

Livello di Trasporto

Introduzione

Livello di trasporto in Internet

UDP - User Datagram Protocol

TCP - Transmission Control Protocol

Meccanismo di ritrasmissione

Controllo del flusso

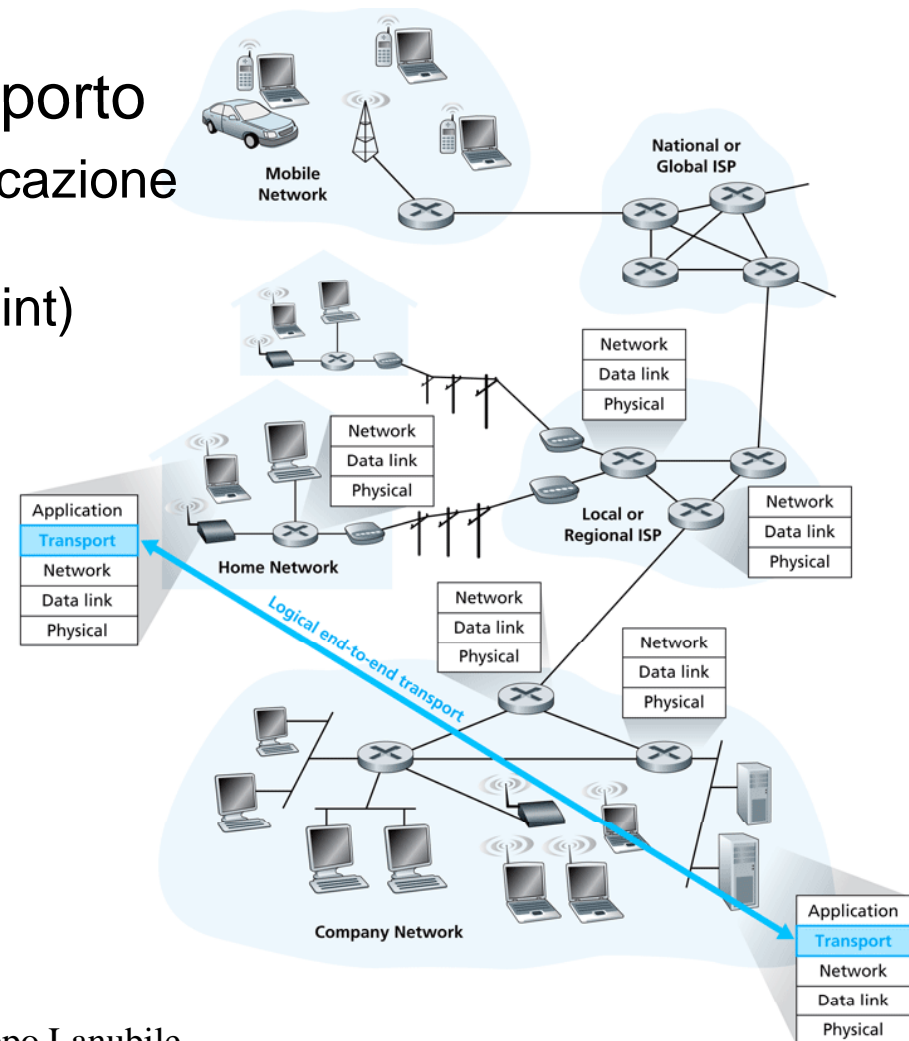
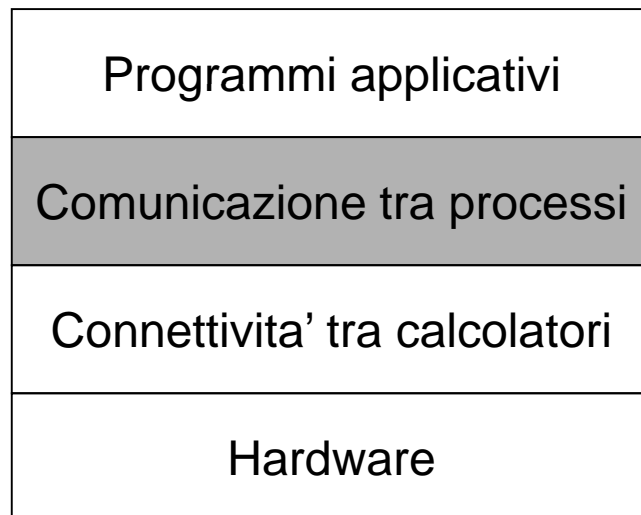
Controllo di congestione

Apertura connessione

Chiusura connessione

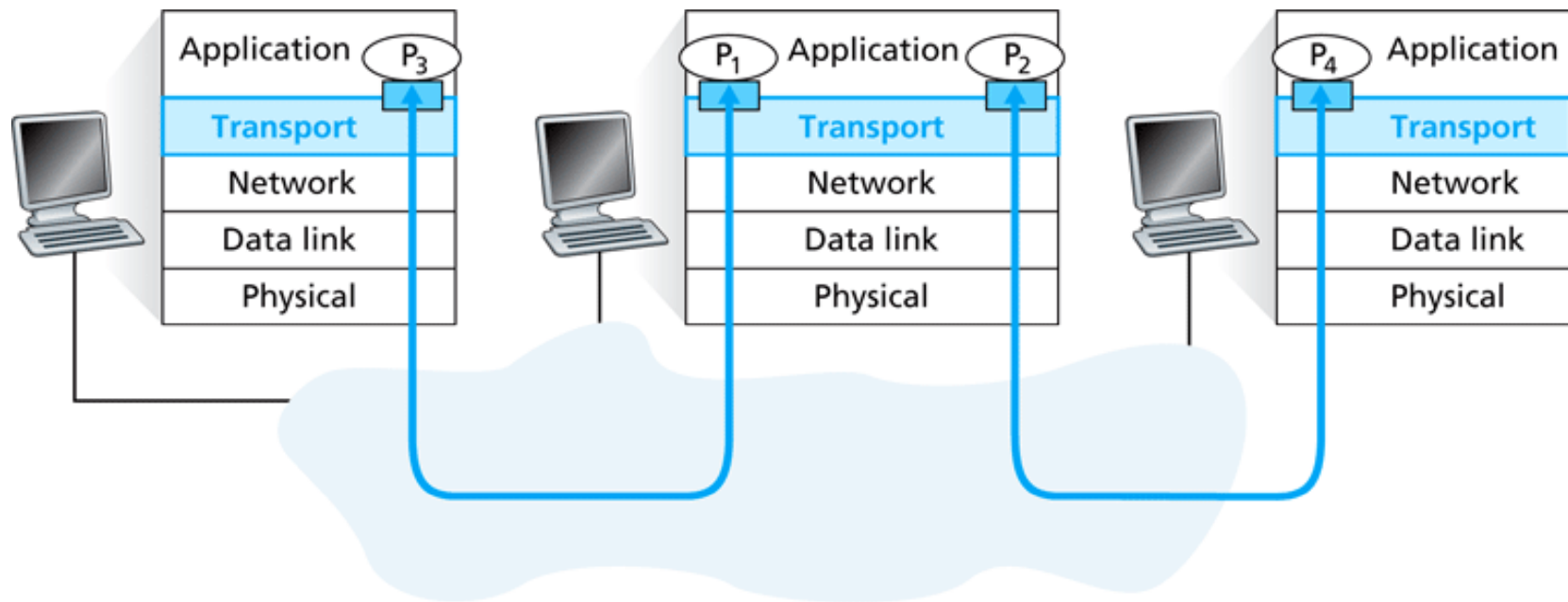
Introduzione

- Obiettivo del livello di trasporto
 - Offrire un canale di comunicazione tra una coppia di processi (comunicazione point-to-point)



Prof. Filippo Lanubile

Figure 3.1 ♦ The transport layer provides logical rather than physical communication between application processes.



Key:



Figure 3.2 ♦ Transport-layer multiplexing and demultiplexing

Livello di trasporto in Internet

- Il protocollo di rete IP fornisce un servizio senza connessione (a datagramma) non affidabile tra gli host

Protocolli di trasporto

- User Datagram Protocol (UDP)
 - servizio senza connessione (datagramma),
 - non affidabile (scoperta errori opzionale e nessun recupero),
 - moltiplicazione/demoltiplicazione: indirizzamento dei processi
- Transmission Control Protocol (TCP)
 - servizio orientato alla connessione (flusso di bytes)
 - decomposizione delle sequenze di byte in messaggi
 - affidabile (scoperta e recupero degli errori)
 - controllo del flusso
 - moltiplicazione/demoltiplicazione: indirizzamento dei processi

Application	Application-Layer Protocol	Underlying Transport Protocol
Electronic mail	SMTP	TCP
Remote terminal access	Telnet	TCP
Web	HTTP	TCP
File transfer	FTP	TCP
Remote file server	NFS	Typically UDP
Streaming multimedia	typically proprietary	UDP or TCP
Internet telephony	typically proprietary	UDP or TCP
Network management	SNMP	Typically UDP
Routing protocol	RIP	Typically UDP
Name translation	DNS	Typically UDP

Figure 3.6 ♦ Popular Internet applications and their underlying transport protocols

Indirizzamento dei processi

- Ogni processo comunicante è associato a un numero di porta a 16-bit, locale e unico rispetto all'host
 - Necessita' di individuare un processo con un indirizzo di rete globalmente unico (IP address e port number)
 - Tutti messaggi contengono due coppie di indirizzi:
 - Mittente: (IP address, port number)
 - Destinatario: (IP address, port number)
 - Il sistema operativo dell'host destinatario consegna al processo un messaggio in arrivo sulla base del port number
- Numeri di porta
- 1 - 255: riservati a servizi standard
 - 21: ftp
 - 23: telnet
 - 25: SMTP
 - 80:http daemon
 - 1 - 1023: disponibili solo a utenti privilegiati
 - 1024 - 4999: per processi automaticamente assegnati
 - 5000 - : solo per processi utente

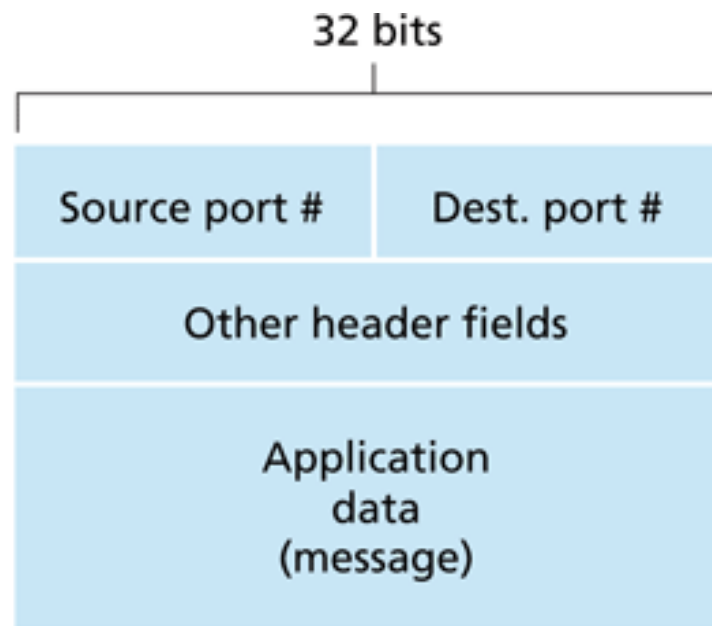


Figure 3.3 ♦ Source and destination port-number fields in a transport-layer segment

