

Il livello trasporto nella rete Internet: TCP e UDP

Mario BALDI

<http://staff.polito.it/mario.baldi>

Silvano GAI

[sgai\[at\]cisco.com](mailto:sgai[at]cisco.com)

Jim KUROSE

<http://www-aml.cs.umass.edu>

Fulvio RISSO

[fulvio.risso\[at\]polito.it](mailto:fulvio.risso[at]polito.it)

Nota di Copyright

Questo insieme di trasparenze (detto nel seguito slide) è protetto dalle leggi sul copyright e dalle disposizioni dei trattati internazionali. Il titolo ed i copyright relativi alle slide (ivi inclusi, ma non limitatamente, ogni immagine, fotografia, animazione, video, audio, musica e testo) sono di proprietà degli autori indicati a pag. 1.

Le slide possono essere riprodotte ed utilizzate liberamente dagli istituti di ricerca, scolastici ed universitari per scopi istituzionali, non a fine di lucro. In tal caso non è richiesta alcuna autorizzazione.

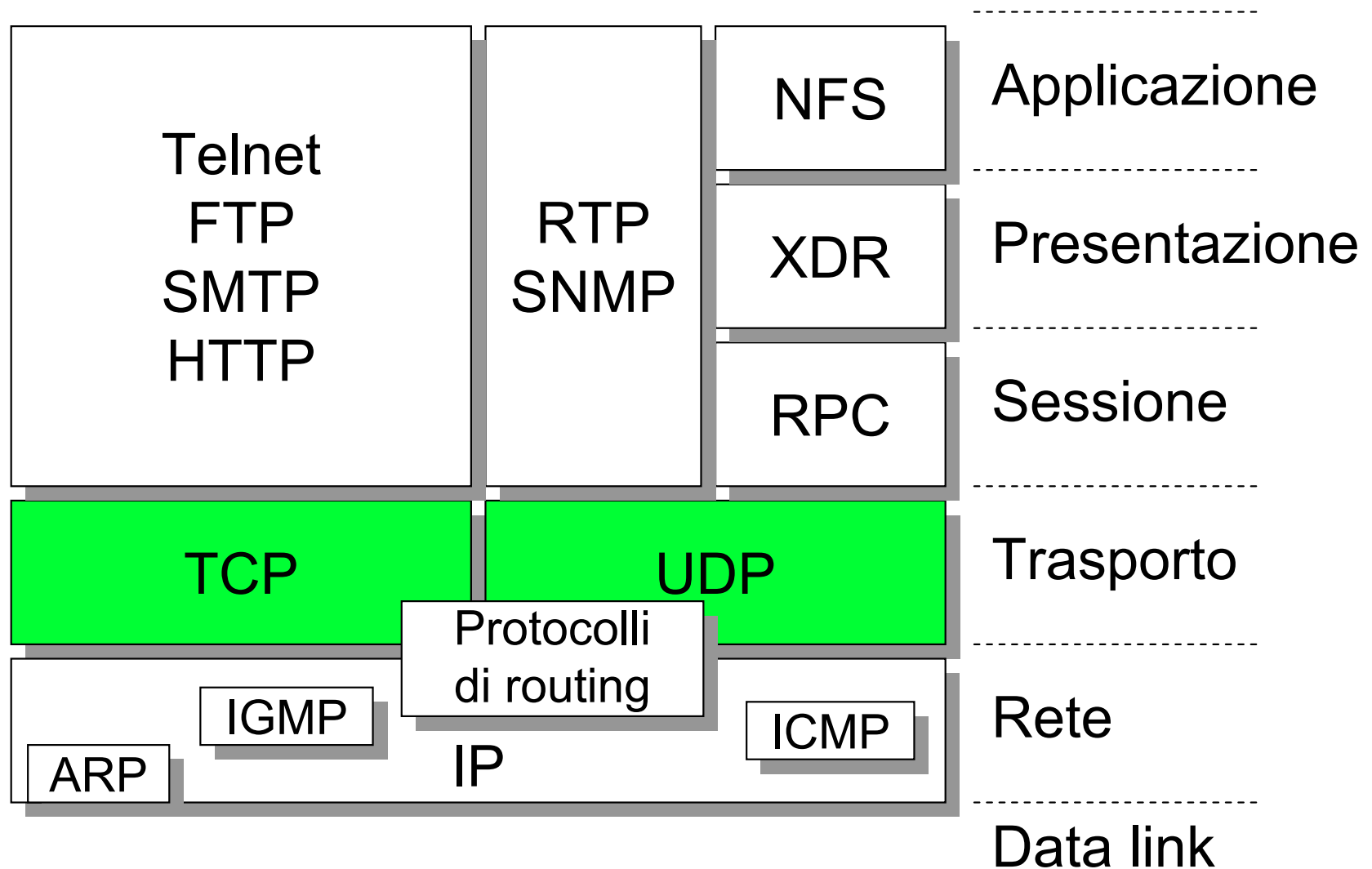
Ogni altra utilizzo o riproduzione (ivi incluse, ma non limitatamente, le riproduzioni su supporti magnetici, su reti di calcolatori e stampate) in toto o in parte è vietata, se non esplicitamente autorizzata per iscritto, a priori, da parte degli autori.

L'informazione contenuta in queste slide è ritenuta essere accurata alla data dell'edizione. Essa è fornita per scopi meramente didattici e non per essere utilizzata in progetti di impianti, prodotti, reti, ecc. In ogni caso essa è soggetta a cambiamenti senza preavviso. Gli autori non assumono alcuna responsabilità per il contenuto di queste slide (ivi incluse, ma non limitatamente, la correttezza, completezza, applicabilità, aggiornamento dell'informazione).

In ogni caso non può essere dichiarata conformità all'informazione contenuta in queste slide.

In ogni caso questa nota di copyright non deve mai essere rimossa e deve essere riportata anche in utilizzi parziali.

