Protocollo TCP

giovedì 10 agosto 2023 16:30

Tramite il protocollo TCP prima di effettuare uno scambio di dati, i processi devono effettuare l'**handshake a tre vie** per stabilire la connessione, ovvero devono inviarsi reciprocamente dei *segmenti preliminari* (3) per stabilire i parametri del successivo trasferimento dei dati veri e propri (dei messaggi).
Lo stato della connessione in TCP risiede completamente nei due sistemi periferici: il protocollo va in esecuzione solo sui sistemi periferici e non negli elementi di rete intermedi *(-> i quali sono completamente ignari della connessione TCP in corso).*
TCP offre un servizio **full-duplex**, ovvero i dati possono fluire in entrami i lati *contemporaneamente*; inoltre TCP viene detta connessione **point-to-point**: ha luogo tra un singolo mittente e un singolo destinatario, non è possibile un **multicast** (più mittenti e/o più destinatari).

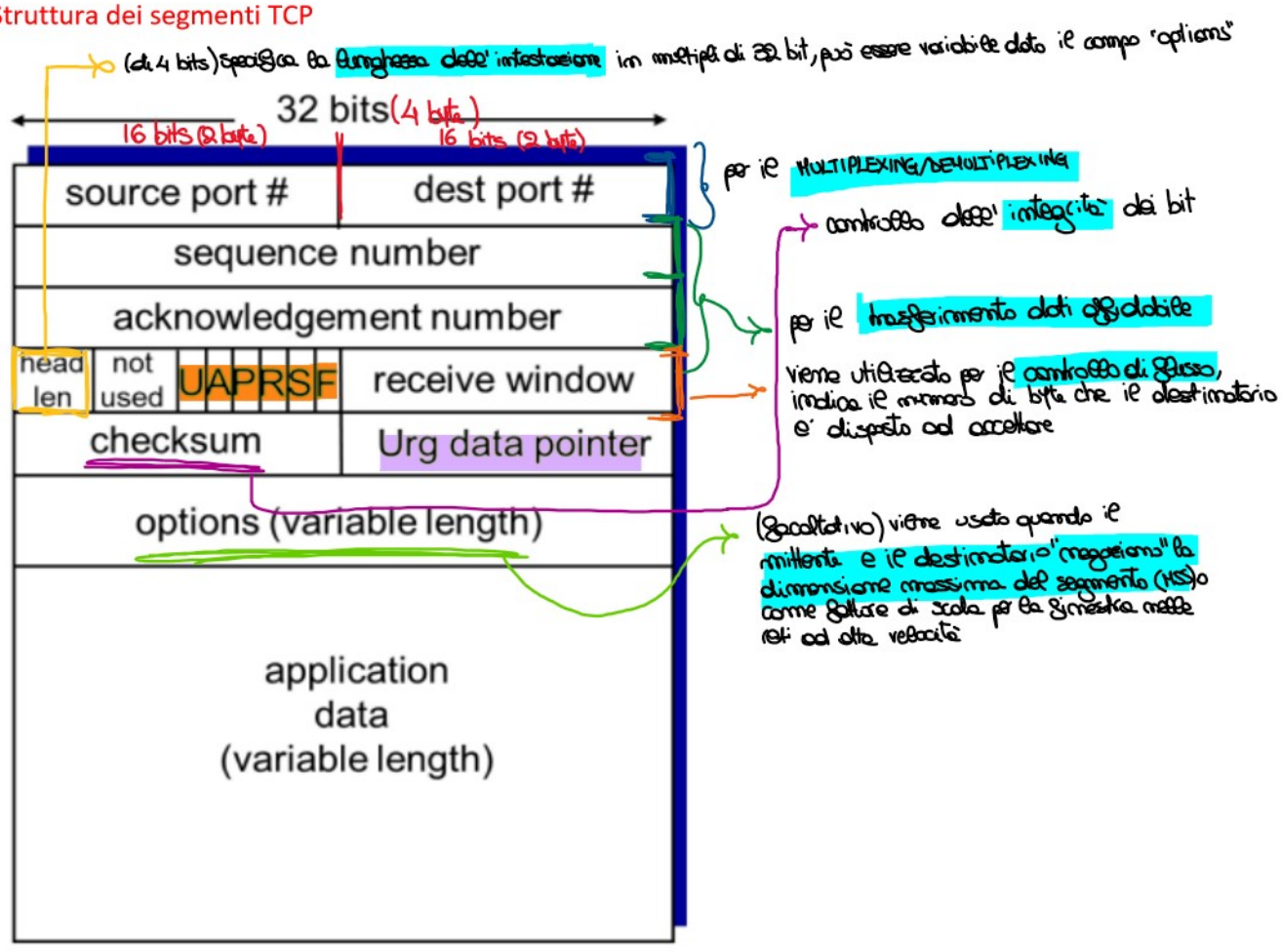
Come si instaura una connessione TCP?

