RETI DI CALCOLATORI – sesto appello - a.a. 2010/2011

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 host A debba trasferire 2000 byte di dati utilizzando TCP a un host B direttamente raggiungibile tramite un collegamento con frequenza di trasmissione di 1 Mbps e con ritardo di propagazione di 2 millisecondi. Indicare – giustificando la risposta – se tale trasferimento può essere completato in 25 millisecondi se il valore di MSS è 500 byte.

Q2. Indicare – giustificando la risposta – se un pacchetto IP che trasporta il comando SMTP "`MAIL FROM <alice@crepes.fr>`" può contenere come indirizzo IP di destinazione l'indirizzo IP del mailserver del mittente del messaggio anziché l'indirizzo IP del mailserver di uno dei destinatari del messaggio, anche nel caso in cui nessuno dei destinatari abbia la sua mailbox sotto crepes.fr.

Q3. Supponiamo che al tempo t il valore di *EstimatedRTT* sia n e che le prime due nuove misurazioni *SampleRTT* che TCP effettua dopo t siano pari a n+2ε e a n-ε, rispettivamente, con ε≠0. Indicare –giustificando la risposta– quanto deve pesare TCP ogni nuova misurazione affinché il valore di *EstimatedRTT* non sia variato dopo avere tenuto conto delle due nuove misurazioni.

Q4. Supponiamo che un router B riceva dal collegamento con un router A un pacchetto IP destinato a un host H (≠B) contenente il comando FTP "`STOR settembre.gif`" e supponiamo che nessun bit del pacchetto sia stato alterato nel trasferimento da A a B ad eccezione di un bit del comando FTP. Indicare – giustificando la risposta – con quale probabilità B rileverà tale errore.

Seconda parte

E1 (6 punti). Descrivere con un automa a stati finiti il comportamento lato mittente di una estensione del protocollo GBN che prevede due modalità (*plain* ed *encoded*) per il trasferimento dei dati. Inizialmente il protocollo trasmette dati in modalità *plain*. Dopo avere ricevuto una esplicita richiesta *encode* dall'applicazione passa a trasferire i dati in modalità *encoded* utilizzando opportuni meccanismi crittografici (che assumiamo siano stati già concordati col lato ricevente) per garantire la segretezza dei dati trasmessi. Il mittente GBN inizia a trasferire i dati in modalità *encoded* solo dopo avere completato con successo il trasferimento dei dati trasmessi in modalità *plain*. Per semplicità ignorare la possibilità che i pacchetti possano essere corrotti durante il trasferimento.

E2 (6 punti). Consideriamo un'implementazione del protocollo link state che utilizza un vettore D di dimensione N per memorizzare le distanze calcolate, un vettore K di dimensione N per tenere traccia dell'insieme S delle destinazioni la cui distanza minima è già stata determinata, un vettore P di dimensione N per tenere traccia dell'immediato predecessore di una destinazione z nel cammino dalla sorgente a z, e un array C di dimensione NxN in cui sono memorizzati i costi dei collegamenti (C[x,y]=NULL in assenza di un collegamento tra x e y).

(a) Scrivere il frammento di codice con cui l'algoritmo aggiorna il vettore D dopo avere selezionato una nuova destinazione w da aggiungere all'insieme S.
(b) Scrivere il frammento di codice con cui l'algoritmo determina – una volta calcolate le distanze minime da tutte le destinazioni – il next hop a cui inoltrare i pacchetti destinati a una destinazione.

Commentare opportunamente i frammenti di codice.

E3 (4 punti). Sia R un router che utilizza il protocollo distance vector e supponiamo che il vettore delle distanze di R contenga $D_R(F)=2x$, $D_R(G)=x$, $D_R(H)=x$ e $D_R(Z)=3x$, dove F, G e H sono gli unici router con cui R ha un collegamento diretto. Supponiamo inoltre che gli ultimi advertisement che R ha ricevuto da F, G e H contenessero rispettivamente $D_F(Z)=x$, $D_G(Z)=2x$ e $D_H(Z)=\infty$. Indicare se è possibile o meno che il solo cambio del costo di uno dei collegamenti di R porti R ad aggiornare il valore di $D_R(Z)$ a 4x. Giustificare la risposta fornita mostrando un esempio nel caso in cui ciò sia possibile oppure spiegando in modo chiaro perché ciò non può avvenire.