I protocolli TCP e UDP

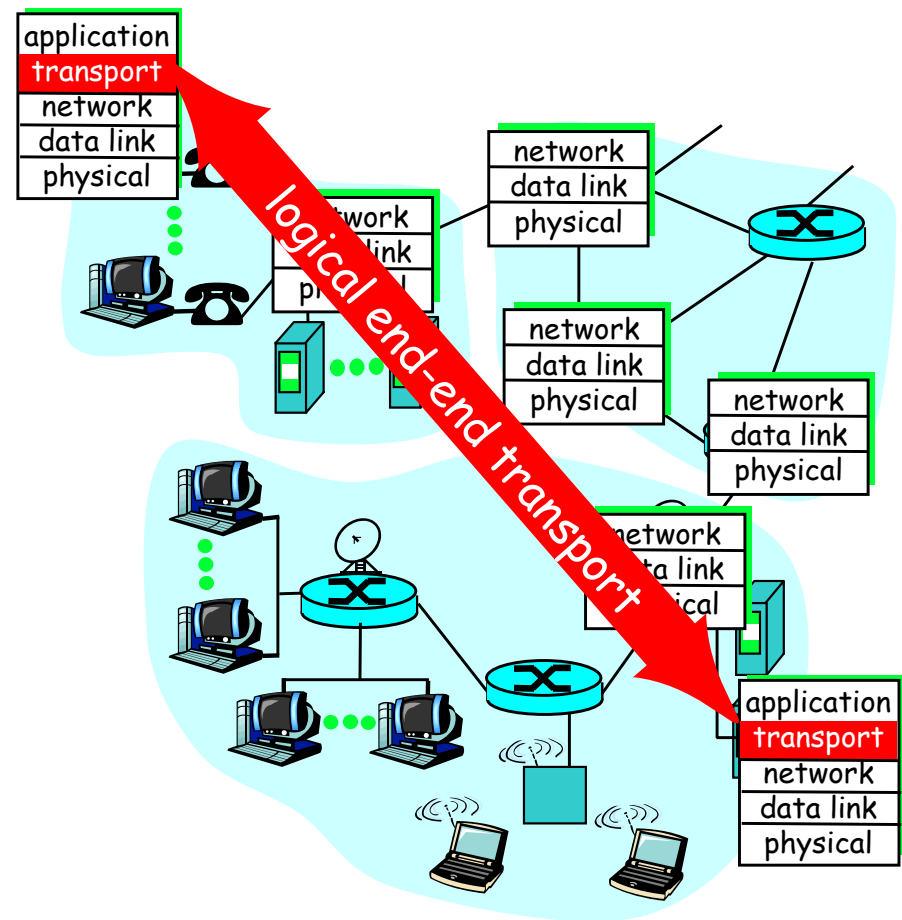


Transport services and protocols

- provide *logical communication* between app' processes running on different hosts
- transport protocols run in end systems (primarily)

transport vs network layer services:

- *network layer*: data transfer between end systems
- *transport layer*: data transfer between processes
 - relies on, enhances, network layer services



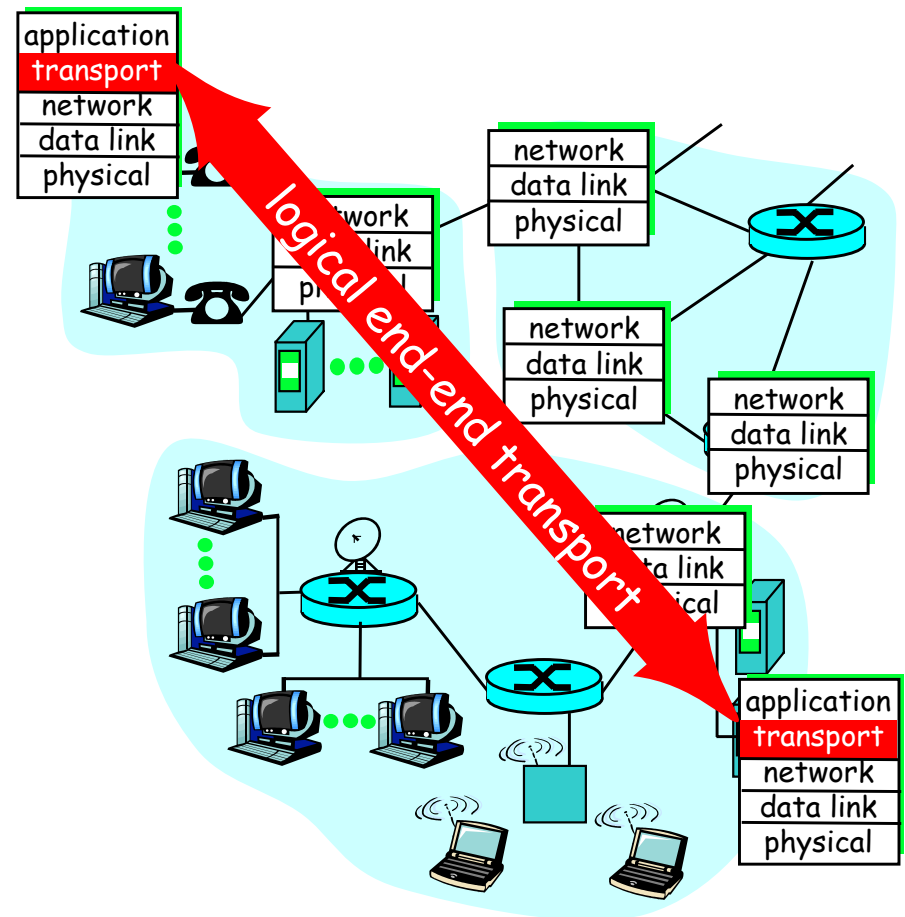
TCP e UDP

- Due protocolli di trasporto alternativi
- Realizzano funzionalità comuni a tutti gli applicativi
- Possono operare simultaneamente con molti applicativi diversi, tramite il concetto di porta

Transport-layer protocols

Internet transport services:

- reliable, in-order unicast delivery (TCP)
 - congestion
 - flow control
 - connection setup
- unreliable ("best-effort"), unordered unicast or multicast delivery: UDP
- services not available:
 - real-time
 - bandwidth guarantees
 - reliable multicast

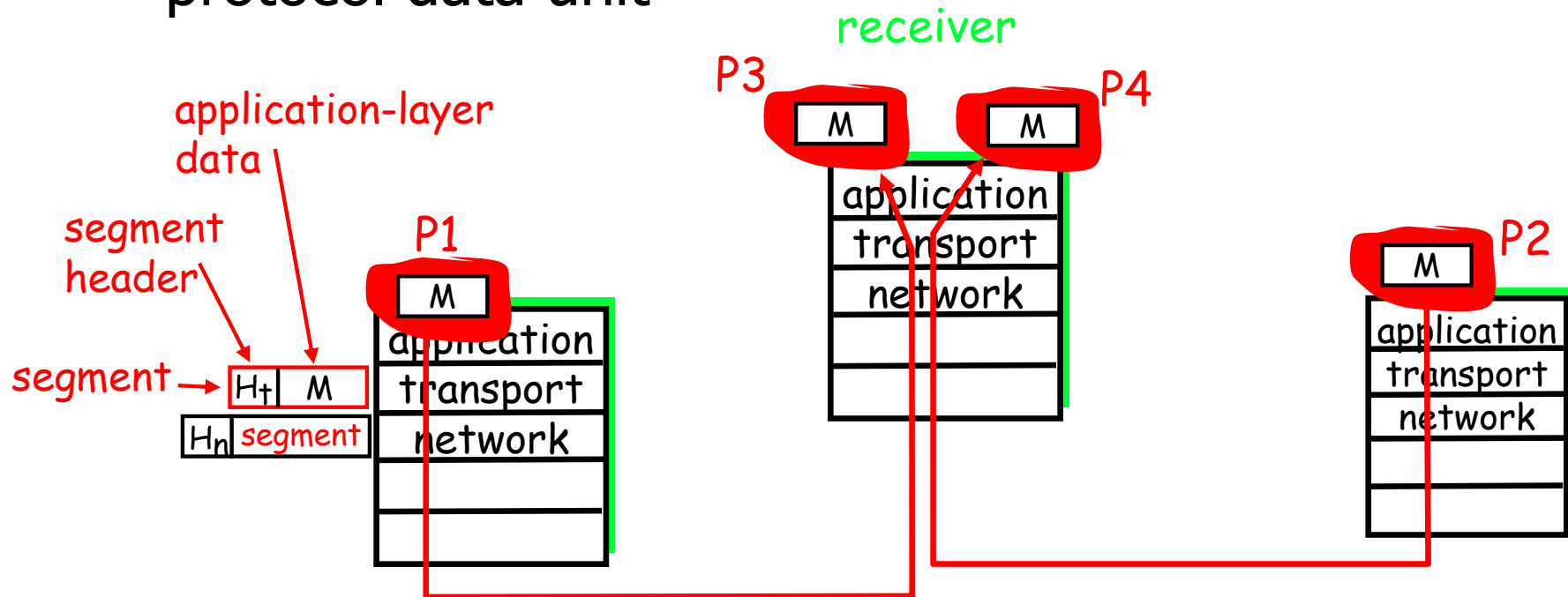


Multiplexing/demultiplexing

Recall: *segment* - unit of data exchanged between transport layer entities

- aka TPDU: transport protocol data unit

Demultiplexing: delivering received segments (TPDUs) to correct app layer processes



