Gruppo di Reti e Telematica Applicata

DIMES
UNIVERSITÀ DELLA CALABRIA

Corso di
Fondamenti di Reti di Telecomunicazioni
LT - ELE / LM-TLC

Lo strato di Trasporto

TCP è un protocollo orientato alla connessione; sono presenti le fasi di instaurazione della connessione, trasferimento dati ed abbattimento della connessione

Prima di iniziare il trasferimento di informazioni verso un utente remoto, deve essere instaurata una connessione, tramite un meccanismo di sincronizzazione noto come three-way handshake (stretta di mano in 3 fasi)

La sincronizzazione serve a risolvere potenziali situazioni anomale, dovute al fatto che IP non è affidabile e quindi i datagrammi possono essere persi, ritardati, duplicati o consegnati fuori sequenza, e che TCP ritrasmette i segmenti persi dopo un certo timeout:

- un segmento appartenente ad una "vecchia" connessione può arrivare ad un host dopo che tra i processi relativi a quel segmento è stata instaurata una nuova connessione
- un host "cade" e perde traccia delle connessioni ancora in atto

Obiettivo

- Evitare la presenza di segmenti duplicati ritardati (stesso numero di sequenza); assicurarsi che un pacchetto sia "morto" e che siano "morti" anche tutti i riscontri relativi a quel pacchetto dopo un tempo T
  (->MSL, maximum segment lifetime)

Soluzione

- Ogni host usa un clock (non sincronizzato con gli altri) che incrementa un contatore binario a passi regolari (la cadenza dipende dalla velocità della rete; a 2 Mbit/s è di 4 ms, cioè 1/(2M/8))
- Il valore del clock è usato per numerare i byte generati
- Il numero di bit del contatore deve essere maggiore o uguale del numero di bit dei numeri di sequenza (32 bit)
- Il clock deve continuare a funzionare anche quando l'host va in "crash"

Requisiti

- Ogni volta che si instaura una nuova connessione l'host trasmittente sceglie in modo pseudo-casuale numero di sequenza iniziale (ISN) di 32 bit da cui partire, ponendolo uguale al valore del suo contatore binario, sincronizzato al clock locale
  - Il numero di sequenza si sceglie in modo pseudo-casuale tra 1 e 2^32 =4.294.967.296
- Gli ottetti successivi sono numerati sequenzialmente a partire dal valore di ISN scelto durante la procedura di 3-way handshake
- Lo spazio dei numeri di sequenza (2^32) deve essere abbastanza ampio in modo che quando i numeri di sequenza si ripetono i vecchi datagrammi con lo stesso numero di sequenza siano scomparsi dalla rete

Dato che si assume che i segmenti restino in rete per non più del Maximum Segment Lifetime (MSL), si vuole che MSL sia minore del tempo di ciclo del clock

La cadenza del clock dipende dalla velocità della rete; a 2 Mbit/s è di 4 ms, cioè 1/(2M/8). Sono necessarie circa 4 ore (4ms x 2^32=4.7 ore) per compiere un ciclo completo di valori di ISN

Dato che MSL è tipicamente minore di 4.7 ore, si può ragionevolmente assumere che il valore di ISN sia unico

- anche a velocità di 100Mbit/s, il tempo di ciclo diventa 5,4 min, cioè ancora maggiore dei valori del MSL che è dell'ordine di decine di sec (120sec)

E' necessario evitare che i numeri di sequenza vengano usati (cioè assegnati a nuovi segmenti) per un tempo T dopo il loro potenziale utilizzo come ISN

Riassumendo: ogni segmento occupa uno o più numeri di sequenza nello spazio da 1 a 2^32; i numeri usati da un segmento sono considerati "occupati" finché passano MSL sec (quite time) dopo un crash per accertarsi che quei segmenti siano scomparsi dalla rete