Il processo (del livello applicativo) client informa il suo livello di trasporto di voler stabilire una connessione verso il dato processo server…
Per stabilire la connessione il client invia un segmento speciale al server, quest'ultimo risponde con un secondo segmento speciale, infine il client invia un ultimo terzo segmento speciale.
I primi due non trasportano payload (dati del livello applicativo) il terzo può invece trasportare informazioni utili.
Questo è l'handshaking a 3 vie! *Una volta instaurata la connessione i processi possono scambiarsi dati.*
TCP utilizza dei buffer ambo i lati (client-server), quando il client fa richiesta di inviare x dati questi verranno messi nel buffer da dove verranno poi man mano prelevati e passati al livello inferiore (di rete).
Lato server TCP riceve un segmento, i dati al suo interno vengono memorizzati nel buffer di ricezione della connessione TCP.
Fenomeno della frammentazione
Introduciamo un limite, ovvero la massima quantità di dati (del livello applicativo) che il segmento TCP (esclusa l'intestazione aggiunta durante l'incapsulamento) può trasportare: MSS (maximum segment size).
L'MSS viene impostato determinando:

1. prima l'unità trasmissiva massima MTU (maximum trasmission unit), ovvero la lunghezza del frame più grande che può essere inviato a livello di collegamento dall'host mittente locale;
2. scegliendo un MSS tale che il segmento TCP (una volta incapsulato in un datagramma IP) stia all'interno di un singolo frame a livello di collegamento.

$$MSS = MTU - size(IP\_header) - size(TCP\_header)$$

Struttura dei segmenti TCP



Generalmente il campo "options" è vuoto, in tal caso la lunghezza dell'intestazione equivale a 20 byte.
Il protocollo TCP vuole ottimizzare l'invio dei messaggi, ad esempio se il vettore IP gli offre una MTU di 1500 byte, TCP li utilizza tutti, dirgli che ho una MTU da 1500 byte significa che prima di far partire un segmento devo collezionare tutti i 1500 byte. Questo però richiede tempo, dobbiamo quindi dire a TCP di mandare subito e non ottimizzare, lo facciamo a livello di applicazione.
TCP offre un meccanismo che utilizza 6 flag (un campo di 6 bits):

- **Bit ACK**, viene usato per indicare che il valore trasportato nel campo di acknowledgement è valido;
- **Bit RST, SYN e FIN** vengono usati per impostare e chiudere la connessione;
- **Bit CWR e ECE**, vengono usati nel controllo della congestione;
- **Bit PSH**, se ha valore 1 (true) indica che il destinatario dovrà inviare immediatamente i dati al suo livello applicativo una volta ricevuti, quindi viene impedito al ricevitore di bufferizzare i dati;
- **Bit URG**, usato per indicare nel segmento la presenza di dati che il mittente a livello superiore ha marcato come "urgenti". Viene indicata la posizione dell'ultimo byte dei dati urgenti nel campo **Urg data pointer**, puntatore ai dati urgenti, di 16 bit. Quindi TCP usa il bit per informare il destinatario che non solo questi dati devono passare immediatamente al livello applicativo ma che devono essere letti subito da esso.

