Corso di Laurea in Ingegneria Informatica

Corso di Reti di Calcolatori

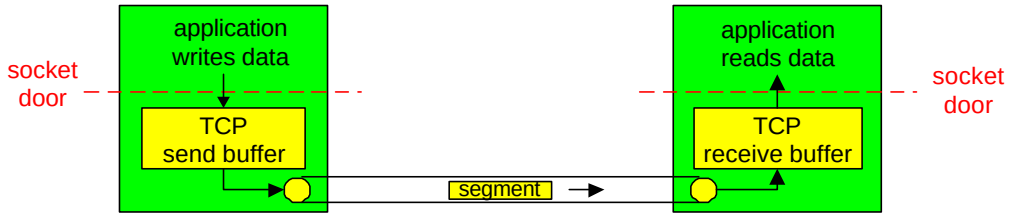
Docente: Simon Pietro Romano
spromano@unina.it

Il livello trasporto:

Il protocollo TCP

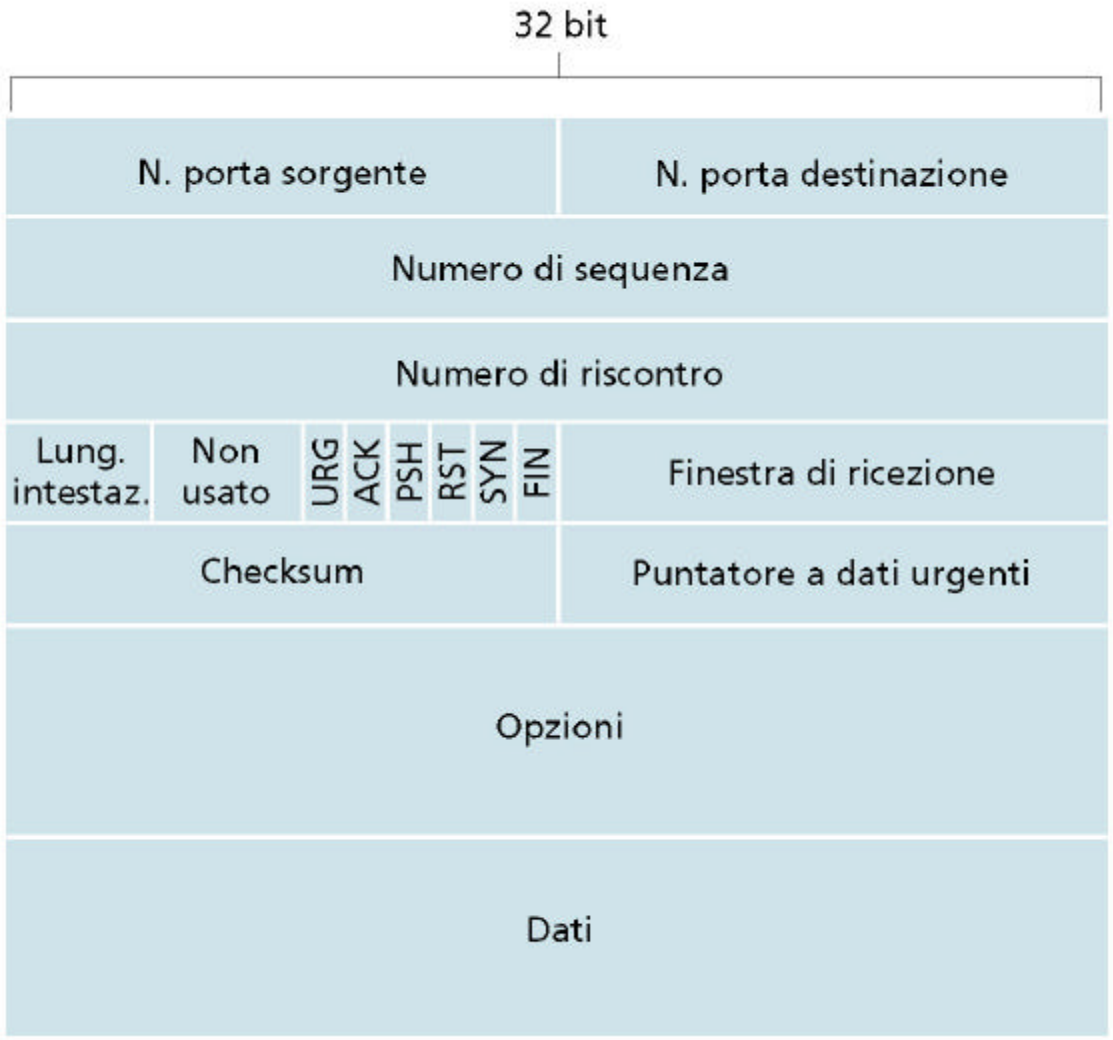
TCP: Transmission Control Protocol

- End-to-end:
  - Una connessione unica tra mittente e ricevente
- Senza errori, sequenza ordinata.
- pipelined:
  - Controllo di flusso e di congestione impostano la TCP window
- *Buffers su mittente e ricevente*



- full duplex data:
  - Flusso di dati bi-direzionale all'interno della stessa connessione
  - MSS: maximum segment size
- connection-oriented:
  - handshaking (scambio di msg di controllo a tre vie) prepara mittente e ricevente prima della comunicazione
- controllo di flusso:
  - Il mittente non invia più di quanto il ricevente non possa accettare

Struttura del segmento TCP



TCP: PDU

- Bit di Codice:
  - Per identificare il tipo di informazione contenuta nel segmento vengono impiegati i 6 bit di codice:
    - URG: Il campo puntatore urgente è valido
    - ACK: Il campo riscontro è valido
    - PSH: Questo segmento richiede una “spinta”
    - RST: Effettua il reset della connessione
    - SYN: Sincronizza i numeri di sequenza
    - FIN: Il trasmettitore ha raggiunto la fine del suo stream di byte

TCP: PDU

- HLEN:
  - Contiene un numero intero che indica la lunghezza dell'intestazione TCP del datagramma. Questa informazione è necessaria perché il campo **opzioni** è di lunghezza variabile
- Porta (provenienza/destinazione):
  - Contengono i numeri di porta di protocollo TCP che identificano gli applicativi alle estremità della connessione