E4 (4 punti). Supponiamo che solo due nodi di una rete Ethernet debbano trasmettere dati e che, dopo avere entrambi riscontrato che il canale è inattivo, essi inizino simultaneamente a trasmettere ciascuno per la prima volta un frame. Determinare – giustificando la risposta – quale è la probabilità che entrambi i nodi riescano a trasmettere con successo il proprio frame soffrendo al più solo una ulteriore collisione.

Traccia della soluzione (sesto appello a.a. 2010/2011)

Q1. No, ciò non è possibile in quanto per trasferire da A a B solo i dati di 4 segmenti full-sized saranno necessari $4 \cdot \left(\frac{4 \cdot 10^3}{10^6} + \frac{2}{10^3}\right) s = 24ms$, a cui tuttavia deve essere aggiunto il tempo necessario per trasferire i bit dei preamboli di tali segmenti del primo segmento dell'handshake, oltre ai tempi di attesa dei riscontri imposti dal controllo di congestione.

Q2. Si, quando il messaggio viene trasferito dall'host del mittente al mailserver del mittente.

Q3. Dopo la prima misurazione *EstimatedRTT=(1-α)n+α(n+2ε)*, mentre dopo la seconda *EstimatedRTT=(1-α)(n+2αε)+α(n-ε)*. Affinché $n+\alpha\varepsilon-2\alpha^2\varepsilon=n$ deve quindi valere $\alpha\varepsilon(1-2\alpha)=0$ ovvero, dato che $\alpha\neq0$ e $\varepsilon\neq0$: $\alpha=1/2$.

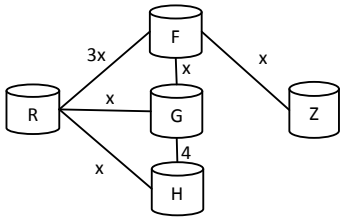
Q4. B non potrà rilevare tale errore in quanto il controllo del checksum IP riguarda solo l'intestazione del datagram.

E2. Assumiamo che K[x]==0 sse x∉ S.

(a)
```
for (x=0;x<N;x++)                    //per ogni destinazione x
      if (K[x]==0 && C[x,w]!=NULL)     //non in S ma direttamente collegata a w
        then if (D[x]>D[w]+C[w,x])
                then { D[x] = D[w]+C[w,x];
                       P[x] = w;
                }
```

(b)
```
nextHop(dest) = {
    r=dest;
    while P[r]≠sorgente do r=P[r];
    return r;
}
```

E3. Si, è possibile. Per esempio nella rete a lato abbiamo che, una volta raggiunto lo stato di quiescenza, il vettore delle distanze di R conterrà $D_R(F)=2x$, $D_R(G)=x$, $D_R(H)=x$ e $D_R(Z)=3x$, e gli ultimi advertisement che R ha ricevuto da F, G e H contenevano rispettivamente $D_F(Z)=x$, $D_G(Z)=2x$ e $D_H(Z)=\infty$. Se a questo punto cade il collegamento RG, R aggiorna la sua distanza da Z in $D_R(Z)=4x$.



.

E1. L'automa ha tre stati (S0, S1 e S2). Lo stato S0 e' identico allo stato descritto nel libro. Ed e' lo stato della sessione plain, S2 e' lo stato della sessione encoded. S1 e' lo stato intermedio del passaggio di sessione Descrivo solo le novita'.

Entro nello stato iniziale S0 con la transizione

Evento PLAIN
Aziione: base = 1, NextSN = 1

Transizione S0 -→ S1

Evento: Encoded && base < NestSN
Azione: empty

Lo stato S1 e' simile come struttura a quella di S0. L'idea che questo sato deve completare l'invio dei dati della sessione plain.

Transioni modificate (S1 →S1)

Evento: rdt_send(data)
Azione: refuse data

Evento: rdt_rcv(rcv_pkt) & rcv_pkt ! corrupted &
getAckNumver(rcv_pkt) + 1 < NextSN
Azione: Base = getAckNumver(rcv_pkt) + 1 StartTimer

Transizione S1 →S2
Evento: rdt_rcv(rcv_pkt) & rcv_pkt ! corrupted &
getAckNumver(rcv_pkt) + 1 = NextSN
Azione: Base = getAckNumver(rcv_pkt) + 1 StopTimer

Lo stato S2 e' identico allo sato S0 (ovvero quello tradizionale)

L'unica differenza sta nella costruzione del pckt da inviare. Assumento che K sia la chiave di sessione e M sia il mac condiviso, H la funzione di hashing La ceazione del pkt diventa
Mkpkt(K(data+NextSN), H(Data+NextSN+M))