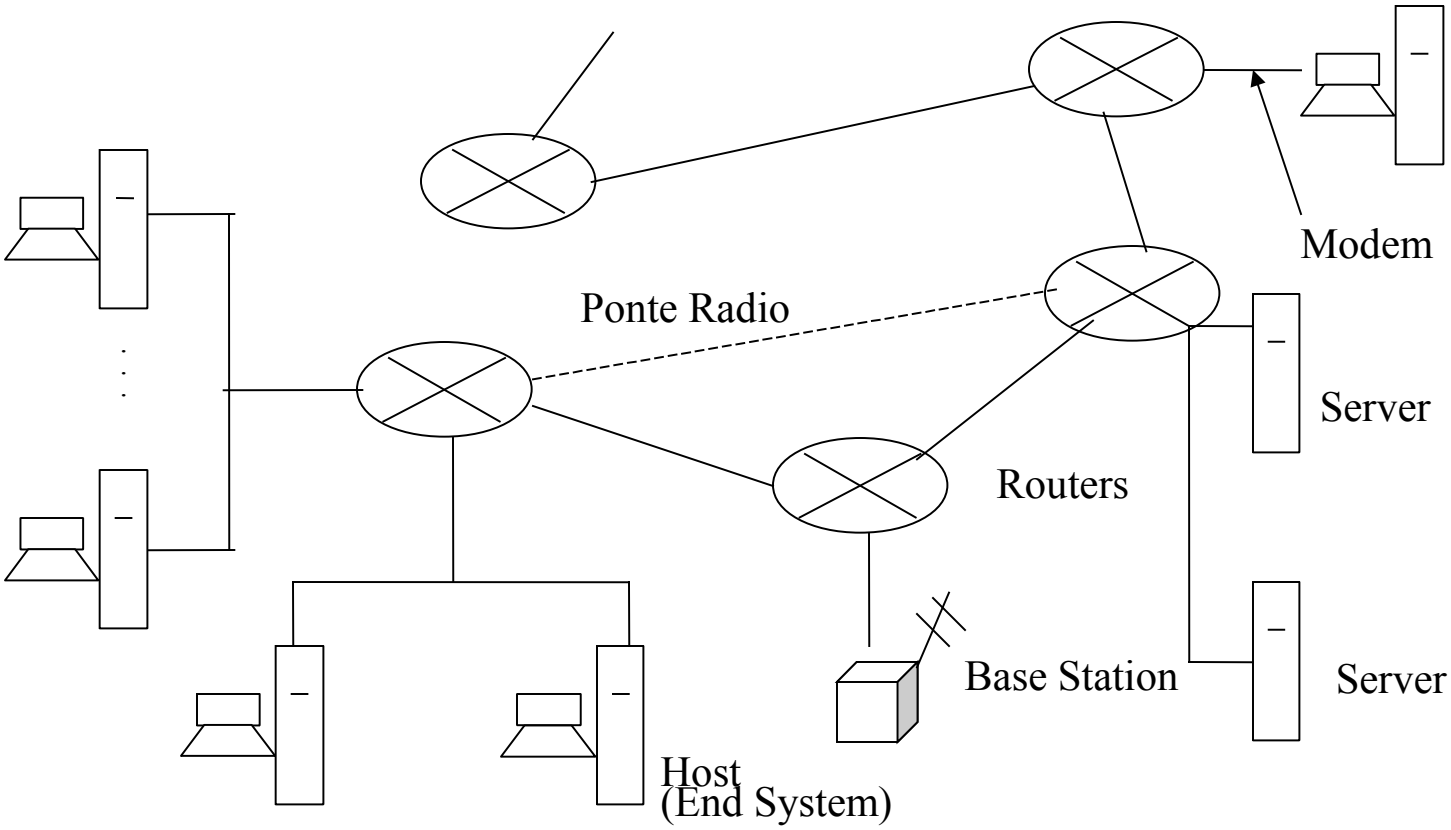
a. LAN con cavo condiviso (obsoleta)



INTERNETWORKS



INTERNET: COMPONENTI FONDAMENTALI



INTERNET: COMPONENTI

Componenti hardware fondamentali di una rete:

- Hosts (end systems): PCs, workstations + servers, PDA, mobile computers and telephones.

- Servers: hosts dedicati a fornire servizi di rete. Ad esempio server http memorizza e fornisce accesso ad un insieme di pagine web.

- Routers: dispositivi che instradano (routing) i messaggi scambiati tra gli hosts. Possiedono più links di collegamento alla rete. Instradano l'informazione in arrivo da un link di ingresso su uno dei links di uscita, scelto secondo un algoritmo di routing

- Communication Link: supporto fisico che permette la comunicazione host/host, host/router, router/router (esempio: cavo coassiale, fibre ottiche, ponti radio…)

INTERNET: APPLICAZIONI DISTRIBUITE

- applicazioni distribuite: un programma distribuito comprende diverse componenti. Ogni componente viene eseguita su un host diverso. Le componenti cooperano scambiandosi dati

- paradigma Client/Server: una componente, il client, in esecuzione su un host (in genere un PC) richiede un servizio ad un'altra componente, il server (in genere in esecuzione su una macchina con maggiore potenza di calcolo)

- richiesta del servizio, ricezione dei risultati richiedono uno scambio di messaggi tra client e server

Esempi di applicazioni client/server

- web
- e-mail
- ftp (file transfer protocol)
- ssh (login remoto)

Paradigma peer to peer

- File sharing P2P (Bittorrent, eMule, Spotify,...)

INTERNET: PROTOCOLLI DI RETE

- Protocollo di rete: insieme di regole che definiscono il formato, l'ordine dei messagi scambiati tra due o più entità comunicanti e le operazioni eseguite in seguito all'invio/ricezione dei messaggi.

- L'invio/ricezione di un messaggio prevede l'esecuzione di diversi protocolli a diversi livelli di astrazione.

- Esempio: richiesta di una pagina ad un web server, protocollo a livello applicazione:

```
client:  GET /public_html/file1 HTTP/1.3
         host: www.unipi.it    ….
server:  ….
     Content-Length: 6821
     Content-type: text/HTML …
```

INTERNET: SUPPORTO PER LE APPLICAZIONI

- La rete definisce un insieme di supporti hardware/software (servizi) per la comunicazione tra gli hosts

- Connection Oriented Service: garantisce che ogni messaggio inviato da un mittente M ad un destinatario D venga recapitato a D e che i messaggi vengono recapitati a D nell'ordine con cui sono stati spediti, cioè se m_1 viene spedito prima di m_2, D riceve m_1 prima di m_2.

- Connectionless Service: non garantisce che un messaggio spedito verrà recapitato. Non garantisce l'ordinamento dei messaggi.

Entrambe i servizi precedenti sono best effort: non forniscono garanzie circa il tempo richiesto per recapitare un messaggio al destinatario.

INTERNET: CONNECTION ORIENTED SERVICE

Connection-Oriented Service:

- client e server si scambiano alcuni messaggi di controllo prima di scambiarsi i dati veri e propri
- poi client e server si scambiano i dati veri e propri (ad esempio GET+risposta dal server)
- lo scambio di questi messaggi consente di allocare strutture dati nel mittente/destinatario necessarie per controllare la trasmissione
- Strutture dati necessarie per gestire la connessione= buffer+variabili di stato
- i messaggi di controllo scambiati sono definiti da un protocollo a livello trasporto, eseguito in modo trasparente rispetto alla applicazione
- Solo i due hosts che hanno stabilito la connessione sono consapevoli della connessione, cioè mantengono le strutture dati per gestirla. I routers non registrano le connessioni stabilite tra gli hosts.

INTERNET: CONNECTION ORIENTED SERVICE

- Connection Oriented Service, servizi forniti
  - Trasferimento di dati Affidabile: Basato sull'utilizzo di messaggi di ack + identificatori di sequenza associati ai messaggi
  - Controllo del flusso: consente al mittente di regolare la velocità di trasmissione dei messaggi in modo da non riempire il buffer del destinatario
  - Controllo della congestione: consente di diminuire la frequenza di invio dei messaggi in modo da evitare la congestione dei routers intermedi
- Servizio fornito dal protocollo TCP (Trasmission Control Protocol)
- Utilizzato da protocollo HTTP, FTP, remote login, e-mail

INTERNET: CONNECTIONLESS SERVICE

- *Connectionless Service:* nessuna garanzia sulla affidabilità della trasmissione

- i pacchetti spediti possono essere:
  - persi durante la trasmissione
  - arrivare fuori ordine
  - sovrascrivere altri pacchetti presenti nel buffer del destinatario e spediti in precedenza(non esiste un meccanismo di controllo del flusso)

- servizio minimale fornito dal protocollo UDP (User Datagram Protocol) a livello trasporto

- utilizzato per la spedizione di dati multimediali (video-conferenza, audio on-demand,…)

ORGANIZZAZIONE A LIVELLI

- Strutturazione a livelli della architettura di rete: consente di modularizzare la progettazione di un sistema complesso (es: una rete)

- Idea generale:

  - partire dai servizi base offerti dai dispositivi hardware L_0 (esempio invio di un bit su un link fisico)
  - definire una sequenza di livelli L_1, …L_i,…L_n che forniscono servizi sempre più astratti
  - il livello L_i utilizza i servizi definiti al livello L_{i-1}
  - il livello L_i offre servizi al livello L_{i+1}
  - i servizi offerti a livello L_i sono implementati mediante un insieme di componenti C_1, …C_i,…C_n che cooperano mediante un protocollo P_i

ORGANIZZAZIONE A LIVELLI

- Per ogni livello L_i, si definisce un protocollo P_i che stabilisce le regole con cui cooperano le componenti a livello L_i

- Il protocollo P_i viene eseguito in modo distribuito dalle componenti

- Le componenti cooperano scambiandosi un insieme di messaggi (Protocol Data Units)

- i-PDU = Protocol Data Units scambiati a livello L_i

- Quando un host A invia un i-PDU all'host B, l'i-PDU viene passato dal Livello L_i al livello L_{i-1}

$\Rightarrow$

il livello L_i sfrutta i servizi del livello L_{i-1} per l'invio dell'i-PDU

ORGANIZZAZIONE A LIVELLI

- quando un host A invia un i-PDU all'host B, l'i-PDU viene passato dal Livello L_i al livello L_{i-1}

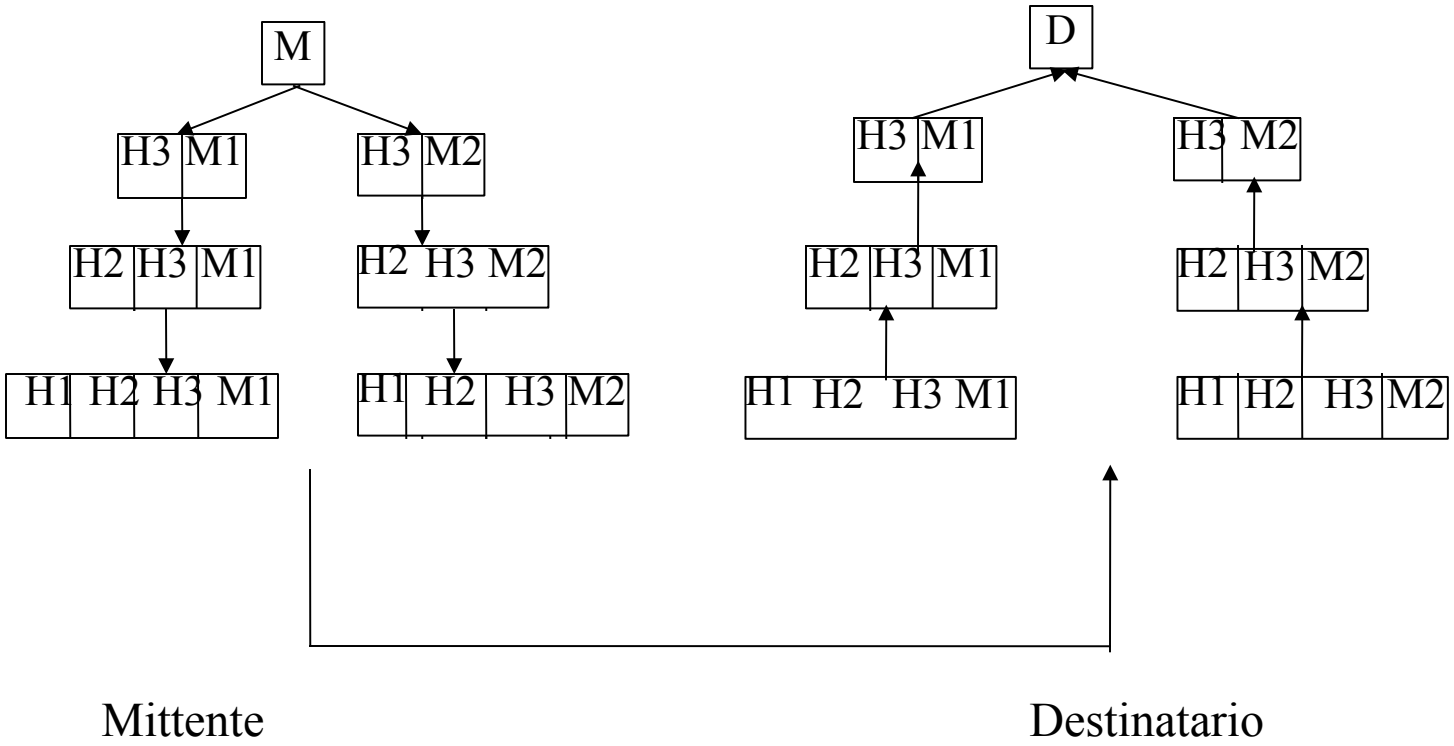
- ogni livello L_i aggiunge alcune informazioni all'i+1-PDU ricevuto dal livello superiore

- informazioni aggiunte dal livello L_i = Header H_i

- ogni header contiene le informazioni necessarie per implementare il servizio offerto dal livello L_i

- Esempio: il livello L_{i-1} garantisce ad L_i che un i-PDU venga recapitato in maniera corretta al destinatario. L'header H_{i-1} contiene informazioni che consentono di implementare la spedizione sicura.