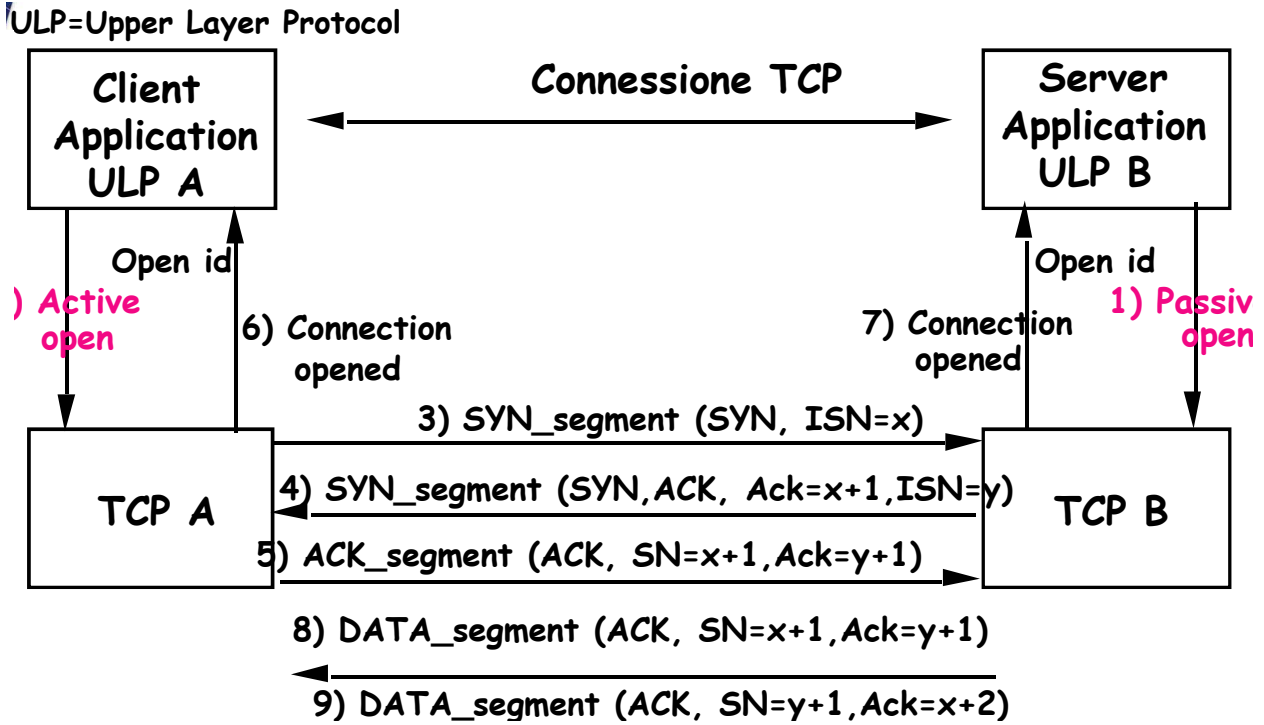
Un meccanismo di tipo three way handshake è necessario perché i numeri di sequenza non sono collegati a un clock globale nella rete

Le due entità TCP interagenti si sincronizzano scambiandosi il proprio numero di sequenza iniziale (ISN), che rappresenta il numero a partire dal quale tutti i byte trasmessi, una volta instaurata la connessione, saranno sequenzialmente numerati

La sincronizzazione consiste nell'invio, da parte di ogni sistema coinvolto nella connessione, del proprio ISN e nella ricezione di una conferma dall'altra parte in un messaggio di acknowledgment:

1) A --> B SYN (Synchronize) my sequence number is X
2) A <-- B ACK your sequence number is X
3) A <-- B SYN (Synchronize) my sequence number is Y
4) A --> B ACK your sequence number is Y

I passi 2 e 3 sono combinati in un unico messaggio, perciò la procedura è nota come three-way (o three-message) handshake


ULP=Upper Layer Protocol
Client Application ULP A
Connessione TCP
Server Application ULP B
Open id
) Active open
6) Connection opened
7) Connection opened
Open id
1) Passiv open
3) SYN_segment (SYN, ISN=x)
TCP A
4) SYN_segment (SYN,ACK, Ack=x+1,ISN=y)
TCP B
5) ACK_segment (ACK, SN=x+1,Ack=y+1)
8) DATA_segment (ACK, SN=x+1,Ack=y+1)
9) DATA_segment (ACK, SN=y+1,Ack=x+2)

1-2: Affinché dal lato server di un servizio applicativo vi sia un processo in ascolto su una porta deve avvenire l'attivazione del TCP mediante una primitiva detta di PASSIVE_OPEN, che serve a predisporre il server alla ricezione di richieste di connessione; dal lato client, quando si vuole effettivamente aprire una connessione deve essere passata al TCP locale una primitiva di ACTIVE_OPEN, che contiene il socket di destinazione