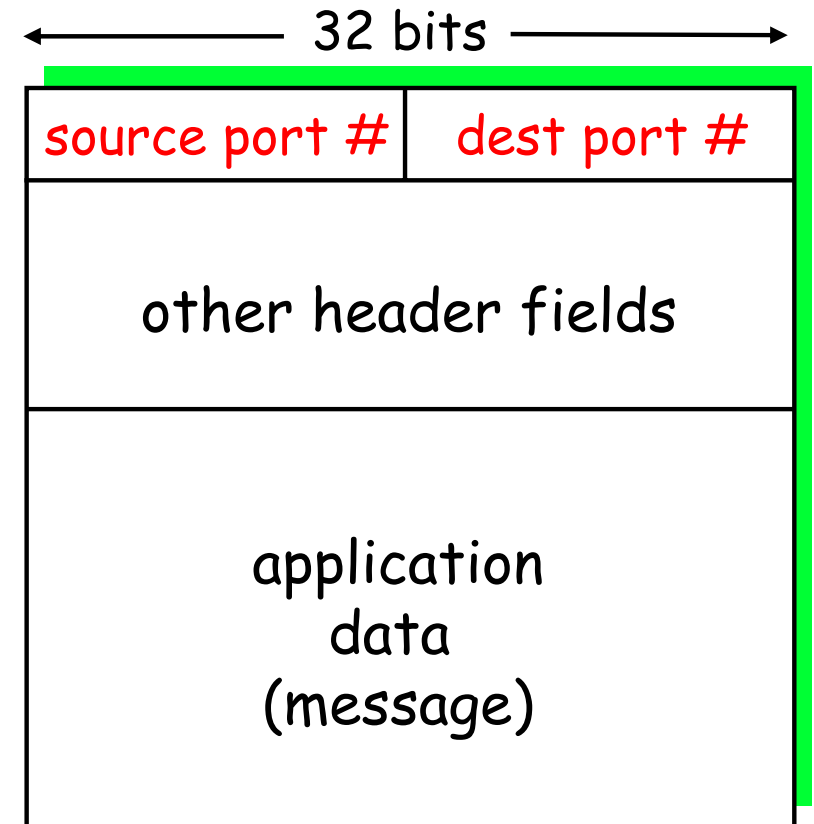
Multiplexing/demultiplexing

Multiplexing:

gathering data from multiple app processes, enveloping data with header (later used for demultiplexing)

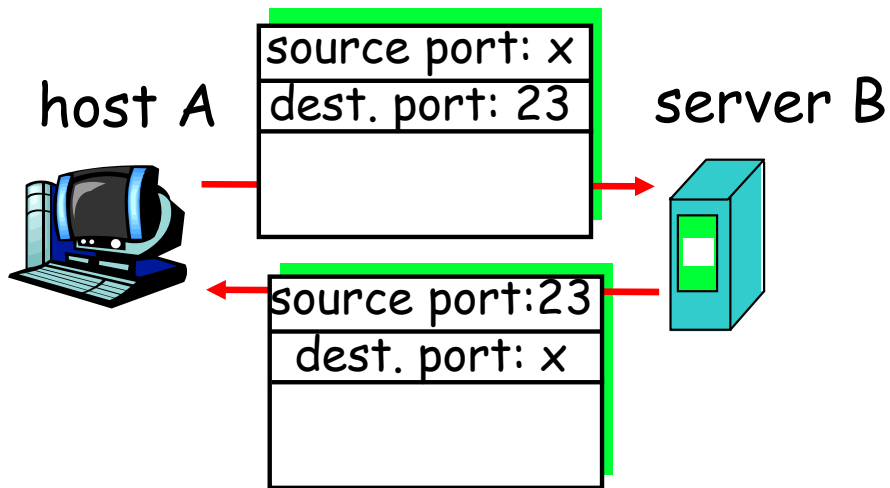
multiplexing/demultiplexing:

- based on sender, receiver port numbers, IP addresses
 - source, dest port #s in each segment
 - recall: well-known port numbers for specific applications



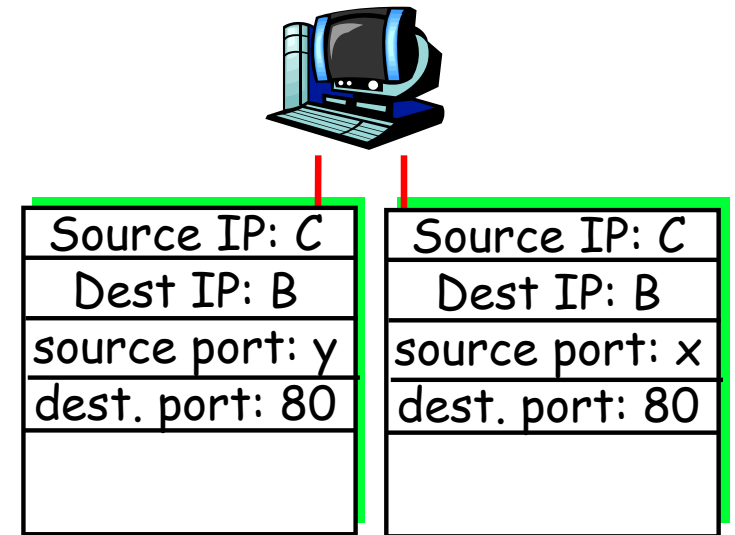
TCP/UDP segment format

Multiplexing/demultiplexing: examples

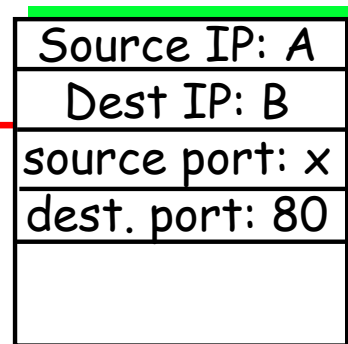


port use: simple telnet app

WWW client
host C



WWW client
host A



WWW
server B

port use: WWW server

Porte TCP e UDP

- Sono il mezzo con cui un programma client indirizza un programma server
- Numero naturale su 16 bit
- 0 ... 1023 = porte privilegiate
- 1024 ... 65535 = porte utente
- Porte statiche
 - quelle dove un server è in ascolto
- Porte dinamiche
 - quelle usate per completare una richiesta di connessione e svolgere un lavoro

Well Known Port

- Sono associate agli applicativi principali, ad esempio:

| servizio | port | TCP | UDP |
|-----------------|-------------|------------|------------|
| daytime | 13 | X | X |
| ftp | 21 | X | |
| telnet | 23 | X | |
| smtp | 25 | X | |
| tftp | 69 | | X |
| gopher | 70 | X | |
| finger | 79 | X | |
| http | 80 | X | |
| pop | 109 | X | |
| nntp | 119 | X | |

- Nonostante le porte UDP e TCP siano indipendenti si è scelto di non avere well known port sovrapposte
 - In questo modo è più semplice gestire servizi che possono essere chiamati sia via UDP che TCP

UDP: User Datagram Protocol [RFC 768]

- Connectionless Transport Protocol
 - No handshaking between UDP sender, receiver
 - Each UDP segment handled independently of others
- “Best effort” service - UDP segments may be:
 - Lost
 - Delivered out of order to app

Same service as IP's?