ORGANIZZAZIONE A LIVELLI

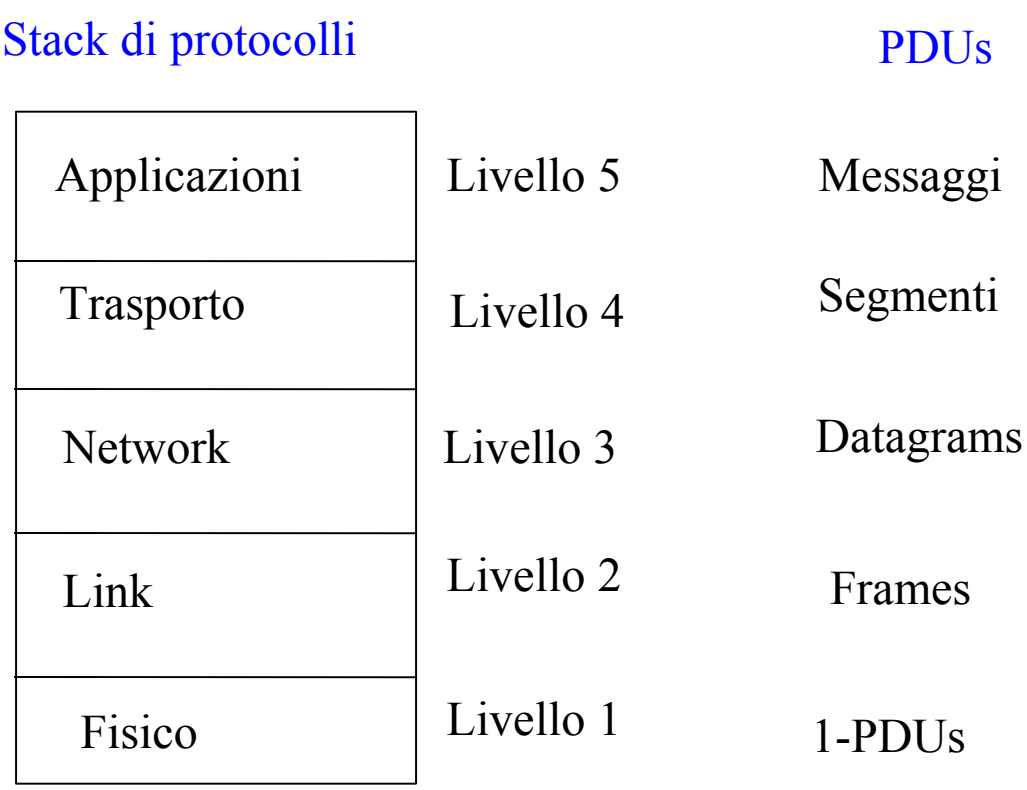


ORGANIZZAZIONE A LIVELLI

Funzioni che possono essere svolte da un livello Li:

- Definizione di connessioni: scambio di alcuni messaggi prima della comunicazione vera e propria
- Definizione di connessioni sicure
- Controllo del flusso: regola il flusso di PDU tra mittente e destinatario in modo da evitare l'overflow dei buffers
- Segmentazione e riassemblaggio dei messaggi: partizionamento di un messaggio (mittente) e successiva ricomposizione del messaggio (destinatario)
- Multiplexing/Demultiplexing

ORGANIZZAZIONE A LIVELLI: TCP/IP



LIVELLO DELLE APPLICAZIONI

- Comunicazione tra due processi (applicazioni) in esecuzione su host diversi

- consente di utilizzare una applicazione nota e quindi il protocollo associato
  - HTTP per l'accesso al WEB
  - FTP (file transfer protocol)
  - SSH (secure shell: accedere ad un sito remoto come si fosse direttamente collegati con un terminale.
  - SMTP (posta elettronica)
  - DNS (Domain Name System)

- creare nuove applicazioni
  - definizione di componenti (C, JAVA,..)
  - comunicazione tra componenti mediante sockets

LIVELLO TRASPORTO

Livello di trasporto: consente di definire un canale logico tra due componenti del livello applicazioni

Protocolli forniti:

TCP (Trasmission Control Protocol): Definisce un canale logico su cui può essere inviato un flusso di bytes. Il canale è affidabile (garantita la consegna ordinata dei messaggi spediti)

UDP: (User Datagram Protocol). Definisce canali logici non affidabili

Protocolli del livello trasporto : end-to-end protocols.

LIVELLO NETWORK

- Protocollo IP = Internet Protocol
- Comunicazioe host to host
- Servizio fornito: routing dei messaggi dal mittente al destinatario
- Internet definisce più algoritmi di routing
  - link state routing
  - distance vector routing
- Livello trasporto solo negli end systems
- Il protocollo IP è supportato sia negli end systems che nei routers

LIVELLO COLLEGAMENTO

- Livello collegamento o Data Link: definisce servizi per inviare un pacchetto tra due nodi della rete (routers o end systems)

- Il servizio fornito dipende dalle caratteristiche fisiche del link specifico

- Esempi: Ethernet per reti locali, wireless etc.

- Servizi offerti: invio di un frame da un nodo all'altro, affidabilità della trasmissione (diverso da affidabilità di TCP)

- Lo stesso datagram può essere trasportato mediante diversi link layer protocols quando attraversa link diversi

LIVELLO FISICO

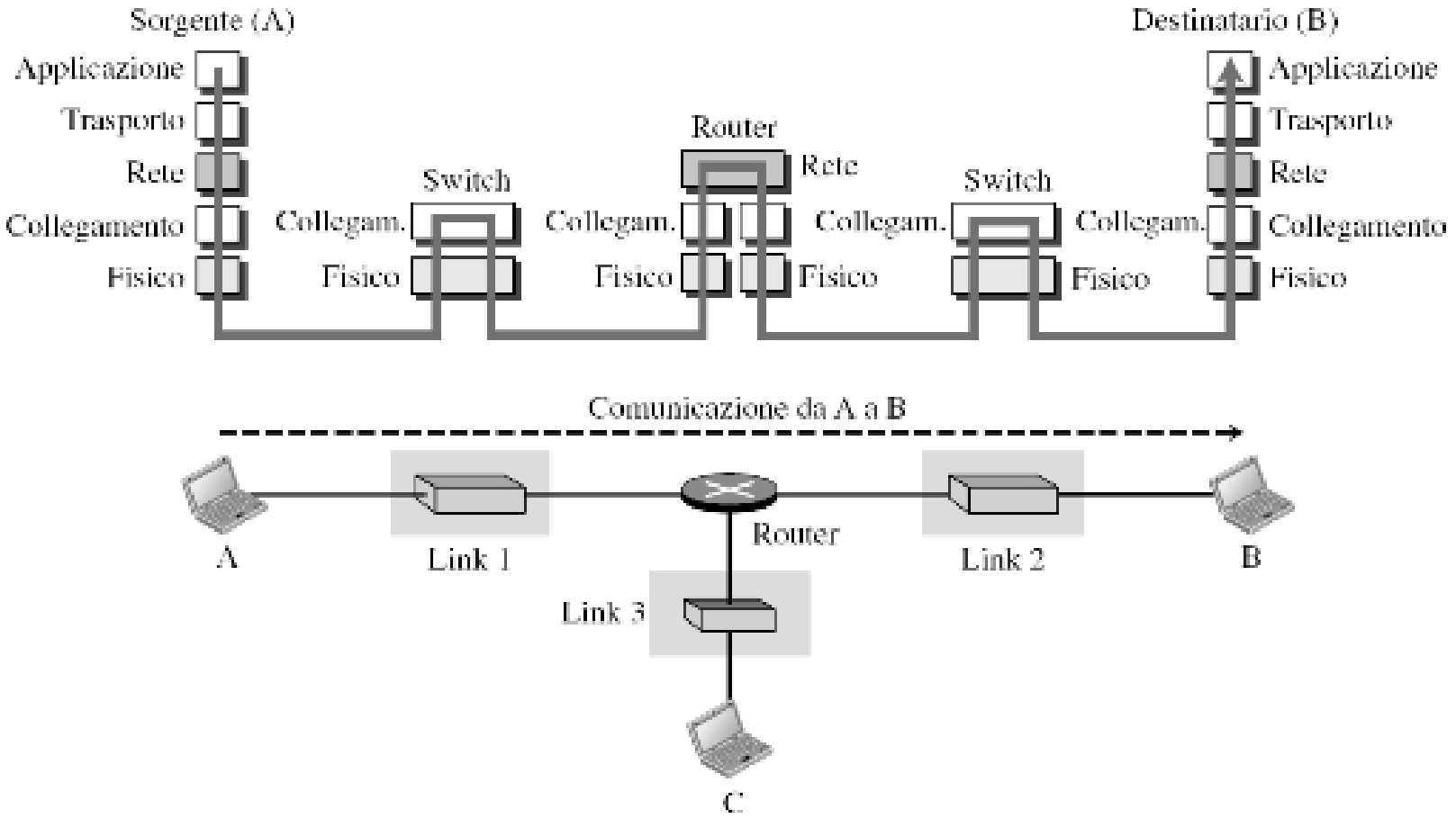
- servizi offerti: invio del singolo bit di un frame tra due nodi (routers o end systems) della rete

- i servizi forniti dipendono dal collegamento fisico che collega i due links

- lo stesso protocollo a livello link può utilizzare protocolli diversi a livello fisico (ad esempio Ethernet definisce protocolli diversi per collegamenti su fibra ottica, doppino di rame,…)

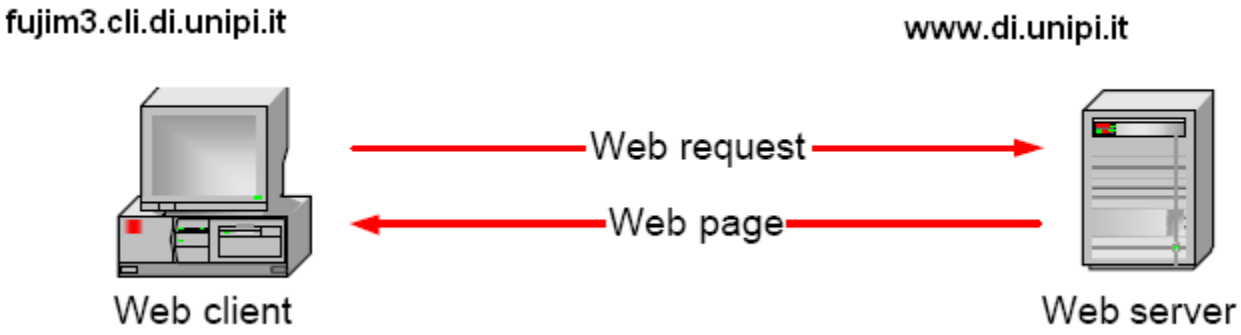
- non lo tratteremo….

COMUNICAZIONE IN INTERNET



TCP/IP NETWORKING: UN ESEMPIO

- Consideriamo una classica applicazione sviluppata su Internet
- Un web browser(web client) in esecuzione sull'host **fujim3.cli.di.unipi.it** richiede una pagina web all'host di nome www.di.unipi.it
- La pagina richiesta è identificata dalla URL www.di.unipi.it/index.html
- Mostriamo cosa accade nella rete dal momento in cui la richiesta è inviata al server al momento in cui la pagina viene restituita al client

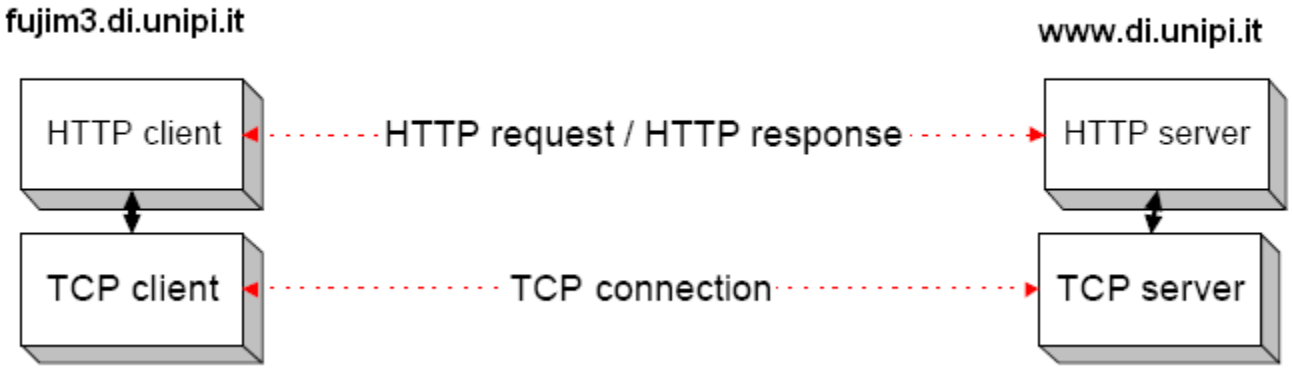


TCP/IP NETWORKING: UN ESEMPIO

- Il web client ed il web server sono programmi che interagiscono mediante HTTP (Hypertext Transfer Protocol)
- HTTP è un protocollo client/server: il web client esegue il programma HTTP client, mentre il web server implementa il programma HTTP server
- In questo modo si definisce un programma distribuito, composto da due componenti che sono eseguite su due nodi remoti della rete e che interagiscono scambiandosi dati
- Quando il browser richiede la pagina web, il client HTTP invia un messaggio HTTP di richiesta
- Quando il web server riceve quel messaggio, ricerca la pagina e la spedisce mediante un messaggio di HTTP reply

