DIPARTIMENTO DI INFORMATICA
Via Salaria 113 - 00198, Roma

SAPIENZA
UNIVERSITÀ DI ROMA

Layering & Application Layer

Reti di Elaboratori
Corso di Laurea in Informatica
Università degli Studi di Roma "La Sapienza"

Prof.ssa Chiara Petrioli

Roberto Petroccia: petroccia@di.uniroma1.it

Parte di queste slide sono state prese dal materiale associato al libro *Computer Networking: A Top Down Approach* , 6th edition.

All material copyright 1996-2009
J.F Kurose and K.W. Ross, All Rights Reserved

Thanks also to Antonio Capone, Politecnico di Milano, Giuseppe Bianchi and Francesco LoPresti, Un. di Roma Tor Vergata

Argomenti della lezione

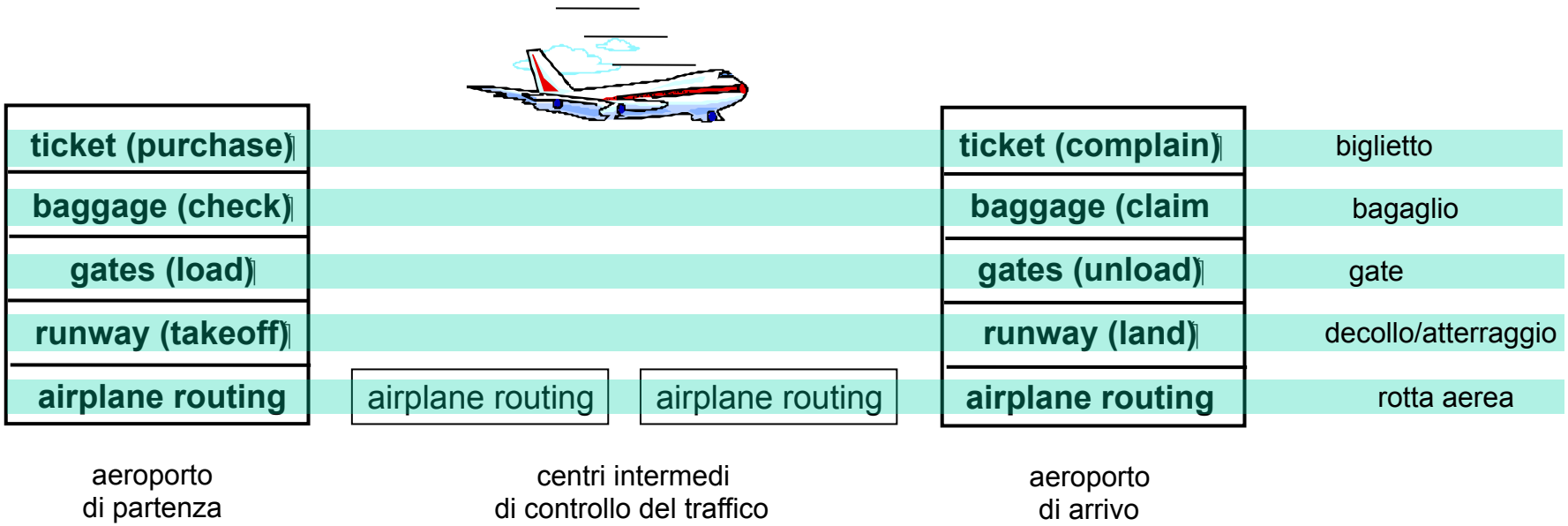
- Layering
  - Stack protocollare (TCP/IP)
  - Incapsulamento dei pacchetti
  - Reti sotto attacco: Sicurezza (accenni)

- Livello applicativo
  - Paradigma architetturale di una applicazione
  - Comunicazione tra processi
  - Servizi di trasporto richiesti dall'applicazione

Layering

- Le reti sono complesse
  - Host, sistemi terminali, server
  - Router, switch
  - svariate tipologie di mezzi trasmissivi
  - applicazioni
  - protocolli
  - hardware, software
- Necessità di un'architettura !!!
  - Strutturazione a livelli

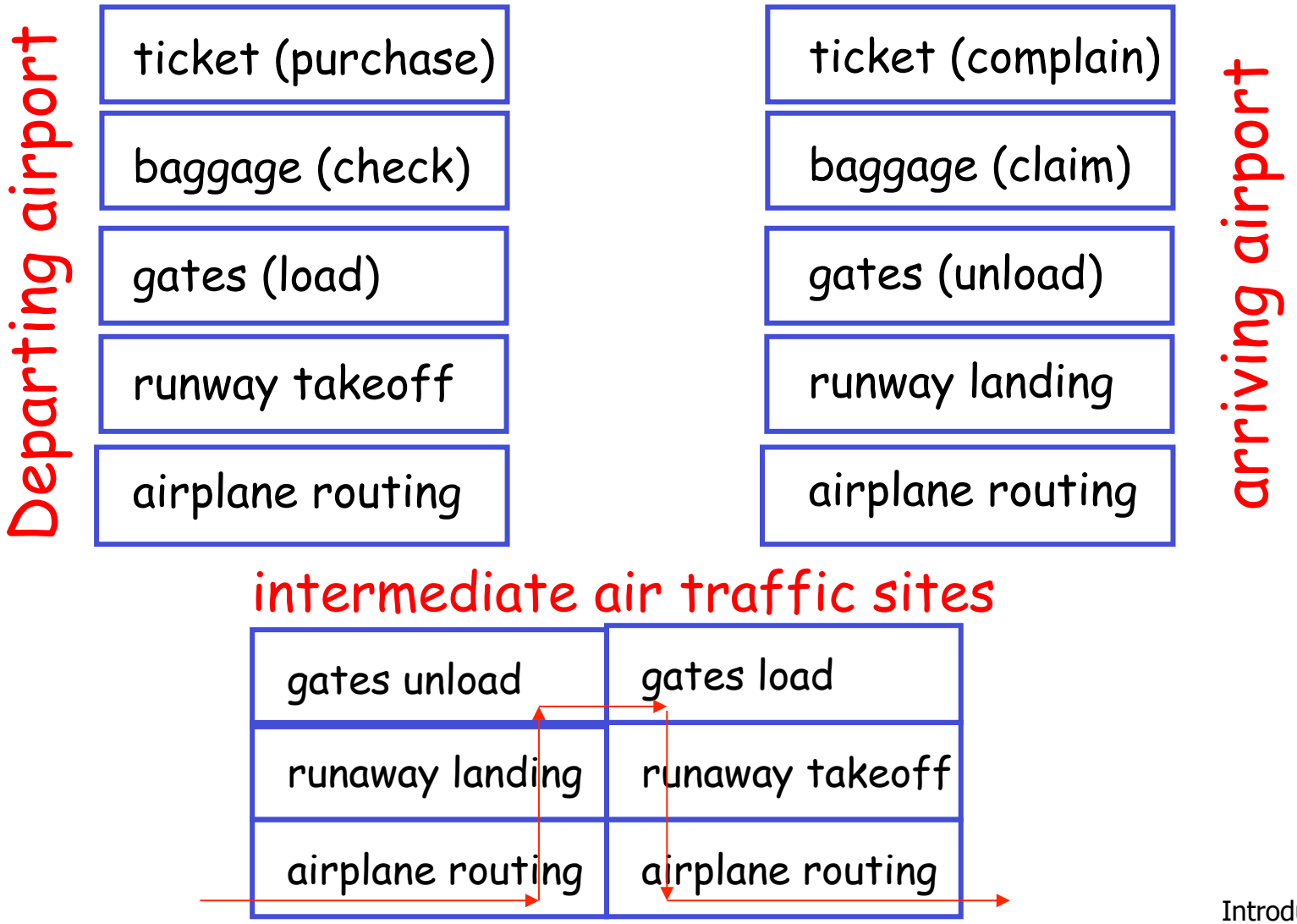
Stratificazione delle funzionalità di una linea aerea



Livelli: ciascun livello realizza un servizio

- effettuando determinate azioni all'interno del livello stesso
- utilizzando i servizi del livello immediatamente inferiore

Distributed implementation of layer functionality



Perché la stratificazione?

Quando si ha a che fare con sistemi complessi:

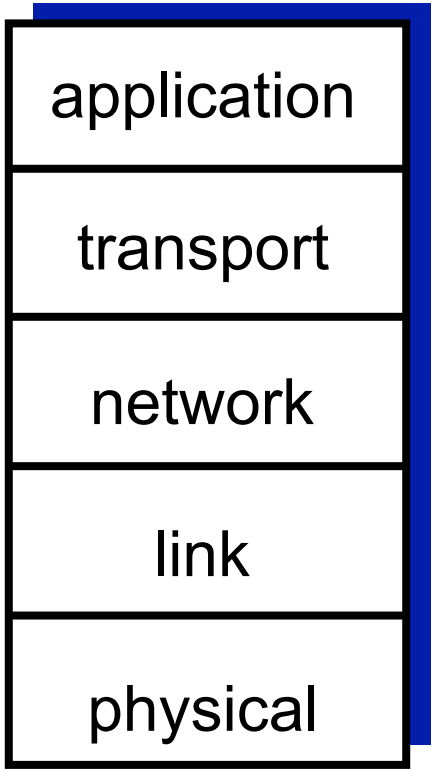
- Una struttura "esplicita" consente l'identificazione dei vari componenti di un sistema complesso e delle loro inter-relazioni
  - Ogni livello realizza un servizio
- La modularizzazione facilita la **manutenzione** e l'**aggiornamento** di un sistema
  - modifiche implementative alla realizzazione di uno dei livelli risultano **trasparenti** al resto del sistema
  - es.: modifiche nelle procedure effettuate al gate non condizionano il resto del sistema

Stratificazione in Internet

- La realizzazione dei servizi di rete è specificata dai protocolli
- I protocolli sono raggruppati in livelli in base al servizio specificato

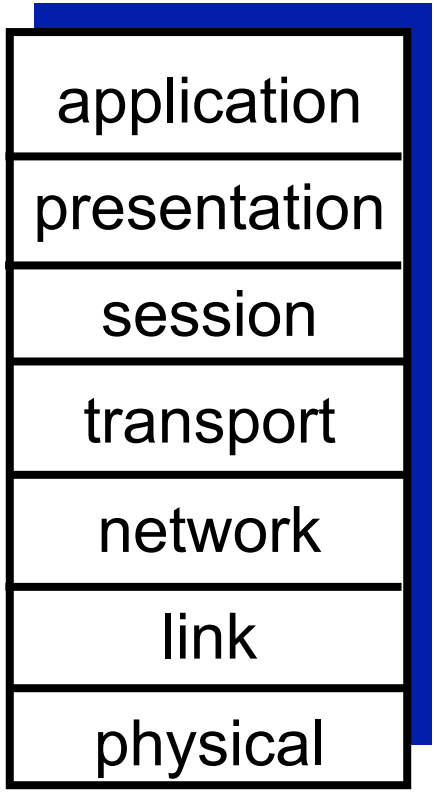
Internet protocol stack (TCP/IP)

- *application:* supporting network applications (data → messages)
  - FTP, SMTP, HTTP
- *transport:* process-process data transfer (data → segments)
  - TCP, UDP
- *network:* routing of datagrams from source to destination (data → datagrams)
  - IP, routing protocols
- *link:* data transfer between neighboring network elements (data → frames)
  - Ethernet, 802.111 (WiFi), PPP
- *physical:* bits "on the wire"