Not exactly; it adds

- Multiplexing of various applications
 - Ports
- Checksum to verify data integrity
 - Optional

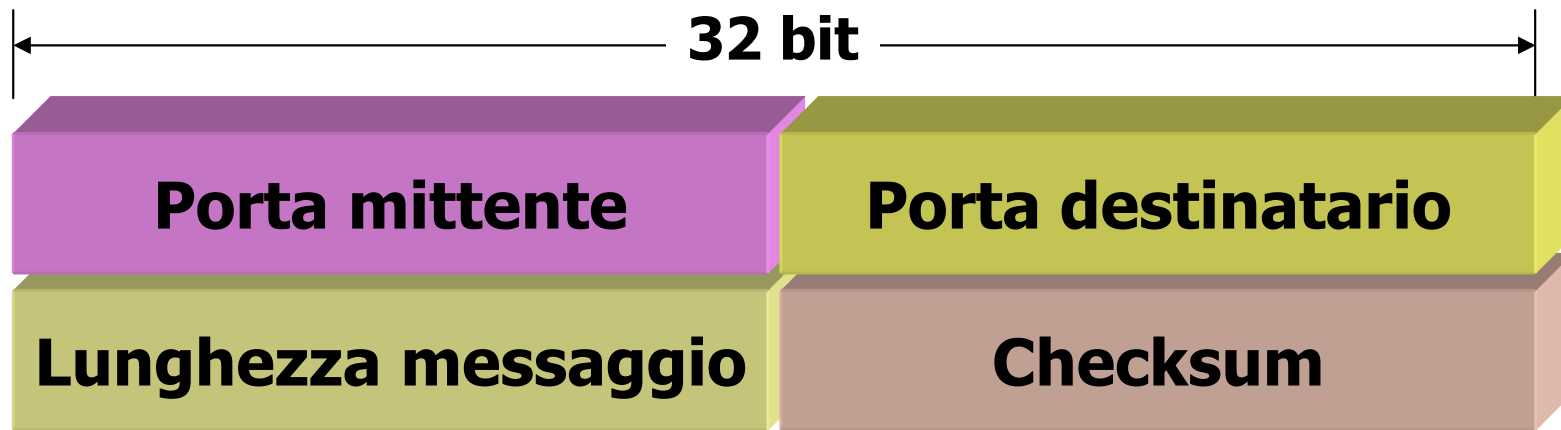
UDP: User Datagram Protocol

- Non prevede controllo di flusso
 - Non è in grado di adattarsi autonomamente a variazioni di carico della rete
 - Non prevede meccanismi di ritrasmissione in caso di errori
 - Potrebbe congestionare la rete Internet

Why UDP?

- No connection establishment
 - Which can add delay
- Simple
 - No connection state at sender, receiver
- Small segment header
 - Low transmission overhead
- No congestion control
 - UDP can blast away as fast as desired

UDP: Formato dell'intestazione



- Checksum e UDP Source sono opzionali: possono essere posti a 0
 - CRC= 0: non interessa il controllo errori sull'header
 - comunicazione monodirezionale

UDP: applicabilità

- Si usa su rete affidabile oppure quando l'affidabilità non è importante
 - NFS (Network File System)
- L'applicazione mette tutti i dati in un singolo pacchetto
 - DNS
 - rwho
 - ruptime
- Non è importante che tutti i pacchetti arrivino a destinazione
 - Applicazioni multimediali
- Eventuali meccanismi di ritrasmissione (se necessari) vengano gestiti direttamente dall'applicazione
 - SNMP (Simple Network Management Protocol)

UDP: applicazioni multimediali

■ Caratteristiche

- I dati devono arrivare entro un tempo limite
 - un pacchetto in ritardo equivale ad un pacchetto perso
- l'applicazione non necessita di ritrasmissione
 - il ritardo introdotto sarebbe inaccettabile
- è necessaria una "banda minima" per la comunicazione

■ UDP è la scelta obbligata

- non ha controllo di flusso

■ RTP (Real Time Transfer Protocol)

- annullamento delle variazioni di ritardo

■ H.323: standard per videoconferenza su IP

- sincronizzazione tra i media
- compatibilità con altre reti (ad esempio ISDN)

UDP checksum

Goal: detect “errors” (e.g., flipped bits) in transmitted segment

Sender:

- treat segment contents as sequence of 16-bit integers
- checksum: addition (1's complement sum) of segment contents
- sender puts checksum value into UDP checksum field

Receiver:

- compute checksum of received segment
- check if computed checksum equals checksum field value:
 - NO - error detected
 - YES - no error detected. *But maybe errors nonetheless?*
More later

TCP: Transmission Control Protocol

- Un protocollo di trasporto:
 - byte-oriented
 - connesso
- Utilizzato da applicativi che richiedono la trasmissione affidabile dell'informazione:
 - telnet
 - ftp (file transfer protocol)
 - smtp (simple mail transfer protocol)
 - http

TCP: funzionalità

- Funzionalità TCP:
 - Supporto della connessione
 - Error Checking
 - Controllo di flusso
 - Multiplazione e demultiplazione
 - Controllo di stato e di sincronizzazione
- TCP garantisce la consegna del pacchetto, UDP no!

TCP: caratteristiche

- Come UDP ha il concetto di porta
- La comunicazione tra due entità TCP fa uso di un circuito virtuale
- Al circuito virtuale è associato un protocollo di trasporto
 - full-duplex
 - acknowledge
 - controllo di flusso
- TCP richiede più banda e più CPU di UDP
 - elaborazione più pesante
 - più informazioni di stato → maggiore memoria
 - più capacità trasmissiva
 - ritrasmissione
 - intestazione più grande

TCP: caratteristiche

- TCP segmenta e riassembla i dati secondo le sue necessità:
 - non garantisce nessuna relazione tra il numero di read e quello di write
- Il TCP remoto deve fornire un acknowledge dei dati, normalmente tramite piggybacking
- Protocollo con sliding window, timeout e ritrasmissione

