

RETI DI CALCOLATORI – secondo appello - a.a. 2011/2012

Per ottenere una valutazione sufficiente dell'intera prova è necessario ottenere una valutazione sufficiente della prima parte.

Prima parte (10 punti)

Q1. Supponiamo che un router A trasmetta un pacchetto p sul collegamento con un router B e che il ritardo di trasmissione d_{trasm} di p sia il doppio del ritardo di propagazione d_{prop} sul collegamento. Indicare –giustificando la risposta– se è possibile o no che metà dei bit di p siano immessi sul collegamento prima che il primo bit di p arrivi a B.

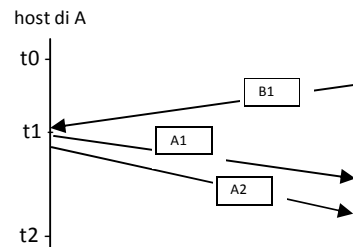
Q2. Supponiamo che l'utente `taz@warner.com` stia utilizzando un host dietro NAT e cerchi di inviare un messaggio di posta elettronica all'utente `goofy@disney.com`, ma digiti erroneamente come indirizzo del destinatario l'indirizzo inesistente `gofy@disney.com`. Considerare il primo pacchetto che verrà spedito contenente un messaggio di errore relativo a ciò e indicare –giustificando la risposta– quale è il protocollo a cui appartiene tale messaggio di errore e i valori dei campi contenenti informazioni di *addressing* nei preamboli IP e TCP di tale pacchetto.

Q3. Indicare –giustificando la risposta– in che modo dovrebbe essere modificato il protocollo ARP per poter funzionare correttamente su LAN che si trovano dietro NAT.

Q4. Determinare il tempo minimo necessario, in termini di RTT tra coppie di nodi, affinché un nodo appartenente a una rete Gnutella riceva il primo messaggio *QueryHit* in risposta a una query che ha generato.

Seconda parte

E1 (6 punti). Supponiamo che al tempo t_0 il TCP di un processo applicativo A abbia 2 MSS di dati in volo, che il valore di *sendBase* sia X , che la dimensione della sua finestra di congestione sia $2MSS$, che si trovi nello stato di *slow start* e che abbia altri 2 MSS di nuovi dati da spedire. Supponiamo inoltre che, quando riceve il segmento B1 che è un riscontro che non contiene dati, il TCP di A invii due segmenti (A1 e A2) contenenti rispettivamente 1MSS e $2/3MSS$ di nuovi dati. Supponendo che nell'intervallo $[t_0, t_2]$ il TCP di A riceva e invii solo i segmenti indicati nella figura a lato e che in tale intervallo non scada nessun timeout, indicare – giustificando la risposta:



- i possibili valori di AckNum e di RcvWin contenuti in B1,
- i possibili valori di SeqNum in A1 e in A2,
- i possibili valori di CongWin, SendBase e NextSeqNum in t2.

Per semplicità supponiamo che tutti i segmenti contenenti dati scambiati da A e B (ad eccezione di A2) contengano 1 MSS di dati.

E2 (6 punti). Supponiamo che un router IPv6 contenga una tabella $nextHop_{v6}$ in cui mantiene le informazioni necessarie per inoltrare pacchetti IPv6. In particolare, $nextHop_{v6}(D) = \langle V, type, nHops, tunnelEnd, tunnelLength \rangle$ dove $nHops$ è il numero complessivo di hop per raggiungere la destinazione D e:

- se $nextHop_{v6}(D) = \langle V, 'v4', nHops, tunnelEnd, tunnelLength \rangle$ allora V è il vicino IPv4 a cui inoltrare i pacchetti IPv6 destinati a D opportunamente incapsulati in pacchetti IPv4 destinati al nodo $tunnelEnd$ e $tunnelLength$ è la lunghezza (in hop) del tunnel, e
- se $nextHop_{v6}(D) = \langle V, 'v6', nHops, NULL, tunnelLength \rangle$ allora V è il vicino IPv6 a cui inoltrare i pacchetti destinati a D e $tunnelLength$ è la lunghezza (in hop) dell'eventuale tunnel presente nella rotta.

Supponendo che un router IPv6 riceva advertisement del tipo $\langle D, Path \rangle$ dove D è una destinazione e $Path$ è un vettore di $Path.length()$ coppie $\langle router, type \rangle$, scrivere il codice con cui tale router aggiorna la sua tabella $nextHop_{v6}$ quando riceve un advertisement da un suo vicino IPv4, in modo da privilegiare rotte che minimizzino la lunghezza del tunnel eventualmente necessario. Per semplicità assumiamo che ogni rotta tra due router IPv6 contenga al più un tunnel.

E3 (4 punti). Decrivere con un automa a stati finiti il comportamento di un nodo ALOHA che, ricevuta una richiesta `sendRequest(destination, data)`, tenta al più tre volte di trasmettere i dati ricevuti, rilevando il completamento con successo della trasmissione con l'evento `frameSent()` ed eventuali collisioni con l'evento `collision()`.

E4 (4 punti). Supponiamo che A possieda un certificato affidabile di B e che utilizzi il protocollo illustrato di lato (dove n è un *nonce* generato da A e n' è un *nonce* generato da B) per inviare a B una chiave di sessione K_s dopo avere autenticato B. Indicare –giustificando la risposta– se un intruso T può venire a conoscenza di K_s senza che né A né B se ne accorgano.