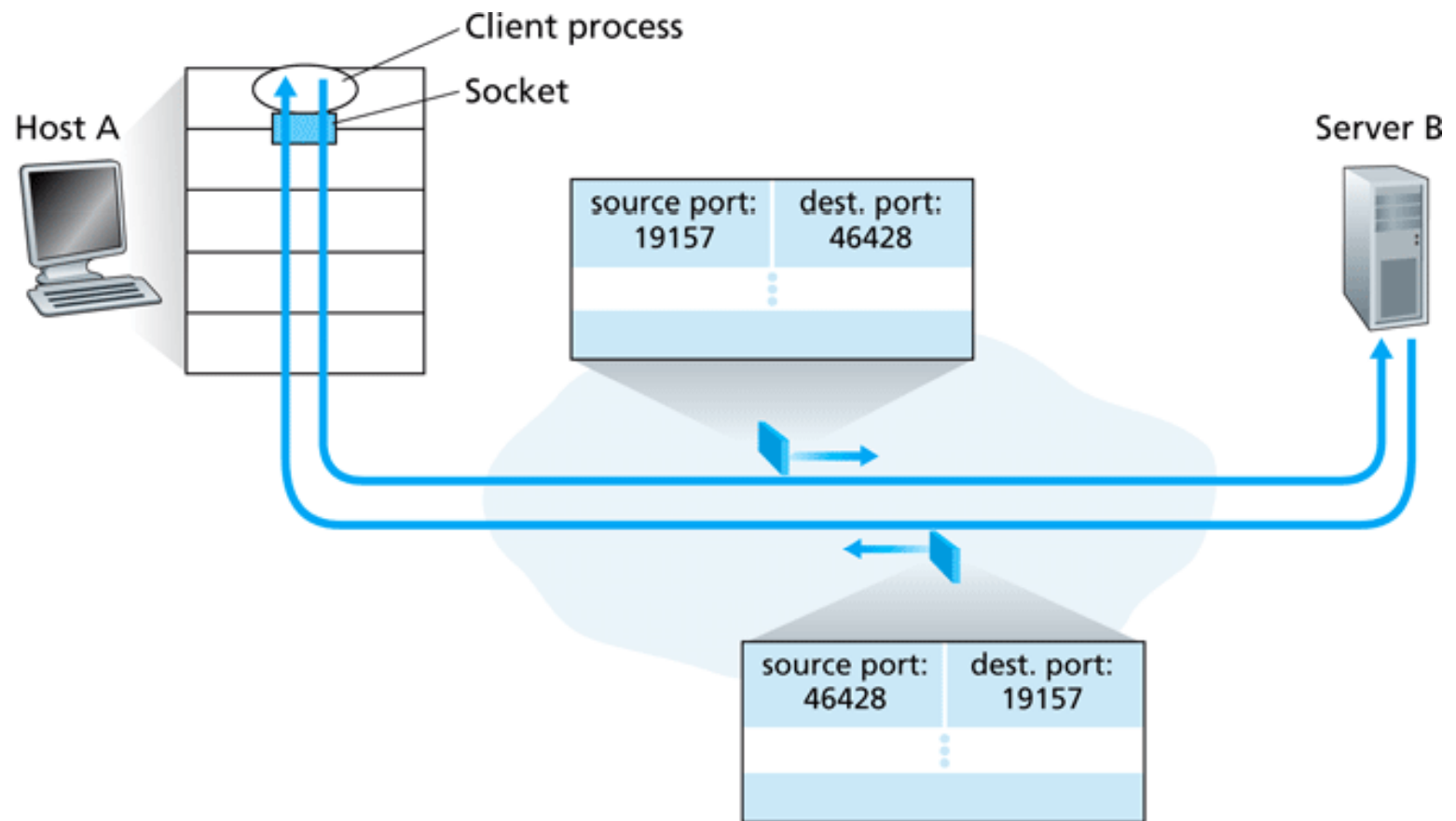


Figure 3.4 ♦ The inversion of source and destination port numbers

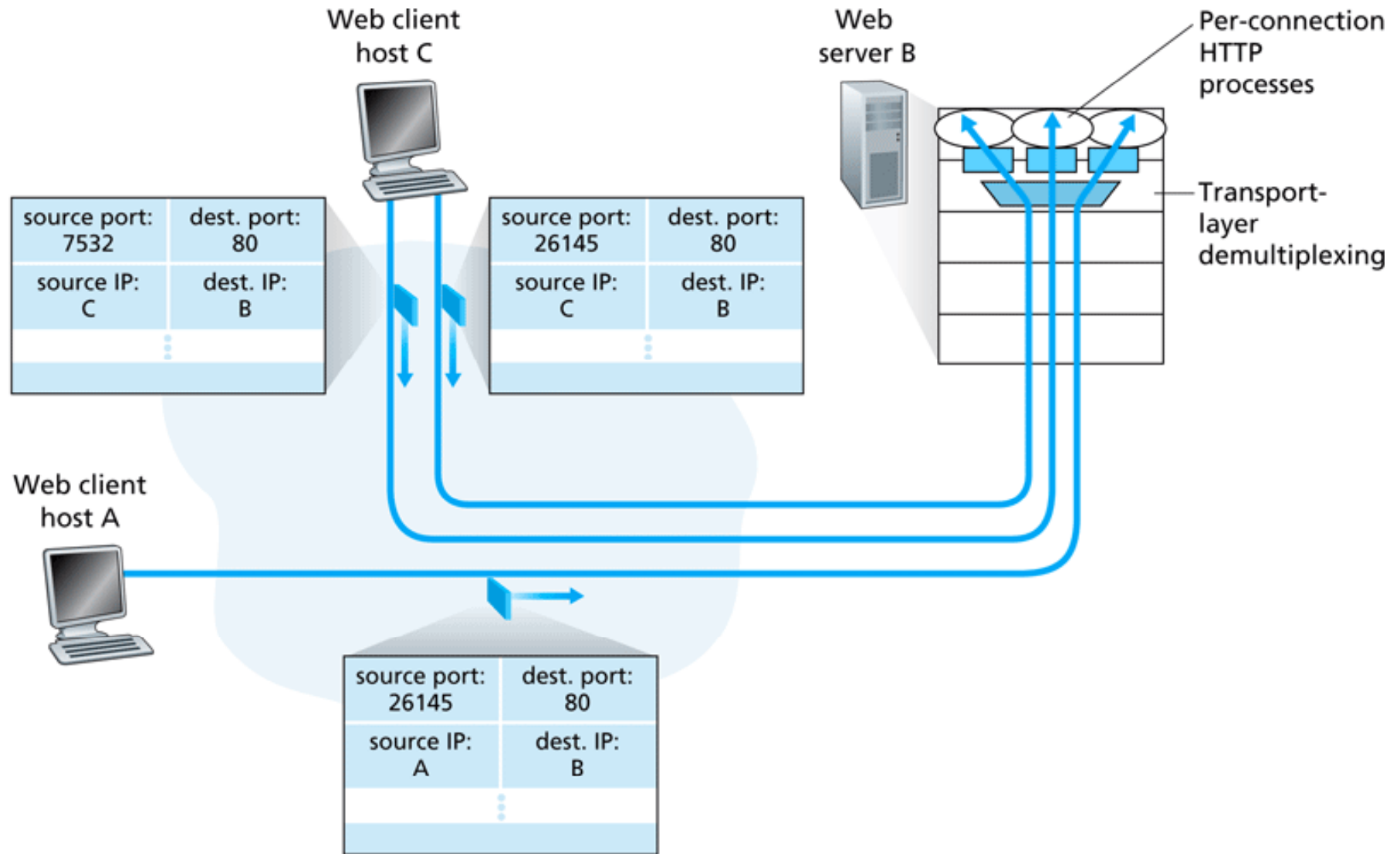


Figure 3.5 ♦ Two clients, using the same destination port number (80) to communicate with the same Web server application

Prof. Filippo Lanubile

UDP - User Datagram Protocol

- Fornisce un servizio punto a punto senza connessione
- Spedizione di datagram (includono l'indirizzo del mittente e del destinatario) con lunghezza max definita dal datagram IP (≤ 64 KB)
- Indirizzi dei processi basati sul numero di porta (16 bit)
- Controllo degli errori opzionale basato su checksum (se non usato e' settato a zero)
- Non fornisce conferma di ricezione dei messaggi
- Nessun controllo del flusso sul mittente
- Usato da SNMP (Simple Network Management Protocol), e da applicazioni basate su semplici servizi richiesta-risposta (RPC)

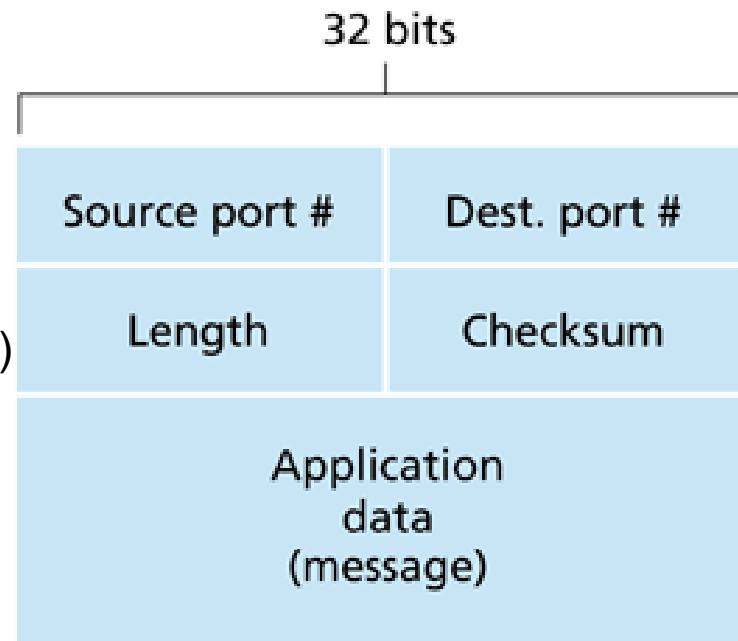


Figure 3.7 ♦ UDP segment structure

TCP - Transmission Control Protocol

Caratteristiche fondamentali

- Comunicazione punto a punto
 - due punti finali che non si interessano di cosa sta nel mezzo
- Comunicazione full duplex
 - scambio di dati in entrambi le direzioni simultaneamente
- Orientato alla connessione
 - un processo richiede una connessione a un altro processo e poi usa la connessione per il trasferimento dei dati
 - Creazione affidabile della connessione
 - i dati di una vecchia connessione non si confondono con le nuove connessioni
 - Distruzione dolce della connessione
 - i dati spediti prima di chiudere una connessione non sono perduti
- Trasferimento affidabile
 - garanzia che i dati arrivano senza errori, nel giusto ordine e senza duplicazioni
- Controllo di congestione
 - previene che un eccesso di dati venga immesso nella rete