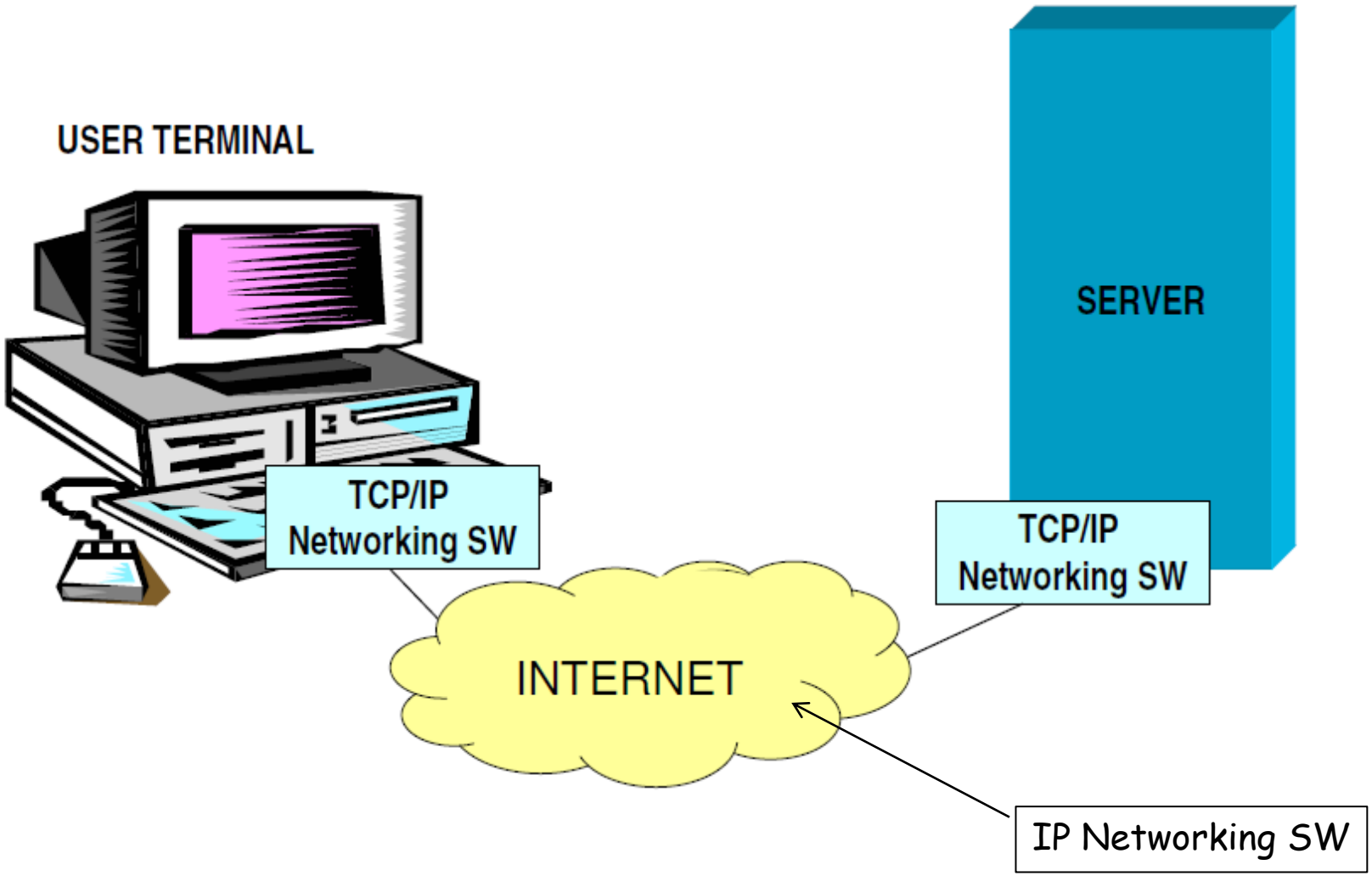


ISO/OSI reference model

- *presentation:* allow applications to interpret meaning of data, e.g., encryption, compression, machine-specific conventions
- *session:* synchronization, checkpointing, recovery of data exchange
- Internet stack "missing" these layers!
  - these services, *if needed,* must be implemented in application
  - needed?



Dove si trova il software di rete?

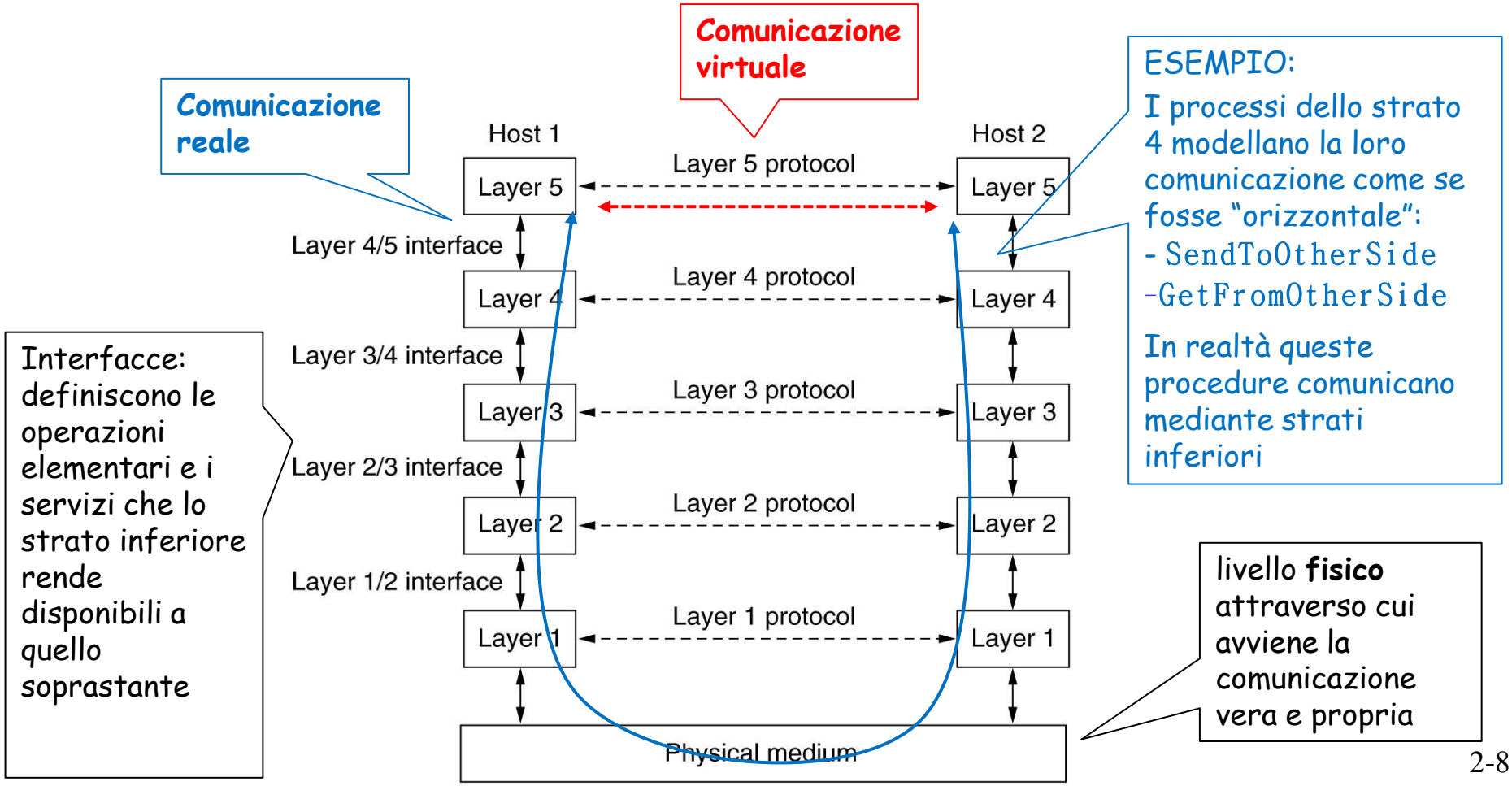


Gerarchia di protocolli

- La rete è organizzata come pila di **strati** (layer) o **livelli**, costruiti l'uno sull'altro
- Lo scopo di ogni strato è quello di **offrire** determinati **servizi** agli strati di livello superiore, nascondendo i dettagli di implementazione
- Lo strato N di un computer è in comunicazione con lo strato N di un altro computer
- Le regole e le convenzioni usate in questa comunicazione sono globalmente note come i **protocolli** dello strato N
- Le entità che formano gli strati sono chiamati **pari** (peer)
- I pari comunicano usando il protocollo
- I dati non sono trasferiti direttamente dallo strato N di un computer allo strato N di un altro computer !!!

Architettura di rete

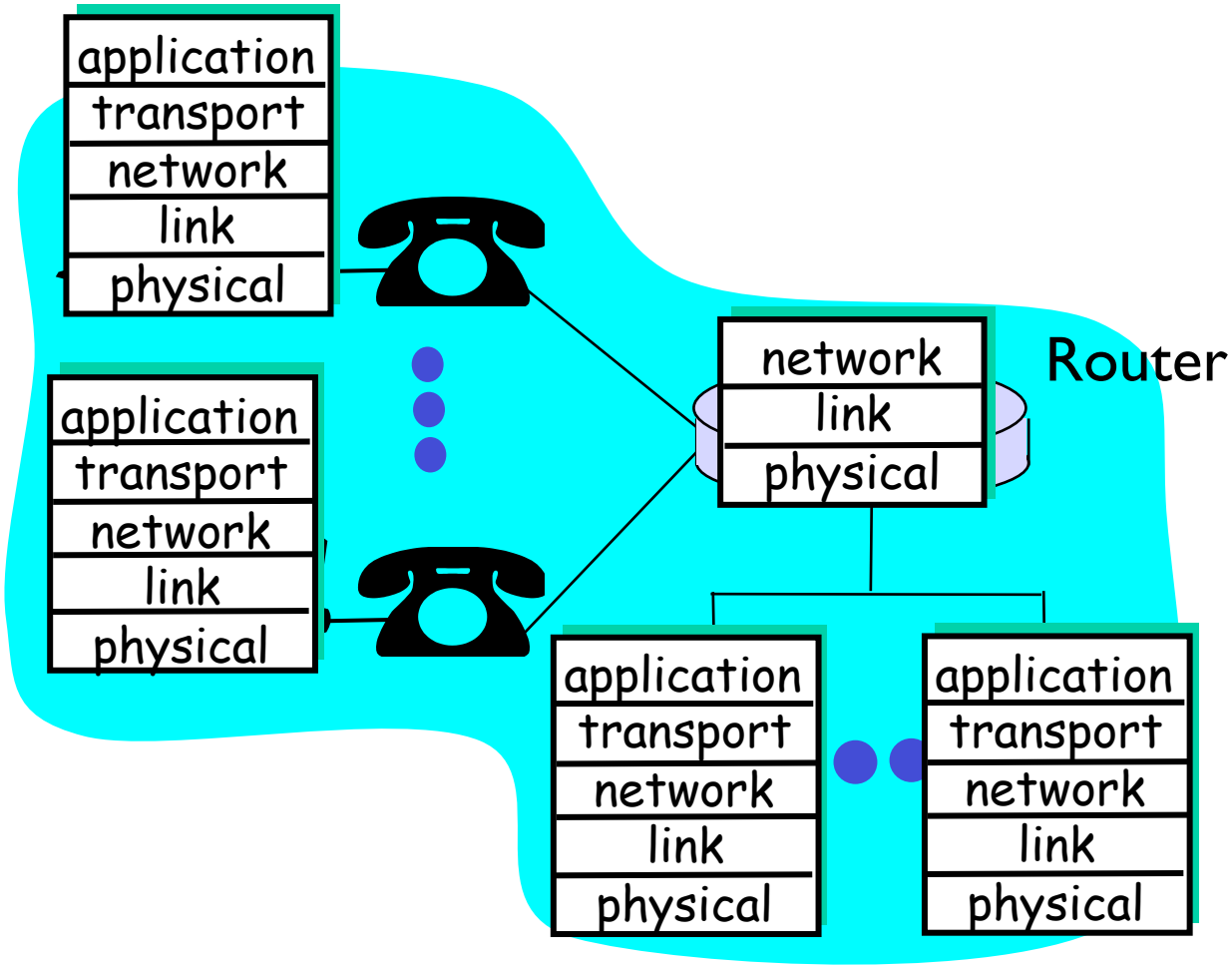
- Ogni strato passa dati e informazioni di controllo allo strato immediatamente **sottostante**, fino a raggiungere quello più basso