TCP/IP NETWORKING: UN ESEMPIO

- HTTP utilizza i servizi definiti dal protocollo di livello trasporto TCP per
  - permettere al client di aprire una connessione con il web server
  - inviare i dati sulla connessione aperta tra client e server
  - sfruttare i servizi offerti da TCP per definire una trasmissione affidabile

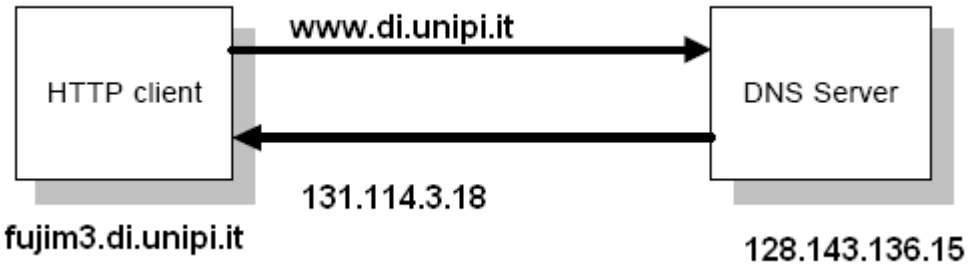


TCP/IP NETWORKING: UN ESEMPIO

- Il client HTTP deve inviare al client TCP l'indirizzo IP del server www.di.unipi.it e la porta su cui si trova in ascolto il server
- Indirizzo IP = stringa di 32 bits che identifica univocamente una l'interfaccia mediante cui un host è connesso ad Internet
- Ogni host possiede un indirizzo IP per ogni interfaccia tramite cui è connesso ad Internet
- Esempio: l'indirizzo IP di www.di.unipi.it è 131.114.3.118
- Ogni byte dell'indirizzo (ottetto) viene scritto come un numero decimale ed i 4 numeri decimali sono separati da un punto
- Porta = stringa di 16 bits che identifica univocamente un'applicazione in esecuzione su un host
- Indirizzo IP + Porta identificano univocamente un'applicazione in esecuzione su un qualsiasi host presente su Internet

INDIVIDUAZIONE INDIRIZZO IP

- L'utente fornisce al browser il nome simbolico dell'host (www.di.unipi.it) oppure una URL che identifica un oggetto sul server
- Il client HTTP deve tradurre il nome simbolico in un indirizzo IP
- La traduzione può essere effettuata utilizzando il servizio di DNS (Domain Name System)
- Il DNS è un servizio distribuito in grado di tradurre nomi in indirizzi IP e viceversa



STABILIRE UNA CONNESSIONE

- I più noti servizi Internet (HTTP servers, posta elettronica, FTP,….) sono associati a porte note (well known port numbers) e sono di solito memorizzati in un file di configurazione
- Esempio: il numero di porta corrispondente al server HTTP è l'80
- il client HTTP invia al client TCP una richiesta per stabilire una connessione sulla porta 80 dell'host di indirizzo 131.114.3.118
- La richiesta di connessione viene gestita dal livello TCP, che utilizza a sua volta i servizi del livello sottostante, il livello IP
- La definizione di una connessione tra il client TCP ed il server TCP richiede tre diversi passi (three way handshake)
  - Il client TCP invia una richiesta di connessione R al server TCP
  - Il server risponde positivamente, se è disposto ad accettare la connessione
  - Il client invia un acknoledgment ed inizia poi ad inviare i dati

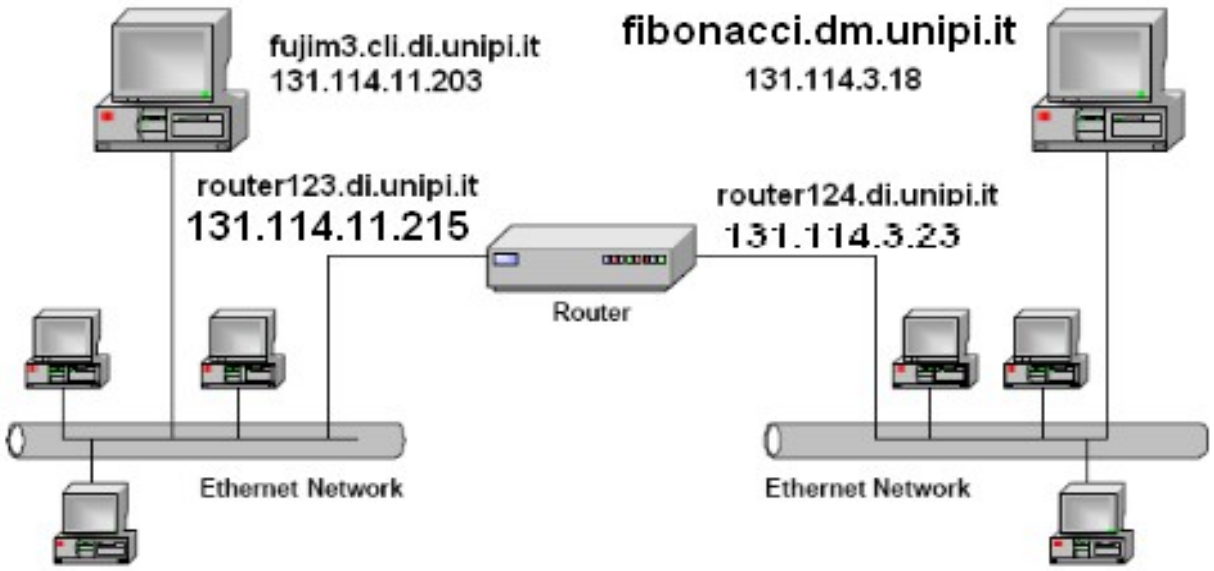
IL LIVELLO DI RETE

- Consideriamo l'invio della richiesta R di connessione
- Il client TCP passa R al livello IP che si occupa di
  - incapsulare la richiesta di connessione all'interno di un pacchetto IP
  - individuare un percorso sulla rete per raggiungere il server (l'host di indirizzo 131.114.3.118)

- Consideriamo la seguente configurazione di rete:
  - L'host fujim3.cli.di.unipi.it e l'host www.dm.unipi.it sono connessi a due diverse reti locali di tipo Ethernet
  - Un router (o gateway) connette le due reti locali. Il router comprende due diverse interfacce, tramite cui è connesso alle due diverse reti locali
  - NOTA BENE: ogni interfaccia del router è caratterizzata da un nome simbolico e da un indirizzo IP

TOPOLOGIA DELLA RETE

Figura 1



Osservazione: ogni interfaccia del router è caratterizzata da un nome simbolico e da un indirizzo IP

IL LIVELLO DI RETE: IL ROUTER

- Router= Dispositivo che dispone di un insieme di interfacce di rete (network interface cards = NIC) in grado di ricevere un pacchetto IP da una delle interfacce ed inoltrarlo su un'altra interfaccia, con l'intento di diminuire la distanza del pacchetto dall'host di destinazione
- Il router utilizza una tabella di routing per l'istradamento dei pacchetti
- Tabella di routing: contiene un insieme di associazioni insieme di indirizzi IP - interfaccia
- Un pacchetto IP diretto verso un certo indirizzo, viene inoltrato verso l'interfaccia corrispondente a quell'indirizzo

IL LIVELLO IP: FUNZIONALITA'

Il livello IP in esecuzione sul client fujim3.cli.di.unipi.it

- riceve dal livello TCP la richiesta di connessione col server ed effettua i seguenti controlli
- Individua se l'host destinatario del pacchetto (Matematica) si trova sulla stessa rete locale del client o meno
  - se gli hosts si trovano sulllo stesso segmento di rete locale, allora il pacchetto viene consegnato direttamente al destinatario
  - se gli hosts non si trovano sulla stessa rete locale il pacchetto viene consegnato ad un router connesso alla rete locale
- Il router di default di un host viene definito al momento della configurazione del sistema (default gateway)
- Ad esempio: il default gateway di fujim3.cli.di.unipi.it è router123.cli.di.unipi.it

IL LIVELLO IP: FUNZIONALITA'

- Per individuare se gli hosts si trovano sulla stessa rete locale, il supporto confronta gli indirizzi IP dei due hosts
- Hosts appartenenti alla stessa rete locale hanno indirizzi IP caratterizzati dallo stesso prefisso (esempio i primi 3 ottetti dell'indirizzo coincidono)
- Nel caso della rete mostrata nella Figura 1, l'host fujim3.cli.di.unipi.it invia il pacchetto IP al router di default router123.cli.di.unipi.it
- Per inviare il pacchetto IP, il livello IP utilizza i servizi offerti dal livello inferiore, il livello data link
- Per poter inviare il pacchetto al router, il pacchetto IP deve essere incapsulato in un frame Ethernets
- Il frame viene poi inviato sulla rete locale utilizzando il protocollo specifico definito a livello data link (in queto caso il protocollo Ethernet)

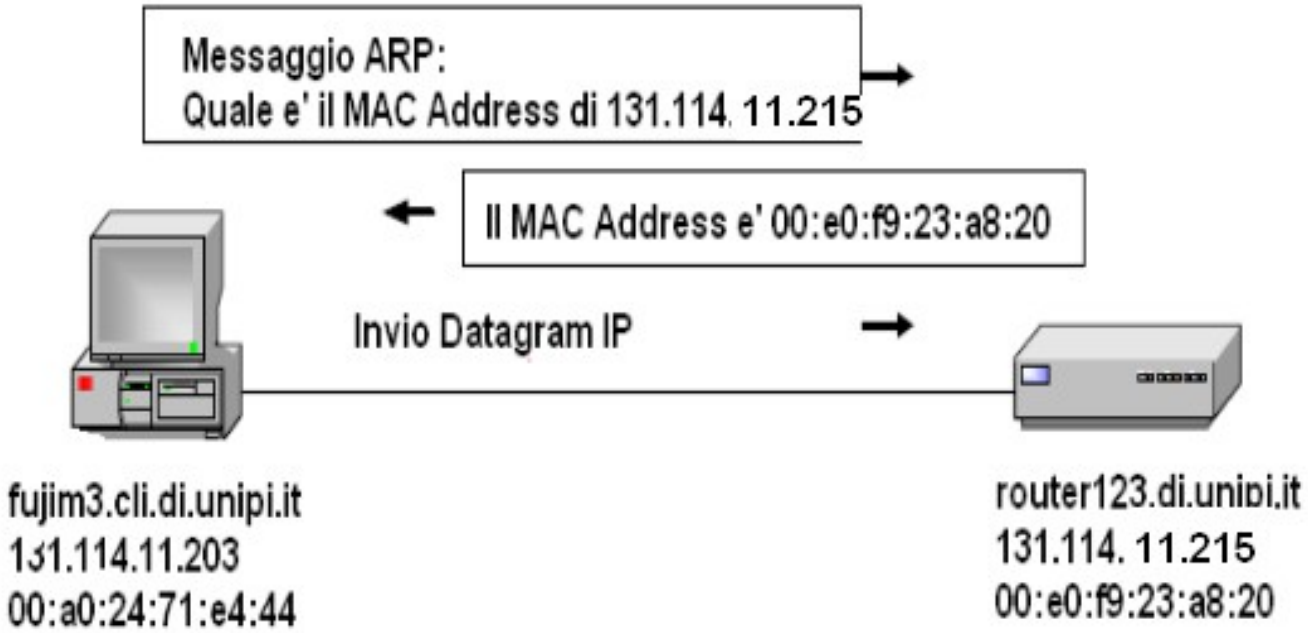
IL LIVELLO DATA LINK

- A livello Ethernet i frames possono essere scambiati tra hosts appartenenti alla stessa rete locale
- Lo schema di indirizzamento a livello Ethernet utilizza MAC Addresses
- MAC (Media Access Control) Address = stringa di 48 bits che identifica univocamente un host connesso ad un segmento di rete Ethernet
- Esempio di MAC Address: 00:a0:24:71:e4:44
- Ogni carattere esadecimale è rappresentato mediante 4 bits.
- Ogni frame Ethernet contiene il MAC Address del mittente ed il MAC Address del destinatario di un frame
- Prima di passare il frame al driver Ethernet, il livello IP deve tradurre l'indirizzo IP del destinatario nel suo MAC Address