TCP - Transmission Control Protocol

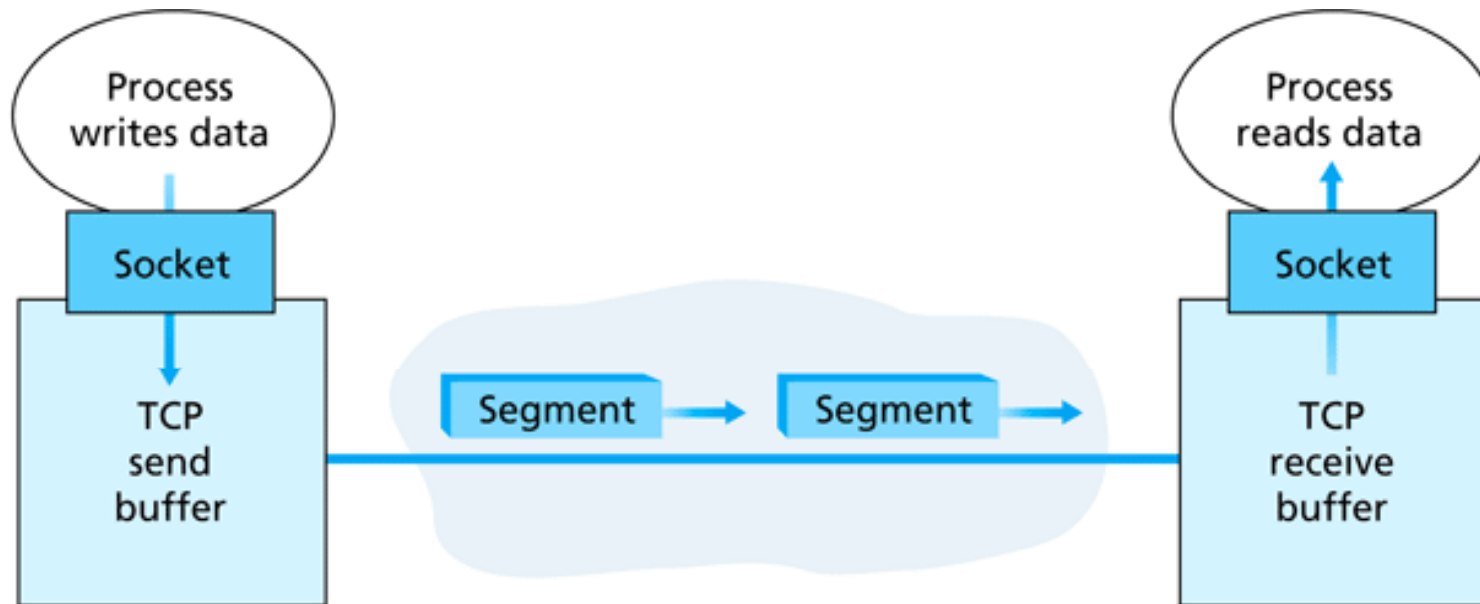
- Segmentazione del flusso di byte
 - il flusso di bytes e' suddiviso in segmenti di lunghezza accettabile per il livello sottostante (IP accetta massimo 64 KB) e poi riassembla i segmenti nella sequenza corretta



Figure 3.30 ♦ Dividing file data into TCP segments

TCP - Transmission Control Protocol

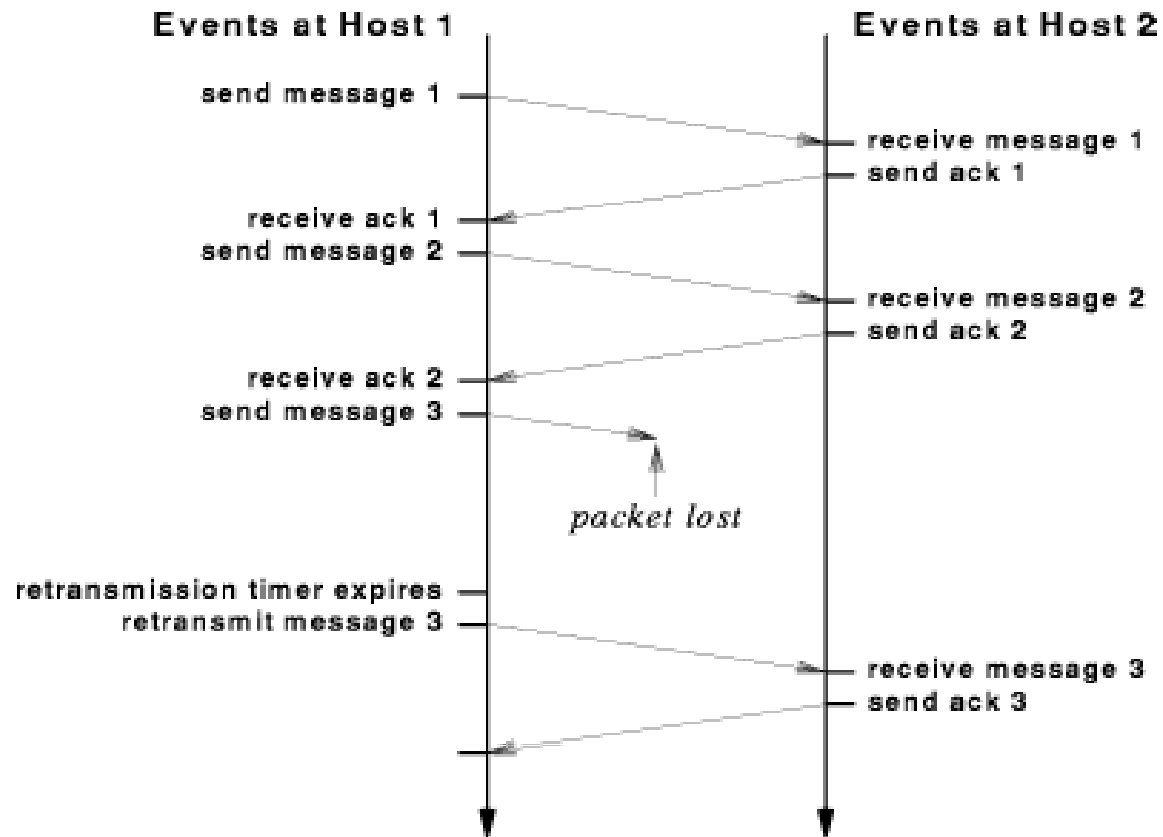
- Interfaccia a stream
 - sequenza continua di bytes (octets): nessuna garanzia che i dati arrivano negli stessi blocchi con cui sono stati spediti
- Controllo di flusso
 - garanzia che il mittente non eccede la capacità di ricezione del destinatario



Prof. Filippo Lanubile

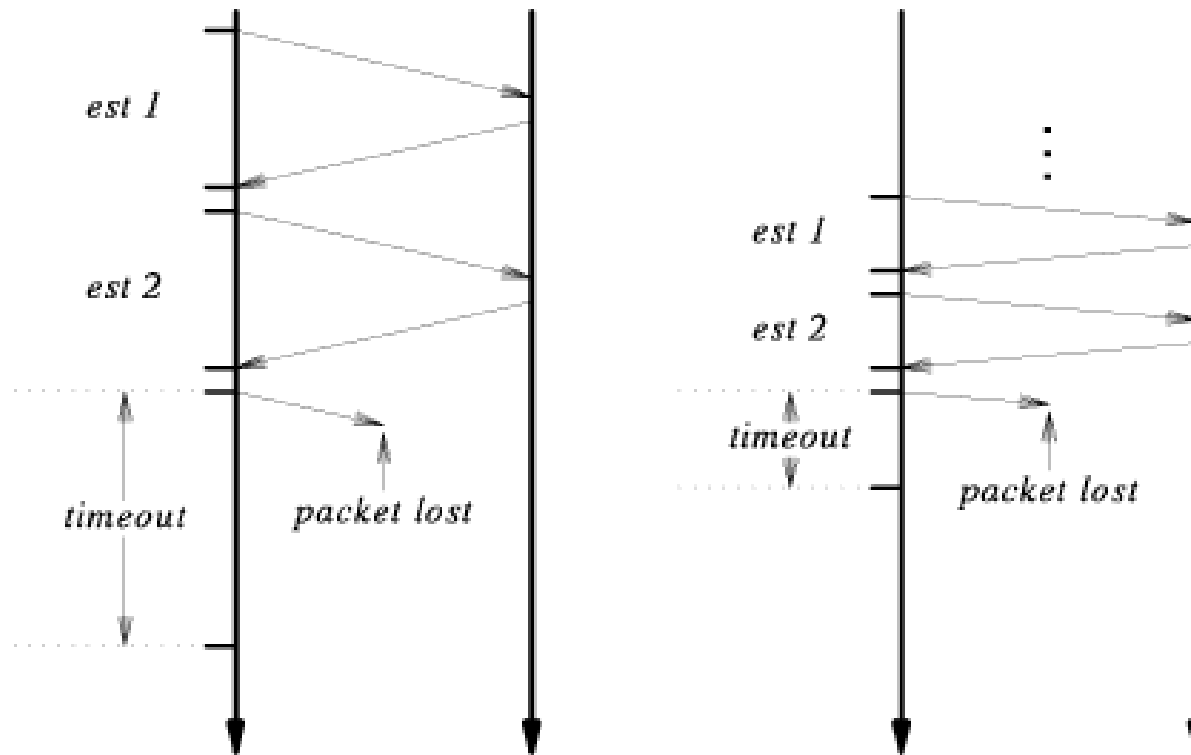
Figure 3.28 ♦ TCP send and receive buffers

Ritrasmissione di pacchetti



- acknowledgment dal receiver per verifica di pacchetti persi
- quanto bisogna aspettare?