Layering: logical communication

Each layer:

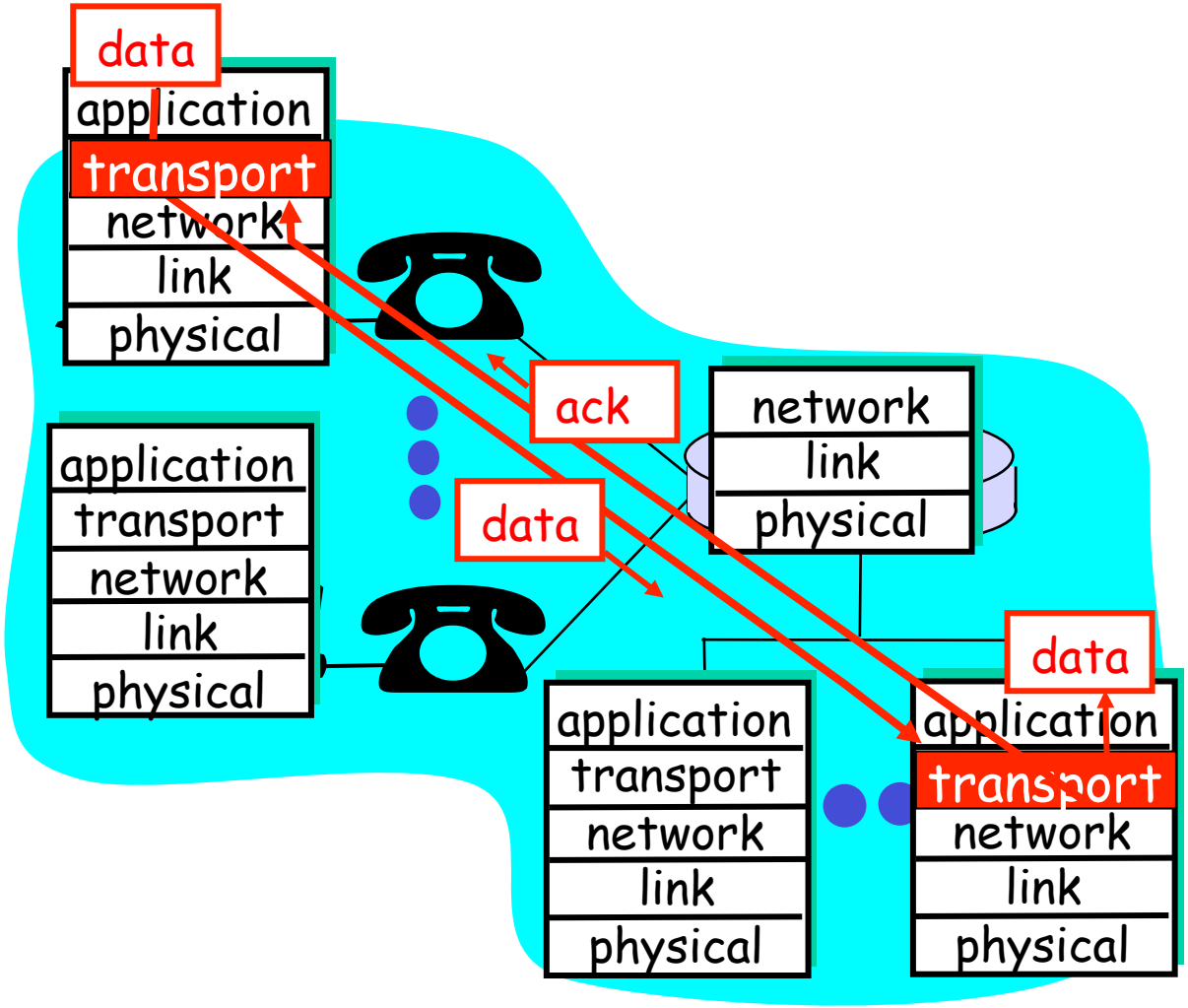
- distributed
- "entities" implement layer functions at each node
- entities perform actions, exchange messages with peers



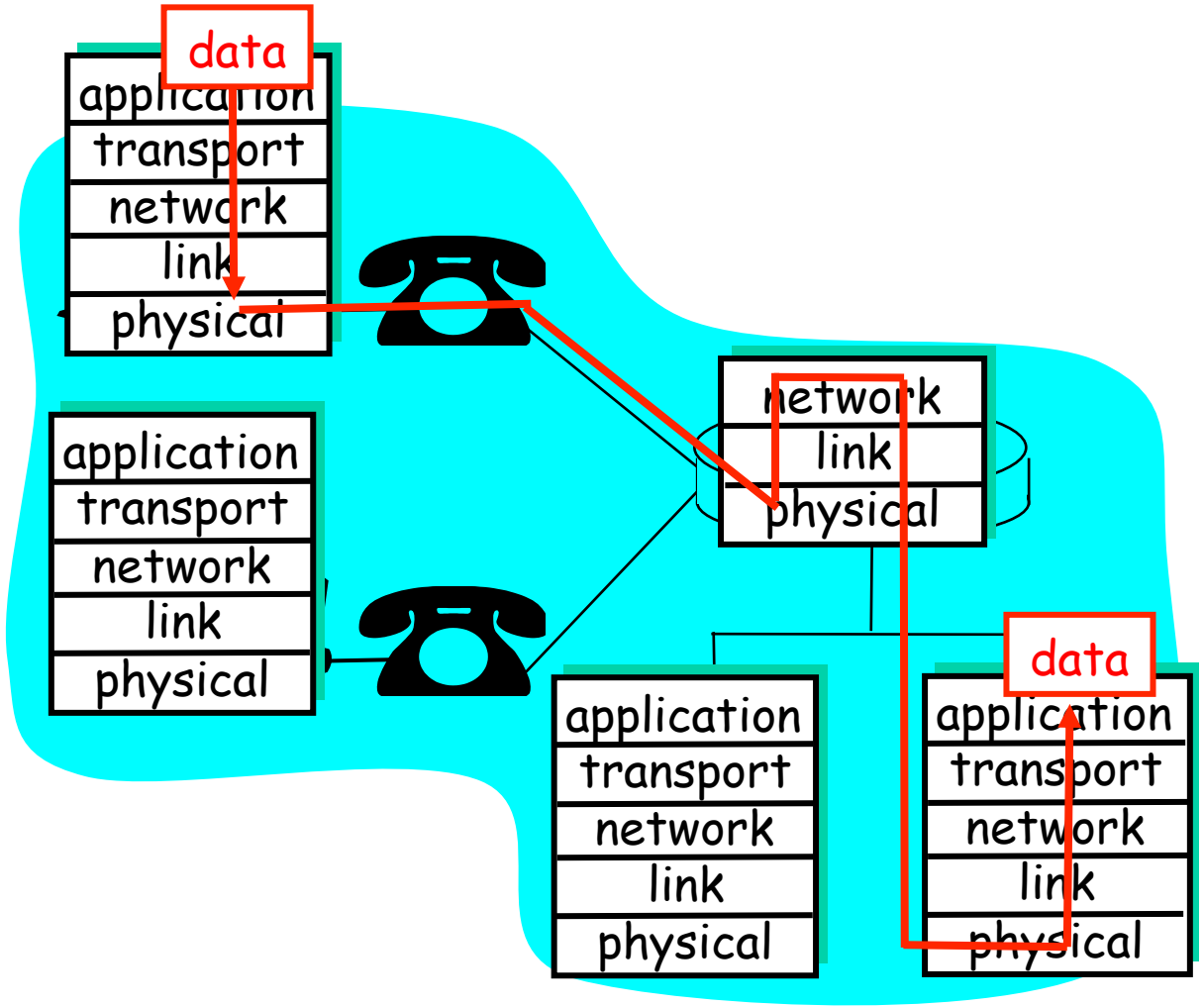
Layering: *logical* communication

E.g.: transport

- take data from app
- add addressing, reliability check info to form "datagram"
- send datagram to peer
- wait for peer to ack receipt
- analogy: post office