TCP: PDU

- Numero sequenziale:
  - questo campo identifica, nello stream di byte del trasmettitore, la posizione dei dati nel segmento. Questo valore è riferito alla stream che fluisce nella medesima direzione del segmento, mentre il **Numero di Riscontro** si riferisce alla stream che fluisce nella direzione opposta
- Numero di riscontro:
  - Contiene il numero sequenziale del byte successivo a quello correttamente ricevuto dalla destinazione. Tale campo è valido solo nei segmenti di riscontro, o nei segmenti utilizzanti la tecnica trasmissiva **Piggy-backing**, e fa riferimento allo stream di dati che fluisce nella direzione opposta a tale segmento

TCP: PDU

- Finestra:
  - Numero intero senza segno di 16 bit che specifica la dimensione del buffer che il TCP ha a disposizione per immagazzinare dati in arrivo. È utilizzato per la gestione dinamica della dimensione della finestra scorrevole
- Puntatore urgente:
  - Il TCP permette la trasmissione fuori banda di dati informativi ad alta priorità. Questi devono essere trasmessi il prima possibile, indipendentemente dalla loro posizione nello stream. Questo campo, se valido, conterrà un puntatore alla posizione, nello stream, dei dati NON urgenti

TCP: PDU

- Checksum:
  - Campo di 16 bit contenente un valore intero utilizzato dal TCP della macchina host di destinazione, per verificare l'integrità dei dati e la correttezza dell'intestazione
    - questa informazione è di essenziale importanza perché il protocollo IP non prevede nessun controllo di errore sulla parte dati del frame
    - per il calcolo del valore checksum il TCP ha bisogno di aggiungere una pseudointestazione al datagramma, per effettuare così un controllo anche sugli indirizzi IP di destinazione e provenienza

TCP: PDU

La Pseudointestazione viene creata e posta in testa al datagramma TCP. Viene inserito in essa un ulteriore byte di zeri per raggiungere un multiplo di 16 bit. Successivamente viene calcolata la checksum su tutto il messaggio così formato, viene scartata la pseudointestazione e passato il datagramma al livello IP. In fase di ricezione, il livello TCP ricrea la pseudointestazione interagendo con l'IP sottostante, calcola la checksum e verifica la correttezza del messaggio ricevuto. In caso di errore il datagramma verrà scartato (e quindi ritrasmesso dal mittente)

TCP: PDU

- Principali opzioni di TCP
  - Maximum TCP payload: durante la fase di connessione, ciascun end-point annuncia la massima dimensione di payload che desidera accettare; la minima tra le due dimensioni annunciate viene selezionata per la trasmissione
  - Window Scale: per negoziare un fattore di scala per la finestra; utile per connessioni a larga banda e/o elevato ritardo di trasmissione
  - Selective Repeat: nel caso in cui un segmento corrotto sia stato seguito da segmenti corretti, introduce i NAK (Not AcKnowledge), per permettere al receiver di richiedere la ritrasmissione di quello specifico segmento; è un'alternativa al "go back n", che prevede la ritrasmissione di tutti i segmenti

TCP: Caratteristiche

- Riscontro e ritrasmissione:
  - Consiste nella ritrasmissione di un segmento se non è giunta conferma entro un tempo massimo (time-out)
- Time-Out:
  - Al momento della trasmissione di un segmento, il TCP attiva un timer

TCP: Caratteristiche

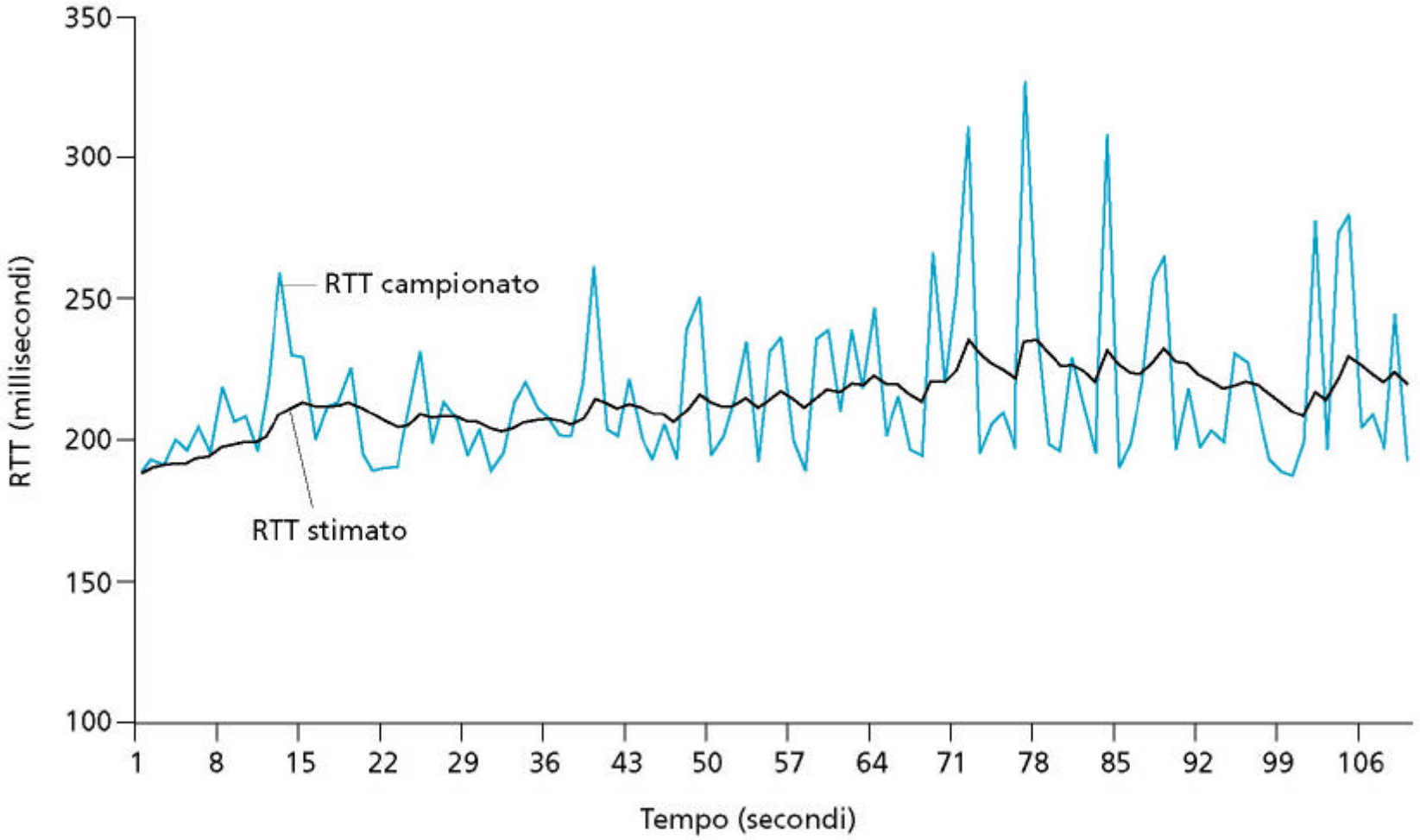
- Nel datagramma di riscontro la destinazione comunica quale byte dello stream si aspetta di ricevere successivamente:
  - I riscontri specificano sempre il numero sequenziale del primo byte non ancora ricevuto
    - » Esempio: in uno stream di 1000 byte segmentato in blocchi di 100 byte, il primo riscontro conterrà il numero sequenziale 101
  - Con questo metodo di riscontro *cumulativo* si ha il vantaggio che la perdita di un riscontro non blocca la trasmissione se confermato dal riscontro successivo