Ritrasmissione adattativa

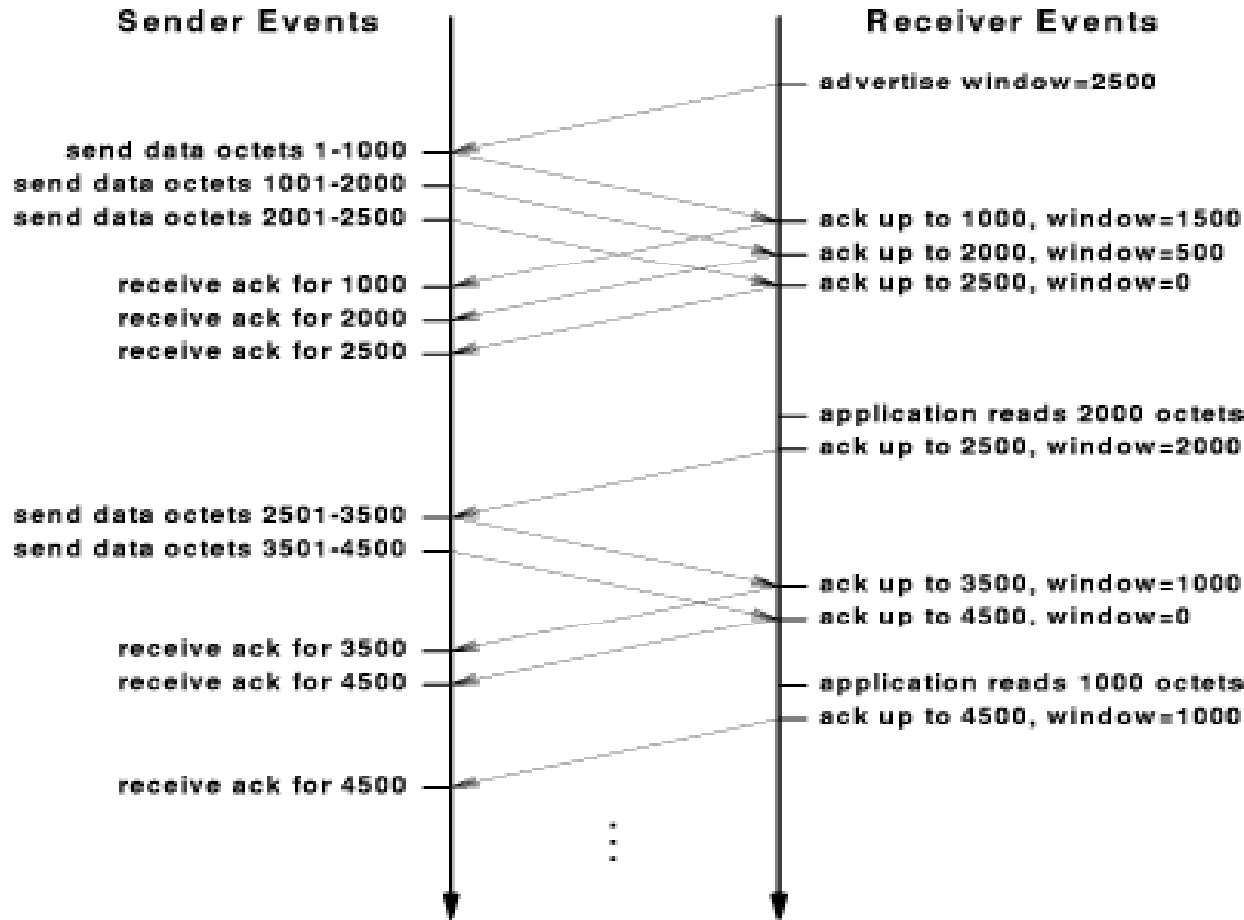


Gestione
dinamica dei
timer di
ritrasmissione

Strategia:
Aspettare
quanto basta
per
considerare
un pacchetto
perso ma non
piu' del
necessario

Soluzione:
algoritmi
basati sul
round trip time

Controllo del flusso mediante finestra scorrevole



Window advertisement:
Il receiver specifica insieme all'ack il numero di byte liberi nel buffer

Il sender usa la dimensione della finestra per determinare l'ammontare dei dati che possono essere spediti senza attendere l'ack

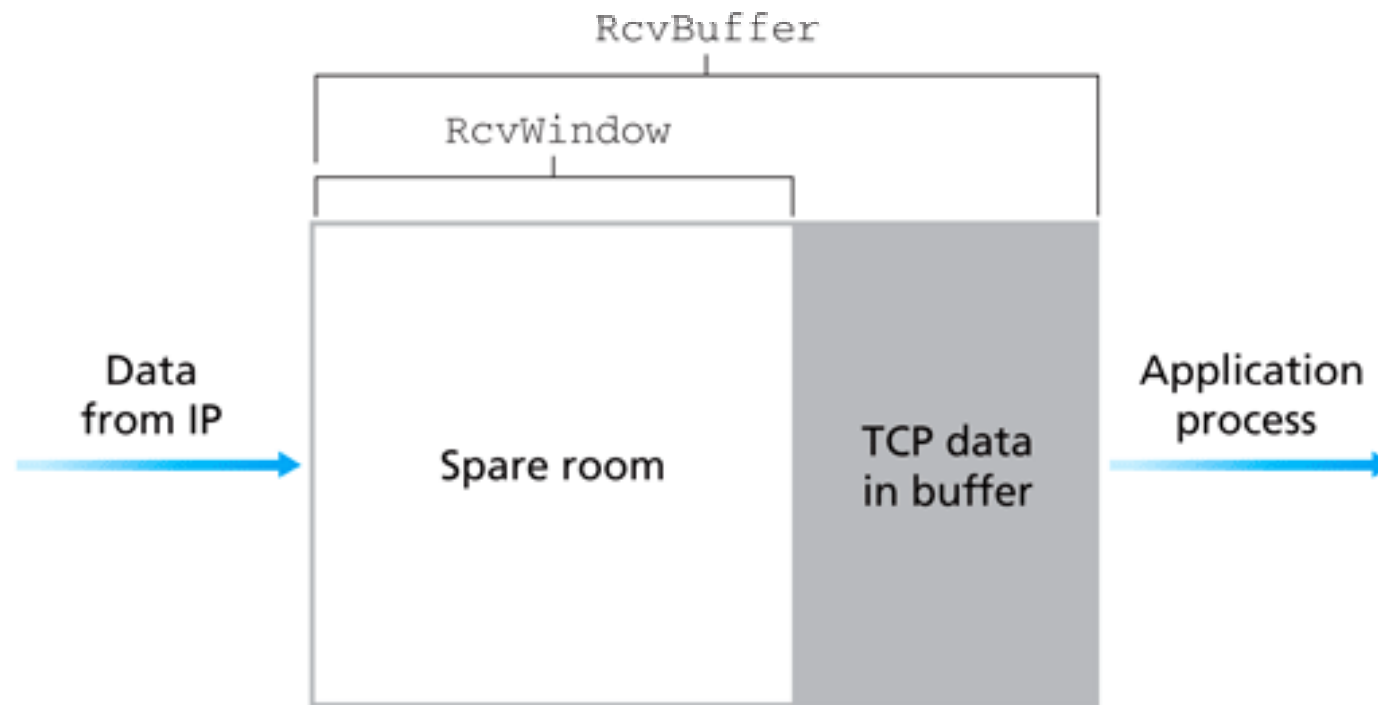
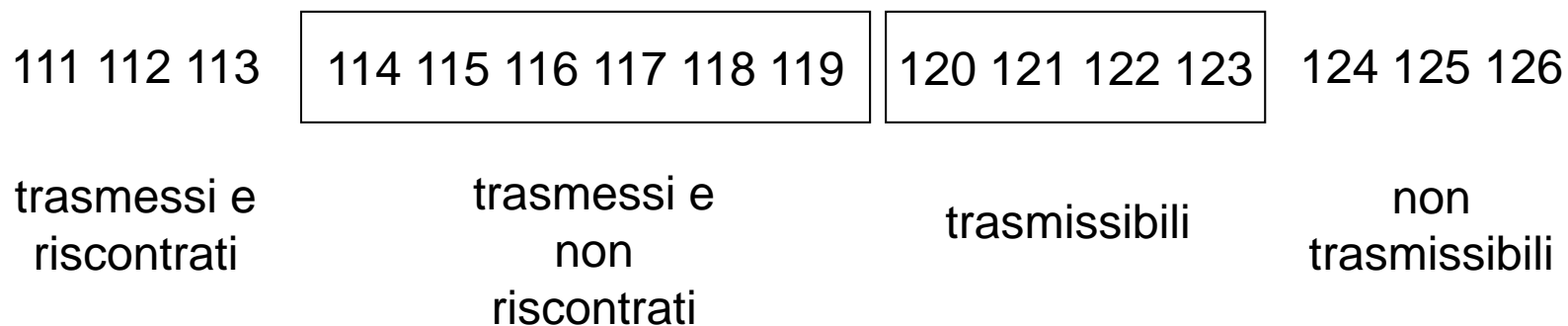


Figure 3.38 ♦ The receive window (RcvWindow) and the receive buffer (RcvBuffer)

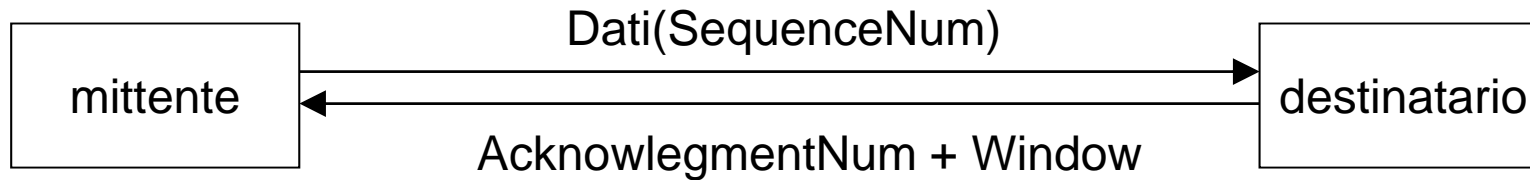
Gestione delle finestre in TCP

- L'algoritmo di finestra scorrevole e' usato da TCP per
 - garantire l'inoltro affidabile e ordinato dei dati
 - realizzare il controllo di flusso
- Caratteristiche
 - ogni byte del flusso complessivo ha un numero di sequenza
 - i riscontri (ack) sono cumulativi
- La dimensione della finestra determina l'ammontare di dati che possono essere trasmessi senza attendere il riscontro (ack)



Formato del segmento TCP

- Il segmento contiene una porzione del flusso di bytes complessivo



- Sequence number: numero di sequenza del primo byte nel segmento TCP
- Acknowledgment number: prossimo byte aspettato (= LastByteAked+1)
- Window:: numero di byte liberi nel buffer del destinatario
- Code bits. SYN, FIN, RESET, PUSH, URG, ACK

0	4	10	16	24	31
SOURCE PORT			DESTINATION PORT		
SEQUENCE NUMBER					
ACKNOWLEDGEMENT NUMBER					
HLEN	NOT USED	CODE BITS	WINDOW		
CHECKSUM			URGENT POINTER		
BEGINNING OF DATA					
⋮					