I dati urgenti si comportano come se viaggiassero su un canale diverso, ci sono proprio dei protocolli in cui questo avviene, dove esiste una banda differente per i dati urgenti, in TCP non c'è: per questo occorre **notificare il ricevitore** e **interrompere altre attività in corso nel momento di ricezione di tale segmento**.
L'*urgent pointer è un offset rispetto al numero di sequenza corrente*, indica qual è l'ultimo byte urgente: il ricevente deve leggere fino a quel byte per essere sicuro che tutti i dati urgenti siano stati consumati, questo perché un messaggio urgente potrebbe essere stato inviato in più segmenti (con sequence number differenti) se eccedono il valore MSS. Quando arriva un segmento urgente vengono svolte due azioni:

1. I segmenti posizionati avanti a lui, in una coda, vengono messi da parte;
2. Vengono mandati due segnali, SIGURG1 e SIGURG2 (piccolo aiutino da parte del sistema operativo), all'applicazione che viene così notificata.

Nota: TCP non ha un apposito canale perché IP (protocollo a livello di rete) non ha di per sé un canale out of bands... FTP è un protocollo out of bands.

Un esempio di urgent è il comando flush: quello che fa l'applicazione è mettere da parte tutto quello che c'è prima del flush, appena quest'ultimo arriva successivamente gli dice di scartare tutto quello che è stato tenuto da parte all'interno del buffer (azione della funzione flush).

Sequence number

Supponiamo che il flusso di dati consista in un file da 500.000 byte, la MSS sia di 1000 byte e il primo byte del flusso sia numerato con 0, TCP quindi costruisce 500 segmenti (500.000/1000) per questo flusso di dati.

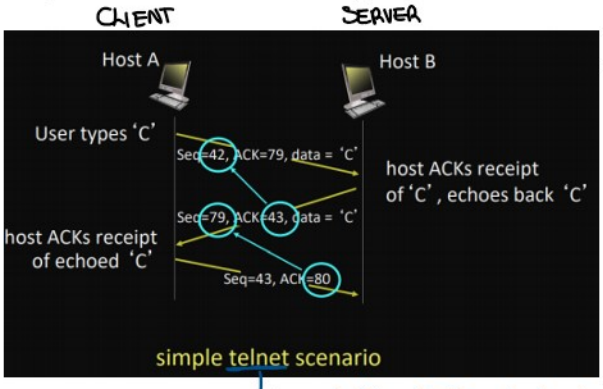
Host A che vuole mandare un flusso di dati all'Host B TCP vede i dati come un flusso continuo di byte ordinati, i numeri di sequenza si applicano al flusso di byte trasmessi e non alla serie di segmenti trasmessi

In generale i partecipanti alle connessioni TCP scelgono a caso un numero di sequenza iniziale, in modo tale da minimizzare la possibilità che un segmento ancora presente nella rete, per via di una connessione precedente tra due host già terminata, venga interpretato in modo errato, ad esempio come un segmento valido in una connessione successiva tra gli stessi due host.

Numeri di acknowledgment

Il numero di acknowledgment, che l'host A scrive nei propri segmenti, è il numero di sequenza del byte successivo che l'Host A attende dall'Host B.
Sappiamo che TCP è full-duplex e questo comporta la gestione di due flussi: il flusso da A verso B e il flusso da B verso A. Ad esempio abbiamo che l'host A ha ricevuto dall'host B i byte da 0 a 535, quindi, quando l'host A invia un segmento all'host B questo avrà come numero di ack 536, lo avrà fino a che non riceverà in ordine il byte 536, può ricevere nel mentre anche i byte successivi che non vengono buttati ma non riceveranno l'ack fino a che non arriveranno in ordine in quel punto: si dice che **TCP è un protocollo che offre acknowledgment comulativi.**

CLIENT SERVER



Numeri sequenziali iniziali:
- 42 client
- 79 server

Lo protocollo a livello applicativo impiegato per il login remoto che usa TCP

simple telnet scenario

↳ protocollo e client applicativo impiegato per il login remoto che usa TCP