Round Trip Time e Timeout

Domanda: A che valore deve essere impostato il timeout?

- Di sicuro sarà maggiore del RTT (Round Trip Time)
- N.B: RTT varia nel tempo
  - Se timeout è scelto troppo breve:
    - timeout prematuro
      - ritrasmissioni ridondanti
      - scarsa efficienza
  - Se timeout è scelto troppo lungo:
    - scarsa efficienza nella gestione delle ritrasmissioni

RTT campionato vs RTT stimato



Calcolo del timeout

EstimatedRTT = (1-a)*EstimatedRTT + a*SampleRTT

- Una media esponenziale pesata (*EWMA: Exponential Weighted Moving Average*) dei campioni:
  - L'influenza di un singolo campione sul valore della stima decresce in maniera esponenziale
  - Valore tipico per a: 0.125

Valore del timeout:

- **EstimtedRTT** più un "margine di sicurezza" proporzionale alla variabilità della stima effettuata:
  - variazione significativa di **EstimatedRTT ->** margine più ampio:

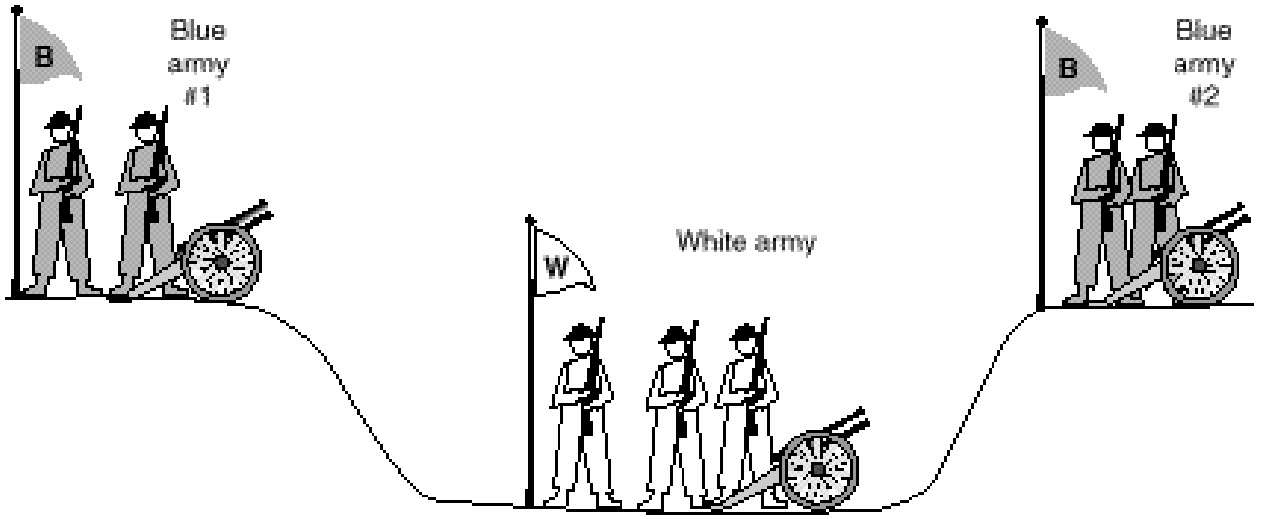
Timeout = EstimatedRTT + 4*DevRTT, dove:

DevRTT = (1-b)*DevRTT + b *|SampleRTT-EstimatedRTT|

NB: Valore raccomandato per b: 0,25

Apertura di una connessione

Il problema dei due eserciti:



TCP Connection Management

Mittente e Ricevente concordano l'apertura della connessione prima di inviare i dati

---

Impostare le variabili del TCP:

- Numeri di sequenza
- Allocare i buffer, impostare un valore iniziale della **`RcvWindow`**

Three way handshake:

Passo 1: client invia segmento di controllo TCP SYN al server

- Specifica il 1° seq #

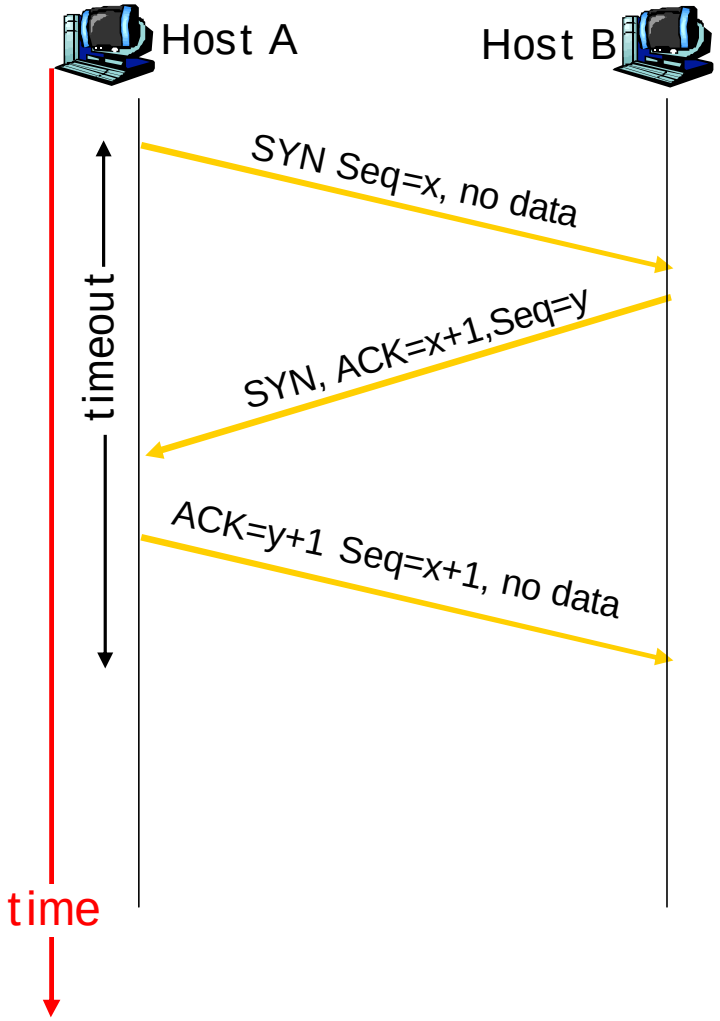
Passo 2: server riceve SYN, rsponde con segmento di controllo SYN/ACK

- ACK del SYN ricevuto
- Alloca buffer
- Specifica il 1°seq. # per la connessione server→client

Passo 3: client riceve SYN/ACK, invia ACK al server

- Connessione instaurata

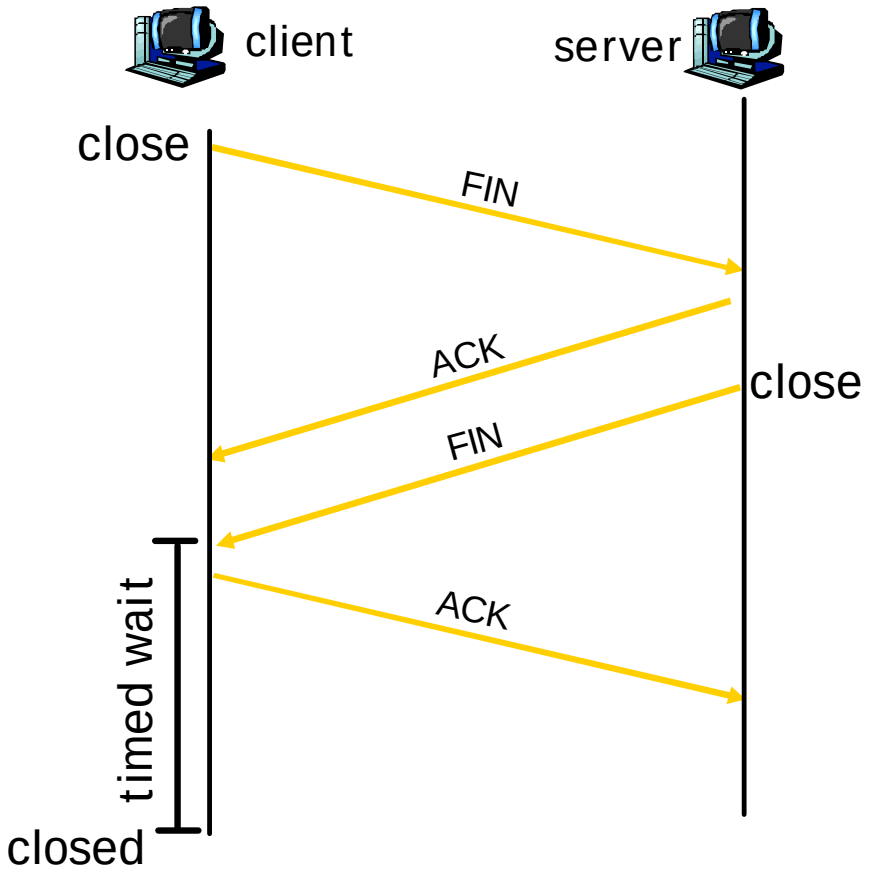
Three way handshake



Chiusura della connessione

Passo 1: client invia segmento di controllo TCP FIN al server

Step 2: server riceve FIN, risponde con ACK. Quindi chiude la connessione, invia ACK al client