Algoritmo di finestra scorrevole rivisitato: controllo di flusso

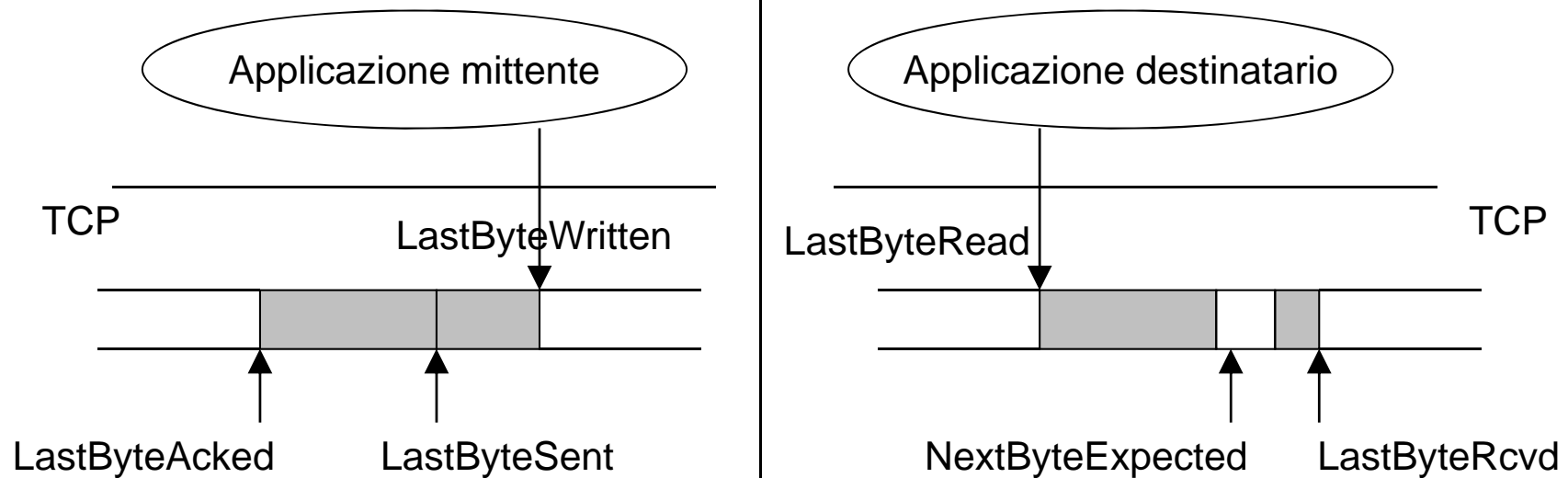
Mittente

- Dimensione buffer: MaxSendBuffer
- $\text{LastByteSent} - \text{LastByteAcked} \leq \text{AdvertisedWindow}$
- $\text{EffectiveWindow} = \text{AdvertisedWindow} - (\text{LastByteSent} - \text{LastByteAcked})$
spedisci se $\text{EffectiveWindow} > 0$
- $\text{LastByteWritten} - \text{LastByteAcked} \leq \text{MaxSendBuffer}$
blocca applicazione mittente se $(\text{LastByteWritten} - \text{LastByteAcked}) + y > \text{MaxSendBuffer}$
- spedisci segmenti sonda (un byte di dati) se $\text{AdvertisedWindow} = 0$

Destinatario

- Dimensione buffer: MaxRcvBuffer
- $\text{LastByteRcvd} - \text{LastByteRead} \leq \text{MaxRcvBuffer}$
- $\text{AdvertisedWindow} = \text{MaxRcvBuffer} - (\text{LastByteRcvd} - \text{LastByteRead})$

Algoritmo di finestra scorrevole rivisitato: inoltro affidabile e ordinato



Mittente

- $\text{LastByteAcked} \leq \text{LastByteSent}$
- $\text{LastByteSent} \leq \text{LastByteWritten}$
- I bytes tra LastByteAcked e LastByteWritten sono nel buffer

Destinatario

- $\text{LastByteRead} < \text{NextByteExpected}$
- $\text{NextByteExpected} \leq \text{LastByteRcvd} + 1$
(= se i dati sono arrivati in ordine altrimenti punta all'inizio del primo intervallo vuoto)
- I bytes tra LastByteRead e LastByteRcvd sono nel buffer

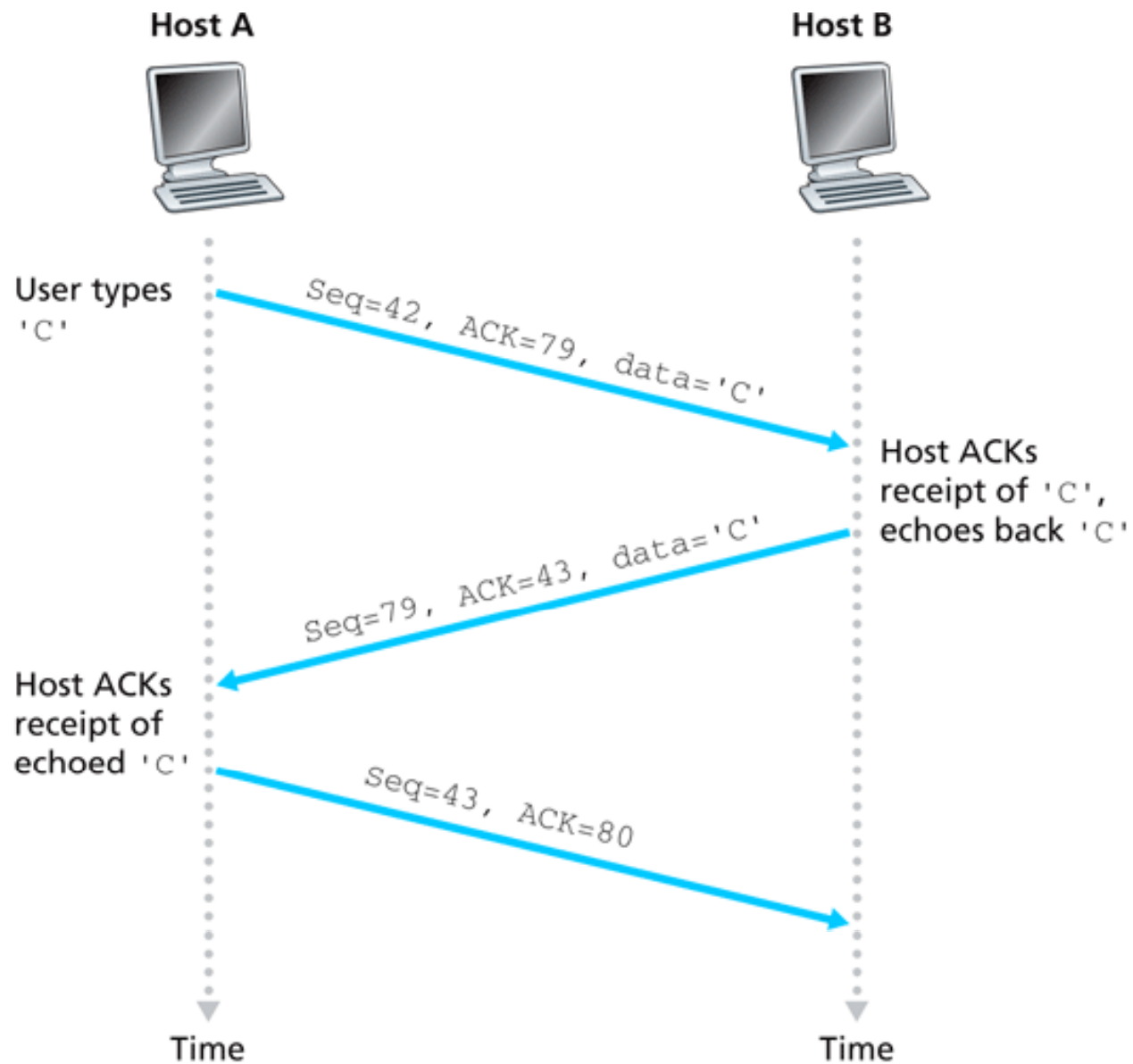


Figure 3.31 ♦ Sequence and acknowledgement numbers for a simple Telnet application over TCP

