RETI DI CALCOLATORI – terzo 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. Indicare –giustificando la risposta– quanti pacchetti IP può ricevere in meno nel caso ottimo un cliente HTTP se specifica il campo `If-Modified-Since` nella richiesta di un file di dimensione (K+0.5) · M, dove K è un intero e M è il valore di MSS nella connessione stabilita col server.

Q2. Supponiamo che al tempo t il TCP di un host A si trovi in congestion avoidance e che il valore della sua variabile CongWin sia 5/3 MSS. Supponiamo che in t il TCP di A riceva un riscontro R non corrotto di tutti i dati che aveva "in volo" su una connessione c e che tale riscontro contenga il valore 8 MSS nel campo RcvWin. Indicare –giustificando la risposta– se dopo avere ricevuto R il TCP di A può o meno inviare 5/2 MSS di nuovi dati sulla connessione c.

Q3. Sia V un router che utilizza il protocollo distance vector e supponiamo che il vettore delle distanze di V contenga $D_V(E)=n$, $D_V(F)=n+1$, $D_V(G)=n+2$ e $D_V(Z)=2n$, dove E, F e G sono gli unici router con cui V ha un collegamento diretto. Supponiamo che V riceva il vettore delle distanze di G contenente $D_G(Z)=n-3$. Indicare –giustificando la risposta– quale è il valore di $D_V(Z)$ determinato da V dopo aver ricevuto il vettore di G.

Q4. Indicare –giustificando la risposta– quale proprietà di SHA-1 viene sfruttata per la generazione di un MAC.

Seconda parte

E1 (6 punti).

(a) Estendere l'automa a stati finiti che descrive il comportamento di un receiver GBN per modellare il caso in cui il receiver disponga di un buffer di ricezione in cui memorizzare solo 1 pacchetto ricevuto non in ordine.

(b) Determinare –giustificando la risposta– quale rapporto deve esistere tra la dimensione dello spazio dei numeri di sequenza e la dimensione della finestra.

E2 (6 punti). Descrivere con uno pseudo-codice il comportamento di un router che riceve un advertisement LSA generato da un router r e contenente numero di sequenza n e costo p del collegamento tra i nodi x e y.

E3 (4 punti). Descrivere con un automa a stati finiti il modo in cui un nodo di una rete token-passing può partecipare alla rigenerazione del token in caso di necessità. Specificare in modo chiaro il significato degli eventi e delle azioni impiegati e utilizzare due stati finali per rappresentare i due diversi modi in cui può terminare con successo la rigenerazione del token. Indicare sotto quali ipotesi la soluzione descritta risolve il problema.

E4 (4 punti). Supponendo che ogni nodo possa fallire indipendentemente dagli altri con probabilità q, indicare –giustificando la risposta– con quale probabilità Chord riesce a garantire che ogni nodo conosca il proprio successore in presenza di fallimenti.

Traccia della soluzione

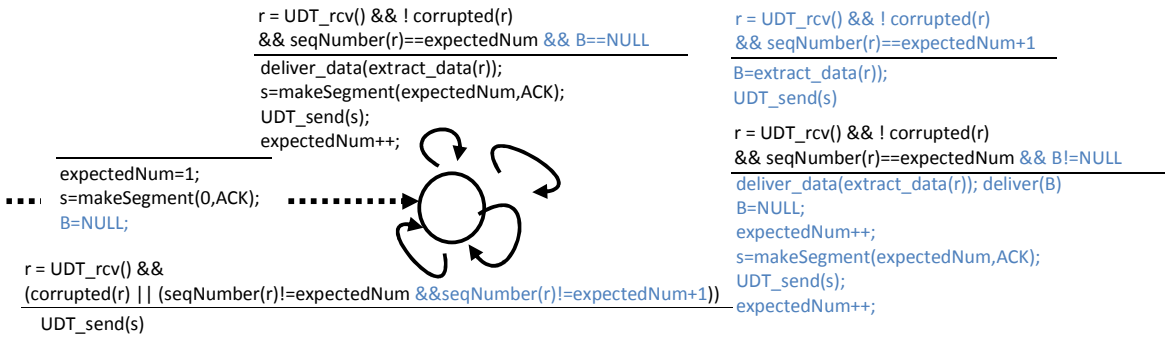
R1. Il caso ottimo è quello in cui il cliente possiede una versione aggiornata del file e non si verificano perdite, corruzioni né frammentazioni dei pacchetti. In tale caso il cliente riceverà la risposta HTTP ("`304 Not Modified`") in 1 solo pacchetto IP. Se invece il cliente non possiede una versione aggiornata del file, il server dovrà inviare almeno (K+1) pacchetti IP per trasmettere il file (oltre a eventuali altri pacchetti dovuti a ritrasmissioni). Nel caso ottimo il cliente riceverà quindi almeno K pacchetti in meno.

