Protocollo TCP - 4

lunedì 21 agosto 2023 11:12

Distinguiamo due principali orientamenti al controllo della congestione:

1. **Controllo di congestione end-to-end**: il livello network non fornisce alcun supporto esplicito al livello di trasporto per il controllo di congestione, quindi questo meccanismo risiede nei sistemi periferici osservando il comportamento della rete;
2. **Controllo di congestione gestito dalla rete:** in questo approccio i router della rete forniscono un feedback esplicito al mittente sullo stato di congestione della rete, come: **(*)** viene trasmesso un avviso diretto dal router al mittente tramite un pacchetto di strozzatura (chokepacket) che dice al mittente "ehi bell'hostone sono congestionato", **(**)** il router imposta un campo in uno dei pacchetti che fluiscono dal mittente al destinatario per indicare la congestione. Alla ricezione di un pacchetto marcato, il destinatario notifica il mittente l'indicazione di congestione *(Nota: questa operazione richiede almeno un RTT)*.

Additive increase, Multiplicative decrease

Il concetto è quello di imporre a ciascun mittente un **limite al tasso d'invio** sulla propria connessione in funzione della congestione di rete percepita… *Come?* ***Per regolare la velocità d'invio dei dati sulla propria connessione, al mittente basta modificare il valore del campo cwnd.***

Il meccanismo di controllo della congestione TCP fa tenere traccia agli estremi dello stato di connessione, tenuto in una variabile aggiuntiva: la **finestra di congestione** (congestion window), indicata con **cwnd**, che pone un vincolo alla velocità di immissione di traffico sulla rete da parte del mittente. La velocità di trasmissione (sinonimo di tasso d'invio) si indica:

cwnd/RTT (byte/s)

La quantità di dati che non hanno ancora ricevuto acknowledgment non può eccedere il minimo tra cwnd e rwnd:

LastByteSent - LastByteAcked ≤ min{cwnd, rwnd}

La finestra di congestione viene decisa tra gli altri dati durante l'avvio della connessione TCP e viene adattata dinamicamente durante la trasmissione dei dati in base alle risposte del destinatario e altri segnali ricevuti.

Se i mittenti TCP trasmettessero tutti insieme troppo velocemente, potrebbero congestionare la rete (portandola alla congestione del terzo scenario…); tuttavia, se invece i mittenti TCP trasmettessero troppo lentamente, potrebbero sottoutilizzare l'ampiezza di banda della rete, in questo caso i mittenti potrebbero trasmettere più velocemente senza congestionare la rete. Bisogna quindi variare la velocità di trasmissione proporzionalmente alla congestione end-to-end identificata…

Algoritmo di controllo di congestione TCP

Principi guida sulla base dei quali è formulato:

- **Un segmento perso implica congestione, quindi i tassi di trasmissione del mittente TCP dovrebbero essere decrementati (trasmetto più lentamente) quando un segmento viene perso;**
- **Un acknowledgment indica che la rete sta consegnando i segmenti del mittente al ricevente e quindi il tasso di trasmissione del mittente può essere aumentato quando arriva un acknowledgment NON duplicato;**

} *si dice che TCP è* ***auto-temporizzato***

- **Rilevamento della larghezza di banda**: la strategia di TCP, per regolare il tasso di trasmissione, è di incrementarlo in risposta all'arrivo degli acknowledgemnt fino a quando non si verifica un "**evento di perdita**", appena quest'ultimo si verifica viene decrementata la velocità.

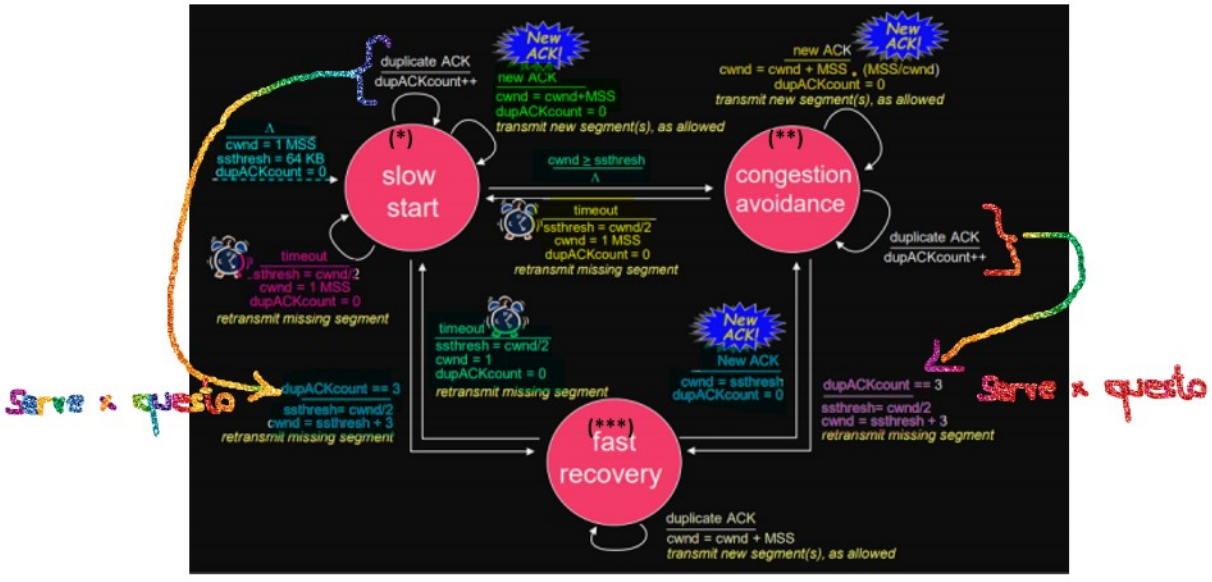
Nota: gli acknowledgment e gli eventi perdita sono ***segnali impliciti****, di conseguenza ogni mittente TCP agisce sulla base di* ***informazioni locali in modo asincrono rispetto agli altri.***