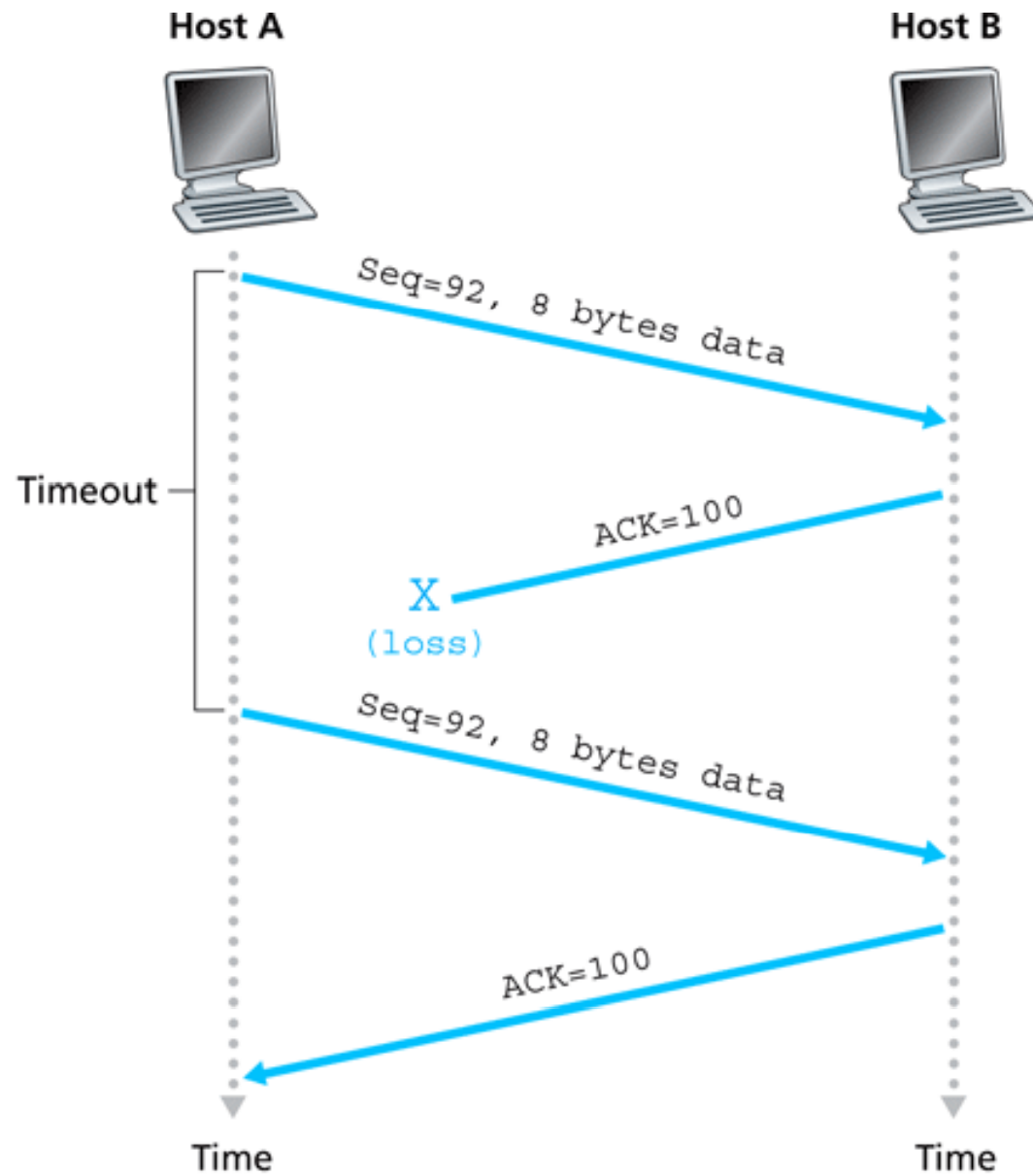


Figure 3.34 ♦ Retransmission due to a lost acknowledgment

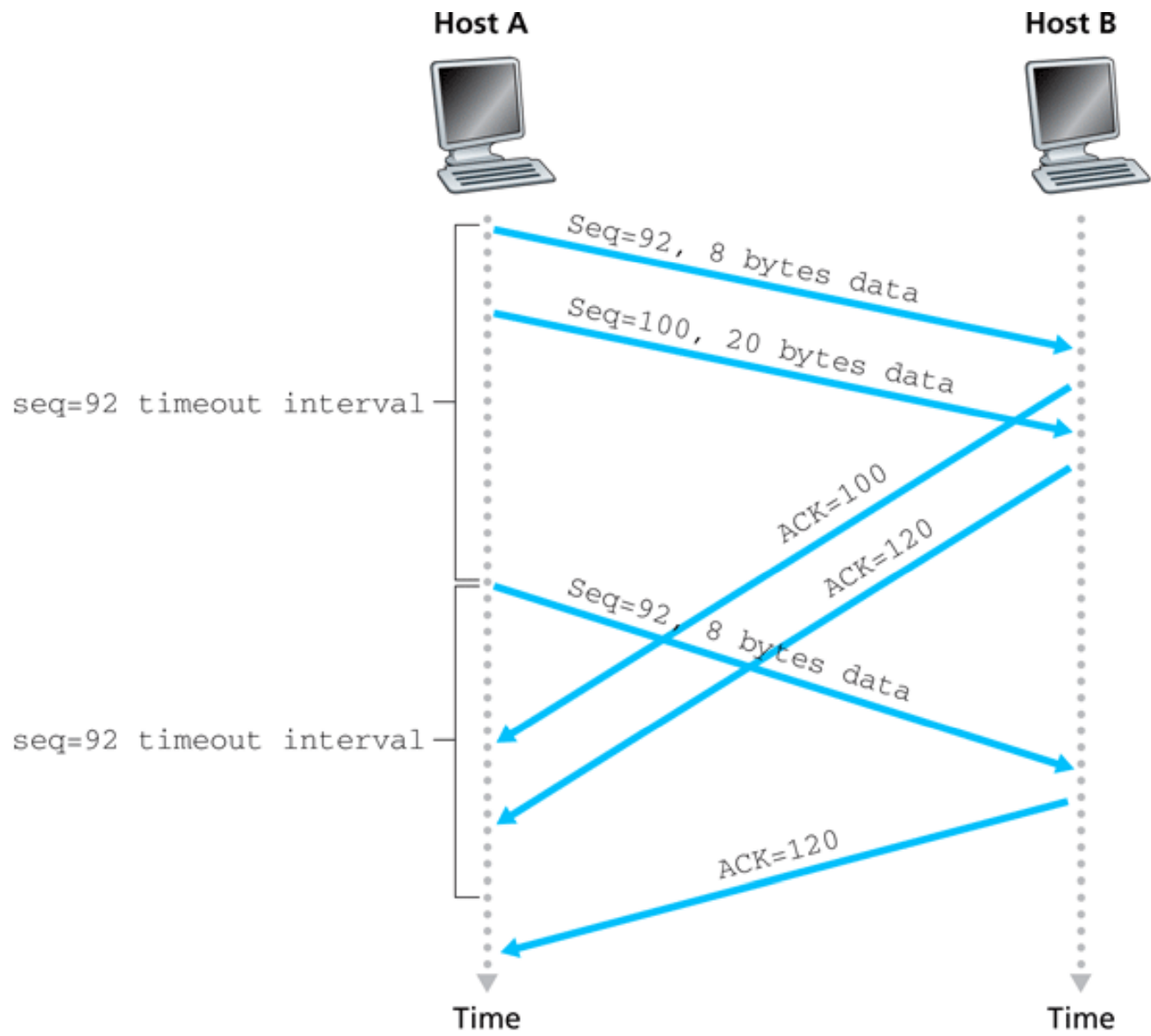


Figure 3.35 ♦ Segment 100 not retransmitted

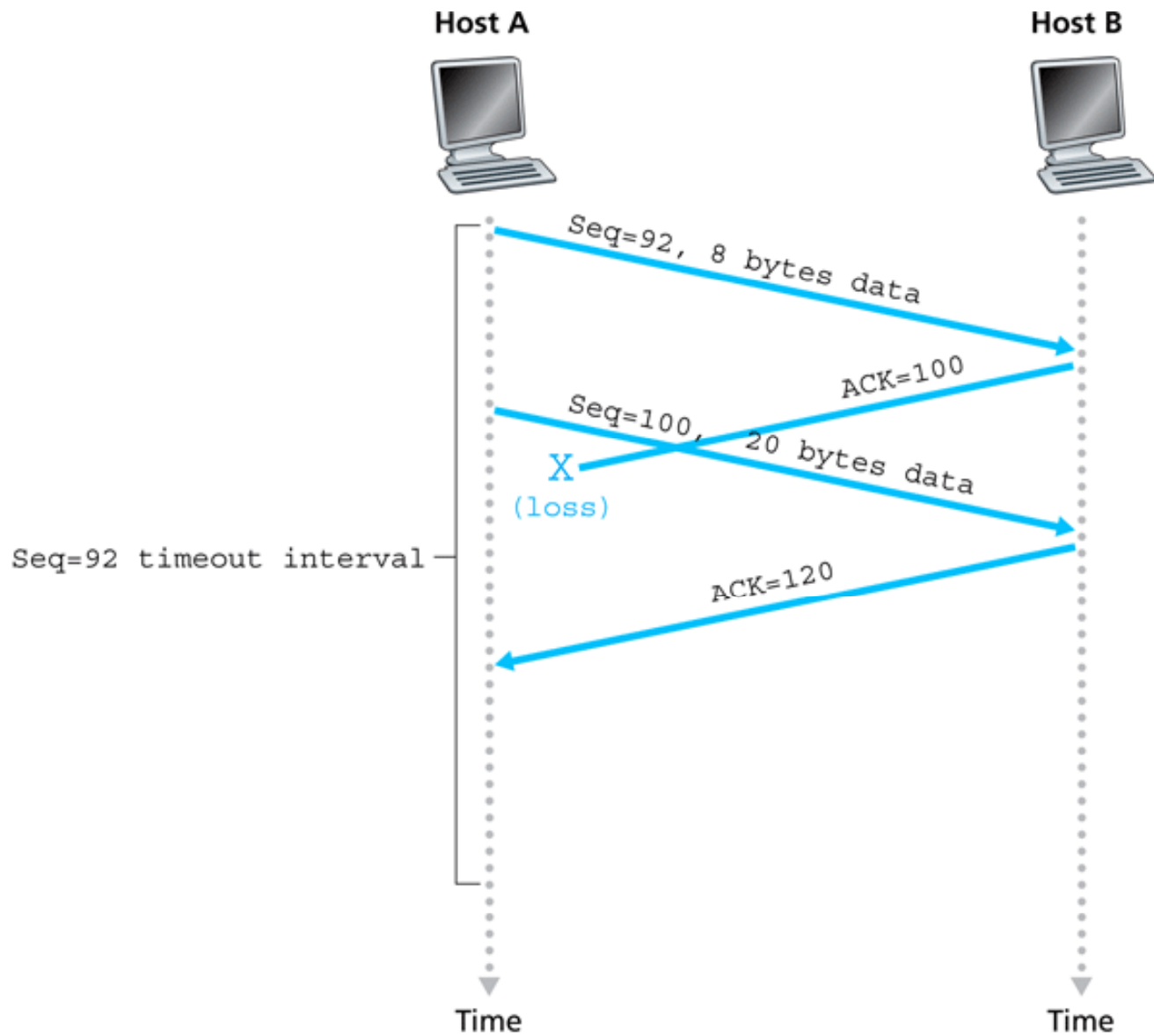


Figure 3.36 ♦ A cumulative acknowledgment avoids retransmission of the first segment.

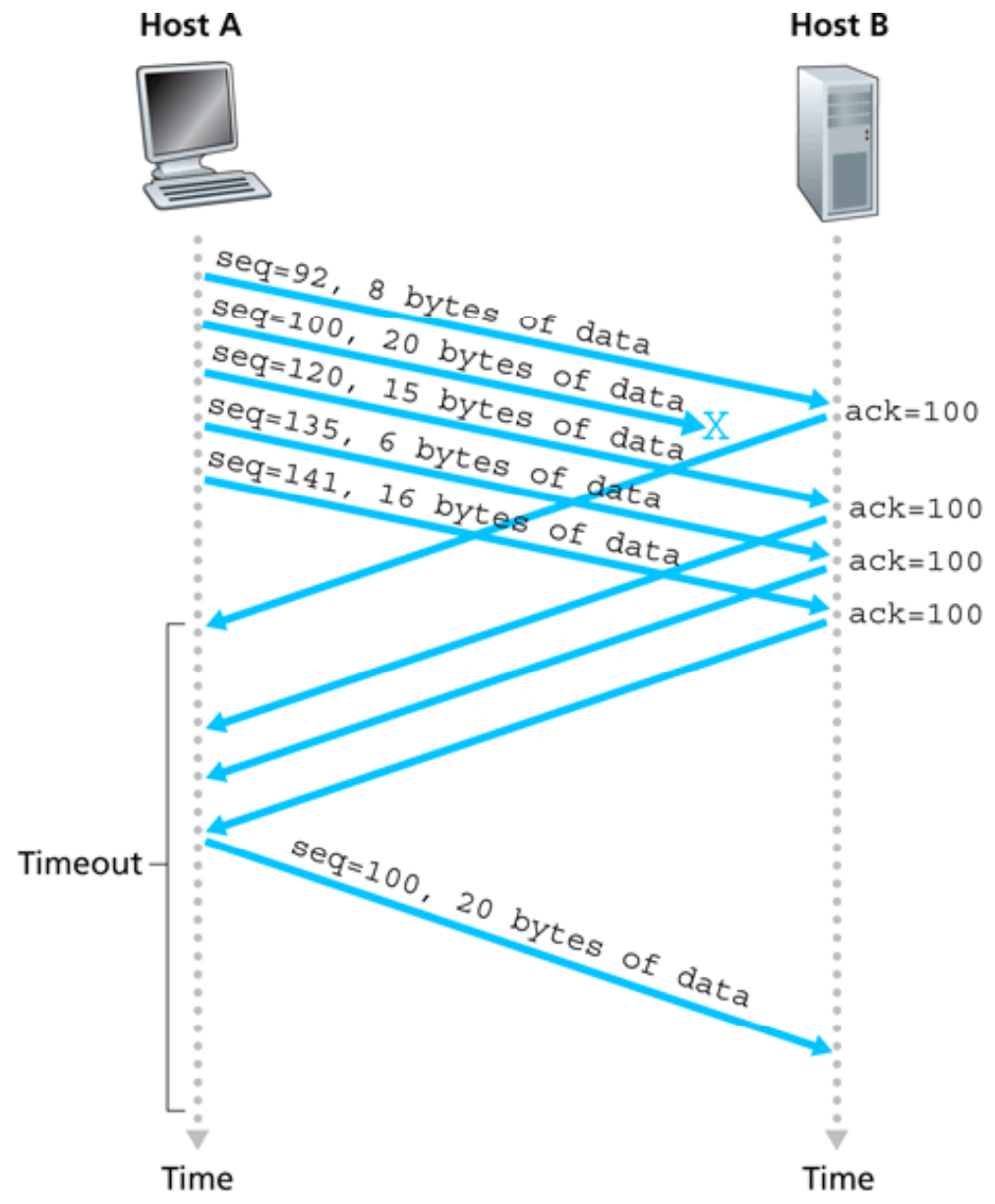


Figure 3.37 ♦ Fast retransmit: retransmitting the missing segment before the segment's timer expires.