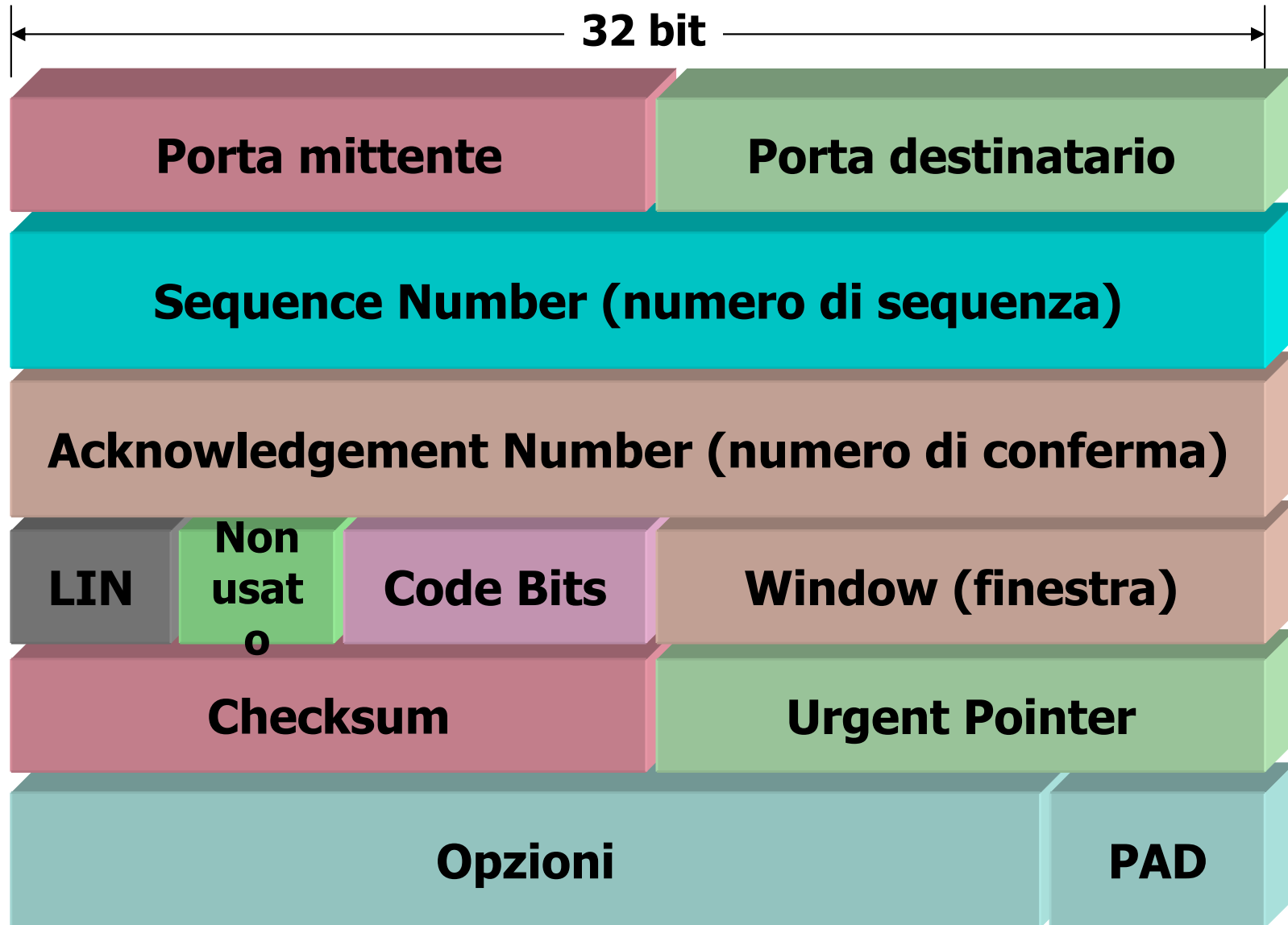
TCP: Sliding Window

- I protocolli a sliding window richiedono di fissare la dimensione della finestra
- In TCP la dimensione della finestra è in byte, non in segmenti
- Il campo "window" del pacchetto TCP indica quanti byte possono ancora essere trasmessi prima di un ACK

TCP: Timeout e Ritrasmissioni

- Per ogni pacchetto trasmesso viene inescato un timer
- Se un timer scatta senza aver ricevuto l'ACK relativo a quel pacchetto questo è ritrasmesso
- Ad ogni ritrasmissione di un pacchetto, sono ritrasmessi anche tutti gli altri pacchetti inclusi nella finestra
 - Può dare origine a congestioni
- Il timer è autoadattante
 - connessioni con basso Round Trip Delay hanno timer bassi
 - reagisce alle condizioni del traffico

TCP: Formato dell'intestazione



TCP: Campi dell'intestazione (1)

- Source Port, Destination Port
- Sequence Number
 - Indica la posizione (in byte) di quel pacchetto all'interno del flusso
 - Il valore iniziale è scelto in maniera casuale all'inizio di ogni connessione ed indipendentemente da ogni macchina
- Acknowledge Number
 - Campo dedicato al piggyback degli acknowledgement
 - numero d'ordine del prossimo byte che il mittente del messaggio si aspetta di ricevere
- Header Length
 - Lunghezza dell'intestazione in multipli di 32 bit

TCP: Campi dell'intestazione (2)

- Reserved
 - Non usato
- Control
 - URG: Urgent Pointer is valid
 - ACK: Acknowledgement field is valid
 - PSH: This segment request a push
 - RST: Reset the connection
 - SYN: Synchronize sequence number
 - FIN: Sender has reached end of its byte stream

TCP: Campi dell'intestazione (3)

■ Window

- Indica quanti byte il mittente del messaggio è in grado di ricevere a partire dall'ultimo byte confermato
- L'host è in grado di ricevere quindi fino al numero di sequenza $ACK + WINDOW$

■ Urgent Pointer

- Indica che nel pacchetto ci sono uno o più byte urgenti
- Lo stack TCP deve far in modo che l'applicazione interessata vada in "urgent mode". Solo dopo che i dati urgenti sono stati tutti consumati, l'applicazione può tornare in "normal mode"
- Tipicamente associati ad eventi asincroni:
 - interrupt, abort, ...

