RETI DI CALCOLATORI – prova di verifica intermedia del 31/10/2016

Per essere ammessi alla prova orale è necessario ottenere una valutazione sufficiente della prima parte.

Prima parte (21 punti)

Q1. Due router, A e B, sono posti agli estremi del tunnel del Monte Bianco, la cui lunghezza è di 11,6 Km, e comunicano direttamente mediante un cavo di quella lunghezza, e in cui la velocità di propagazione è $2 \cdot 10^8$m/s. Inoltre, la velocità di trasmissione R dei due router è 16 Mbps. Al tempo t, il router A inizia ad inviare un pacchetto a B, e B finisce di ricevere tutto il pacchetto al tempo t+22808 microsecondi. Indicare – giustificando la risposta – qual'è la lunghezza L del pacchetto.

Q2. Un host deve risolvere il nome simbolico engineering.vanderbilt.edu, il cui indirizzo IP **non** è noto al suo resolver. Supponendo che la gerarchia dei name server abbia 4 livelli, indicare – giustificando la risposta - il numero di messaggi DNS che, nel caso peggiore, circoleranno in internet per risolvere tale nome simbolico, se (a) si utilizza ad ogni livello una risoluzione ricorsiva, oppure (b) una risoluzione iterativa.

Q3. Indicare – giustificando la risposta - quante connessioni TCP vengono stabilite tra un client e un server FTP, se il client chiede al server la lista dei file disponibili, quindi scarica tre di quei file, ed infine ne richiede la cancellazione.

Q4. Un client C chiede la pagina web www.acme.com/home/products.html al server B di www.acme.com con una GET che è contenuta in un segmento TCP lungo X byte. Indicare -giustificando la risposta – i valori dei campi *sequence number, ack number,* e dei flags *A* ed *S,* in ciascuno dei segmenti che C e B si scambiano per aprire la connessione nell'ipotesi che l'ack finale dell'apertura della connessione sia inviato in piggybacking assieme alla GET. Si supponga che in B ed in C rwnd sia molto grande, che non scada alcun timeout, che non ci siano errori di trasmissione, che nessun segmento vada perduto, e che il numero di sequenza iniziale di C sia 1111 e quello di B sia 2222.

Q5. Consideriamo lo scenario precedente, e supponiamo che C abbia ricevuto, al tempo t, da B, 3 dei 4 segmenti di Y byte contenenti la pagina richiesta, ed invii (al tempo t) al server B una richiesta di chiusura di connessione. Il server riceve tale richiesta prima di inviare il quarto ed ultimo segmento della pagina, ed effettua una chiusura di tipo half-close. Indicare -giustificando la risposta – i valori dei campi *sequence number, ack number,* e dei flags *A* ed *F* in ciascuno dei segmenti che C e B si scambiano dal tempo t in poi. Si supponga che in B ed in C rwnd sia molto grande, che non scada alcun timeout, che nessun segmento vada perduto, e che non ci siano errori di trasmissione.

Q6. Si consideri il seguente scenario TCP in cui, per semplicità, non sono mostrati i segmenti inviati o *re*-inviati dal sender TCP al receiver TCP, che si suppone essere quelli necessari per avere i riscontri descritti di seguito. Al tempo t0 il TCP di un host A ha una connessione già stabilita, per la quale ha 4 segmenti full sized in volo e nessun nuovo dato da spedire, il primo byte dei segmenti in volo è il byte Y, ssthresh=6.5 MSS, cwnd=5 MSS. Inoltre, non ha ricevuto nessun riscontro duplicato. Tra il tempo t0 e il tempo t1 riceve 7 riscontri: i primi due non duplicati e con acknumber uguale a Y+1 MSS per il primo, e Y+3 MSS per il secondo. I seguenti 4 riscontri sono tutti duplicati, ed infine, al tempo t1, riceve un settimo riscontro con acknumber uguale a Y+4 MSS. Si supponga che non scatti alcun timeout tra t0 e t1. Indicare, per ciascun riscontro ricevuto, lo *stato* del TCP e i valori di *ssthresh* e *cwnd*, giustificando la risposta.

Q7. Si consideri lo scenario del quesito precedente, modificato come descritto sotto. Anche in questo caso, per semplicità, non sono mostrati i segmenti *re*-inviati dal sender TCP al receiver TCP (ma sono indicati i segmenti inviati e contenenti nuovi dati, vedere sotto), che si suppone essere quelli necessari per avere i riscontri descritti di seguito. Questa volta, al tempo t0 il TCP di A ha anche 2 MSS di nuovi dati da spedire. Inoltre, si supponga che subito dopo il secondo e subito dopo il quinto riscontro ricevuto, A invii due segmenti contenenti nuovi dati, il primo di 1 MSS (subito dopo il secondo riscontro) e il secondo di 0.5 MSS (subito dopo il quinto riscontro). Indicare – giustificando la risposta – il valore di *rwnd* ricevuto in ciascuno dei 7 riscontri.