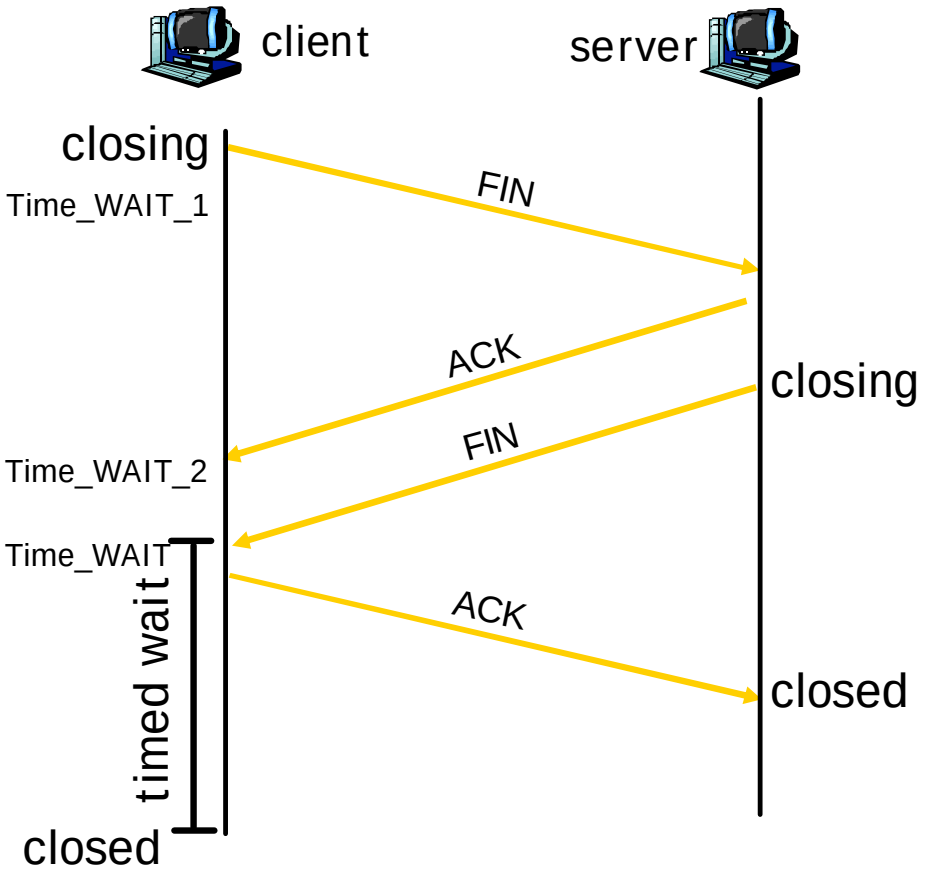
Chiusura della connessione

Passo 3: client riceve FIN, risponde con un ACK

- Attende in uno stato TIMED_WAIT (nel caso in cui l'ultimo ACK vada perso, e riceve un ulteriore FIN dal server)

Passo 4: server, riceve ACK. Chiude la connessione

N.B: una piccola variante al Passo 2: il sender invia ACK e FIN contemporaneamente all'interno dello stesso segmento



Numeri sequenza ed ACKs

Seq. number:

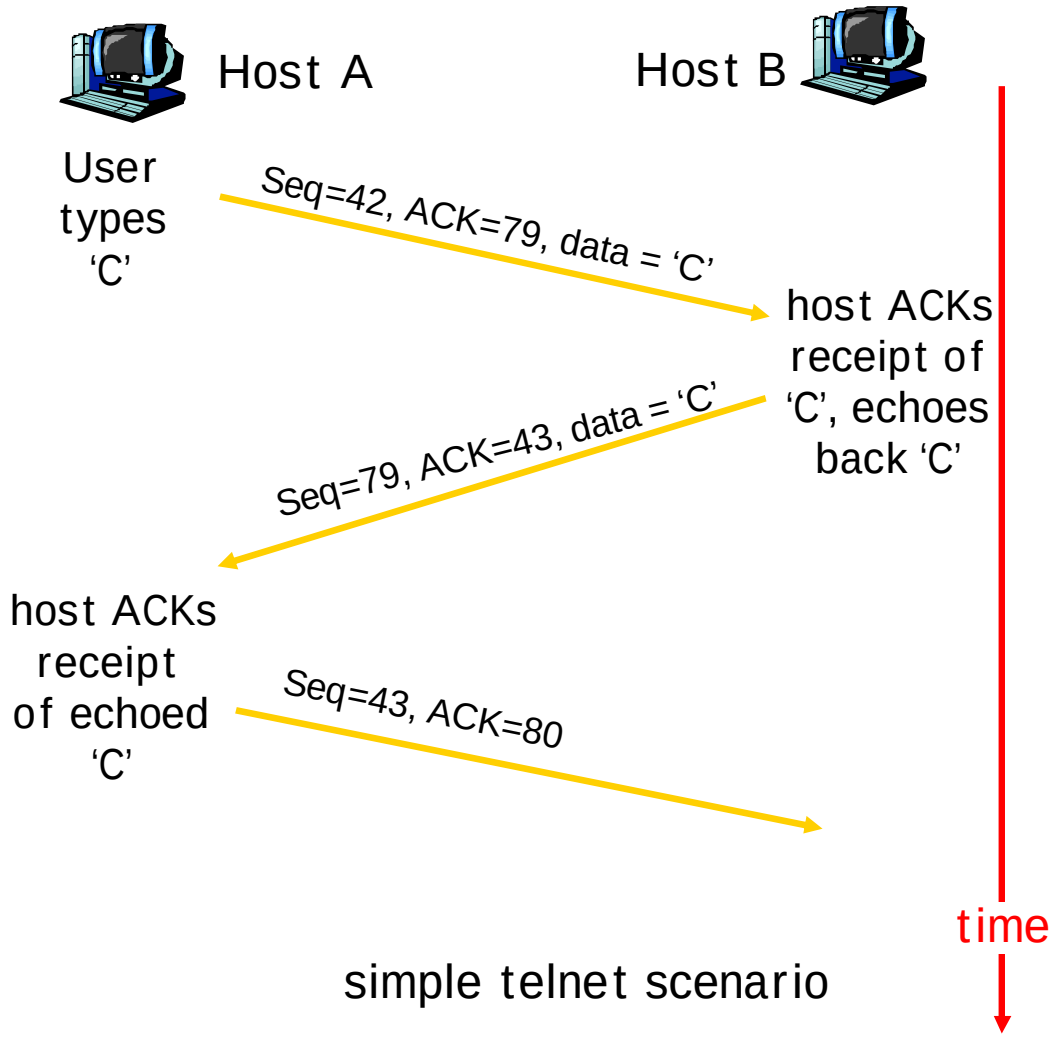
- Numero del primo byte del segmento in procinto di essere inviato

ACK:

- Numero del byte successivo atteso dall'altro host
- ACK

Q: in che modo il ricevente gestisce I segmenti fuori ordine?