IL LIVELLO DATA LINK

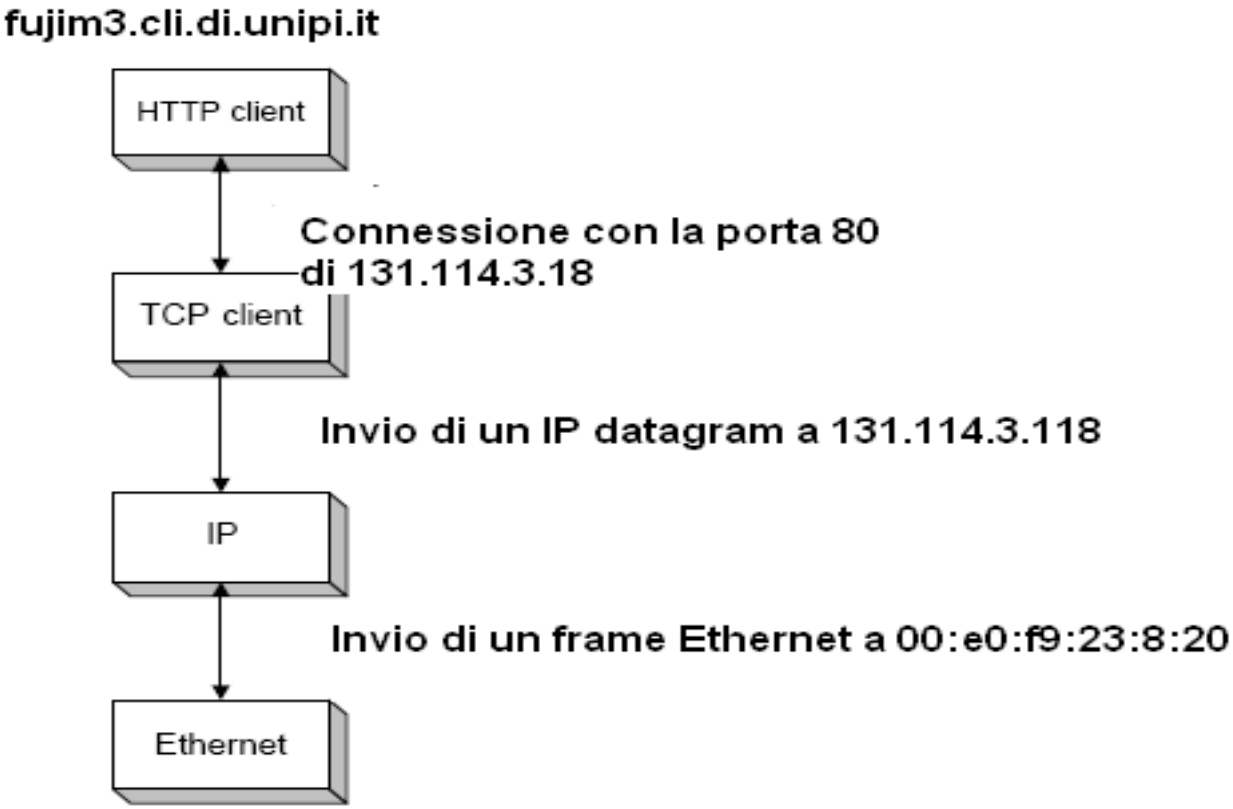
- La traduzione da indirizzo IP a MAC Address è supportata dall'Address Resolution Protocol (ARP)
- Il protocollo ARP prevede che la richiesta del MAC address di un host caratterizzato da un certo indirizzo IP sia inviata in broadcast sulla rete Ethernet
- L'host caratterizzato da quell'indirizzo IP invia un messaggio di risposta contenente il proprio MAC Address
- Il pacchetto IP viene incapsulato in un frame Ethernet
- Il livello IP dell'host invia il MAC Address del router al driver Ethernet, che spedisce il frame che incapsula il pacchetto al router

IL LIVELLO DATA LINK

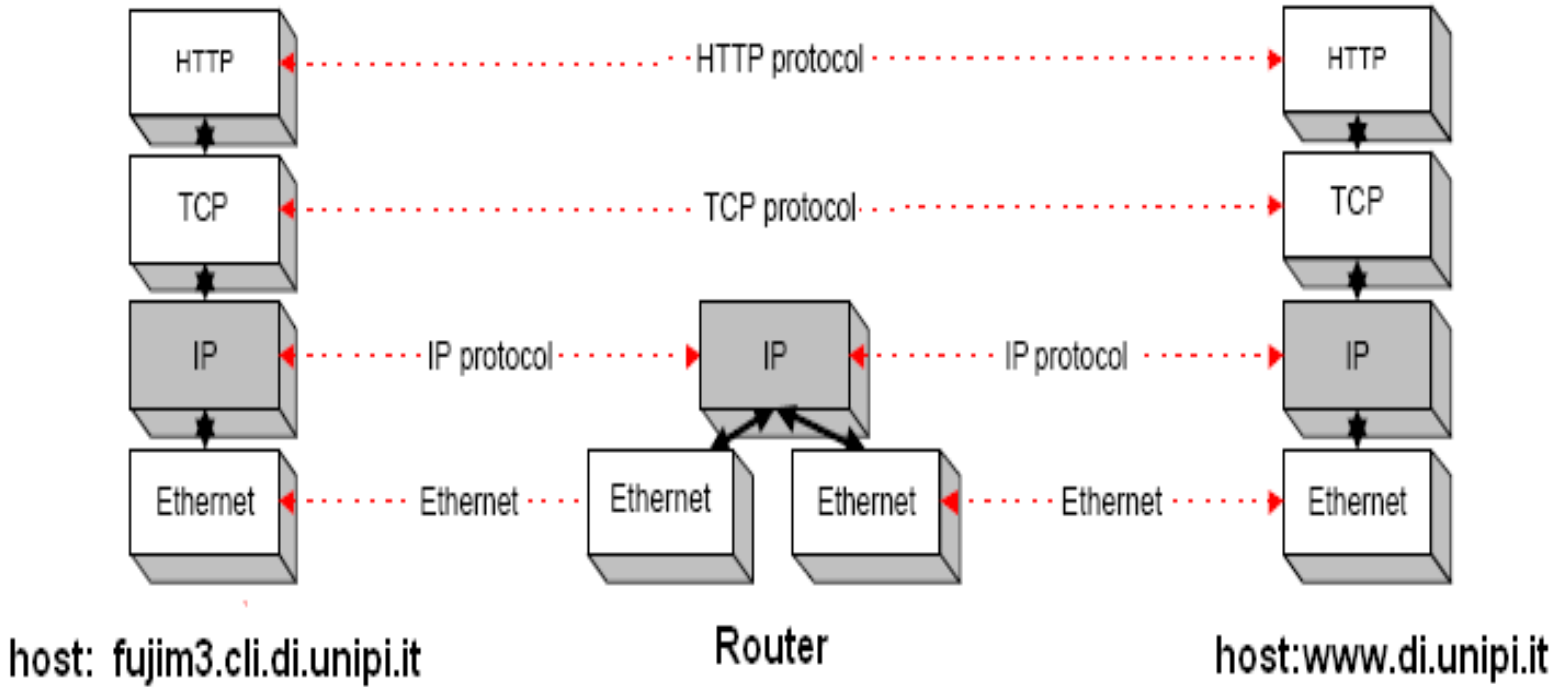


LO STACK TCP/IP

Internet Protocol Stack



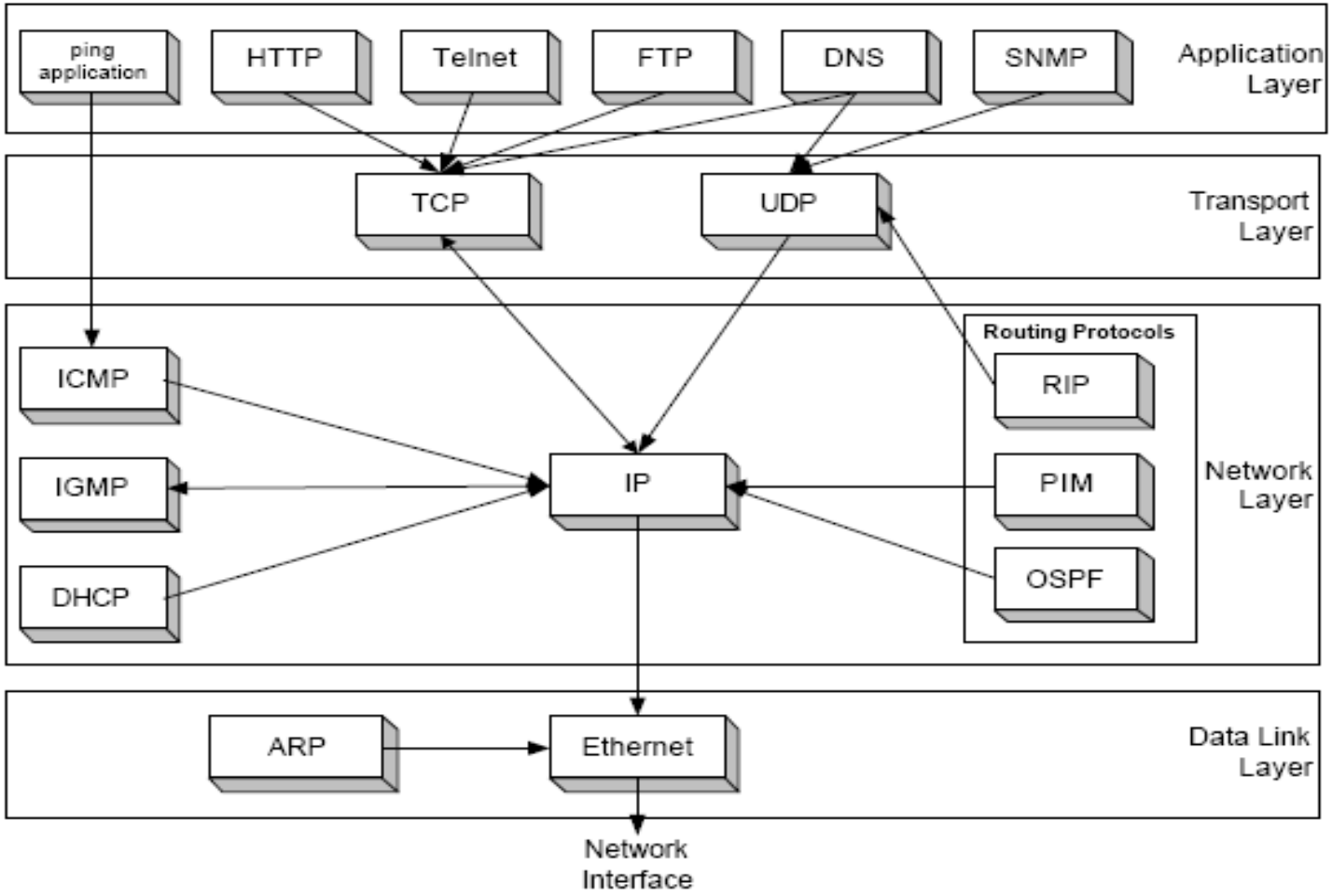
LO STACK TCP/IP



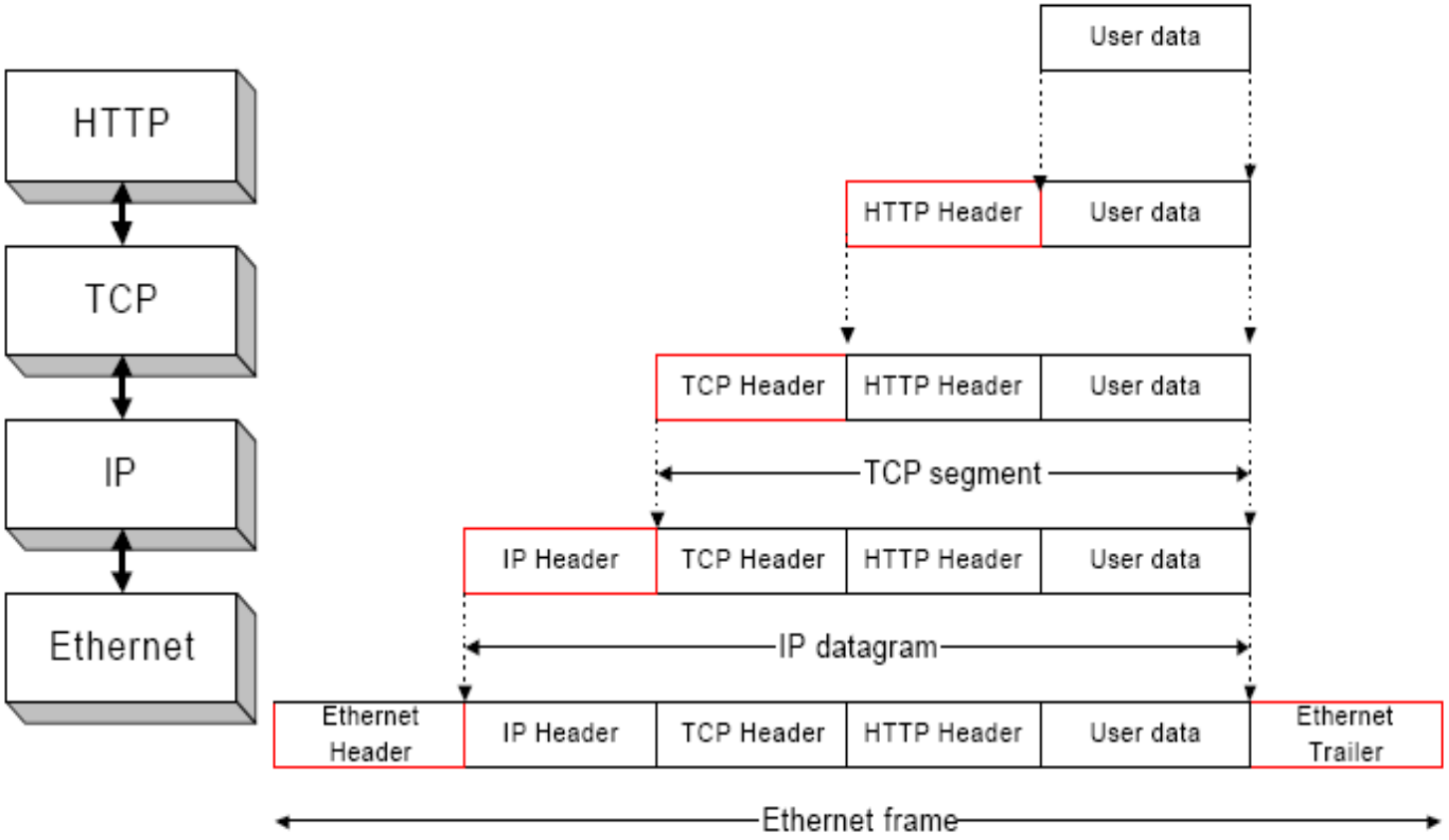
LO STACK TCP/IP

- Ad ogni livello dello stack TCP/IP esiste un programma che interagisce con il programma del livello corrispondente in esecuzione su un altro host
- Livelli applicazione/trasporto: end-to-end layers. A questo livello non è visibile l'esistenza di routers tra i due end-hosts
- Si possono utilizzare diversi protocolli data link in segmenti diversi della rete (ad esempio Ethernet, PPP= Point to Point Protocol,...)
- Le funzionalità del livello data link sono implementate in parte a livello software, in parte a livello hardware