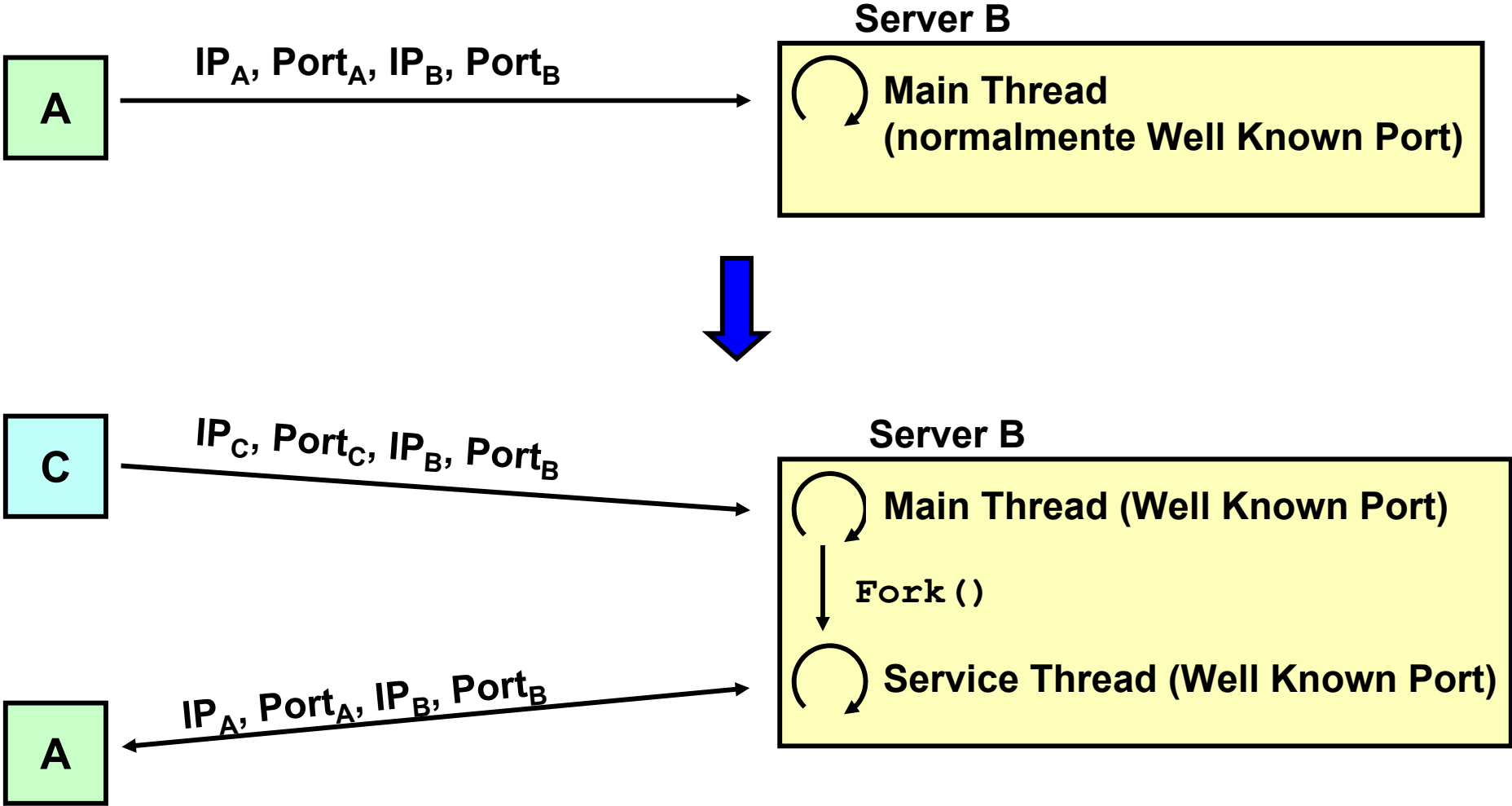
■ Options

- Maximum Segment Size (normalmente pari alla MTU)

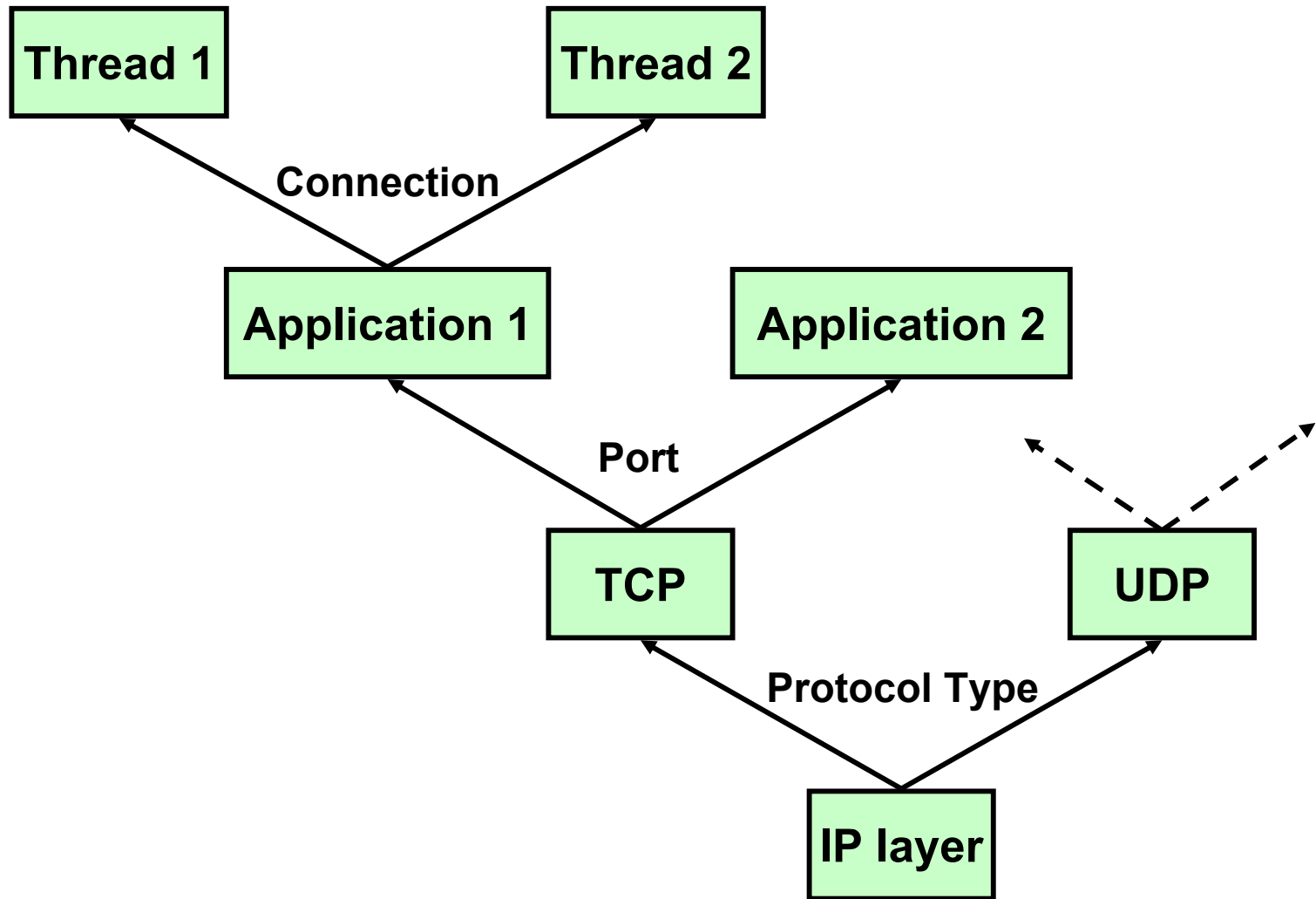
Connessioni

- TCP identifica un “canale” di comunicazione con il nome di connessione
- Sono identificate dalla quadrupla:
 - <IP client><IP server><Port client><Port Server>
 - | 32 bit | 32 bit | 16 bit | 16 bit |
- Questa soluzione permette
 - A client diversi di accedere allo stesso servizio sullo stesso server
 - Allo stesso client di attivare più sessioni dello stesso servizio
 - Viola il modello a livelli
 - informazioni di livello 3 usate a livello 4