3. La prima trama inviata dal TCP del lato client è una trama di SYN (synchronize) caratterizzata dal bit di SYN posto a 1. Il numero di sequenza contenuto nel campo ISN è il numero iniziale scelto dal TCP del client per la nuova connessione

4. In risposta, il TCP del server nell'host destinatario invia una trama di SYN-ACK con i bit di SYN e ACK posti a 1. ISN contiene il numero di sequenza iniziale scelto dal TCP del server per la nuova connessione dal valore del suo contatore. Il campo Ack Number contiene il riscontro della corretta ricezione della trama precedente (valore di ISN ricevuto più 1)

5. L'apertura della connessione viene completata con l'invio di un segmento vuoto con l'ACK finale contenente il riscontro del TCP client della corretta ricezione della trama di SYN-ACK del TCP server; con il numero di sequenza riscontrato pari al numero di sequenza in trasmissione dell'host destinatario + 1

6, 8. Quindi il TCP sender comincia a trasmettere dati, dopo aver inviato la primitiva di Connection Opened al livello applicativo. Si noti che il SN dei messaggi 5 e 8 è lo stesso perché l'invio degli ACK non contribuisce a incrementare il valore di SN

7, 9. L'host destinatario comincia a trasmettere solo dopo aver ricevuto quest'ultimo terzo segmento e aver inviato al proprio livello applicativo la primitiva di Connection Opened

Quando il TCP trasmittente invia il primo segmento (SYN) per instaurare una connessione, può inserire nel segmento un'informazione sulla massima dimensione del campo dati di utente di un segmento (MSS) che è in grado di trattare

Il TCP ricevente risponde comunicando la propria MSS; con questo scambio di informazioni le due entità TCP stabiliscono il valore di MSS comune (il minore tra i due)

La scelta della MSS dipende da due fattori:

- la dimensione della memoria a disposizione delle entità TCP
- la dimensione della Maximum Transfer Unit (MTU) della sottorete a cui è connesso l'end-system; l'MTU è resa nota all'entità IP e TCP dal software che interfaccia TCP/IP alla sottorete (detto driver di rete)

Il calcolo dell'MSS avviene sottraendo all'MTU la dimensione degli header IP e TCP (in genere le opzioni non si usano e gli header hanno dimensione fissa di 20 byte)

Se il TCP trasmittente non specifica nel primo segmento (SYN) il valore di MSS, si usa il valore di default di MSS pari a 536 byte

Un valore di MSS di 536 byte, sommata ai 40 byte di header IP + TCP, dà luogo a un datagramma IP di dimensione pari a 576 byte, ovvero un datagramma che tutti i sistemi di Internet devono necessariamente accettare e gestire

Oggi è diffusa la tendenza a usare valori molto più grandi dell'MSS per aumentare l'efficienza e la velocità del collegamento (con IPv6 si arriva ad una MTU di 1280 byte)

Il rilascio di una connessione avviene mediante un meccanismo analogo di three-way handshake modificato

Una connessione TCP è bi-direzionale e può essere vista come due flussi di dati indipendenti per ciascuna direzione; nel rilascio, le 2 vie sono chiuse indipendentemente

Quando un programma applicativo non ha più dati da trasmettere ordina al TCP di chiudere la connessione "in una direzione". Il TCP finisce di trasmettere gli eventuali dati rimasti nel buffer e ne aspetta il riscontro, quindi invia un messaggio di FIN (bit di FIN settato a 1) al TCP ricevente

Il TCP ricevente invia un messaggio ACK di avvenuta ricezione al TCP trasmittente e poi avvisa il suo livello applicativo che non ha più dati da trasferirgli (questa fase può richiedere tempo per l'interazione umana)

Intanto i dati possono continuare a fluire nella direzione ancora aperta della connessione, e i riscontri devono continuare a essere inviati dal TCP che aveva chiesto la chiusura della connessione. Questo finché anche l'altra direzione della connessione verrà chiusa

Normalmente per il rilascio di una connessione servono 4 segmenti: 2 FIN e 2 ACK, però può succedere che il primo ACK e il secondo FIN siano contenuti nello stesso segmento e il numero totale di segmenti necessari si riduce così a 3