Layering: physical communication

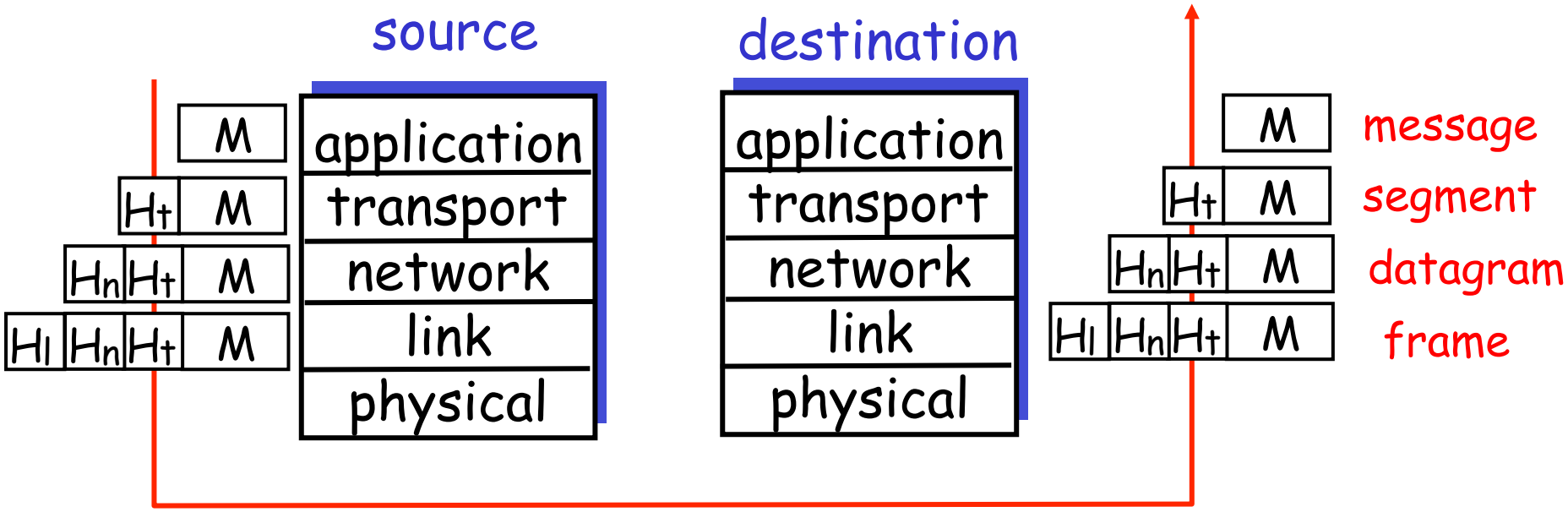


Encapsulation

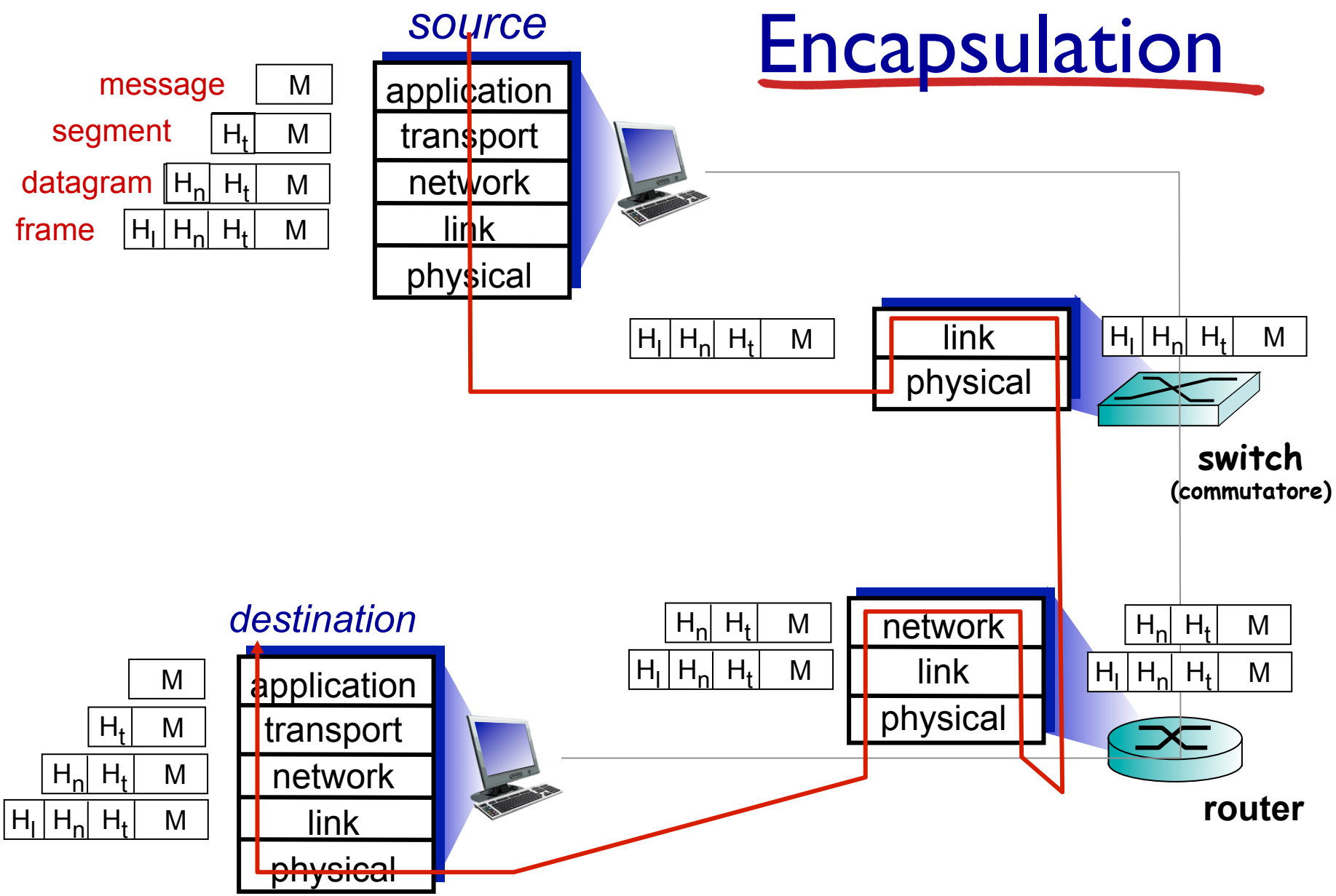
Protocol layering and data

Each layer takes data from above

- adds header information to create new data unit
- passes new data unit to layer below

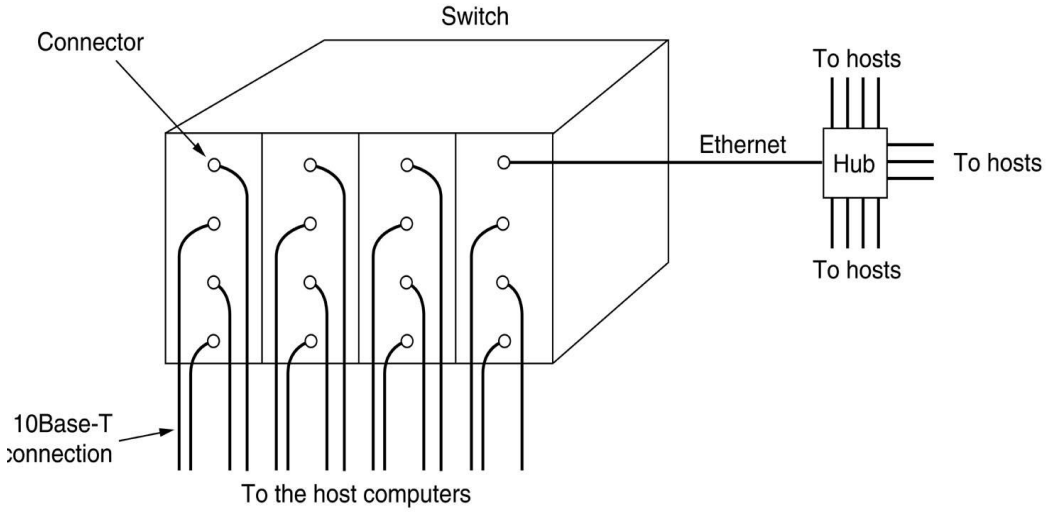


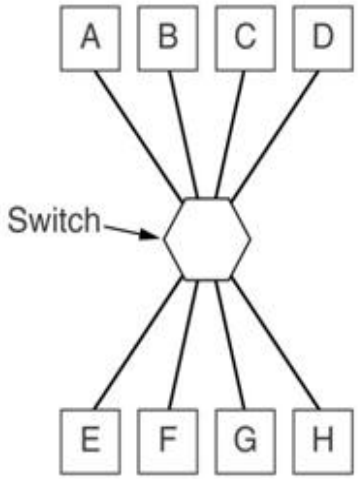
Encapsulation



Switch (commutatore)

- Instrada pacchetti al livello 2 (collegamento)
- Utilizzato per collegare singoli computer all'interno di una rete LAN



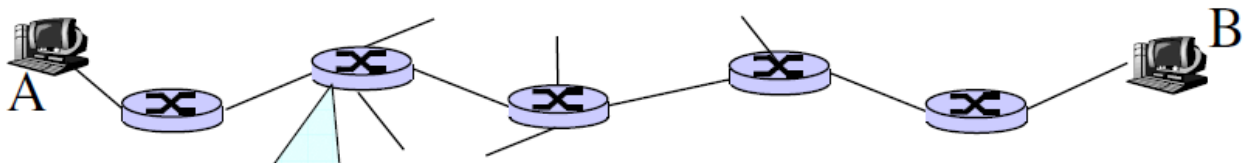


(c)

Router

- Instrada pacchetti al livello 3 (rete)

- Forward a chunk of information (called *packet*) arriving on one of its communication links to one of its outgoing communications link (the *next hop* on the source-to-destination path)



forwarding

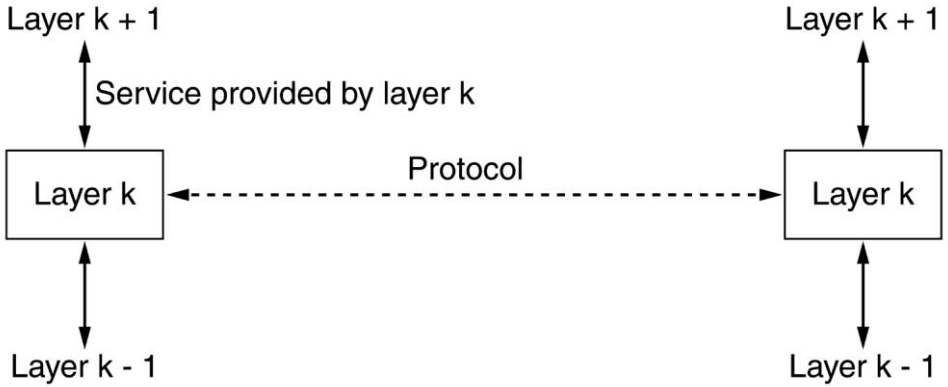
-Receives the packet
-Based on a routing table and the destination address, computes the 'next hop' to the destination
-Forwards the packet to the next hop
-The process of computing and maintaining the routing table is called Routing

Routing table

Dest. Address	Next Hop

Servizi e protocolli

- Sono concetti ben distinti
- Un <u>servizio</u> è un insieme di primitive che uno strato offre a quello superiore
  - Definisce quali operazioni lo strato è in grado di offrire, ma non dice nulla di come queste operazioni sono implementate
  - E' correlato all'interfaccia tra due strati, dove quello inferiore è il provider del servizio, mentre quello superiore è l'utente
- Un <u>protocollo</u> è un insieme di regole che controllano il formato e il significato dei pacchetti, o messaggi scambiati tra le entità pari all'interno di uno strato



Layering: pros

- Vantaggi della stratificazione
  - Modularita'
    - Semplicita' di design
    - Possibilita' di modificare un modulo in modo trasparente se le interfacce con gli altri livelli rimangono le stesse
    - Possibilita' per ciascun costruttore di adottare la propria implementazione di un livello purche' requisiti su interfacce soddisfatti
  - Gestione dell'eterogeneita'
    - Possibili moduli 'diversi' per realizzare lo stesso insieme di funzioni, che riflettano l'eterogeneita' dei sistemi coinvolti (e.g. diverse tecnologie trasmissive, LAN, collegamenti punto-punto, ATM etc.)
    - Moduli distinti possibili/necessari anche se le reti adottassero tutte la stessa tecnologia di rete perche' ad esempio le applicazioni possono avere requisiti diversi (es. UDP e TCP). All'inizio TCP ed IP erano integrati. Perche' adesso sono su due livelli distinti?

Layering: cons

- Svantaggi della stratificazione
  - A volte modularita' inficia efficienza
  - A volte necessario scambio di informazioni tra livelli non adiacenti non rispettando principio della stratificazione

Chapter 1: roadmap

1.1 what *is* the Internet?

1.2 network edge

- end systems, access networks, links

1.3 network core

- packet switching, circuit switching, network structure

1.4 delay, loss, throughput in networks

1.5 protocol layers, service models

1.6 networks under attack: security

1.7 history

Network security

- field of network security:
  - how bad guys can attack computer networks
  - how we can defend networks against attacks
  - how to design architectures that are immune to attacks
- Internet not originally designed with (much) security in mind
  - *original vision:* "a group of mutually trusting users attached to a transparent network" ☺
  - Internet protocol designers playing "catch-up"
  - security considerations in all layers!