Le tre fasi principali quindi sono:

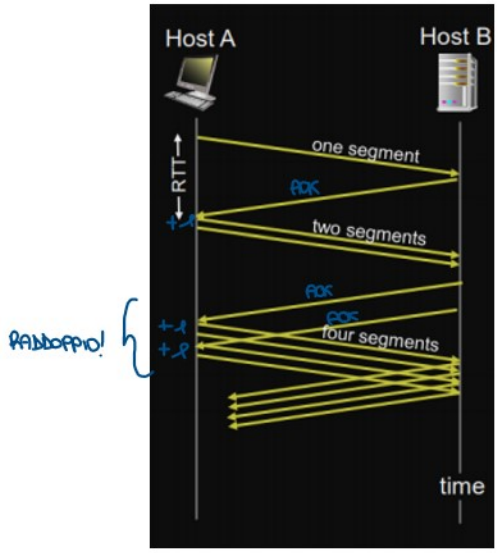
1. **Slow start;**
2. **Congestion avoidance;**
3. **Fast recovery;**

} TCP Tahoe

} TCP Reno



(*) Fase iniziale: quando si stabilisce una connessione, il valore cwnd viene in genere inizializzato a 1 MSS, comporta una velocità di invio iniziale di circa 1 MSS/RTT. Quindi il valore di cwnd parte da 1 MSS e viene incrementato di 1 ogni qualvolta che un segmento trasmesso riceve l'ACK, per capire visivamente succede questo:



Questo ha come effetto il raddoppio della velocità di trasmissione a ogni RTT.
Crescita esponenziale della velocità di trasmissione in questa fase

La crescita esponenziale della velocità di trasmissione termina se ricadiamo nei seguenti casi:

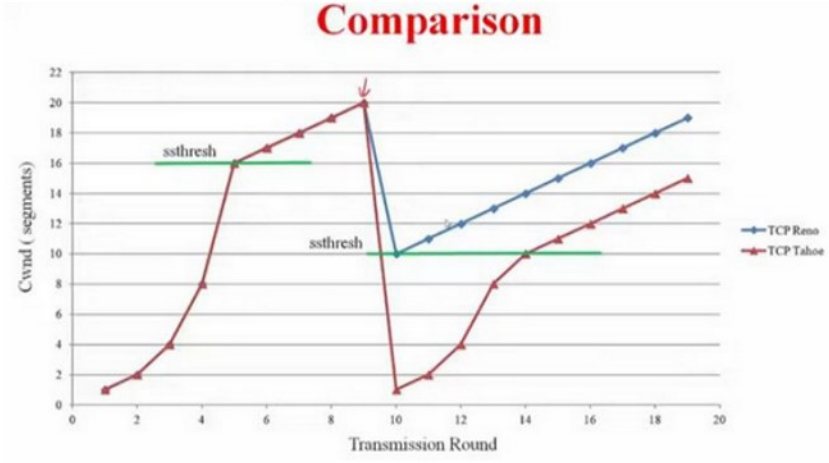
- Ev. Perdita, sintomo di congestione, indicato dall'avvenimento di timeout,in questo caso il mittente TCP pone il valore di cwnd ad 1 e inizia nuovamente il processo di slow start. Inoltre viene inizializzato il valore di una seconda variabile di stato chiamata **ssthresh (slow start treshold)** a **cwnd/2**, ovvero metà del valore che aveva la finestra di congestione quando viene rilevata una congestione di rete;
- Quando il **valore ssthresh** è già stato impostato precedentemente *(al punto precedente)* e **cwnd è pari o supera questo valore, la fase di slow start termina e TCP entra nella fase di congestion avoidance**;
- Ev. Perdita indicato da 3 ACK duplicati, in questo caso TCP opera una fast retransmit (ritrasmissione rapida) ed entra nella fase di fast recovery.