PROTOCOLLI DEFINITI PER OGNI LIVELLO



INCAPSULAMENTO DEI DATI



INCAPSULAMENTO DEI DATI

- Quando i dati vengono passati da un livello ad un livello più basso, ai dati viene aggiunto un insieme di informazioni di controllo (header)
- Solo il livello data link aggiunge un postfisso ai dati (trailer)
- Aggiunta di header e di trailer = incapsulamento dei dati
- Dati contenuti tra l'header ed il trailer = payload
- Header+payload+trailer (se presente) costituiscono un protocol data-unit (PDU)
- Datagram= protocol data unit a livello UDP ed a livello IP
- Pacchetto= utilizzato per riferire un generico PDU (a qualsiasi livello)
- La dimensione dei dati cresce con l'aumentare del livello
- L'header di un certo livello viene elaborato solamente dal protocollo del livello corrispondente

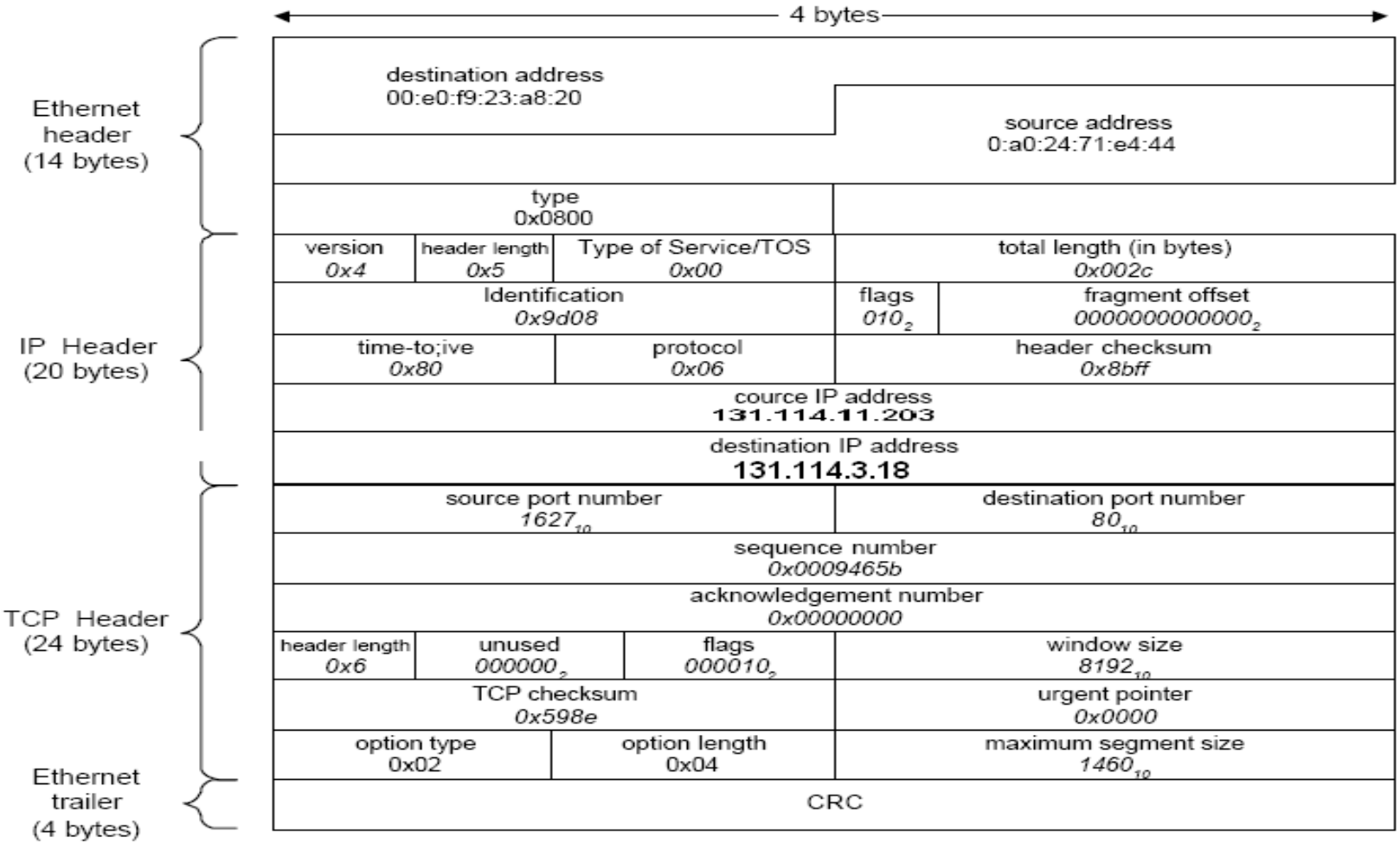
INCAPSULAMENTO DEI DATI

- Alcuni campi contenuti negli header
  - Indirizzo mittente e destinatario (livello IP, livello data link)
  - Campi per il controllo degli errori
  - Lunghezza del payload
  - Lunghezza dell'header, se questo ha lunghezza variabile
- Lato mittente: ogni livello aggiunge un header prima di passare il PDU al livello successivo
- Lato destinatario, per ogni livello L:
  - si elimina l'header corrispondente ad L e si passa il payload del PDU al livello sovrastante
  - È necessario decidere a quale protocollo di livello superiore consegnare il payload (demultiplexing)

INCAPSULAMENTO DEI DATI

- Esempi di demultiplexing
  - Un device driver Ethernet deve decidere se assegnare il payload di un frame al protocollo ARP oppure al protocollo IP
  - Il livello IP deve decidere se il payload è destinato al livello UDP, TCP oppure a qualche altro protocollo
- Demultiplexing: utilizza un campo dell'header della PDU, che identifica un protocollo di livello più alto oppure un processo applicativo
- Ogni protocollo utilizza un campo di demultiplexing

FORMATO DI UN FRAME ETHERNET



FORMATO DI UN FRAME

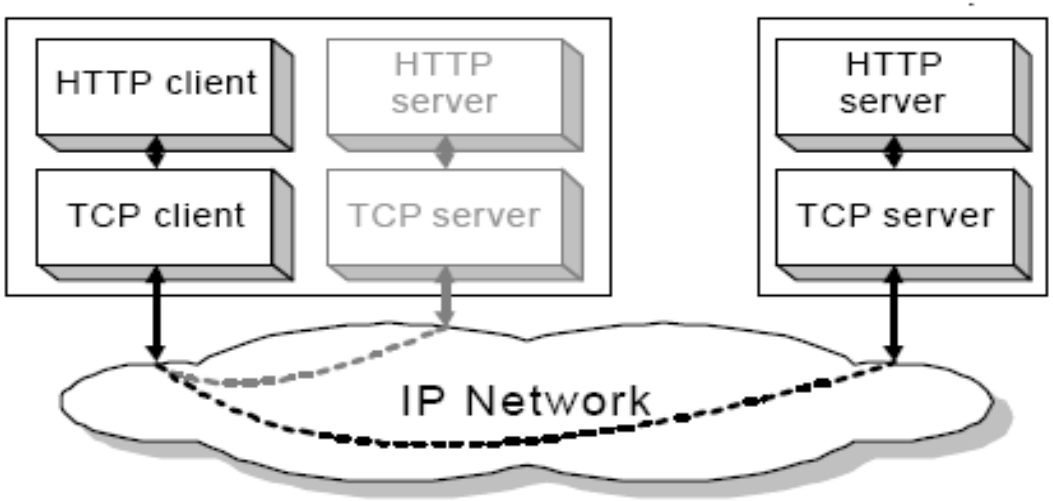
- I primi 14 bytes del frame rappresentano l'header Ethernet e rappresentano
  - i MAC address di mittente e destinatario
  - Gli ultimi due bytes codificano il campo per il demultiplexing. Esempio: il valore 0x0800 indicano che il payload del frame è un datagram IP. In questo caso il driver deve passare il payload al livello IP
- I secondi 20 bytes rappresentano l'header IP. Alcuni campi significativi:
  - Indirizzo IP di mittente e del destinatario
  - Campi per la gestione della frammentaione
  - Campo campo protocol è utilizzato per il demultiplexing

FORMATO DI UN FRAME

- TCP header: 24 bytes
- Alcuni campi significativi
  - La porta del mittente e del destinatario. Sono i campi utilizzati per il demultiplexing,in quanto identificano il protocollo o l'applicazione che devono ricevere il payload
  - Campi utilizzati al momento in cui si richiede la connessione (flags)
  - Checksum per il controllo degli errori

RAPPRESENTAZIONI DELLA RETE A LIVELLI DIVERSI

- Ogni livello definisce una rappresentazione astratta della rete.
- A livello applicazione/TCP la rete viene vista come una singola rete IP, in cui non sono visibili i routers
- L'astrazione definita da questi livelli consente di trattare nello stesso modo comunicazioni locali (dirette allo stesso host) o in esecuzione su hosts diversi



TECNICHE DI INSTRADAMENTO DEI MESSAGGI

- Come avviene la comunicazione tra gli end systems mediante i routers intermedi?

- Diverse tecniche:

  - commutazione di circuito
  - commutazione di pacchetto
    - Datagram Networks (Datagram analogo a Telegramma)
    - Virtual Circuits

- Internet: Rete a commutazione di pacchetto, utilizza datagrams

- ATM: Rete a commutazione di pacchetto, utilizza virtual circuits

COMMUTAZIONE DI CIRCUITO

Commutazione di circuito: è sempre presente un collegamento dedicato, chiamato circuito, tra due dispositivi.

- se HA vuole comunicare con HB
  - si stabilisce un cammino P tra HA ed HB
  - si riserva un insieme di risorse lungo P (buffers, banda dei links) prima di iniziare la comunicazione C
  - le risorse rimangono riservate a C per tutta la sua durata

$\Rightarrow$

Poichè le risorse sono dedicate a quella trasmissione si può garantire una velocità di trasmissione stabilita

Nota: i routers intermedi tengono traccia della connessione tra gli end system il che implica che si realizza una connessione stretta tra HA ed HB $\neq$ dal servizio connection oriented offerto da TCP.

COMMUTAZIONE DI CIRCUITO

Tecniche per l'implementazione di un insieme di circuiti sullo stesso link

FDM Frequency Division Multiplexing
- circuiti diversi utilizzano frequenze diverse

TDM Time Division Multiplexing
- Tempo diviso in frames
- Ogni frame temporale è diviso in slots
- circuiti diversi utilizzano slot temporali diversi all'interno dello stesso frame

Vantaggi vs. svantaggi: garanzia del servizio vs. utilizzazione limitata delle risorse della rete

COMMUTAZIONE DI PACCHETTO

- i messaggi scambiati tra gli hosts vengono divisi in pacchetti.
- i pacchetti vengono spediti sui links di comunicazione ed instradati dai routers (packet switches)
- le risorse utilizzate per inoltrare un pacchetto non vengono riservate, ma vengono allocate on demand

$\Rightarrow$

- se le risorse richieste non sono disponibili un pacchetto può essere bloccato per un tempo non prevedibile a priori

$\Rightarrow$

non si può garantire un livello minimo di servizio

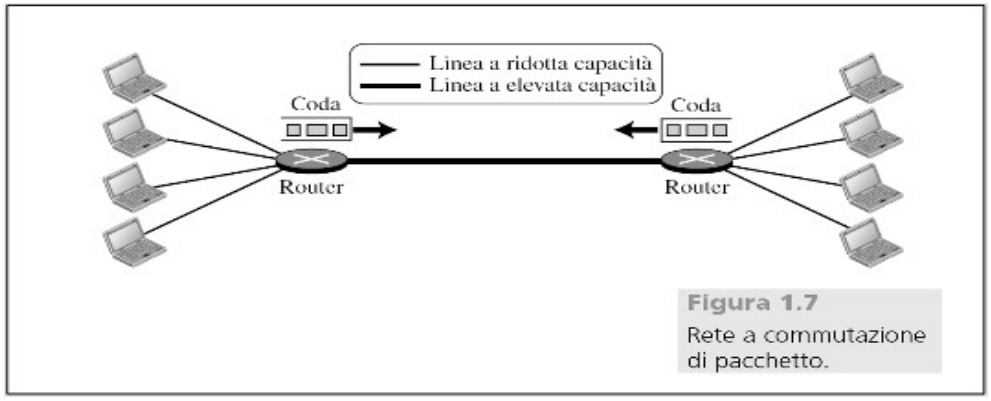


Figura 1.7
Rete a commutazione di pacchetto.