- A: Nelle RFC non è specificato, ampia libertà ai programmatori



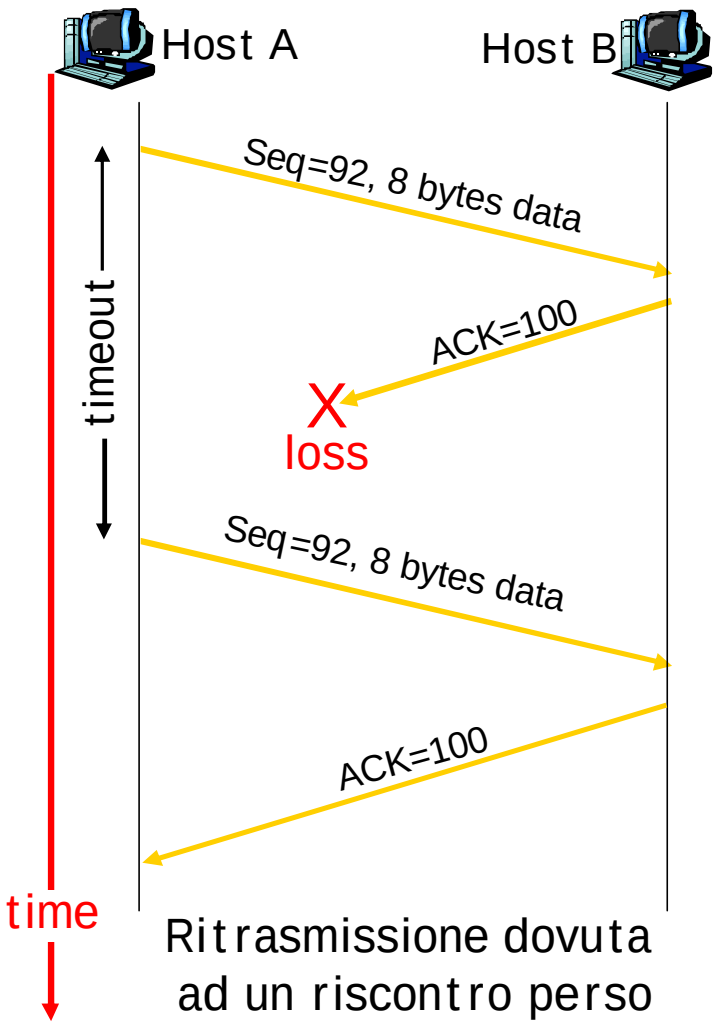
simple telnet scenario

Un sender TCP semplificato

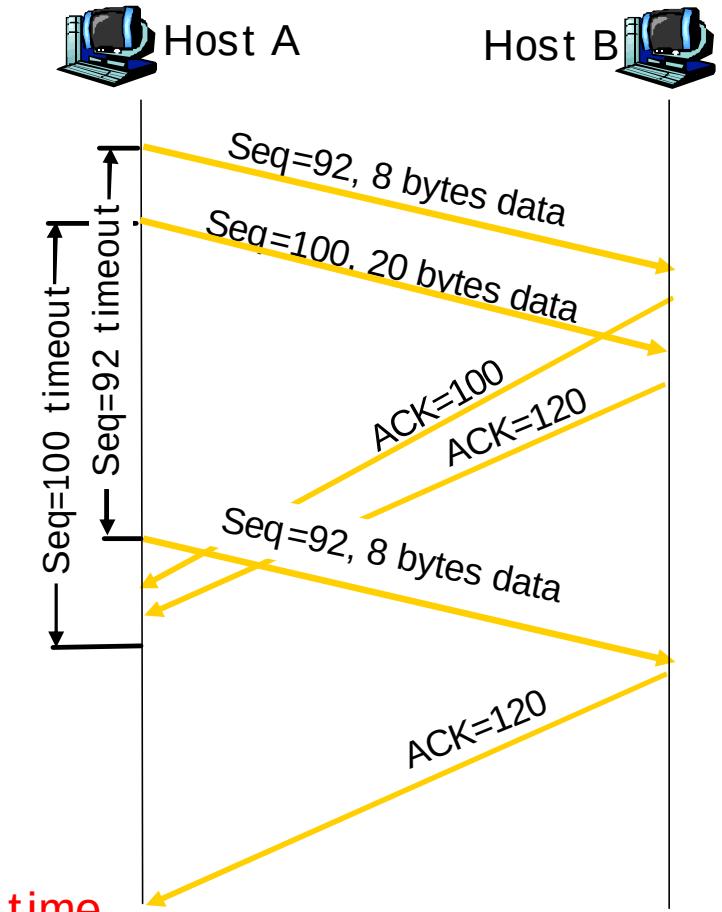
/* Si è assunto che il sender non sia limitato dal controllo di flusso o di congestione del TCP, che i dati da sopra siano di dimensioni inferiori all'MSS e che il trasferimento dei dati avvenga in una sola direzione.*/

```
NextSeqNum=InitialSeqNumber
SendBase=InitialSeqNumber

loop (forever) {
   switch(event)

      event: data received from application above
         create TCP segment with sequence number NextSeqNum
         if (timer currently not running)
            start timer
         pass segment to IP
         NextSeqNum=NextSeqNum+length(data)
         break;

      event: timer timeout
         retransmit not-yet-acknowledged segment with
            smallest sequence number
         start timer
         break;

      event: ACK received, with ACK field value of y
         if (y > SendBase) {
            SendBase=y
            if (there are currently any not-yet-acknowledged
               segments)
               start timer
            }
            break;

   } /* end of loop forever */
```