Connessioni multiple

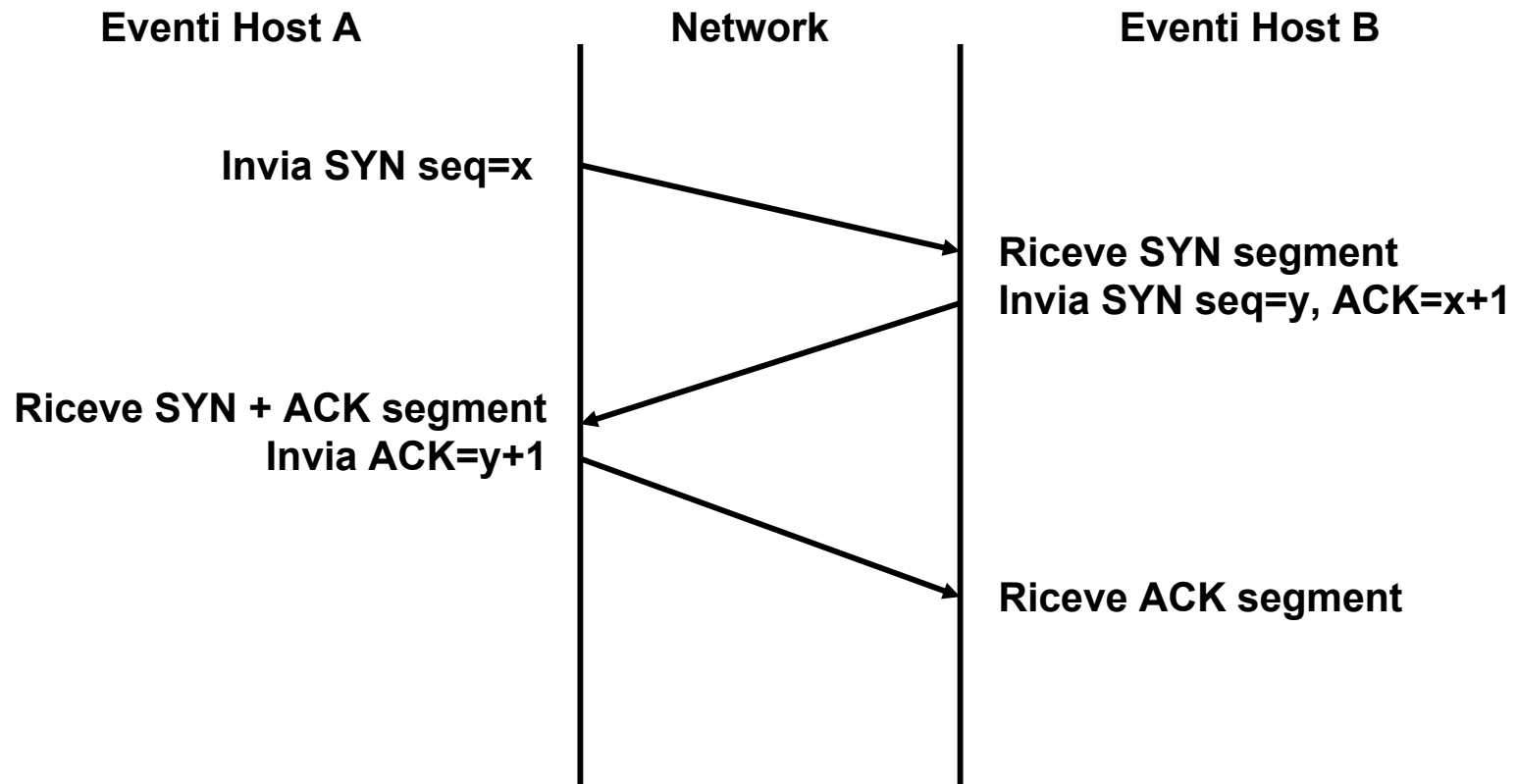


Demultiplexing



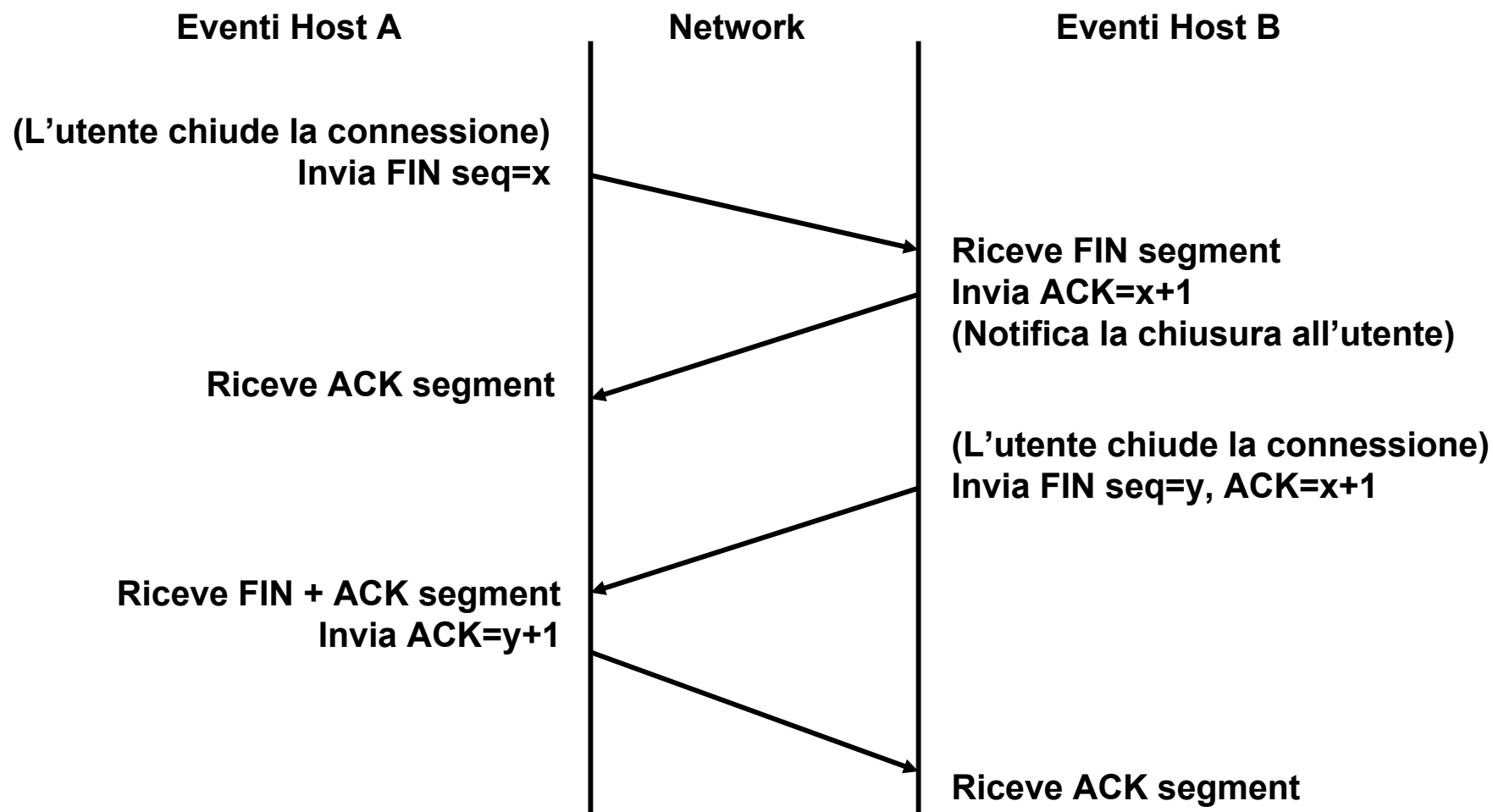
Apertura della connessione

- Three way handshake



Chiusura della connessione

■ Three way handshake modificato



Gestione della connessione: End-to-end flow control

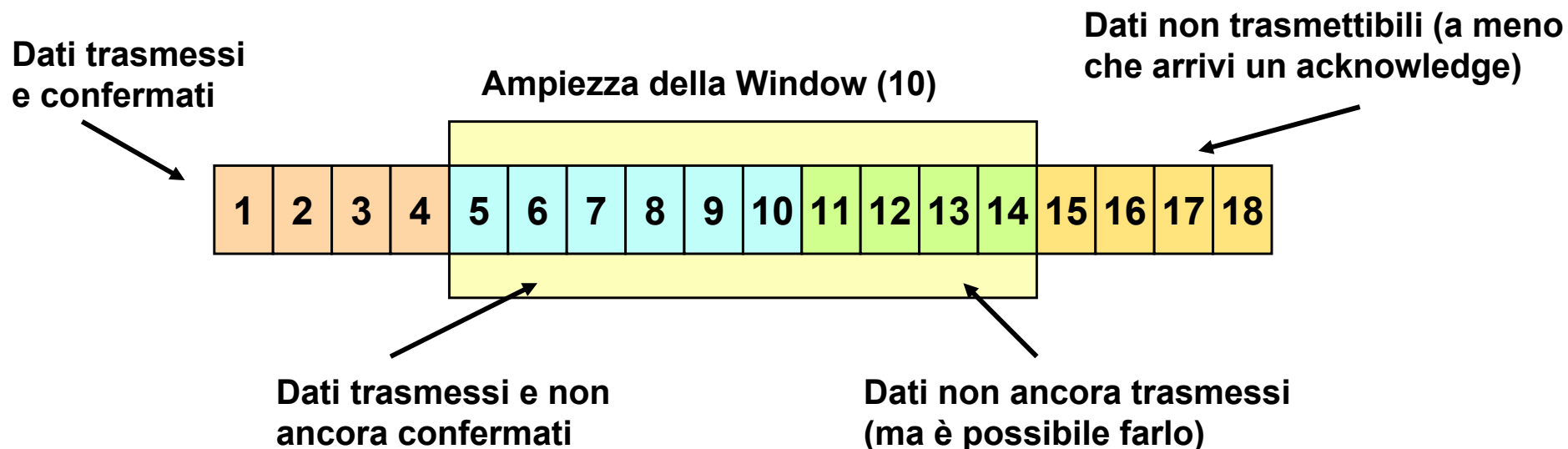
- Controlla che il flusso di dati end to end sia accettabile
 - evita che il mittente trasmetta troppo velocemente se il destinatario non è in grado di riceverli
 - insufficiente potenza elaborativa o memoria
 - controlla che i pacchetti arrivino correttamente
 - si occupa della ritrasmissione dei pacchetti persi o erronei
 - elimina i pacchetti duplicati
- È basato su due meccanismi
 - Sliding window
 - regola la quantità di dati trasmessi la cui ricezione non è stata ancora confermata
 - Round Trip Estimate
 - stima il tempo necessario ad un pacchetto per raggiungere la destinazione e alla conferma per arrivare al mittente (round trip delay)
 - permette di determinare quando procedere alla ritrasmissione

Gestione della connessione: Congestion Control

- La congestione è rilevata nel momento in cui un retransmission timer espira
- La versione base di TCP ritrasmette l'intera finestra di quando un timer espira
- Questo può causare gravi congestioni della rete (congestion collapse):
 - Nell'ottobre 1986 Arpanet fu bloccata da una congestione (da 32 kbs a 40 bps)
- Soluzioni
 - Multiplicative decrease window size
 - Slow start

Sliding Window

- Controlla la quantità di dati che è possibile trasmettere prima di ricevere la conferma di ricezione
 - una finestra di N byte (N-window) consente la trasmissione di N byte prima che giunga l'acknowledge del primo trasmesso
- Vincola l'efficienza (throughput) sui lunghi percorsi
 - se la finestra non è più larga della memoria di canale, il trasmettitore si ferma



Sliding Window

- In TCP la Window ha dimensione variabile
 - Deve essere stabilito un valore iniziale
 - può essere diverso per trasmettitore e ricevitore
 - viene negoziato all'apertura della connessione
 - All'apertura della connessione il ricevitore comunica al trasmettitore la dimensione della window ammessa
 - Durante lo scambio dati il ricevitore può diminuire la dimensione della window se il buffer di ricezione del TCP si sta riempiendo
 - il livello superiore non sta prelevando i dati dal livello TCP
 - si usa il campo window dei pacchetti trasmessi nella direzione opposta

Sliding Window

- In TCP la dimensione della finestra è in byte, non in segmenti
 - Il campo window del pacchetto TCP indica quanti byte il mittente è in grado di ricevere a partire dall'ultimo byte confermato
- L'acknowledge è dato in forma "cumulativa"
 - L'arrivo di un ack relativo al byte X indica implicitamente che tutti i byte precedenti sono stati ricevuti
 - È un vantaggio se la connessione ha basso tasso di perdita
 - si diminuisce il traffico di controllo
 - Può essere svantaggioso nel caso di elevato tasso di perdite non consecutive
 - può essere ritrasmessa una sequenza di byte anche se questa era precedentemente arrivata correttamente

Retransmission Timer

- Tempo massimo per cui si aspetta la conferma di avvenuta ricezione (acknowledgement)
- Alla trasmissione di ogni segmento viene fatto partire un timer