R2. No, non può perché, ricevuto R, aggiorna la dimensione di CongWin a CongWin + (MSS/Congwin x MSS) = 34/15 MSS < 5/2 MSS.

R3. V determinerà il nuovo valore di $D_V(Z)$ applicando la formula di Bellman-Ford: $D_V(Z)=\min\{C(V,E)+ D_E(Z), C(V,F)+D_F(Z), C(V,G)+n-3\}$ dove C(X,Y) indica il costo del collegamento XY tra il nodo X e il nodo Y. Se i costi dei collegamenti VE, VF e VG non sono cambiati, V modificherà quindi il valore di $D_V(Z)$ solo se C(V,G)<n+3.

R4. SHA-1 viene utilizzata per codificare un messaggio m con una chiave di autenticazione s in un hash crittografico SHA-1(m+s) detto MAC. La proprietà di SHA-1 sfruttata per la generazione di un MAC è che dovrebbe essere computazionalmente impossibile trovare due messaggi x e y diversi, tali che SHA-1(x)=SHA-1(y).

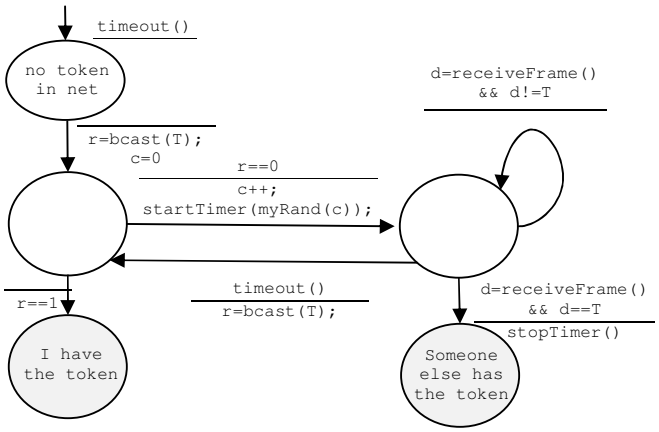
E1.



(b) Sia N la dimensione della finestra. Nel caso pessimo il mittente mittente considera in volo i segmenti [X,X+N-1] mentre il ricevente li considera invece ricevuti e attende quindi di ricevere X+N oppure X+N+1. La dimensione dello spazio dei numeri di sequenza deve quindi essere almeno N+2.

E2.
```
/* Indichiamo con lastFrom[r] il numero di sequenza più alto contenuto negli advertisement
generati da r finora ricevuti dal router (lastFrom[r]=NULL se non ha ancora ricevuto nessun
advertisement generato da r) e con A.age() il valore del campo "età" contenuto nell'advertisement A.
Indichiamo inoltre con C l'array contenente i costi dei collegamenti.
Quando il router riceve da un vicino v un advertisement A generato da r contenente numero di
sequenza n e costo p del collegamento tra i nodi x e y: */
if (lastFrom[r]==NULL || lastFrom[r]<n)
    then {lastFrom[r]=n;
          if A.age()<maxAge then {"incrementa age di A";
                                  forEach vicino w!=v do send(w,A); //inoltra advertsiment
                                 }
          if C[x,y]!=p then {C[x,y]=p;  C[y,x]=p;                  //aggiorna array costi collegam.
                             nextHops=recomputeLS(C);              //ricalcola cammini minimi
                            }
    }
```

E3. Supponiamo che la necessità di rigenerare il token sia rilevata con un evento `timeout()` (causato dallo scadere di un timer avviato subito dopo avere inoltrato il token). Il token può essere rigenerato semplicemente prevedendo che ogni nodo cerchi di generarlo inviando un messaggio di broadcast e risolvendo eventuali collisioni ad esempio con l'algoritmo di regressione esponenziale di CSMA/CD. Supponiamo di avere a disposizione le operazioni `int bcast(data)` (per spedire un frame in broadcast, ottenendo 0 se si è generata collisione e 1 altrimenti), `void startTimer(int)` e `void stopTimer()` (per avviare e arrestare il timer), `int myRand(int)` (per ottenere un intero casuale nell'intervallo $[0,2^x-1]$) e `data receiveFrame()` (per ricevere un frame). Sia `T` la codifica del token.



E4. Ogni nodo Chord mantiene una lista di successori di dimensione r, contenente i primi r successori del nodo. La probabilità che tutti gli r successori di un nodo falliscano simultaneamente è q^r e quindi la probabilità con cui Chord garantisce che ogni nodo conosca il proprio successore in presenza di fallimenti è $(1- q^r)$.