Alcuni scenari di rilievo - 1

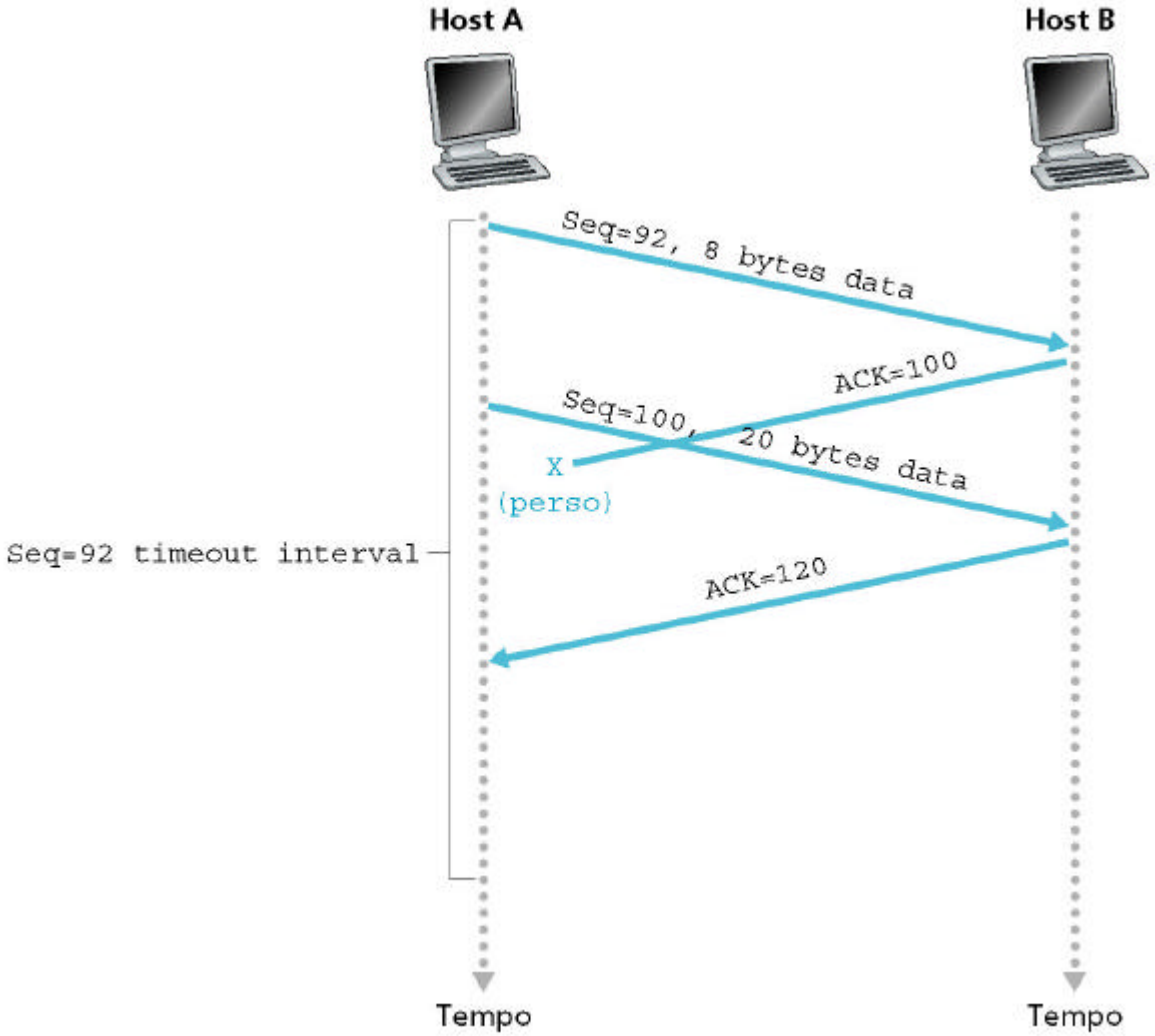


Ritrasmissione dovuta
ad un riscontro perso



Timeout prematuro,
azione dell' ACK cumulativo:
il segmento 100 non è ritrasmesso

Alcuni scenari di rilievo - 2



Il riscontro cumulativo evita la ritrasmissione del primo segmento...

Modifiche tipiche del TCP - 1

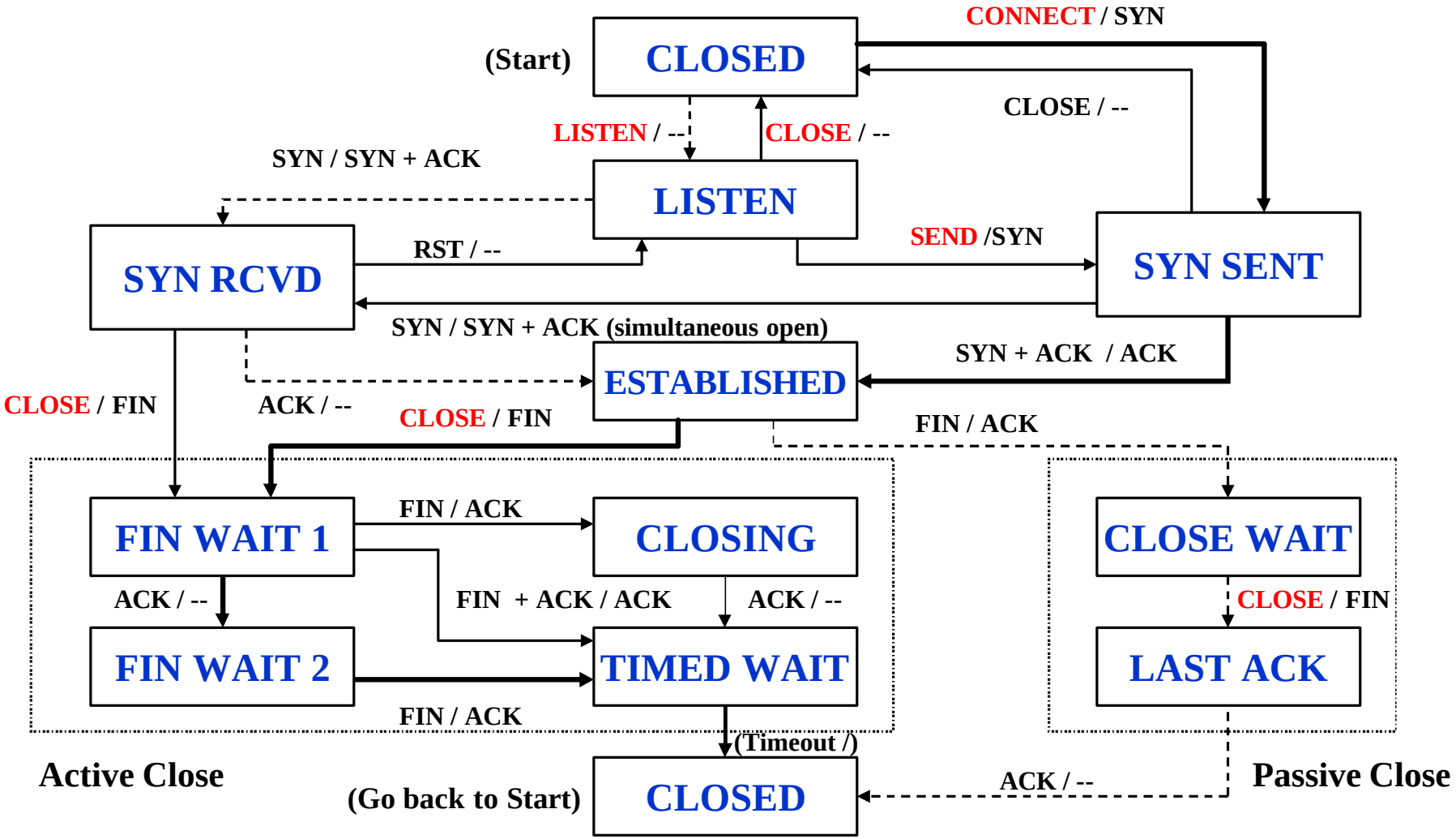
- Raddoppio dell'intervallo di timeout:
  - Allo scadere di un timeout:
    - si imposta il prossimo intervallo al doppio del valore precedente (invece di usare la stima di RTT)
      - Crescita esponenziale degli intervalli dopo ogni ritrasmissione
  - Quando il timer viene riavviato (ricezione di un ACK o di nuovi dati dall'applicazione):
    - l'intervallo di timeout viene nuovamente configurato in funzione dei valori più recenti di `EstimatedRTT` e `DevRTT`