Controllo di congestione

- Un traffico eccessivo può causare la perdita di messaggi
 - i protocolli di trasporto rispondono con ritrasmissione
 - Un'eccessiva ritrasmissione causa una maggiore congestione fino al collasso della rete
- TCP interpreta la perdita di messaggi come spia della congestione e previene il collasso
 - il mittente ritarda la trasmissione dei messaggi e la raddoppia fino a che si è raggiunta la meta' dell'advertising window e non ci sono altre perdite

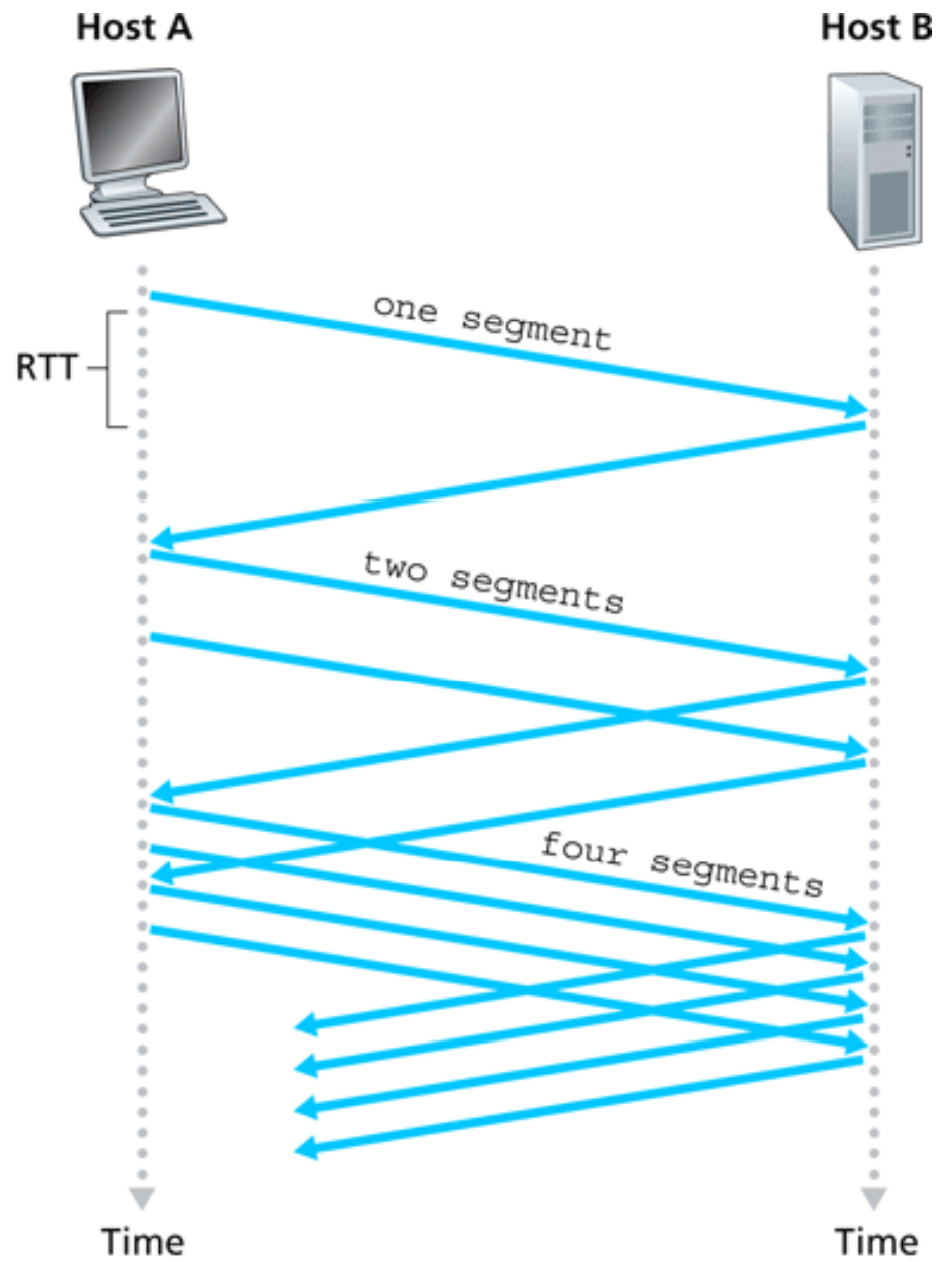


Figure 3.52 ♦ TCP slow start

Creazione della connessione

Requisiti

- Creazione affidabile della connessione
 - i dati di una vecchia connessione non si devono confondere con le nuove connessioni

Soluzione: three-way handshake

- Coinvolge lo scambio di tre segmenti tra client e server
 - lo scambio permette un accordo su alcuni parametri: per TCP i sequence number iniziali dei rispettivi flussi di byte
 - Uso del flag (code bit) SYN

Three-way handshake
per l'apertura
di una
connessione

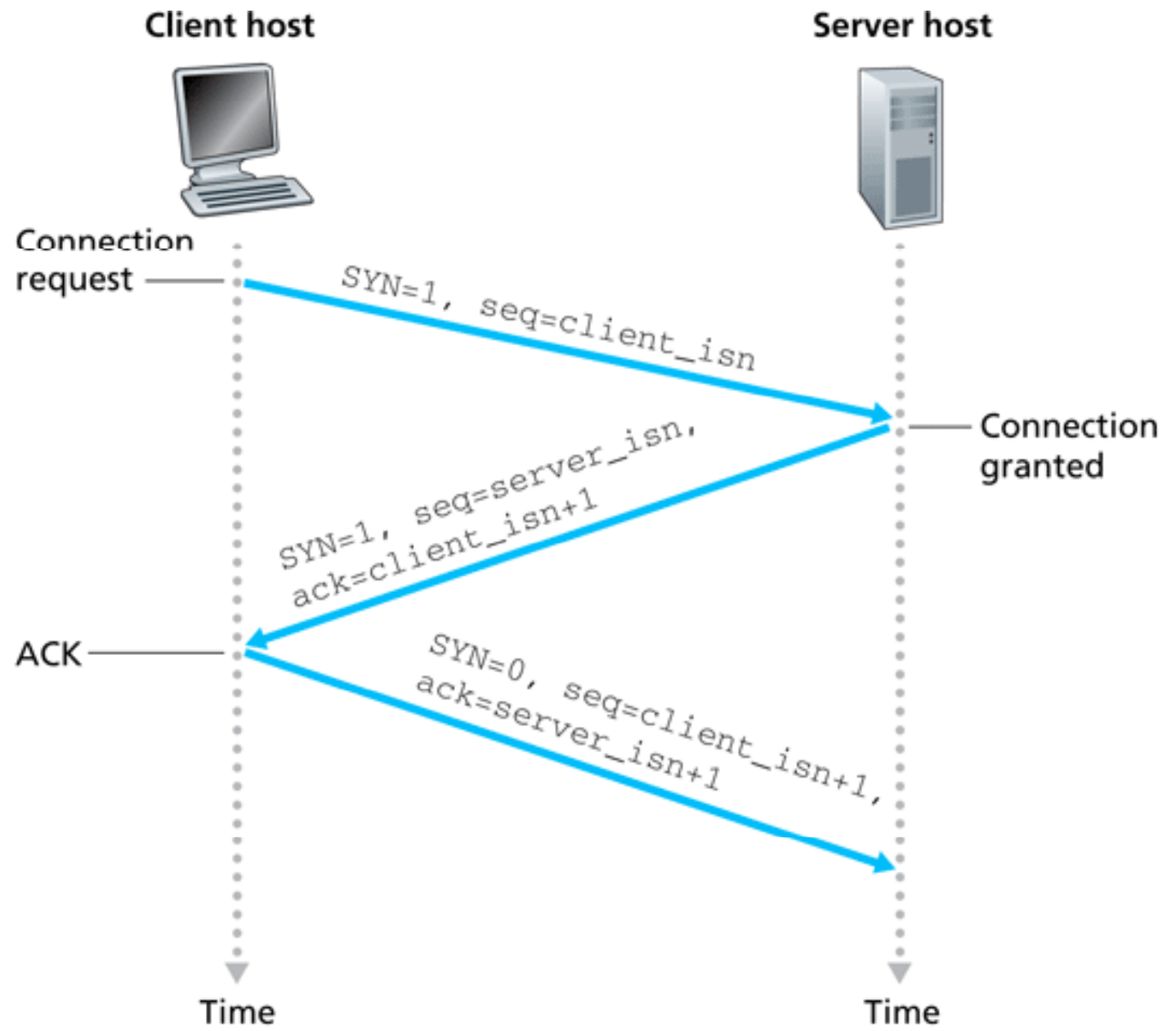


Figure 3.39 ♦ TCP three-way handshake: segment exchange

Distruzione della connessione

Requisiti

- Distruzione dolce della connessione
 - i dati spediti prima di chiudere una connessione non devono essere persi

Soluzione

- Coinvolge lo scambio di quattro segmenti tra client e server
 - uso del flag (code bit) FIN