Seconda parte (9 punti)

E1. Si consideri una variante del protocollo *Go Back n* in cui il *receiver* è dotato di una finestra ampia N segmenti dove memorizza segmenti non corrotti ricevuti anche fuori ordine, purché il loro numero di sequenza sia compreso nella finestra. Quando il receiver riceve corretto il segmento più vecchio atteso, passa al processo di livello superiore il messaggio contenuto in quel segmento e, se li ha ricevuti, anche quelli dei segmenti successivi presenti nella finestra, purché in ordine, ed invia al sender un riscontro cumulativo per tali segmenti. Per le altre azioni, il receiver si comporta come nel protocollo standard. Descrivere, mediante un automa a stati finiti che utilizza pseudocodice (come nell'automa [GBN] del materiale didattico) e non con descrizione delle attività a parole, il comportamento del receiver della variante del protocollo Go Back n sopra descritta.

Traccia della soluzione

Q1. Il ritardo di propagazione è di $11{,}6x10^3/2x10^8$=58 microsecondi. Quindi, T_{trasm} + 58 = 22808 microsecondi. Pertanto, T_{trasm}=22750 microsecondi. Siccome L=T_{trasm}xR, si ha che L=$22750x10^{-6}x16x10^6$=22750x16=364000 bit = 45500 byte.

Q2. Nel caso **peggiore**:

(a) bisogna percorrere tutto l'albero dei name server, dal name server locale fino ad un root server, e poi da questo al name server autoritativo, che nel caso peggiore è il name server locale per engineering.vanderbilt.edu, per poi tornare indietro: in totale sono 6+6 messaggi a cui vanno aggiunti i 2 messaggi dal resolver al name server locale del client, e viceversa: quindi 14.

(b) anche ora i messaggi saranno 14, perchè i name server coinvolti saranno 6, nel caso pessimo, più i 2 messaggi dal resolver al name server locale del client, e viceversa.

Q3. Le connessioni saranno cinque:
- una connessione di controllo persistente per lo scambio di comandi e risposte FTP,
- una connessione dati (non persistente) per il trasferimento della lista dei file e
- tre connessioni dati (non persistenti) per il trasferimento dei tre file.

Q4. C→B: (SYN) seqnum=1111, acknum=????, A=F, S=T; **B→C: (SYN+ACK)** seqnum=2222, acknum= 1112, A=T, S=T; **C→B: (ACK del SYN+ACK in piggybacking alla GET)** seqnum=1112, acknum= 2223, A=T, S=F; **commento:** *questo segmento è lungo X byte. Dopo, B invia a C 3 segmenti lunghi Y byte ciascuno, facendo incrementare il suo seqnum fino a 2223+2Y (l'ultimo dei 3 segmenti conterrà i byte da 2223+2Y fino a 2223+3Y-1).*

Q5. Si parte dalla situazione descritta alla fine del commento precedente. In questa soluzione, l'ack del FIN di C viene mandato a se stante. Poteva anche essere in piggyback con l'ultimo segmento inviato da B (nel qual caso, non si doveva decrementare il campo seqnum). ***C→B (segmento FIN):*** *seqnum=1112+X, acknum= 2223+3Y, A=T, F=T;* ***B→C (ACK del FIN):*** *seqnum=2223+3Y-1, acknum= 1112+X+1, F=F, A=T;* ***B→C (ultimo segmento di Y byte):*** *seqnum=2223+3Y, acknum=1112+X+1, A=T, F=F;* ***B→C (segmento FIN):*** *seqnum=2223+4Y, acknum= 1112+X+1, A=T, F=T;* ***C→B (ACK finale):*** *seqnum=1112+X+1, acknum= 2223+4Y+1, A=T, F=F;*

Q6. *Primo riscontro*: stato = slow start; cwnd = 6; ssthresh = 6.5 ; *secondo riscontro*: stato = congestion avoidance; cwnd = 7; ssthresh = 6.5 ; *terzo e quarto riscontro*: come il secondo riscontro; *quinto riscontro*: stato = fast recovery; cwnd = 6.5; ssthresh = 3.5; *sesto riscontro:* stato = fast recovery; cwnd = 7.5; ssthresh = 3.5; *settimo riscontro*: stato = congestion avoidance; cwnd = 3.5; ssthresh = 3.5

Q7. Nota: se si hanno in volo X MSS di dati quando arriva un riscontro con valore di rcwnd minore di X, questo non può avere effetto sui dati inviati (non possiamo mica farceli restituire dalla rete!). Quindi in alcuni casi, la soluzione esatta prevede un "<=" invece di un "=" . *Primo riscontro*: rcwnd <=3MSS; *secondo riscontro*: rcwnd = 2MSS; *terzo e quarto riscontro*: rcwnd <= 2MSS; *quinto riscontro*: rcwnd = 2.5MSS; *sesto riscontro*: rcwnd <= 2.5MSS; *settimo riscontro*: rcwnd<=1.5MSS

E1. /* Utilizziamo un vettore rcvSgmt per memorizzare i (dati dei) segmenti ricevuti e un vettore isReceivedSgmt per ricordare quali segmenti sono già stati ricevuti. */