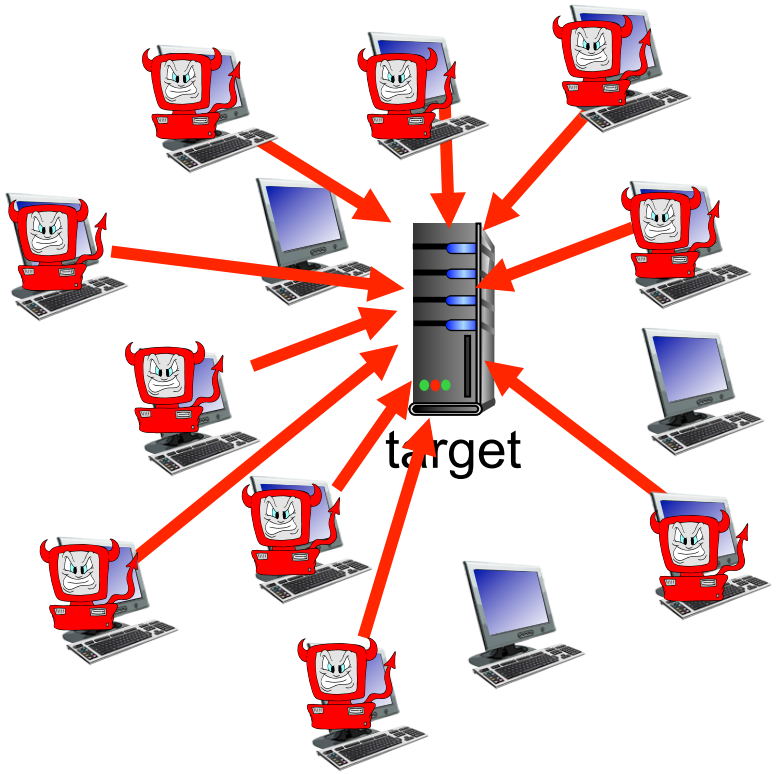
Bad guys: put malware into hosts via Internet

- malware can get in host from:
  - *virus:* self-replicating infection by receiving/executing object (e.g., e-mail attachment)
  - *worm:* self-replicating infection by passively receiving object that gets itself executed
- spyware malware can record keystrokes, web sites visited, upload info to collection site
- infected host can be enrolled in botnet, used for spam. DDoS attacks

Bad guys: attack server, network infrastructure

Denial of Service (DoS): attackers make resources (server, bandwidth) unavailable to legitimate traffic by overwhelming resource with bogus traffic

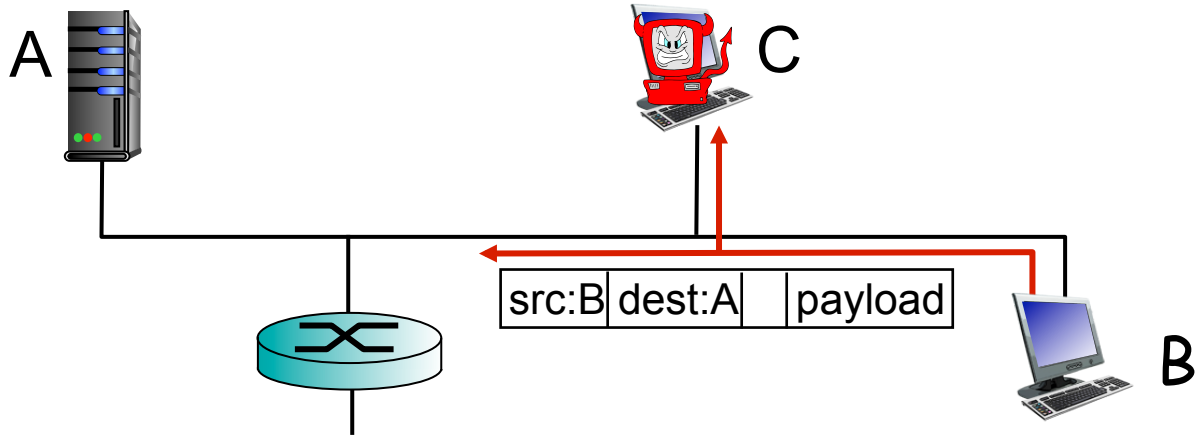
1. select target
2. break into hosts around the network (see botnet)
3. send packets to target from compromised hosts



Bad guys can sniff packets

packet "sniffing":

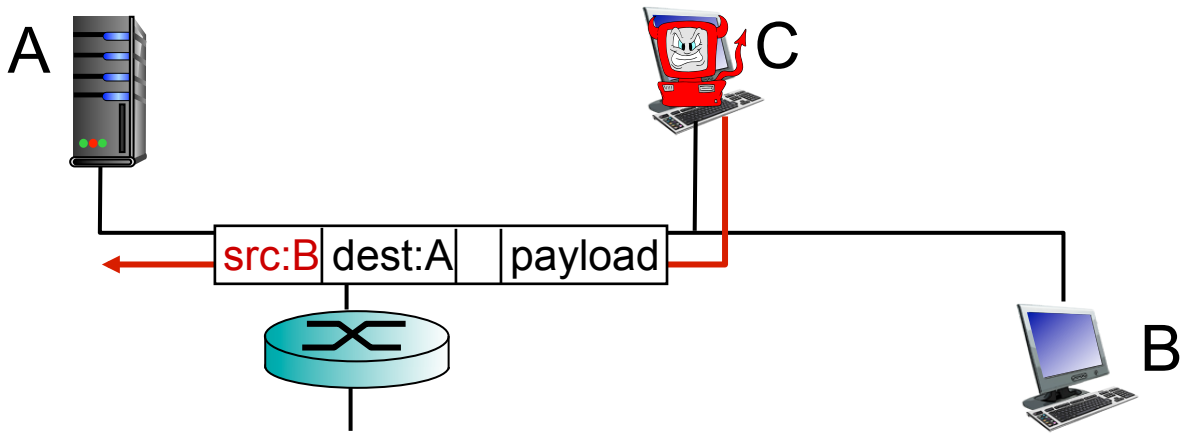
- broadcast media (shared ethernet, wireless)
- promiscuous network interface reads/records all packets (e.g., including passwords!) passing by



- wireshark software used for end-of-chapter labs is a (free) packet-sniffer

Bad guys can use fake addresses

IP spoofing: send packet with false source address



… lots more on security (throughout, Chapter 8)

Chapter 1: roadmap

1.1 what *is* the Internet?

1.2 network edge

- end systems, access networks, links

1.3 network core

- packet switching, circuit switching, network structure

1.4 delay, loss, throughput in networks

1.5 protocol layers, service models

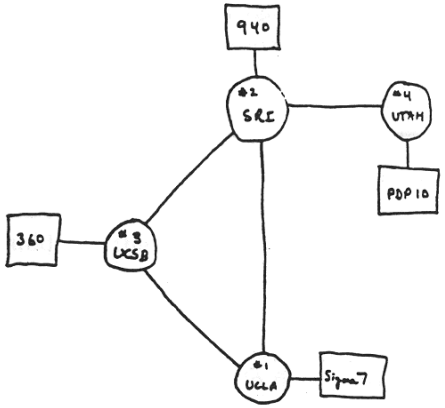
1.6 networks under attack: security

1.7 history

Internet history

1961-1972: Early packet-switching principles

- 1961: Kleinrock - queueing theory shows effectiveness of packet-switching
- 1964: Baran - packet-switching in military nets
- 1967: ARPAnet conceived by Advanced Research Projects Agency
- 1969: first ARPAnet node operational
- 1972:
  - ARPAnet public demo
  - NCP (Network Control Protocol) first host-host protocol
  - first e-mail program
  - ARPAnet has 15 nodes



THE ARPA NETWORK

Internet history

1972-1980: Internetworking, new and proprietary nets

- 1970: ALOHAnet satellite network in Hawaii
- 1974: Cerf and Kahn - architecture for interconnecting networks
- 1976: Ethernet at Xerox PARC
- late70's: proprietary architectures: DECnet, SNA, XNA
- late 70's: switching fixed length packets (ATM precursor)
- 1979: ARPAnet has 200 nodes

Cerf and Kahn's internetworking principles:

- minimalism, autonomy - no internal changes required to interconnect networks
- best effort service model
- stateless routers
- decentralized control

define today's Internet architecture

Internet history

1980-1990: new protocols, a proliferation of networks

- 1983: deployment of TCP/IP
- 1982: smtp e-mail protocol defined
- 1983: DNS defined for name-to-IP-address translation
- 1985: ftp protocol defined
- 1988: TCP congestion control
- new national networks: Csnet, BITnet, NSFnet, Minitel
- 100,000 hosts connected to confederation of networks

Internet history

1990, 2000's: commercialization, the Web, new apps

- early 1990's: ARPAnet decommissioned
- 1991: NSF lifts restrictions on commercial use of NSFnet (decommissioned, 1995)
- early 1990s: Web
  - hypertext [Bush 1945, Nelson 1960's]
  - HTML, HTTP: Berners-Lee
  - 1994: Mosaic, later Netscape
  - late 1990's: commercialization of the Web

late 1990's – 2000's:

- more killer apps: instant messaging, P2P file sharing
- network security to forefront
- est. 50 million host, 100 million+ users
- backbone links running at Gbps

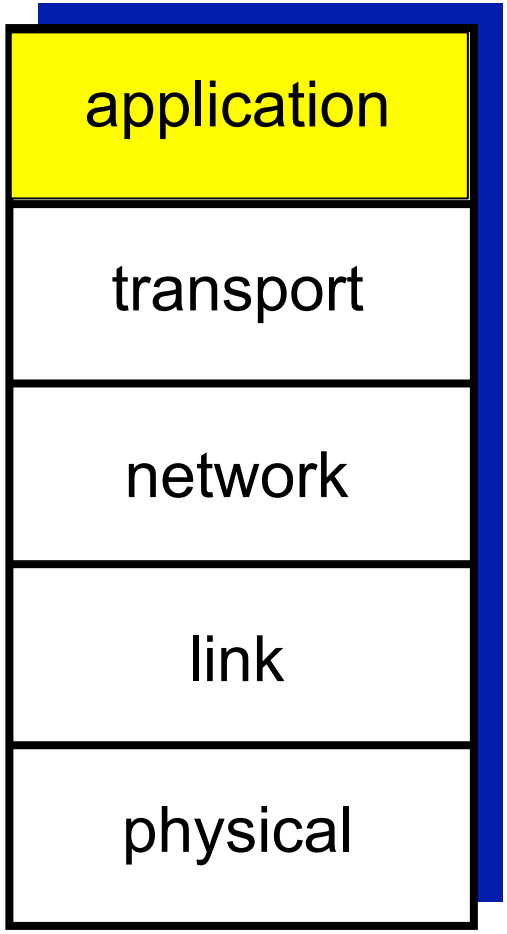
Internet history

2005-present

- ~750 million hosts
  - Smartphones and tablets
- Aggressive deployment of broadband access
- Increasing ubiquity of high-speed wireless access
- Emergence of online social networks:
  - Facebook: soon one billion users
- Service providers (Google, Microsoft) create their own networks
  - Bypass Internet, providing "instantaneous" access to search, emai, etc.
- E-commerce, universities, enterprises running their services in "cloud" (eg, Amazon EC2)

Livello applicativo

- Principi delle applicazioni di rete
- Web e HTTP
- FTP
- Posta elettronica
  - SMTP, POP3, IMAP
- DNS
- Applicazioni P2P
- Programmazione delle socket con TCP
- Programmazione delle socket con UDP



Alcune applicazioni di rete

- Posta elettronica
- Web
- Messaggistica istantanea
- Autenticazione in un calcolatore remoto
- Condivisione di file P2P
- Giochi multiutente via rete
- Streaming di video-clip memorizzati
- Telefonia via Internet
- Videoconferenza in tempo reale