Doppio
handshake per la
chiusura di una
connessione

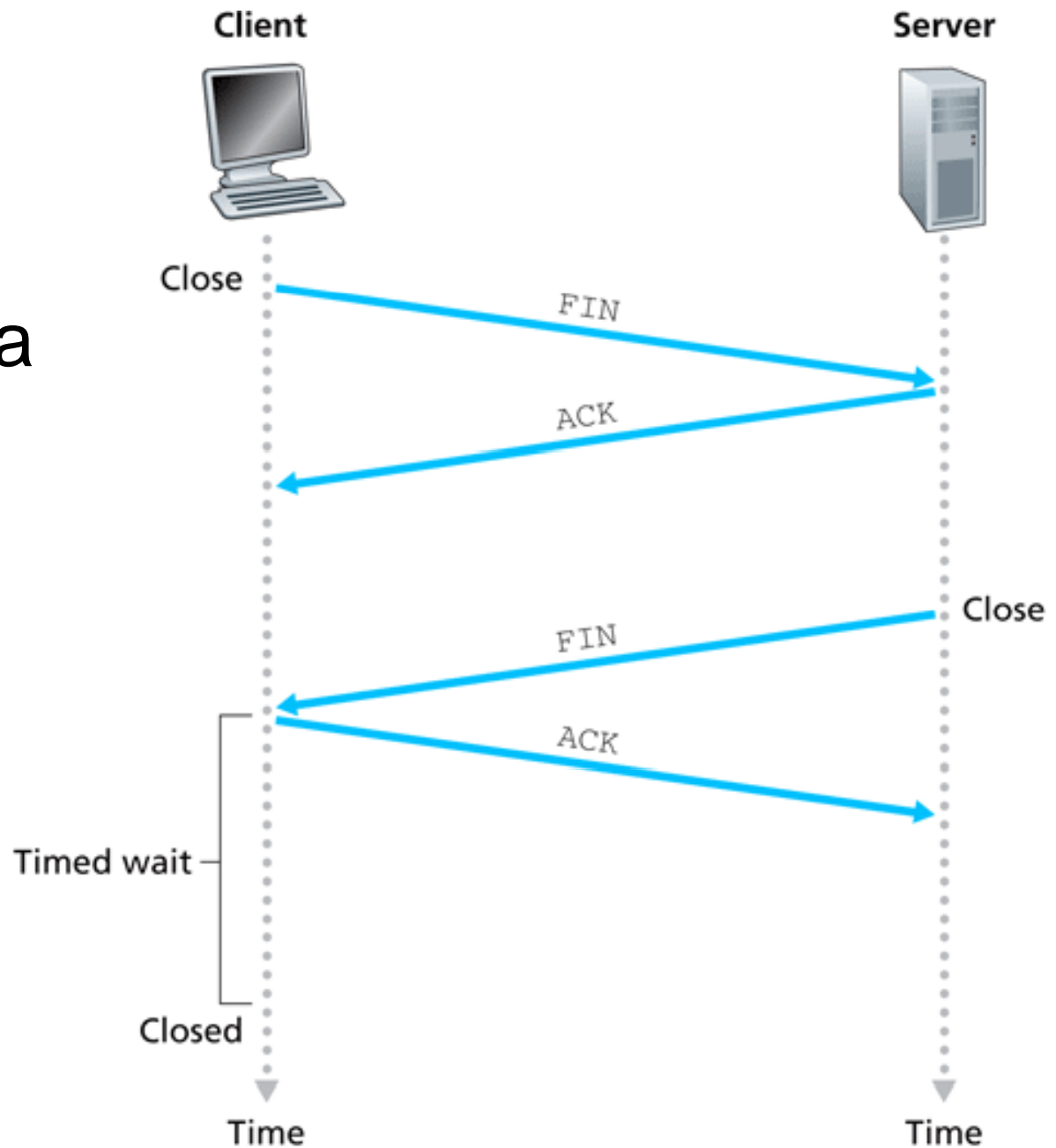


Figure 3.40 ♦ Closing a TCP connection

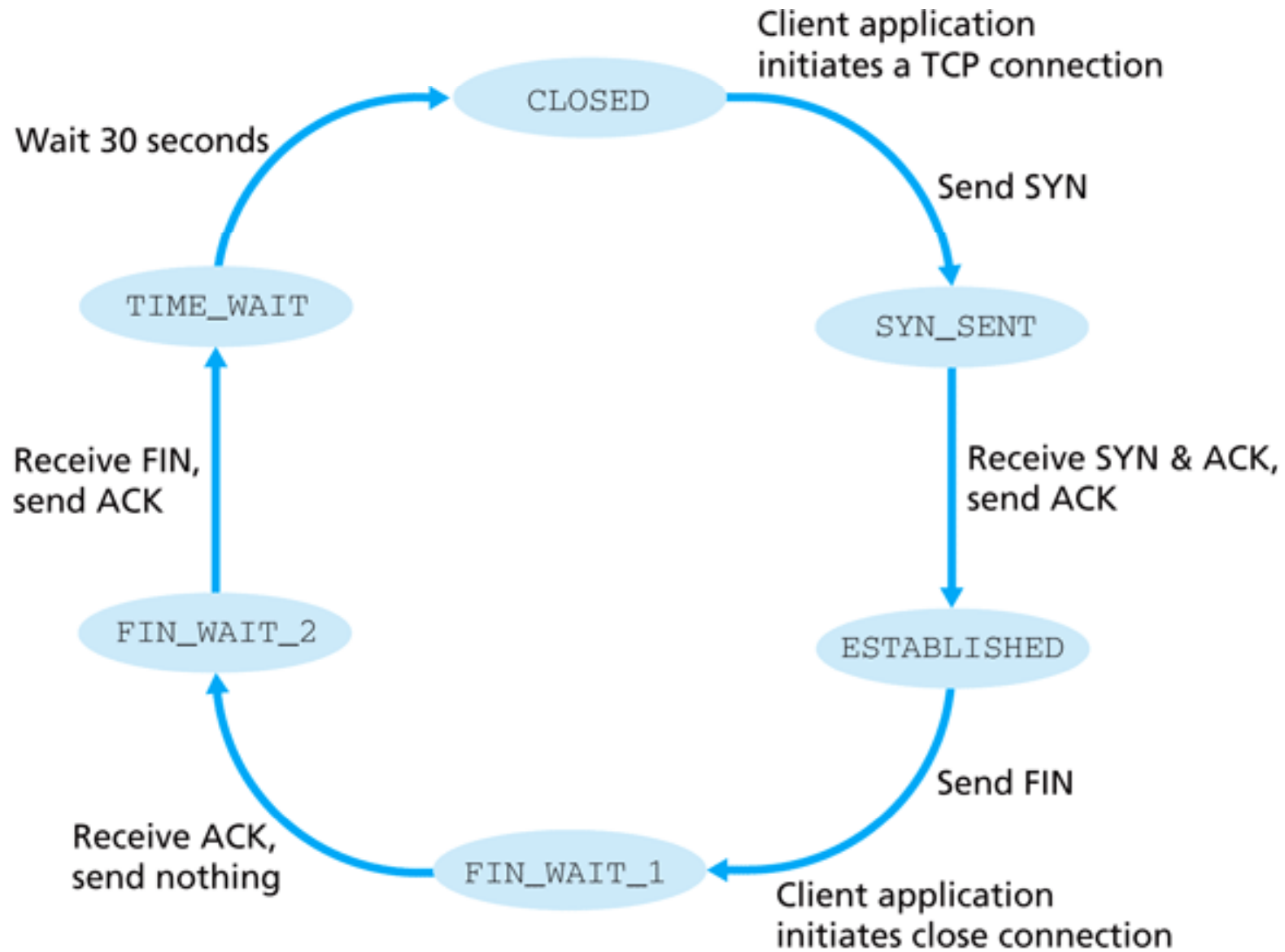


Figure 3.41 ♦ A typical sequence of TCP states visited by a client TCP

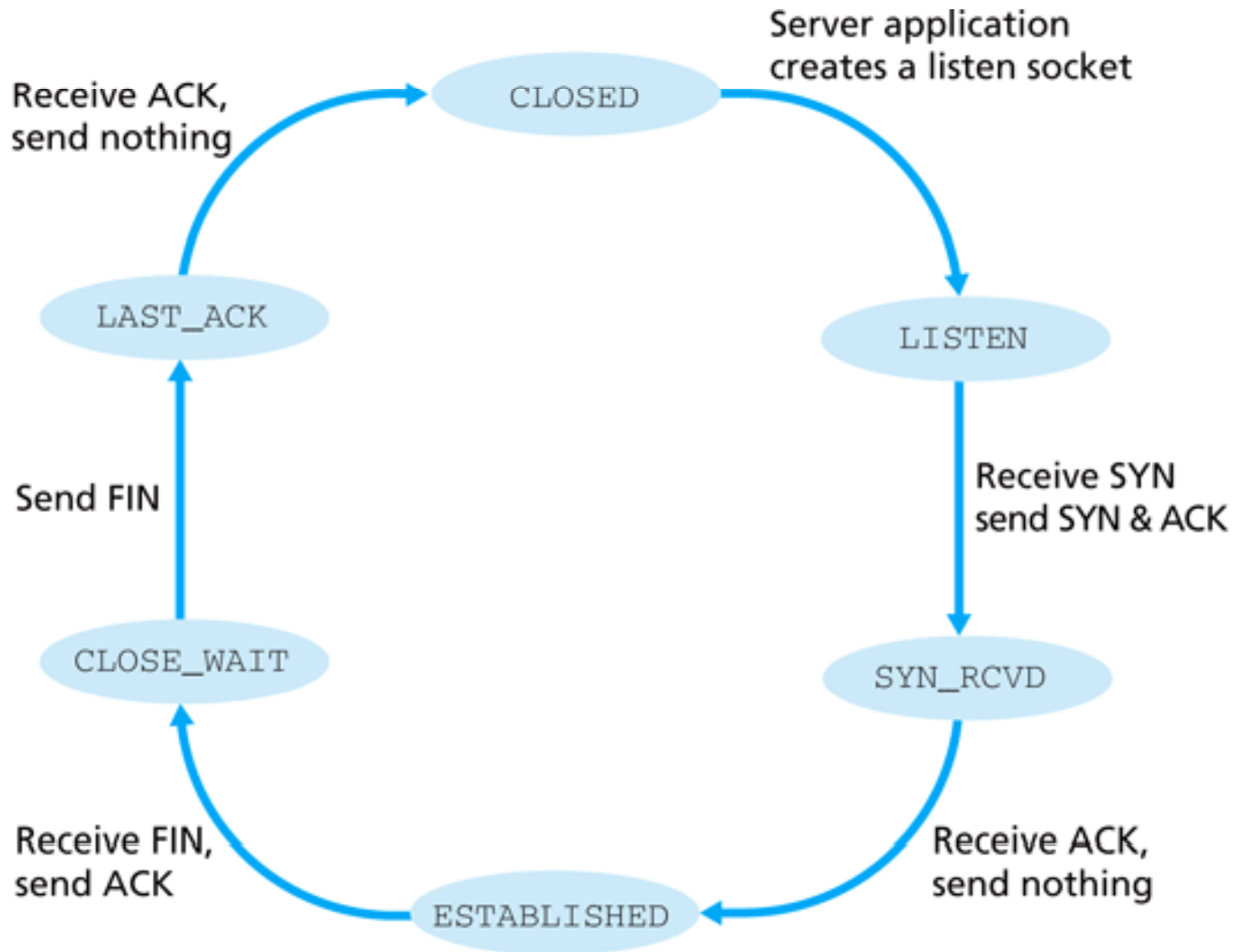


Figure 3.42 ♦ A typical sequence of TCP states visited by a server-side TCP

Diagramma di macchina a stati