- Fornisce una forma limitata di controllo della congestione:
  - Il mittente, in caso di supposta congestione (perdita di un segmento), ritrasmette ad intervalli sempre più lunghi

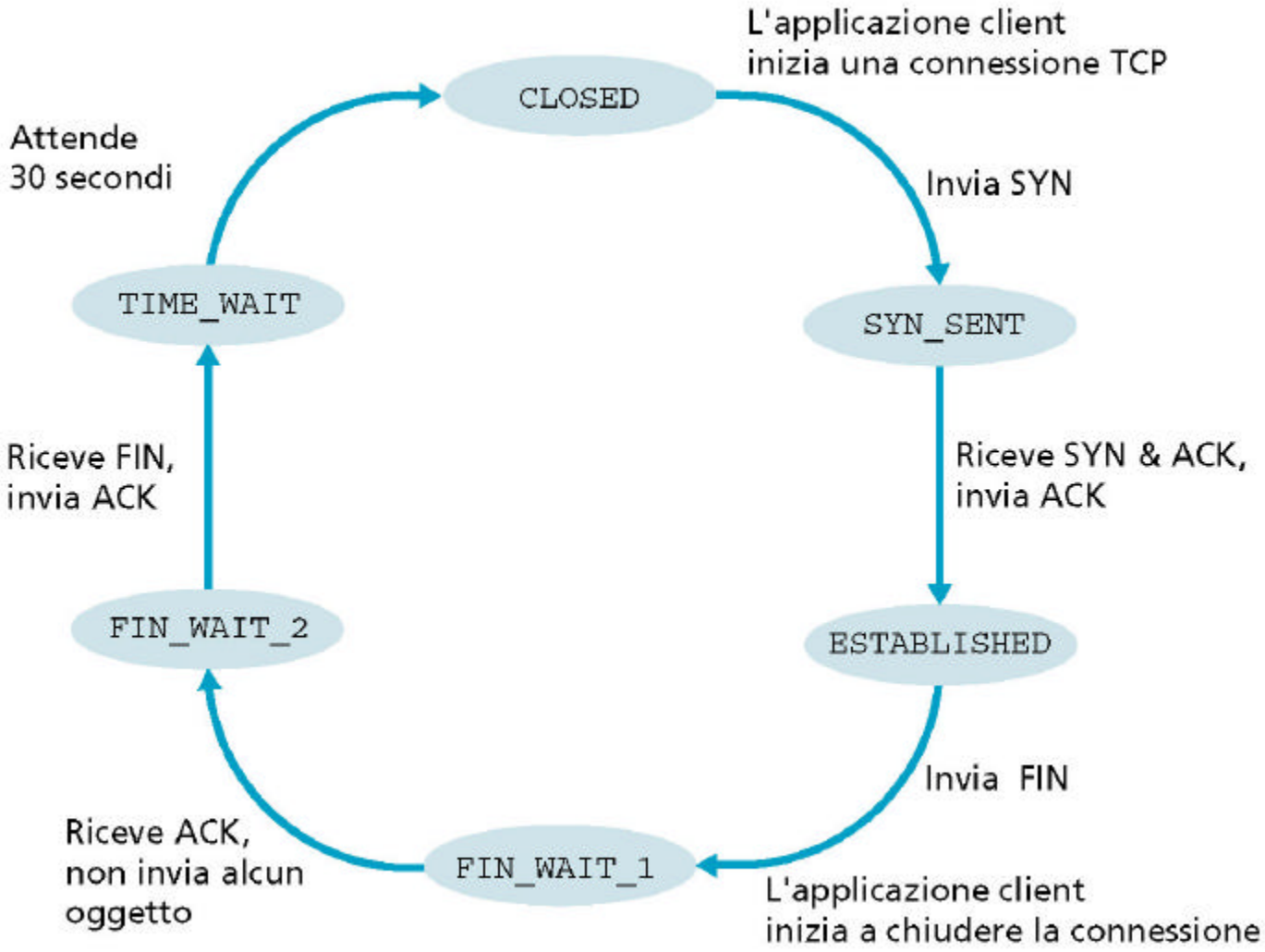
Modifiche tipiche del TCP - 2

- Ritrasmissione veloce:
  - ACK duplicati:
    - Consentono di rilevare la perdita di un pacchetto prima del timeout
      - un receiver che rileva un “buco” nei segmenti ricevuti (ricezione di un segmento con numero di sequenza maggiore di quello atteso):
        - » invia un nuovo riscontro per l’ultimo byte di dati che ha ricevuto correttamente
      - poiché il mittente spesso manda molti segmenti contigui, se uno di tali segmenti si perde, ci saranno molti ACK duplicati contigui:
        - » un sender che riceve tre ACK duplicati per gli stessi dati assume che il segmento successivo a quello riscontrato tre volte è andato perso ed effettua, quindi, una ritrasmissione prima della scadenza del timeout

Diagramma degli stati del TCP



Sequenza tipica degli stati nel client



Sequenza tipica degli stati nel server