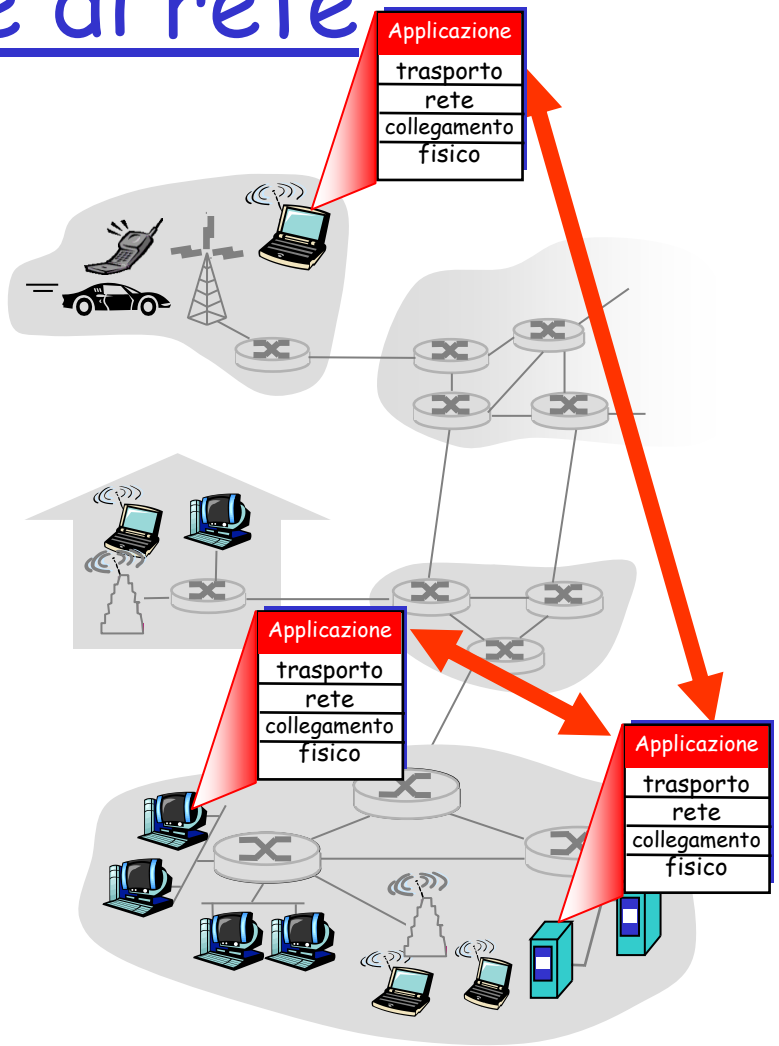
Creare un'applicazione di rete

Scrivere programmi che

- girano su sistemi terminali diversi
- comunicano attraverso la rete
- Ad es. il software di un server Web comunica con il software di un browser

software in grado di funzionare su più macchine

- non occorre predisporre programmi per i dispositivi del nucleo della rete, quali router o commutatori Ethernet
- I programmi applicativi sono indipendenti dalla tecnologia che c'è sotto



Esempio: email

Una nuova applicazione di rete

- Supponiamo di voler creare una nuova applicazione di rete, allora è necessario avere un piano architetturale:

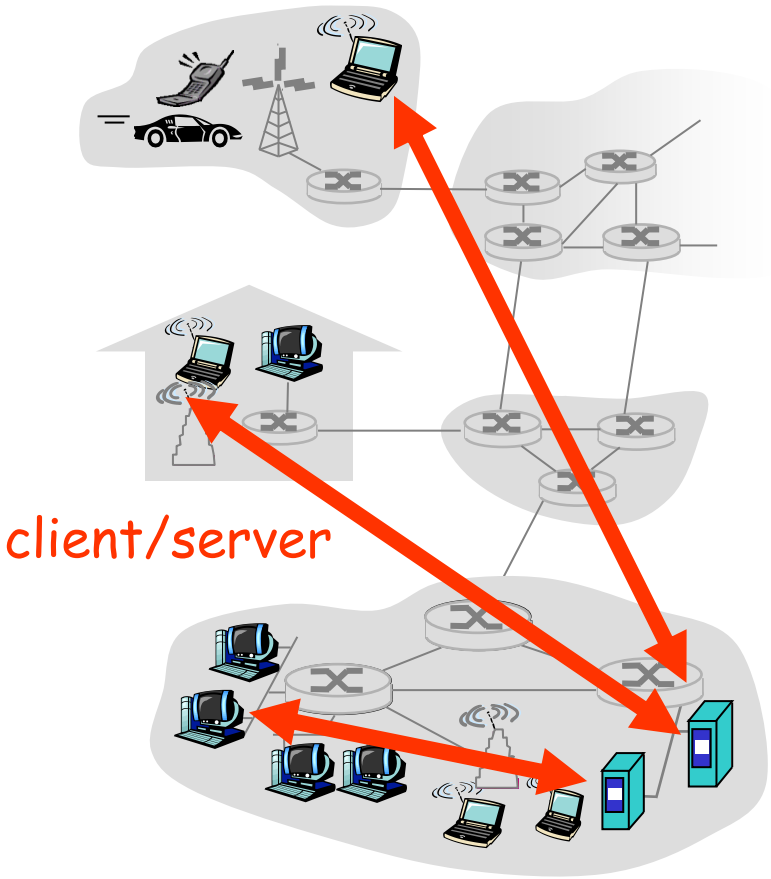
- Che tipo di **architettura** si vuole creare (client-server, peer-to-peer)?
- Come **comunicano** i processi dell'applicazione?
- Che tipo di **servizi** (di rete) richiede l'applicazione (affidabilità, banda)?

Architettura dell'applicazione

Paradigma:

- Client-server
- Peer-to-peer (P2P)
- Architetture ibride (client-server e P2P)

Architettura client-server



server:

- host sempre attivo
- indirizzo IP fisso
- server farm per creare un potente server virtuale

client:

- comunica con il server
- può contattare il server in qualunque momento
- può avere indirizzi IP dinamici
- non comunica direttamente con gli altri client

Client-server paradigm

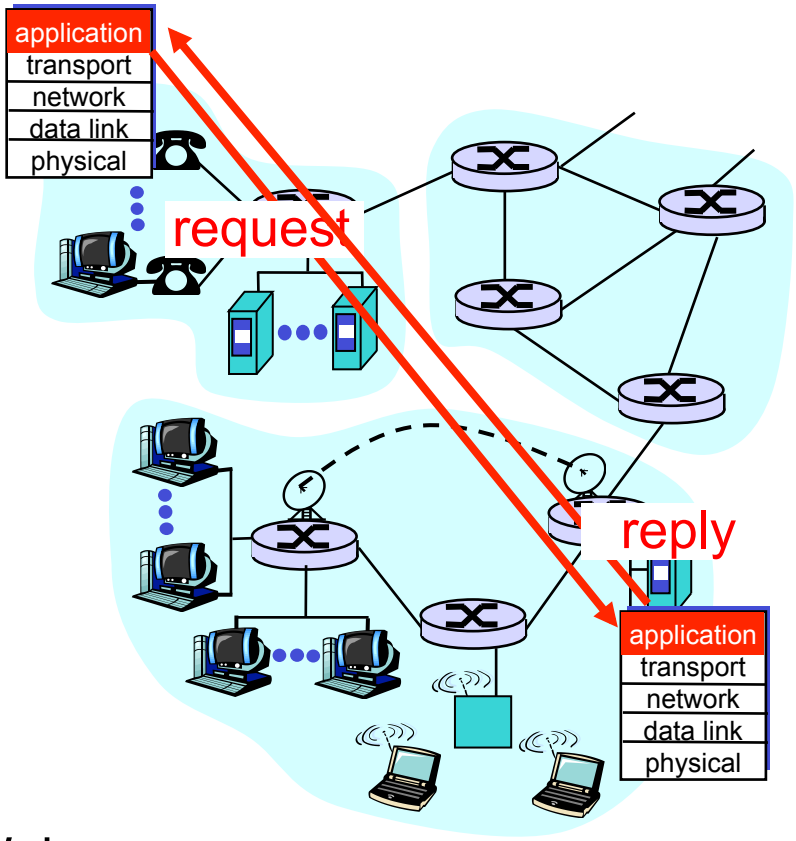
Typical network app has two pieces: *client* and *server*

Client:

- initiates contact with server ("speaks first")
- typically requests service from server,
- Web: client implemented in browser; e-mail: in mail reader

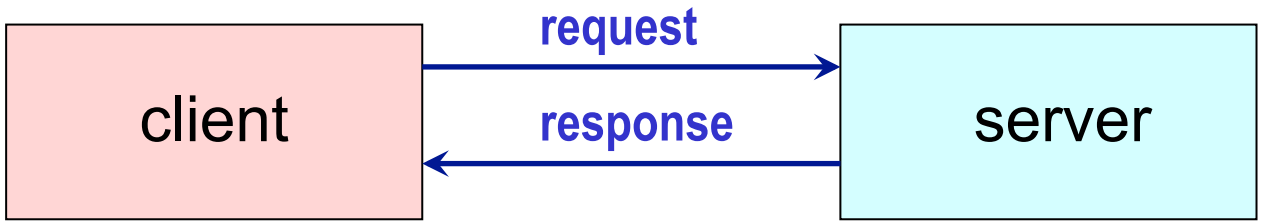
Server:

- provides requested service to client
- e.g., Web server sends requested Web page, mail server delivers e-mail



Architettura Client-Server

- Un processo client è solo in grado fare richieste di servizio (informazioni) e di interpretare le risposte
- Un processo server ha solo il compito di interpretare le richieste e fornire le risposte
- Se è necessario nello stesso host sia fare richieste che fornire risposte vengono usati due processi, client e server.
- Un protocollo applicativo per un'architettura client-server rispecchia questa divisione di ruoli e prevede messaggi di richiesta (request) generati dal lato client e messaggi di risposta (response) generati dal server



Programmi Client e Server

- Distinguiamo tra programma (software) e processo, istanza del programma in esecuzione su un host
- Un processo server è in esecuzione a tempo illimitato sul proprio host (daemon) e viene attivato mediante una *passive open*
- Un processo client viene attivato solo al momento di fare le richieste e viene attivato mediante una *active open* su richiesta dell'utente o di altro processo applicativo
- la passive open del server fa si che da quel momento il server accetti richieste dai client
- l'active open del client richiede l'indicazione dell'indirizzo e della porta del client, ed inizia la connessione TCP per mettere in comunicazione il client e il server