 - se il timer espira prima di aver ricevuto la conferma di ricezione di quel segmento, avviene la ritrasmissione
- Un segmento è considerato confermato anche se lo è un segmento che lo segue nel flusso dati

Retransmission Timer

- Il timer è autoadattante
 - la durata dipende dalle condizioni del traffico
 - connessioni con basso Round Trip Delay hanno timer brevi
 - si adatta ad eventuali variazioni del round trip time medio evitando ritrasmissioni
 - ad ogni ack ricevuto il valore del timer viene ricalcolato per tenere conto delle variazioni di round trip time
- Tutta la Window viene ritrasmessa
 - può dare origine a congestioni
 - intervengono i meccanismi di Congestion Control

Congestion Control:

Multiplicative decrease window size

- Nel momento in cui si verifica una congestione non è opportuno ritrasmettere l'intera window
 - questo può portare ad un peggioramento della congestione
- Si adotta un meccanismo di ritrasmissione di tipo esponenziale decrescente
 - la dimensione della window viene dimezzata
 - se il retransmission timeout espira nuovamente, la diminuzione della window viene iterata
 - si procede fino a quando la ritrasmissione permessa è di un solo segmento

Congestion Control:

Multiplicative decrease window size

- Si definiscono 2 nuove window
 - receiver_window: numero di segmenti di dimensione massima che il ricevitore è in grado di ricevere
 - congestion_window: esprime il numero di segmenti inviabili nel caso di congestione
- In ogni momento si ha:
 - $\text{Allowed_window} = \min(\text{receiver_window}, \text{congestion_window})$
- Se non ci sono congestioni la congestion window corrisponde alla receiver window
- In caso di congestione
 - congestion_window viene dimezzata (fino alla dimensione minima di 1 messaggio)
 - il retransmission_timer dei messaggi che sono inclusi nella finestra viene moltiplicato per un fattore N

Congestion Control:

Multiplicative decrease window size

■ Backoff timer

- il retransmission timer dei segmenti che si trovano nella window viene aumentato in modo esponenziale (raddoppiato) ogni volta che si ha un timeout
 - anche quando la window ha la dimensione di 1 segmento
- si evitano troppe ritrasmissioni nel caso in cui le conferme siano semplicemente in ritardo

Congestion Control: Slow Start

- Garantisce una crescita graduale del bit rate
- Entra in azione sia dopo una congestione sia all'apertura di una connessione
 - Nel caso si provenga da una congestione non è opportuno riportare il bit rate a quello precedente la congestione in quanto può dare origine a oscillazioni
- Congestion Window
 - Viene inizializzata ad 1 segmento
 - Viene incrementata all'arrivo di ogni conferma (crescita lineare)
 - Inizio: congestion = 1 → trasmette 1 segmento
 - Riceve 1 ack → congestion = 2 → può trasmettere fino a 2 segmenti
 - Riceve 2 ack → congestion = 4 → ... fino a 4 segmenti
 - ...

Congestion Control: Slow Start

- Congestion avoidance
 - Raggiunta una dimensione pari alla metà della window in uso quando si è verificata la congestione, si entra in una fase di congestion avoidance
 - in questa fase la congestion window viene incrementata linearmente
 - 1 segmento per ogni finestra ricevuta correttamente
- Ad un certo punto inizia a prevalere la receiver_window
- Questo algoritmo è stato proposto da Nagle (1987)