COMMUTAZIONE DI PACCHETTO

Come viene instradato un pacchetto all'interno di un router:

- il router deve riceve un intero pacchetto da un link di ingresso prima di spedirlo su un link di uscita (store and forward)
- esiste un insieme di buffers associati ai links di ingresso/uscita
- se il link su cui deve essere instradato il pacchetto risulta occupato, il pacchetto deve essere memorizzato nella coda associata al link di uscita (queueing delay)
- perdita di pacchetti = può avvenire quando il buffer associato al link su cui il pacchetto deve essere instradato è pieno. Viene scartato il pacchetto in arrivo o uno dei pacchetti memorizzati nel buffer.

COMMUTAZIONE DI MESSAGGIO

commutazione di messaggio = caso particolare della tecnica della commutazione di pacchetto

- il massaggio spedito da un host non viene diviso in segmenti
- ogni pacchetto contiene un intero messaggio

Packet switching : trasmissione pipeline

Message switching : trasmissione sequenziale

Packet switching richiede una latenza minore nella spedizione dei messaggi

INSTRADAMENTO DEI PACCHETTI

- Tecniche di commutazione: strategie con cui si gestisce il flusso dei pacchetti al fine di garantire instradamento e consegna al destinatario.

- Approcci per il l'istradamento dei pacchetti:

  - datagram networks: il routing viene effettuato in base all'indirizzo del destinatario contenuto nel pacchetto

  - virtual circuit networks: viene costruito un circuito virtuale tra mittente e destinatario. I messaggi scambiati vengono instradati su questo circuito

- Internet: Packet Switched Datagram Network

- ATM: Packed Switched Virtual Circuit Network

COMMUTAZIONE DI PACCHETTO DATAGRAM NETWORKS

- Datagram Networks forniscono un servizio analogo al servizio postale (datagram= telegram)
- ogni pacchetto contiene un header in cui è contenuto l'indirizzo del destinatario
- l'indirizzo possiede una struttura gerarchica (es: indirizzo sottorete + indirizzo host)
- ogni router sceglie il link di uscita su cui instradare un pacchetto in arrivo in base all'indirizzo contenuto nell'header del pacchetto (tabelle di routing)
- i routers non mantengono informazioni circa le connessioni effettuate tra hosts

COMMUTAZIONE DI PACCHETTO
DATAGRAM NETWORKS

Caratteristiche principali reti a commutazione di pacchetto

- non si stabilisce alcuna connessione (a livello di routers) tra mittente e destinatario (nota bene: questo non vieta che si stabilisca una trasmissione connection oriented tra gli host a livello trasporto!!)

- quando un host invia un pacchetto non ha alcun modo di sapere ne se sarà recapitato ne se il destinatario è operativo
  - ogni pacchetto viene inoltrato in modo indipendente dai pacchetti precedenti

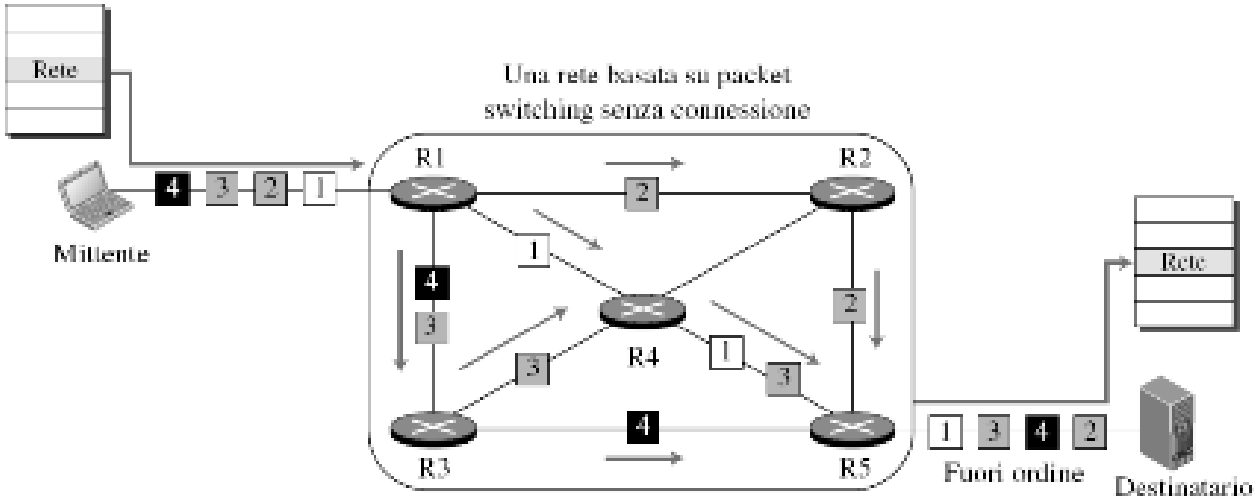
    $\Rightarrow$

  due pacchetti scambiati tra gli stessi hosts possono segure percorsi diversi (esempio: le tabelle di routing vengono modificate dinamicamente, sono previsti più percorsi per la stessa destinazione)

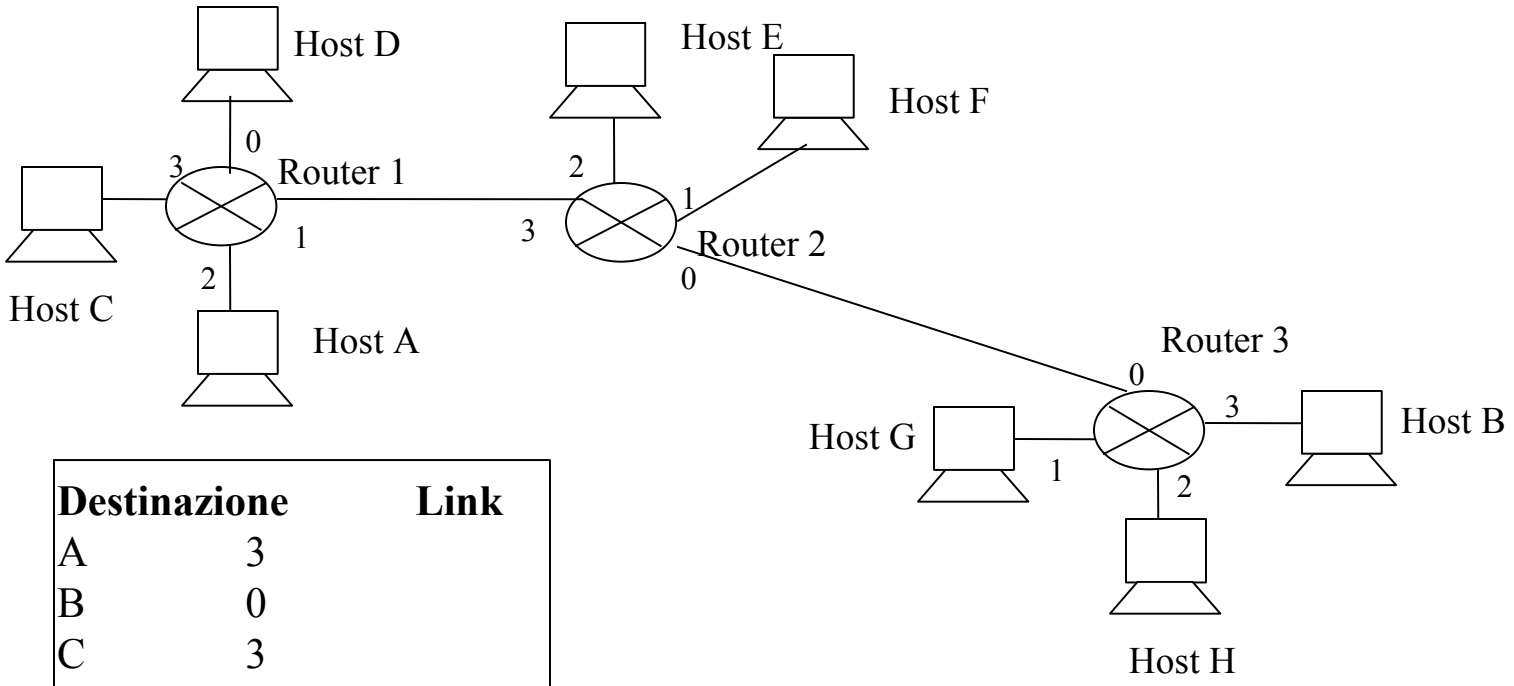
- tecnica tollerante ai guasti

COMMUTAZIONE DI PACCHETTO
DATAGRAM NETWORKS

Figura 4.3 Una rete basata su packet switching senza connessione.



DATAGRAM NETWORKS: ROUTING



Destinazione	Link
A	3
B	0
C	3
D	3
E	2
F	1
G	0
H	0

Router 2:
Tabella di Routing

DATAGRAM NETWORKS: STRUTTURA DI UN ROUTER

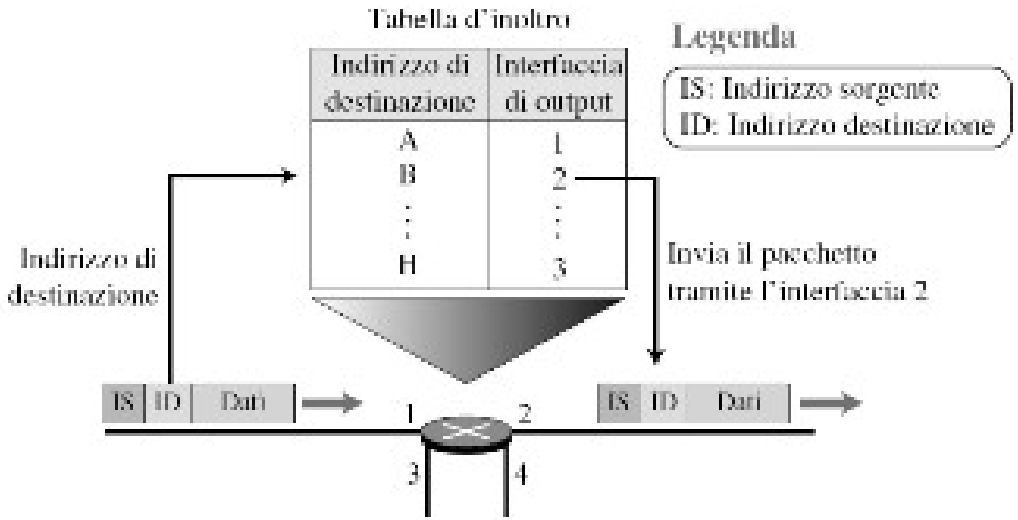


Figura 4.4

Processo d'inoltro di un router in una rete priva di connessione.

COMMUTAZIONE DI PACCHETTO: CIRCUITI VIRTUALI

- Prima dell'invio dei dati viene creato un circuito virtuale tra mittente e destinatario
- Questo percorso verrà usato da tutti i pacchetti appartenenti alla stessa comunicazione che verranno via via spediti
- ogni circuito virtuale ha un identificatore e ogni pacchetto contiene come unica informazione di controllo l'identificatore di circuito virtuale.
- i nodi della rete instradano i pacchetti in base al circuito virtuale di appartenenza, cosicchè tutti i pacchetti di un messaggio seguono lo stesso percorso e giungono in ordine a destinazione.
- IMPORTANTE: differenza con commutazione di circuito
  - il circuito virtuale non alloca risorse.
  - il circuito virtuale CV individua un cammino, ma risorse presenti su quel cammino non sono dedicate unicamente a CV
  - permettono la gestione dinamica delle risorse della rete che ottimizza l'uso della rete stessa

COMMUTAZIONE DI PACCHETTO: CIRCUITI VIRTUALI

Svantaggi

- overhead iniziale per stabilire il circuito virtuale e gestione di un protocollo più complesso
- in caso di guasti, occorre ricostruire il circuito virtuale

Vantaggi

- risparmio di spazio nell'header del pacchetto (numero circuito virtuale vs. indirizzo dell'host)
- è affidabile in quanto è possibile fornire lungo il circuito controllo di errore e di sequenza;
- è veloce in quanto in ciascun nodo non è necessario prendere decisioni complesse in merito all'instradamento;
- è efficiente per comunicazioni lunghe poichè i pacchetti viaggiano con poche informazioni di controllo.

COMMUTAZIONE DI PACCHETTO: CIRCUITI VIRTUALI

Fase di set-up del circuito virtuale

- Il mittente invia un pacchetto di richiesta di connessione
- I nodi intermedi che ricevono tale pacchetto da una interfaccia di ingresso, devono
  - decidere su quale interfaccia di uscita rilanciare il pacchetto
  - memorizzare tale interfaccia, in modo che altri pacchetti dati, appartenenti alla stessa comunicazione, seguano lo stesso percorso
- L'assegnazione di un numero univoco ad ogni circuito virtuale è impensabile per reti geografiche su larga scale
- Ogni nodo intermedio sceglie un identificatore unico per ogni circuito virtuale e tale identificatore è significativo solo in locale

COMMUTAZIONE DI PACCHETTO: CIRCUITI VIRTUALI

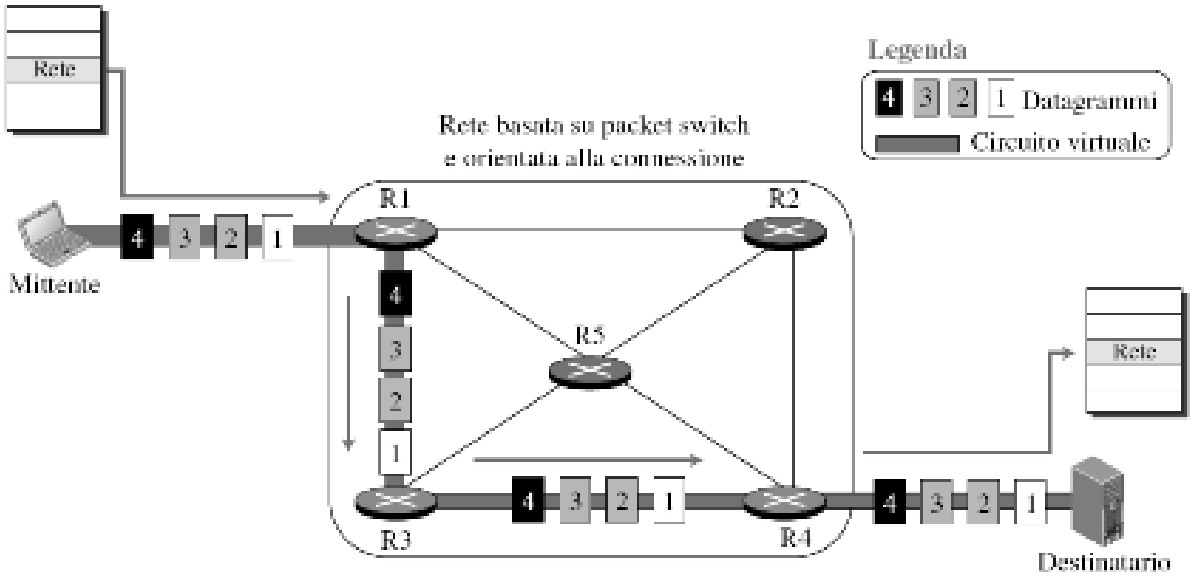


Figura 4.5

Rete basata su circuiti virtuali.