Data center e Server virtuali

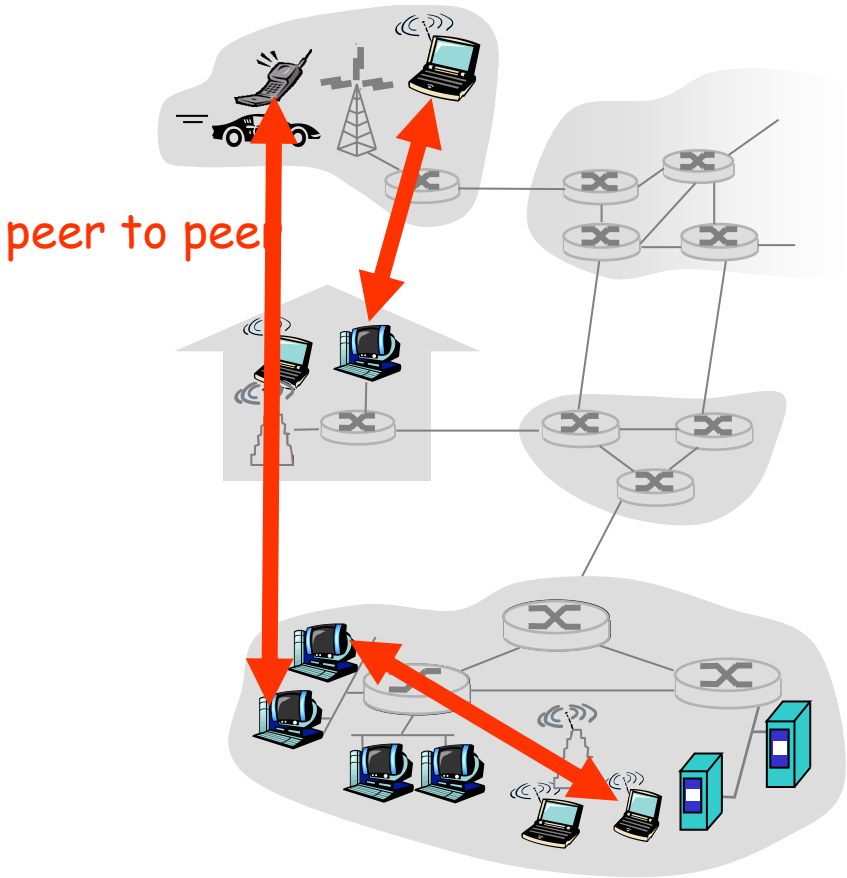
- Spesso un server in esecuzione su un singolo host può non essere in grado di gestire un numero elevato di client (Google, Bing, Amazon, e-Bay, Facebook, Twittter, etc.)
- Esistono i cosiddetti **data center**, ossia un insieme di più host fisicamente collegati tra loro che creano un **server virtuale** più potente di un server normale su un singolo host**.**
- Più data center possono essere interconnessi tra loro (Google ha 30-50 data center sparsi per il mondo) che collaborano tra loro e si scambiano informazioni per gestire le richieste degli utenti ai diversi servizi
- Un data center può avere centinaia o migliaia di server che devono essere alimentati e continuamenti gestiti per un corretto funzionamento
- Banda aggiuntiva è necessaria per lo scambio di informazioni tra i data center

Architettura P2P pura

- non c'è un server sempre attivo
- coppie arbitrarie di host (peer) comunicano direttamente tra loro
- i peer non devono necessariamente essere sempre attivi, e cambiano indirizzo IP

Facilmente scalabile

Difficile da gestire



Spesso si usa un sistema ad incentivi

Ibridi (client-server e P2P)

Skype

- Applicazione P2P di Voice over IP
- Server centralizzato: ricerca indirizzi della parte remota
- Connessione client-client: diretta (non attraverso il server)

Messaggistica istantanea

- La chat tra due utenti è del tipo P2P
- Individuazione della presenza/location centralizzata:
  - l'utente registra il suo indirizzo IP sul server centrale quando è disponibile online
  - l'utente contatta il server centrale per conoscere gli indirizzi IP dei suoi amici

Comunicazione tra Processi

Processo: programma in esecuzione su di un host.

- All'interno dello stesso host, due processi comunicano utilizzando schemi interprocesso (definiti dal SO).
- processi su host differenti comunicano attraverso lo scambio di messaggi

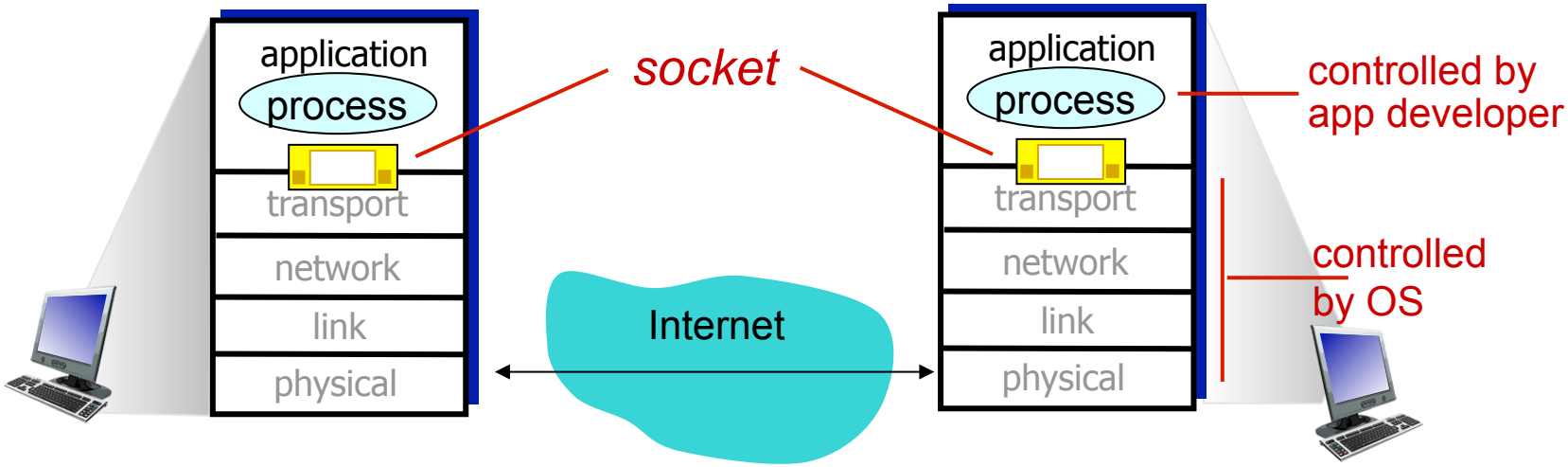
Processo client: processo che dà inizio alla comunicazione

Processo server : processo che attende di essere contattato

- Nota: le applicazioni con architetture P2P hanno processi client e processi server

Sockets

- process sends/receives messages to/from its socket
- socket analogous to door
  - sending process shoves message out door
  - sending process relies on transport infrastructure on other side of door to deliver message to socket at receiving process

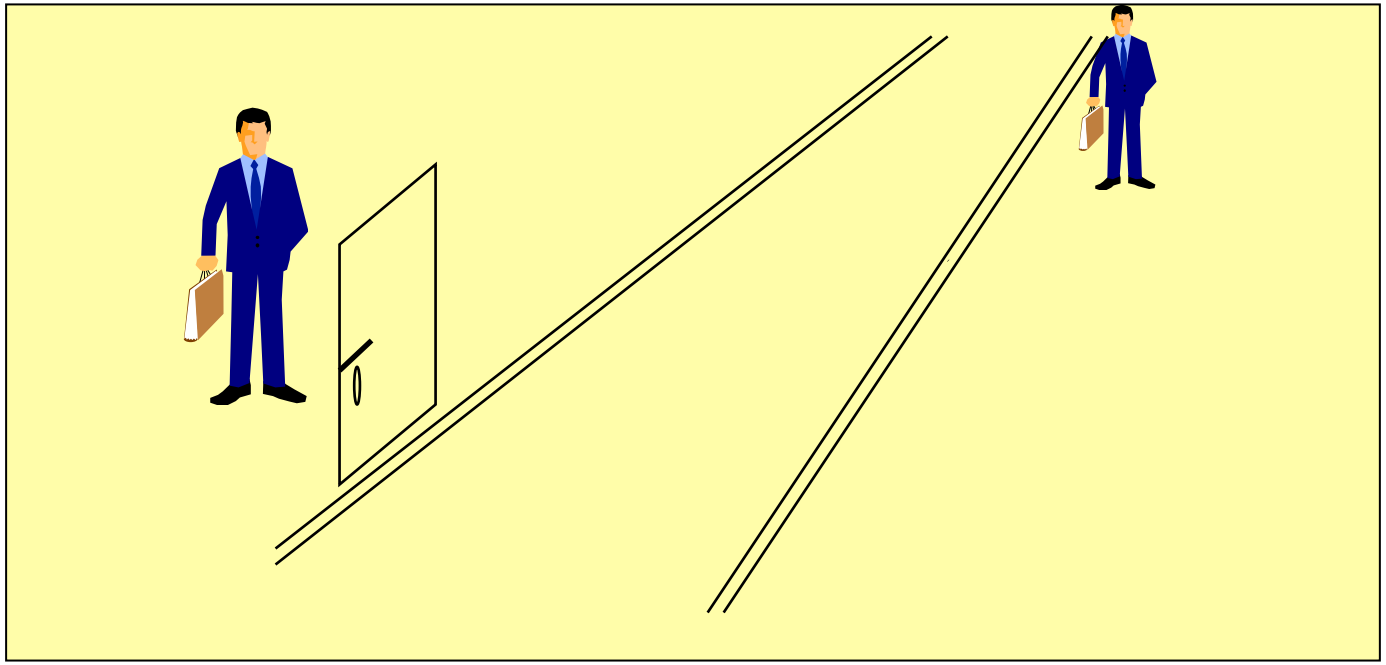


Indirizzamento dei processi

- Affinché un processo su un host invii un messaggio a un processo su un altro host, il mittente deve identificare il processo destinatario.
- Un host ha un indirizzo IP univoco a 32 bit (es. 127.0.0.1)
- D: È sufficiente conoscere l'indirizzo IP dell'host su cui è in esecuzione il processo per identificare il processo stesso?
- Risposta: No, sullo stesso host possono essere in esecuzione molti processi.
- L'identificatore comprende sia l'indirizzo IP che i numeri di porta associati al processo in esecuzione su un host.
- Esempi di numeri di porta:
  - HTTP server: 80
  - Mail server: 25
- Per inviare un messaggio HTTP al server gaia.cs.umass.edu:
  - Indirizzo IP: 128.119.245.12
  - Numero di porta: 80

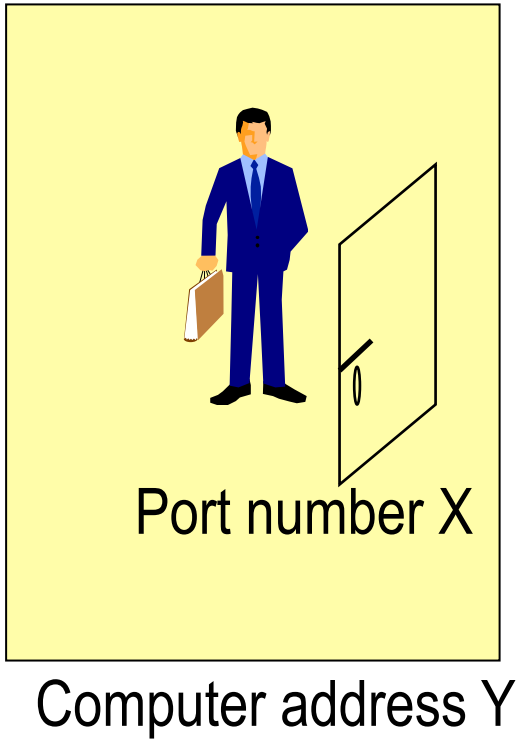
N.B. Il numero di porta è una delle informazione contenute negli header di livello di trasporto per capire a quale applicazione bisogna riportare il messaggio

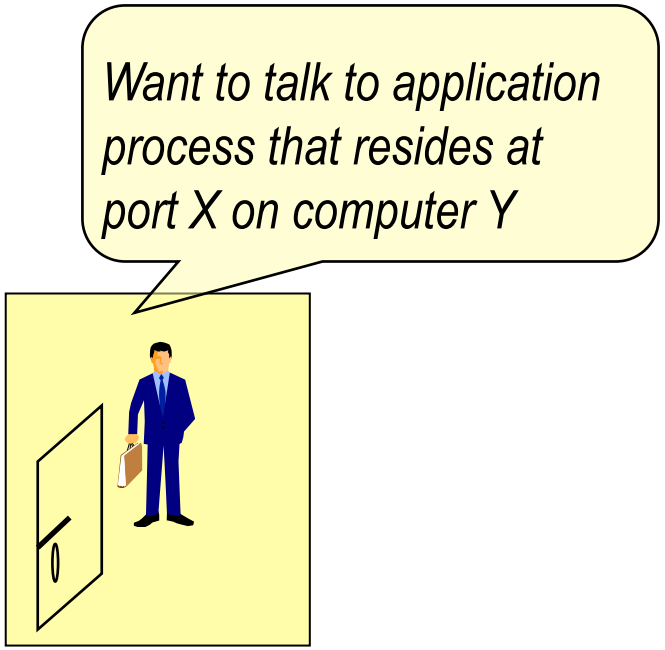
Port numbers



the “address” of the SW process inside the computer!