- La chiusura di una connessione avviene attraverso i seguenti passi:

1) A ------> B FIN (ho finito, non ho più dati)

2) A <------ B ACK (OK)

3) A <------ B FIN (ho finito anch'io)

4) A ------> B ACK (OK)

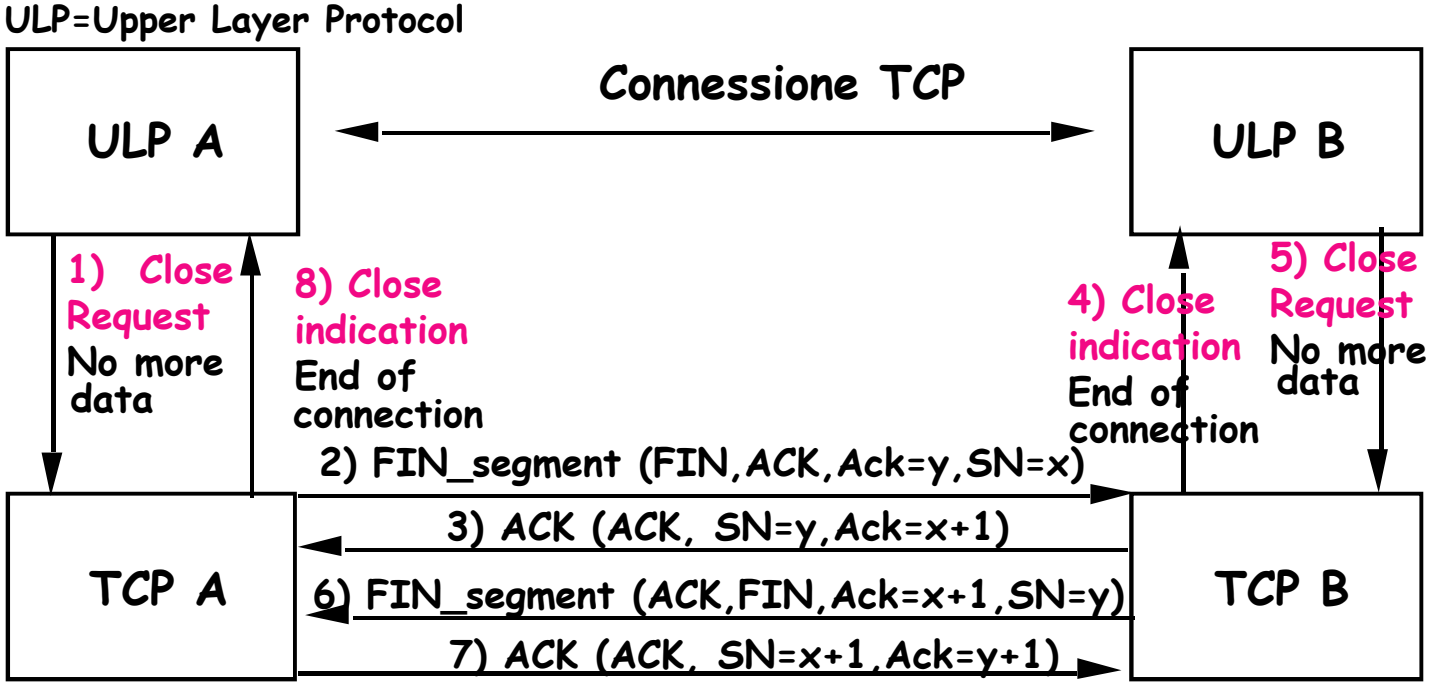
- oppure

1) A ------> B FIN (ho finito, non ho più dati)

2) A <------ B ACK (ho ancora dei dati da trasmettere)

3) A <------ B FIN (ho finito anch'io)

4) A ------> B ACK (OK)


ULP=Upper Layer Protocol
ULP A
Connessione TCP
ULP B
1) Close Request
No more data
8) Close indication
End of connection
4) Close indication
End of connection
5) Close Request
No more data
2) FIN_segment (FIN,ACK,Ack=y,SN=x)
3) ACK (ACK, SN=y,Ack=x+1)
6) FIN_segment (ACK,FIN,Ack=x+1,SN=y)
7) ACK (ACK, SN=x+1,Ack=y+1)
TCP A
TCP B

Il rilascio di una connessione è più complesso della sua instaurazione; infatti informare un'applicazione che è pervenuta una richiesta di chiudere una direzione della connessione ed ottenere una risposta potrebbe richiedere un tempo considerevole (se implica, ad es., un'interazione umana)

Il meccanismo usato non garantisce la corretta chiusura in ogni caso e il TCP rimedia con dei timeout (pari a 2 MSL). Se il pacchetto di FIN non è seguito da un ACK, dopo un tempo sufficiente la connessione full duplex viene chiusa. L'altra parte può non accorgersi della chiusura immediatamente ma prima o poi si renderà conto che nessuno risponde e a sua volta chiude.