COMMUTAZIONE DI PACCHETTO: CIRCUITI VIRTUALI

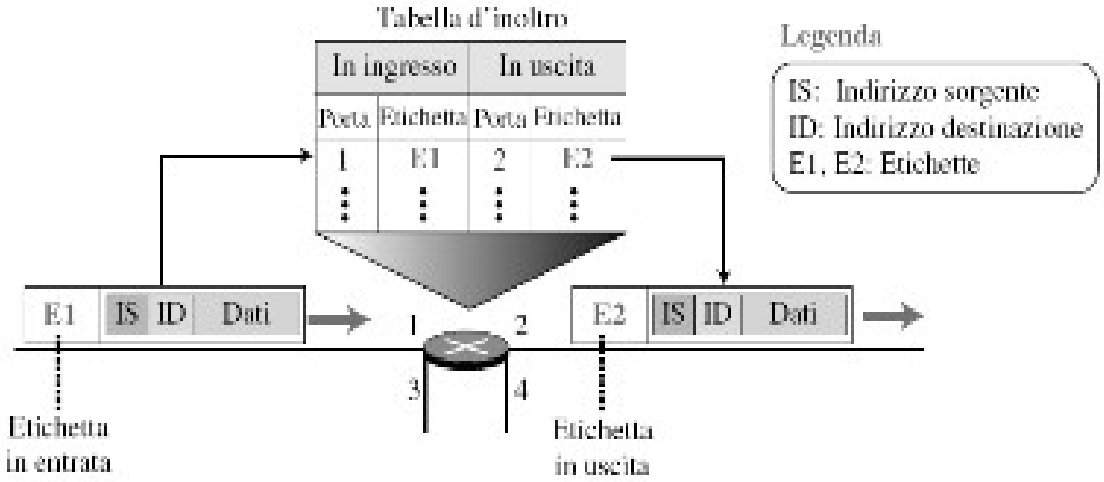
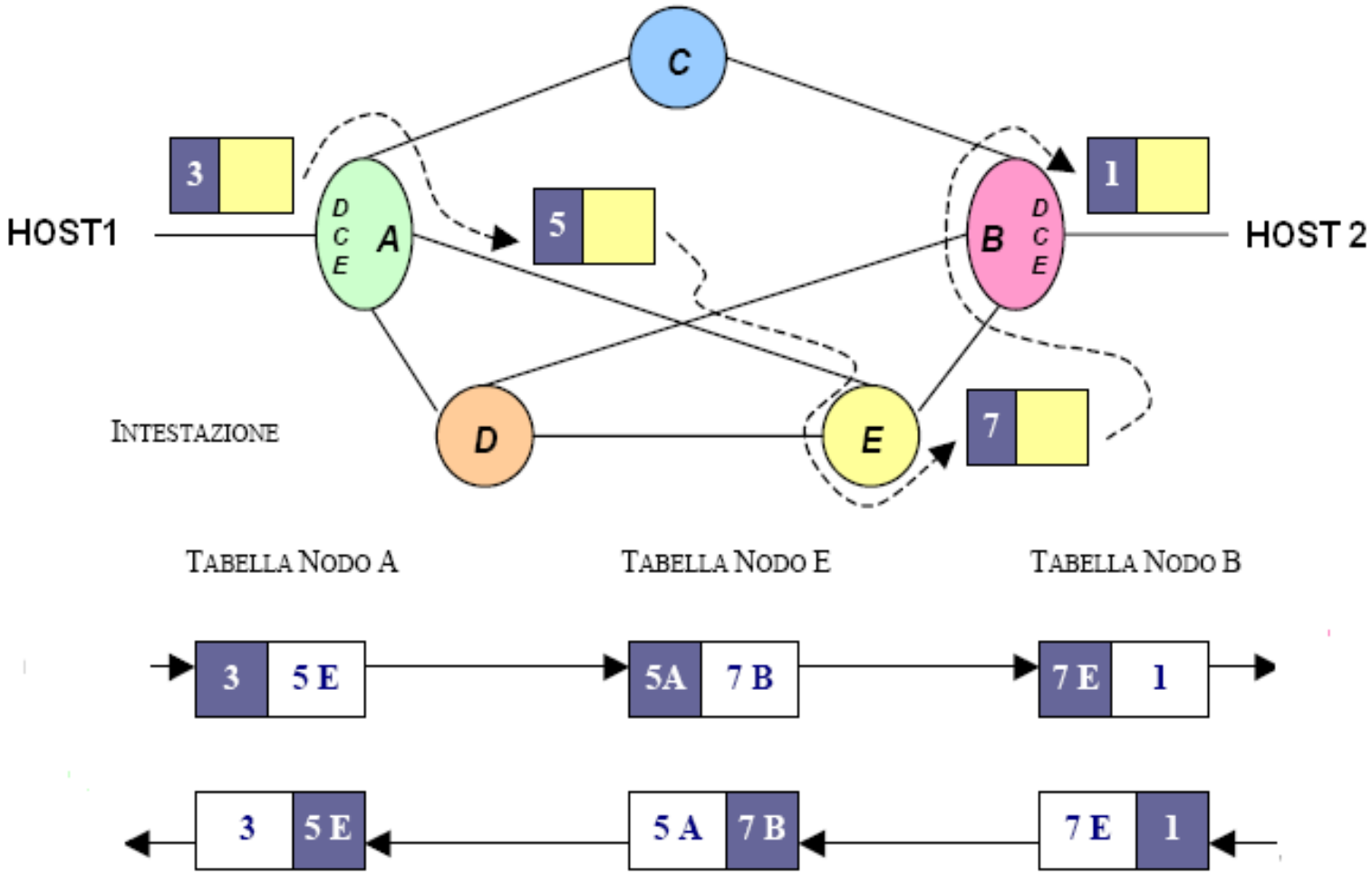


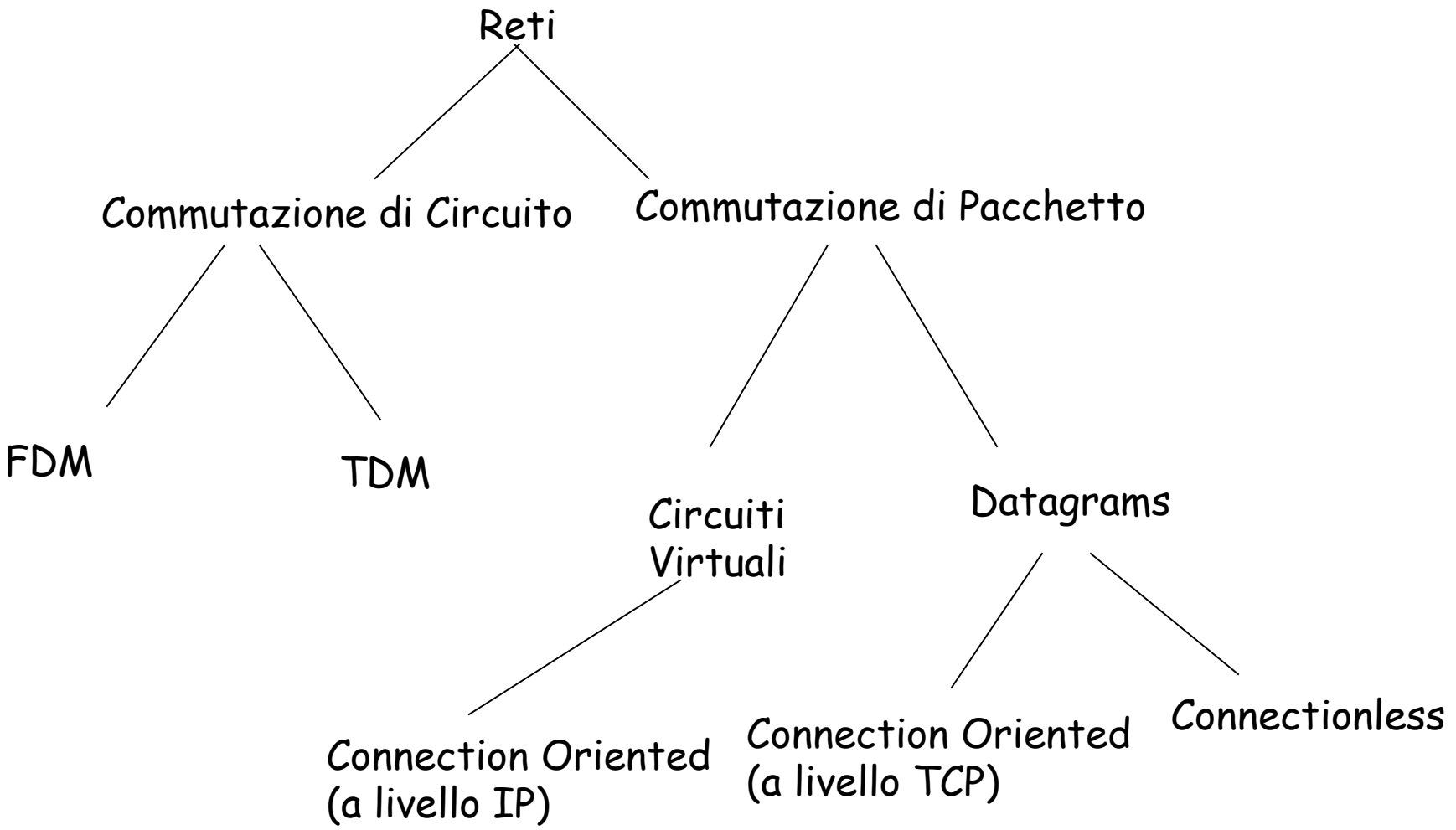
Figura 4.6

Inoltro di un router in una rete a circuito virtuale.

COMMUTAZIONE DIPACCHETTO: CIRCUITI VIRTUALI



SCHEMA RIASSUNTIVO



INDIRIZZI IP: STRUTTURA

- Indirizzo IP: identificatore associato ad una interfaccia di rete a livello IP. Quando l'interfaccia viene connessa alla rete, l'indirizzo IP deve essere unico globalmente su tutta la rete

- non possono esistere due interfacce diverse sulla rete con lo stesso indirizzo IP

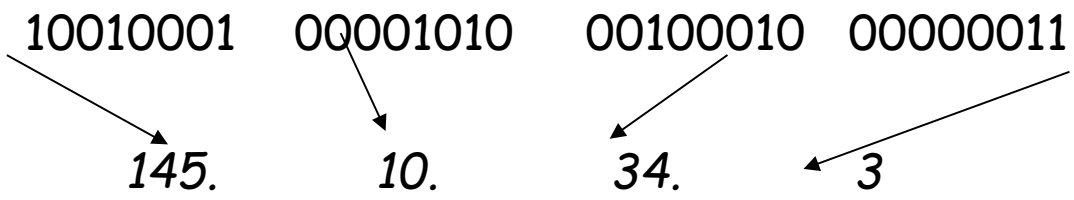
- eccezione: le interfacce di hosts appartenenti a reti private (intranets) non devono necessariamente avere indirizzi univoci

- ogni host su una rete IPv4 è identificato da un numero rappresentato su 4 bytes indirizzo IP dell'host (assegnato da IANA).
- IPv6

- se un host, ad esempio un router, presenta più di una interfaccia sulla rete vengono definiti più indirizzi IP per lo stesso host, uno per ogni interfaccia

INDIRIZZI IP

Esempio di indirizzo IPv4:



- ognuno dei 4 byte, viene interpretato come un numero decimale senza segno
- ogni valore varia da 0 a 255 (massimo numero rappresentabile con 8 bits)
- evoluzione: in IPv6 indirizzo IP comprende 16 bytes
- indirizzi disponibili in IPv4, $2^{32} = 4.294.967.296$
  - alcuni indirizzi sono riservati: ad esempio indirizzi di loopback, individuano il computer su cui l'applicazione è in esecuzione

INDIRIZZI IP: ASSEGNAZIONI STATICA VS. DINAMICA

Assegnazione degli indirizzi IP:

- Statica: ad un host viene assegnato un indirizzo IP fisso (utilizzata ad esempio per i servers)
- Dinamica: l'indirizzo IP di un host può variare dinamicamente.