(**)**Congestion avoidance**: quando TCP entra nello stato di congestion avoidance, il valore di cwnd è parti al valore di ssthresh, ovvero cwnd/2.
In questa fase TCP adotta un approccio più conservativo: incrementa i 1 MSS (al posto che raddoppiare) ad ogni RTT, **l'incremento è infatti lineare**.
La crescita lineare durante la congestion avoidance termina se ricadiamo nei seguenti casi:

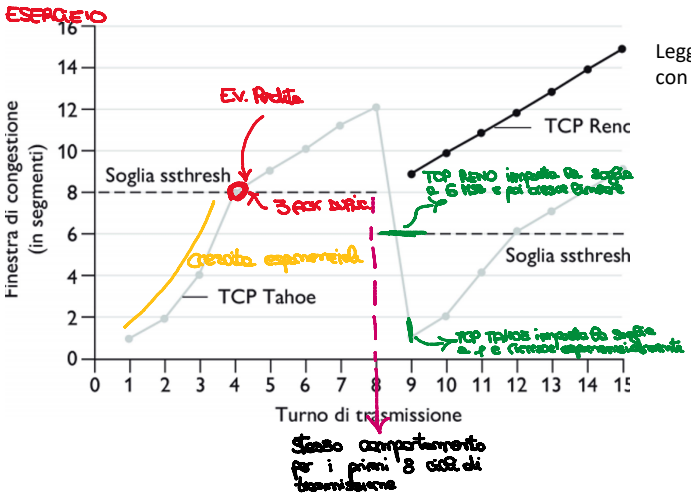
- Ev. Perdita indicato da un timeout, si comporta come slow start;
- Ev. Perdita indicato da 3 ACK duplicati:
  - **TCP Reno**, in questo caso la rete continua a consegnare segmenti dal mittente al ricevente, per cui la risposta di TCP, a questo tipo di evento, è quella di dimezzare il valore di cwnd:
    cwnd/2 + 3 MSS *(per tenere conto 3 segmenti duplicati ricevuti).*
    Successivamente imposta il valore di ssthresh pari a metà del valore di cwnd al momento del ricevimento dei tre ACK duplicati; infine TCP entra nello stati di fast recovery;
  - **TCP Tahoe**, la variabile ssthresh viene impostata a cwnd/2 e riparto con cwnd = 1 (MSS).

(***)Fast recovery: durante questa fase, il valore di cwnd è incrementato di 1 MSS per ogni ACK duplicato ricevuto relativamente al segmento perso (+3 degli ACK duplicati nella fase di entrata).
Quando successivamente arriva un ACK per uno dei segmenti persi TCP riduce il valore di cwnd e torna nella fase di congestion avoidance. Se invece si verifica un timeout vi è una transizione dallo stato di fast recovery a quello di slow start.
Fast recovery è un componente raccomandato ma non obbligatorio di TCP.

TCP RENO VS TCP TAHOE



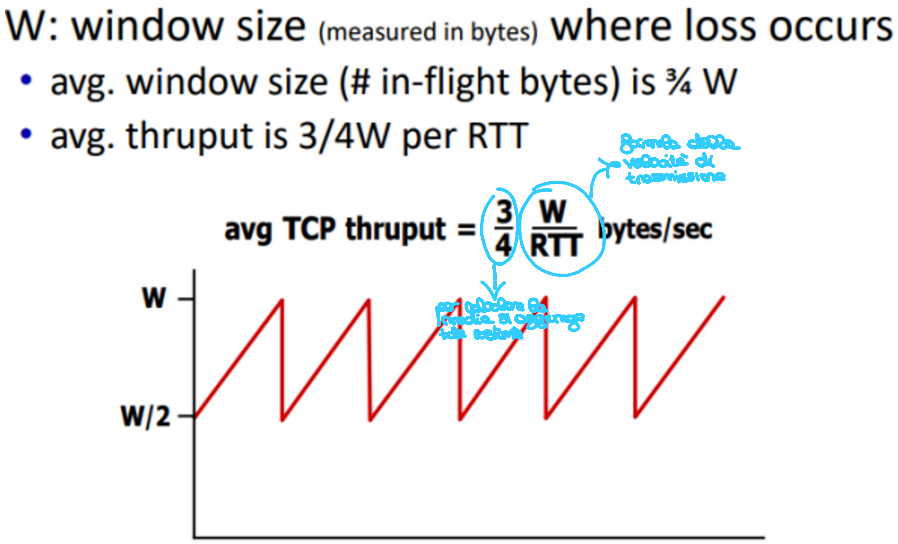
ESERCIZIO



Legge che mette in relazione il troughput della connessione con il tasso di perdita:

TCP throughput = (1.22·MSS)/(RTT· √L)

Il controllo della congestione AIMD da luogo ad un comportamento del valore cwnd a dente di sega:



la velocità di trasimissione varia tra *W/(2·RTT)* e *W /RTT* ; in pratica il primo è nel momento dopo *Ev. Perdita* e W viene quindi dimezzato, il secondo (l'estremo superiore) è il momento esatto in cui W è aumentato più che poteva fino alla congestione. È un ciclo che si ripete in continuazione!