Se una connessione non può essere chiusa secondo la procedura normale a causa di situazioni anomale, o se un programma applicativo è forzato a chiudere immediatamente una connessione, TCP prevede una procedura di reset

Tale procedura consiste nell'inviare un segmento con il bit RST "settato". Alla ricezione di tale segmento la connessione è immediatamente terminata senza scambio di ulteriori messaggi e TCP ne informa i programmi applicativi

La strategia utilizzata dal TCP per il trasferimento dati usa riscontri solo positivi (PAR, Positive Acknowledge with Retransmission) ed è basata sull'uso di finestre in trasmissione ed in ricezione

- rispetto a meccanismi come Stop & Wait che prevedono un riscontro per ogni unità dati, i meccanismi a finestra permettono di incrementare la portata del collegamento

TCP vede il flusso di dati in trasmissione come una sequenza di byte e quindi la finestra in trasmissione opera a livello di byte, invece che a livello di trama o pacchetto come in X.25

La finestra in trasmissione specifica il numero di byte che possono essere inviati senza ricevere un riscontro

Analogamente, la finestra in ricezione specifica il numero di byte che possono essere accettati fuori sequenza

In TCP, la dimensione della finestra in ricezione coincide con quella in trasmissione; quindi la dimensione di finestra specifica sia il numero di byte che un'entità TCP può trasmettere senza ricevere riscontri, sia il numero di byte che un'entità TCP può ricevere fuori sequenza

La dimensione della finestra di trasmissione non è decisa dal mittente, ma è comunicata al mittente dal destinatario (nel campo Window di 16 bit) "ogni volta" che questo emette un segmento; quindi la dimensione della finestra varia dinamicamente nel tempo

Per questo motivo la finestra è chiamata in TCP Advertised Window (finestra "pubblicizzata" dal destinatario al mittente)

All'inizio di una connessione, quando il mittente non ha ancora trasmesso dei dati, emette un numero di byte pari alla dimensione W della finestra; quindi si interrompe in attesa di un riscontro

Il TCP prevede esclusivamente riscontri (ACK) positivi

I riscontri possono essere "cumulativi", cioè il destinatario conferma la ricezione dell'ultimo byte di una sequenza di dati ricevuta "completamente" in modo corretto, ovvero senza errori e senza elementi mancanti

In particolare, il destinatario comunica al mittente il "prossimo" byte che si aspetta di ricevere (nel campo Acknowledgement Number), significando così che "tutti" i byte precedenti sono stati ricevuti correttamente, e la nuova dimensione della finestra W

Alla ricezione di un riscontro (con Ack Number pari a x) il mittente sposta in avanti la finestra ed è autorizzato a inviare, senza attesa di riscontro, i byte con numero di sequenza compreso tra x e W+x

Il sender tiene traccia del successivo numero di sequenza da usare nella variabile SND.NXT e del più vecchio numero di sequenza non riscontrato in SND.UNA; il receiver tiene traccia del successivo numero di sequenza da accettare nella variabile RCV.NXT

Quando il sender crea un segmento e lo trasmette, fa avanzare SND.NXT; quando il ricevitore accetta un segmento fa avanzare RCV.NXT e invia un riscontro; quando il sender riceve un riscontro, fa avanzare SND.UNA

Se il flusso di dati è momentaneamente "idle" e tutti i dati inviati sono stati riscontrati le 3 variabili sono uguali

La differenza tra i valori di queste variabili è una misura del ritardo nella comunicazione; la quantità di cui le variabili sono fatte avanzare è la lunghezza dei dati nel segmento

Si noti che una volta che la connessione è stabilita tutti i segmenti devono trasportare informazioni di riscontro