Addressing scheme



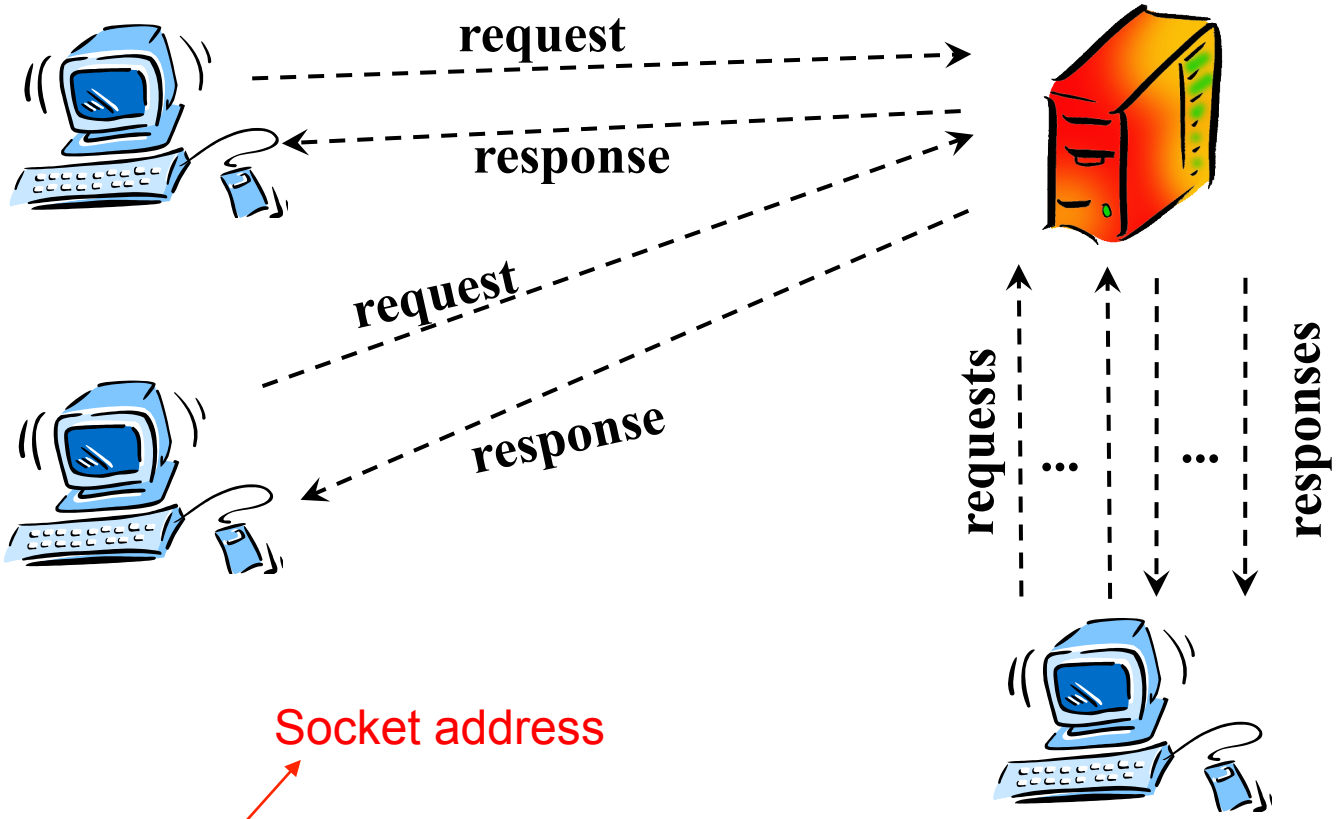


Port numbers

- 16 bit address (0-65535)
- well known port numbers for common servers
  - FTP 20, TELNET 23, SMTP 25, HTTP 80, POP3 110, … (full list: RFC 1700)
- number assignment (by IANA)
  - 0 not used
  - 1-255 reserved for well known processes
  - 256-1023 reserved for other processes
  - 1024-65535 dedicated to user apps

Programmi Client e Server

- Normalmente più client possono inviare richieste ad uno stesso server
- Un client può fare più richieste contemporanee

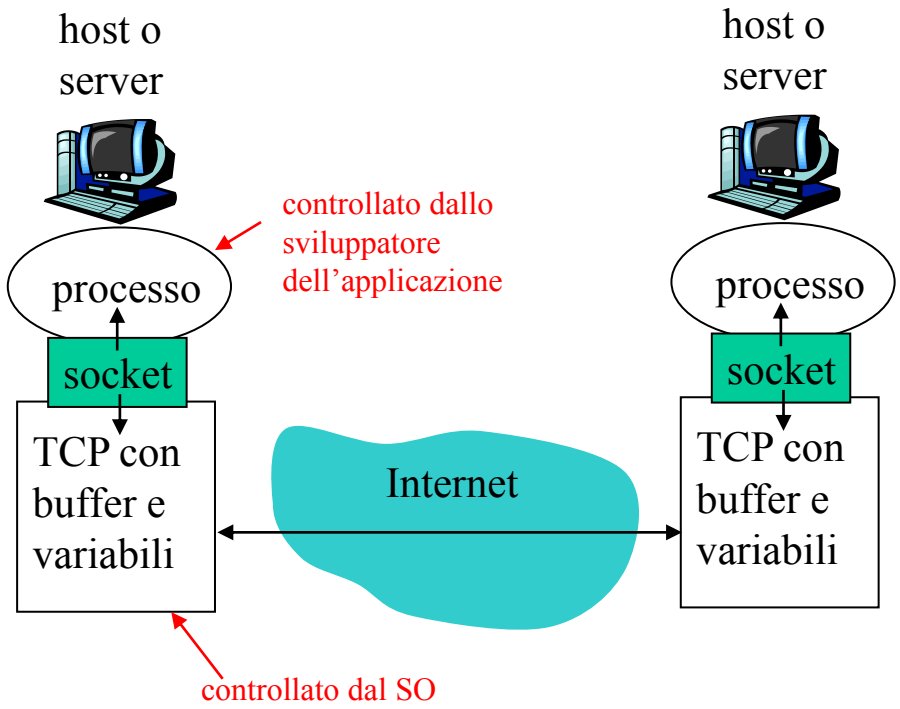


Socket address

- Un flusso di informazioni tra due processi identificato da una quadrupla *(indirizzo IP sorgente, porta sorgente, indirizzo IP destinazione, porta destinazione)*
- Di solito il client NON USA un well known port #
  - OS assegna un num. di porta disponibile

Socket

- un processo invia/riceve messaggi a/da la sua socket
- una socket è analoga a una porta
  - un processo che vuole inviare un messaggio, lo fa uscire dalla propria "porta" (socket)
  - il processo presuppone l'esistenza di un'infrastruttura esterna che trasporterà il messaggio attraverso la rete fino alla "porta" del processo di destinazione



- Una socket è l'interfaccia tra il livello di applicazione e il livello di trasporto
- API: (1) scelta del protocollo di trasporto; (2) capacità di determinare alcuni parametri (approfondiremo questo aspetto più avanti)

Quale servizio richiede l'applicazione?

Perdita di dati

- alcune applicazioni (ad esempio, audio) possono tollerare qualche perdita
- altre applicazioni (ad esempio, trasferimento di file, telnet) richiedono un trasferimento dati affidabile al 100%

Temporizzazione

- alcune applicazioni (ad esempio, telefonia Internet, giochi interattivi) per essere "realistiche" richiedono piccoli ritardi
- Applicazioni come la posta elettronica non hanno particolari requisiti di temporizzazione

Throughput

- alcune applicazioni (ad esempio, quelle multimediali) per essere "efficaci" richiedono un'ampiezza di banda minima
- altre applicazioni ("le applicazioni elastiche") utilizzano l'ampiezza di banda che si rende disponibile

Sicurezza

- Cifratura, integrità dei dati, ...

Requisiti del servizio di trasporto di alcune applicazioni comuni

Applicazione	Tolleranza alla perdita di dati	Throughput	Sensibilità al tempo
Trasferimento file	No	Variabile	No
Posta elettronica	No	Variabile	No
Documenti Web	No	Variabile	No
Audio/video in tempo reale	Sì	Audio: da 5 Kbps a 1 Mbps Video: da 10 Kbps a 5 Mbps	Sì, centinaia di ms
Audio/video memorizzati	Sì	Come sopra	Sì, pochi secondi
Giochi interattivi	Sì	Fino a pochi Kbps	Sì, centinaia di ms
Messaggistica istantanea	No	Variabile	Sì e no

Servizi dei protocolli di trasporto Internet

Servizio di TCP:

- ***Orientato alla connessione:*** è richiesto un setup fra i processi client e server
  - *Somiglia al* ***sistema telefonico:*** come per eseguire una telefonata, l'utente deve stabilire una connessione, usarla e quindi rilasciarla.
  - Funziona come un tubo: il trasmettitore vi spinge oggetti (bit) a una estremità e il ricevitore li prende dall'altra. L'ordine è conservato ovvero i bit arrivano nella sequenza con cui sono stati trasmessi.
- *trasporto affidabile* fra i processi d'invio e di ricezione
- *controllo di flusso:* il mittente non vuole sovraccaricare il destinatario
- *controllo della congestione:* "strozza" il processo d'invio quando le rete è sovraccaricata
- *non offre:* temporizzazione, garanzie su un'ampiezza di banda minima, sicurezza (alcune possono essere implementate, Es. SSL)

Servizi dei protocolli di trasporto Internet

Servizio di UDP:

- ***Senza connessione:*** non è richiesto alcun setup fra i processi client e server
- trasferimento dati inaffidabile fra i processi d'invio e di ricezione
  - *Somiglia al **sistema postale**:* Ogni messaggio è instradato attraverso il sistema postale in modo indipendente dagli altri
  - E' possibile che due messaggi mandati alla stessa destinazione arrivino in tempi diversi
- non offre: setup della connessione, affidabilità, controllo di flusso, controllo della congestione, temporizzazione né ampiezza di banda minima e sicurezza

D: Perché esiste UDP?

Applicazioni Internet: protocollo a livello applicazione e protocollo di trasporto

Applicazione	Protocollo a livello applicazione	Protocollo di trasporto sottostante
Posta elettronica	SMTP [RFC 2821]	TCP
Accesso a terminali remoti	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
Trasferimento file	FTP [RFC 959]	TCP
Multimedia in streaming	HTTP (es. YouTube) RTP [RFC 1889]	TCP o UDP
Telefonia Internet	SIP, RTP, proprietario (es. Skype)	Tipicamente UDP anche TCP

Protocollo a livello di applicazione

Definisce:

- **Tipi** di messaggi scambiati, ad esempio messaggi di richiesta e di risposta
- **Sintassi** dei tipi di messaggio: quali sono i campi nel messaggio e come sono descritti
- **Semantica** dei campi, ovvero significato delle informazioni nei campi
- **Regole** per determinare quando e come un processo invia e risponde ai messaggi

Protocolli di pubblico dominio:

- Definiti nelle RFC
- Consentono l'interoperabilità
- Ad esempio, HTTP, SMTP

Protocolli proprietari:

- Ad esempio, Skype