Assegnazione dinamica:

- utilizzata per hosts che si collegano in modo temporaneo alla rete (es collegamento telefonico). Viene assegnato un nuovo indirizzo IP al momento del collegamento
- utilizza servers DHCP (Dynamic Host Configuaration Protocol)
- dal punto di vista del programmatore di rete: tenere presente che l'indirizzo IP di un host può cambiare in relazione ad esecuzioni diverse della stessa applicazione, ed anche durante una stessa esecuzione della applicazione

INDIRIZZI IP

Classfull Addressing: un indirizzo IP strutturato secondo una gerarchia a due livelli

Network Prefix + Host Number

Network Prefix : identifica la rete in cui si trova l'host

Host Number : identifica il perticolare host su quella rete

- tutti gli host di una rete presentano lo stesso prefisso
- due hosts in reti diversi possono avere lo stesso host number
- è possibile specificare network prefixes di lunghezza diversa, in questo modo si possono individuare reti di dimensioni diverse
  - maggiore la dimensione del network prefix, minore la il numero di hosts presenti sulla sotto rete)
- di solito ad un'organizzazione viene assegnato un prefisso P. L'organizzazione può quindi assegnare liberamente indirizzi alle interfacce degli hosts e dei routers di propria competenza, utilizzando P come prefisso

CLASSI INDIRIZZI IP

- IPv4 definisce 5 classi di indirizzi IP (A,B,C,D,E)
  - Classi A,B,C sono caratterizzate da diverse lunghezze dei network prefixes (quindi da diverso numero di hosts per rete)..
  - Classe D utilizzata per indirizzi di multicast (la vedremo meglio in seguito).
  - Classe E riservata
- La classe determina il confine tra il network prefix e l'host number
- Ogni classe è individuata dalla configurazione dei primi bits dell'indirizzo IP (esempio indirizzi in classe A hanno il primo bit dell'indirizzo =0
- Il meccanismo del classfull addressing semplifica le operazioni effettuate dal router

INDIRIZZI IP RISERVATI

- Indirizzi IP utilizzati per interfacce di rete presenti su LAN private

Indirizzi	Classe
10.X.X.X	A
172.16.X.X - 172.31.X.X	B
192.168.X.X	C

- Gli indirizzi 127.X.X.X individuano la rete di loopback, ovvero una rete fittizia costituita dall'interfaccia di rete stessa.

- Se si invia un pacchetto ad uno di tali indirizzi IP, questo transita dal livello applicazione al livello network dello stack TCP/IP. A questo livello si riconosce l'indirizzo di loopback e il pacchetto risale lo stack TCP/IP fino al livello applicazione dell'host stesso

- Il pacchetto viene ricevuto su una porta diversa rispetto a quello su cui è stato spedito

INDIRIZZI IP RISERVATI

- Un indirizzo IP che contiene tutti 1 nella parte host specifica un indirizzo di broadcast. Esempio 148.143.255.255 indica tutti gli host della rete 148.143

- Gli indirizzi IP con tutti i bit dell'host id a 0 sono utilizzati per identificare la rete nel suo complesso: per esempio, l'indirizzo IP 192.167.58.0 identifica la rete 192.167.58.X e viene utilizzato nelle routing tables.

PROGRAMMAZIONE DI RETE: INTRODUZIONE

Programmazione di rete:
sviluppare applicazioni definite mediante due o più processi in esecuzione su hosts diversi, distribuiti sulla rete. I processi cooperano per realizzare una certa funzionalità

- Cooperazione: richiede lo scambio di informazioni tra i processi
- Comunicazione = Utilizza protocolli (= insieme di regole che i partners della comunicazione devono seguire per poter comunicare)
- Alcuni protocolli utilizzati in INTERNET:
  - TCP (Trasmission Control Protocol) un protocollo connection-oriented
  - UDP (User Datagram Protocol) protocollo connectionless

PROGRAMMAZIONE DI RETE: INTRODUZIONE

Per identificare un processo con cui si vuole comunicare occorre

- la rete all'interno della quale si trova l'host su cui e' in esecuzione il processo
- l'host all'interno della rete
- il processo in esecuzione sull'host

Identificazione della rete e dell'host = definita dal protocollo IP, Internet Protocol

Identificazione del Processo = utilizza il concetto di porta

- Porta = Intero da 0 a 65535

IL PROTOCOLLO IP

Il Protocollo IP (Internet Protocol) definisce

- un sistema di indirizzamento per gli hosts
- la definizione della struttura del pacchetto IP
- un insieme di regole per la spedizione/ricezione dei pacchetti

Versioni del protocollo IP sono attualmente utilizzate in Internet

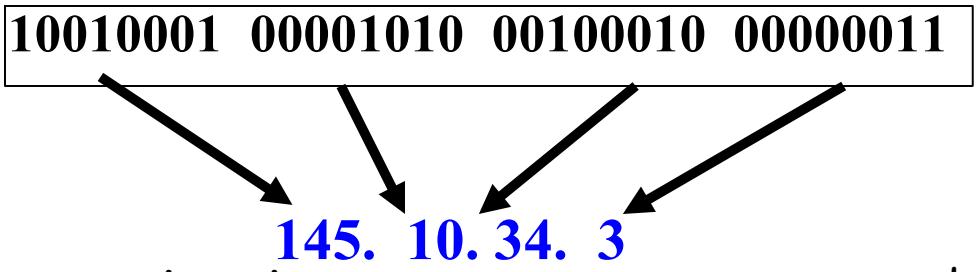
- IPV4 (IP Versione 4)
- IPV6 (IP versione 6)
  - Introduce uno spazio di indirizzi di dimensione maggiore rispetto ad IPV4

INDIRIZZAMENTO DEGLI HOSTS

- Ogni host di una rete IPV4 o IPV6 è connesso alla rete mediante una o più interfacce
- Ogni interfaccia è caratterizzata da un indirizzo IP
- Indirizzo IP
  - IPV4: numero rappresentato su 32 bits (4 bytes)
  - IPV6:numero rappresentato su 128 bits (16 bytes)
- se un host, ad esempio un router, presenta più di una interfaccia sulla rete, allora si hanno più indirizzi IP per lo stesso host, uno per ogni interfaccia
- Multi-homed hosts: un host che possiede un insieme di interfacce verso la rete, e quindi da un insime di indirizzi IP
  - gateway tra sottoreti IP
  - routers

INDIRIZZI IP

Un indirizzo IPV4



- ognuno dei 4 bytes, viene interpretato come un numero decimale senza segno
- ogni valore varia da 0 a 255 (massimo valore su 8 bits)

Un indirizzo IPV6

- 128 bits
- rappresentato come una sequenza di 16 cifre separate da due punti

3:6:0:0:0:0:6:45:0:0:9:56:67:0:0:1

INDIRIZZI IP E NOMI DI DOMINIO

- Gli indirizzi IP semplificano l'eleborazione effettuata dai routers, ma sono poco leggibili per gli utenti della rete

- Soluzione: assegnare un nome simbolico unico ad ogni host della rete
  - si utilizza uno spazio di nomi gerarchico

    esempio: fujih0.cli.di.unipi.it (host fuji presente nell'aula H alla postazione 0, nel dominio cli.di.unipi.it)
  - livelli della gerarchia separati dal punto.
  - nomi interpretati da destra a sinistra (diverso dalle gerarchia di LINUX)
- Corrispondenza tra nomi ed indirizzi IP = Risoluzione di indirizzi IP
- La risoluzione viene gestita da un servizio di nomi

DNS = Domain Name System

INDIRIZZAMENTO A LIVELLO DI PROCESSI

- Su ogni host possono essere attivi contemporaneamente più servizi (es: e-mail, ftp, http,…)
- Ogni servizio viene incapsulato in un diverso processo
- L'indirizzamento di un processo avviene mediante una porta
- Porta = intero compreso tra 1 e 65535. Non è un dispositivo fisico, ma un' astrazione per individuare i singoli servizi (processi).
- Porte comprese tra 1 e 1023 riservati per particolari servizi.
- Linux :solo i programmi in esecuzione su root possono ricevere dati da queste porte. Chiunque può inviare dati a queste porte.

  Esempio: porta 7 echo
  porta 22 ssh
  porta 80 HTTP
- In LINUX: controllare il file /etc/services

SOCKETS

- Socket: astrae il concetto di 'communication endpoint'
- Individuato da un indirizzo IP e da un numero di porta
- Socket in JAVA: istanza di una di queste classi
  - Socket
  - ServerSocket
  - DatagramSocket
  - MulticastSocket

JAVA: GESTIRE INDIRIZZI IP

Classe **JAVA. NET.InetAddress** (importare JAVA.NET).

Gli oggetti di questa classe sono strutture con due campi

- hostname = una stringa che rappresenta il nome simbolico di un host
- indirizzo IP = un intero che rappresenta l' indirizzo IP dell'host

La classe InetAddress:

- non definisce costruttori
- fornisce tre metodi statici per costruire oggetti di tipo InetAddress
  - `public static InetAddress.getByName (String hostnme) throws UnKnownHostException`
  - `public static InetAddress.getAllByName(String hostname) throws UnKnownHostException`
  - `public static InetAddress getLocalHost( ) throws UnKnownHostException`

JAVA: GESTIRE INDIRIZZI IP

```
public static InetAddress getByName (String hostname)
                                    throws UnKnownHostException
```

- cerca l'indirizzo IP corrispondente all'host di nome hostname e restituisce un oggetto di tipo

  InetAddress = nome simbolico dell'host + l'indirizzo IP corrispondente

  reverse resolution: può essere utilizzata anche per tradurre un indirizzo IP nel nome simbolico corrispondente
- in generale richiede una interrogazione del DNS per risolvere il nome dell'host
  - il computer su cui è in esecuzione l'applicazione deve essere connesso in rete
- può sollevare una eccezione se non riesce a risolvere il nome dell'host (ricordarsi di gestire la eccezione!)

JAVA: GESTIRE INDIRIZZI IP

Esempio:

```
try  { InetAddress address =
        InetAddress.getByName("www.google.it");
       System.out.println (address);
     }
catch (UnKnownHostException e)
     {System.out.println ("Non riesco a risolvere il
                          nome"); }
```

JAVA: GESTIRE INDIRIZZI IP

```
public static InetAddress [ ] getAllByName (String hostname)
                                    throws  UnKnownHostException
```

utilizzata nel caso di hosts che posseggano piu indirizzi (es: web servers)

```
public static InetAddress getLocalHost ()
                        throws UnKnownHostException
```

per reperire nome simbolico ed indirizzo IP del computer su cui è in esecuzione l'applicazione

Getter Methods = Per reperire i campi di un oggetto di tipo InetAddress (non effettuano collegamenti con il DNS $\Rightarrow$ non sollevano eccezioni)

```
public String getHostName ( )
public byte [ ] getAddress ( )
public String getHostAddress ( )
```

JAVA: GESTIRE INDIRIZZI IP

```
public static InetAddress getByName (String hostname)
                                throws UnKnownHostException
```

se il valore di hostname è l'indirizzo IP (una stringa che codifica la dotted form dell'indirizzo IP)

- la getByName non contatta il DNS
- il nome dell'host non viene impostato nell'oggetto InetAddress
- il DNS viene contattato solo quando viene richiesto esplicitamente il nome dell'host tramite il metodo getter getHostName( )
- la getHostName non solleva eccezione, se non riesce a risolvere l'indirizzo IP.

JAVA: GESTIRE INDIRIZZI IP

- accesso al DNS: operazione potenzialmente molto costosa
- i metodi descritti effettuano caching dei nomi/indirizzi risolti.
  - nella cache vengono memorizzati anche i tentativi di risoluzione non andati a buon fine (di default: per un certo numero di secondi)
- se creo un InetAddress per lo stesso host, il nome viene risolto con i dati nella cache (di default: per sempre)
- permanenza dati nella cache: per default alcuni secondi se la risoluzione non ha avuto successo, tempo illimitato altrimenti. Problemi: indirizzi dinamici.....
- `java.security.Security.setProperty` consente di impostare il numero di secondi in cui una entrata nella cache rimane valida, tramite le proprietà

  `networkaddress.cache.ttl`

  `networkaddress.cache.negative.ttl`

```
java.security.Security.setProperty("networkaddress.cache.ttl
  ",”0”);
```

NSLOOKUP

- implementazione in JAVA dell'utility UNIX **`nslookup`**
- **`nslookup`**
  - consente di tradurre nomi di hosts in indirizzi IP e viceversa
  - i valori da tradurre possono essere forniti in modo interattivo oppure da linea di comando
  - si entra in modalità interattiva se non si forniscono parametri da linea di comando
  - consente anche funzioni più complesse (vedere LINUX)

NSLOOKUP

```
import java.net.*;
import java.io.*;

public class HostLookUp {
   public static void main (String [ ] args} {
    if (args.length > 0) {
        for (int i=0; i<args.length; i++) {
              System.out.println (lookup(args[i])) ;
                                                  }
        }
    else {System.out.println("Error"); }
```

NSLOOKUP

```
public class nslookup {
    private static String lookup(String host) {
        InetAddress node;
        try { node =InetAddress.getByName(host);
                System.out.println(node);
                if  (isHostName(host))
                      {return node.getHostAddress( );}
                 else{
                       return node.getHostName ( );}
               }
        catch (UnknownHostException e)
                          {return "non ho trovato l'host";}
                                }
```

NSLOOKUP

```
private static boolean isHostName (String host)

    {char[ ] ca = host.toCharArray();
    for (int i = 0; i<ca.length; i++)
        { if(!Character.isDigit(ca[i])) {
            if (ca[i] != '.') return true; }
        }
    return false;
    }
```