RETI DI CALCOLATORI – prova scritta dell'11/07/2017

Per essere ammessi alla prova orale è necessario ottenere una valutazione sufficiente della prima parte e una votazione totale di 15 o superiore.

Prima parte (15 punti)

Q1. Indicare gli indirizzi IP ottenuti dal protocollo DNS di un client, e indicare, per ciascun server DNS utilizzato, e per ciascun nome simbolico sotto riportato, i resource record utilizzati per ottenerli (nell'ordine del loro utilizzo), relativamente ai due nomi simbolici www.scholar.google.com e www.myjeeves.ask.com, supponendo di avere a disposizione i seguenti server DNS (nei resource record il campo TTL non è mostrato perché ininfluente per il quesito):

–*server DNS locale al client, di indirizzo 209.105.12.4, con (tra gli altri) i resource record*:
(mailserver.ask.com, 205.17.23.23, A, IN)
(ask.com, dns.ask.com, NS, IN)
(google.com, dns.google.com, NS, IN)
(dns.ask.com, 205.17.3.98, A, IN)
(dns.google.com, 206.11.15.1, A, IN)

–*server DNS di indirizzo 205.17.3.98, con (tra gli altri) i seguenti resource record*:
(ask.com, mailserver.ask.com, MX, IN)
(www.ask.com, w3.ask.com, CNAME, IN)
(www.myjeeves.ask.com, w3.myjeeves.ask.com, CNAME, IN)
(w3.myjeeves.ask.com, web.myjeeves.ask.com, CNAME, IN)
(w3.ask.com, web.ask.com, CNAME, IN)
(mailserver.ask.com, 205.17.200.8, A, IN)
(web.ask.com, 205.17.3.12, A, IN)
(proxy.ask.com, 205.17.3.11, A, IN)
(web.myjeeves.ask.com, 205.17.3.10, A, IN)

–*server DNS di indirizzo 206.11.15.1 con (tra gli altri) i resource record*:
(google.com, dns.google.com, NS,IN)
(w3.scholar.google.com, web.scholar.google.com, CNAME, IN)
(www.scholar.google.com, 206.11.17.7, A, IN)
(www2.google.com, 206.11.25.4, A, IN)
(www.sports.google.com, 206.11.15.5, A, IN)
(web.news.google.com, 206.11.67.67, A, IN)
(dns.sports.google.com, 113.66.101.1, A, IN)
(proxy.google.com, 206.11.25.4, A, IN)
(web.google.com, 206.11.25.5, A, IN)

Q2. Al tempo t, il TCP di un host si trova nello stato congestion avoidance e cwnd=13/6. Qual'è il valore di ssthresh al tempo t? Giustificare la risposta.

Q3. Un sistema autonomo AS che contiene 8000 hosts e quattro server (web, email, ftp e DNS), adotta al suo interno indirizzi privati del gruppo 192.168.0.0/16, ed usa per i suoi router NAT tutti gli indirizzi pubblici del gruppo 114.113.87.24/29. Qual'è il numero massimo di connessioni TCP uscenti da AS e relative agli host che possono essere contemporaneamente attive nell'AS? Giustificare la risposta.

Q4. L'host M invia un datagram D IPv6 al server S (che usa IPv6) lungo il percorso mostrato sotto, in cui R_1, R_2, R_3, R_4 sono router IPv6 e tra R_2 ed R_3 il percorso attraversa una parte in cui sono presenti solamente router IPv4, e pertanto utilizza la tecnica del tunneling in tale parte. Nel tunnel, D subisce una frammentazione. Indicare – giustificando la risposta – chi, tra M, S, R_1, R_2, R_3, R_4 effettua il riassemblaggio di D.

Q5. Quali chiavi e per quali funzioni devono usare il mittente ed il destinatario per utilizzare il protocollo PGP nella sua versione che garantisce la massima sicurezza (relativamente a PGP)?

La seconda parte è sull'altra faccia del foglio.

Seconda parte (15 punti)

E1 (7 punti). I computer di una rete locale WiFi utilizzano due thread distinti, il thread mittente ed il thread destinatario, per realizzare il protocollo CSMA/CA. Descrivere, con uno pseudocodice, le azioni svolte dal thread **destinatario** del computer IO, limitatamente alla ricezione di un frame da un altro computer. Nello pseudocodice, si utilizzino (tra le altre) le procedure/funzioni:
- *receive(F)* per ricevere un frame F dalla rete (bloccante);
- send*(F)* per inviare un frame F nella rete;
- *settimer(T)* per settare il timer con il valore T e farlo partire;
- *wait (E)* per aspettare l'evento E;
- *check(F)* che restituisce il valore *true* se e solo se il frame F è corretto.

E2 (8 punti). Si consideri una estensione del protocollo **Go Back N** in cui, oltre alle consuete azioni, si vuole realizzare anche il **controllo del flusso** tra mittente e destinatario, utilizzando la tecnica usata in **TCP**. Per questo, ogni riscontro ricevuto ha anche il campo *rwnd* che contiene il valore della finestra di ricezione inviato dal receiver. Questo valore viene restituito dalla function *getrwnd(A)*, dove A è un riscontro. La finestra di invio ha dimensione N ed il valore iniziale di rwnd è *vi*.

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 **sender** della variante del protocollo Go Back N sopra descritta.

Traccia della soluzione

Q1. Per www.scholar.google.com: nel server locale al client si guardano nell'ordine la terza e la quinta riga. Da questa si deduce che bisogna andare al server 206.11.15.1. Qui si guarda la terza riga che contiene l'indirizzo IP cercato: 206.11.17.7
Per www.myjeeves.ask.com: nel server locale al client si guardano nell'ordine le righe seconda e quarta. Questa porta al server 205.17.3.98. Qui si recupera l'informazione cercata guardando la terza, la quarta e la nona ed ultima riga da cui si deduce l'indirizzo IP desiderato: 205.17.3.10

Q2. Siccome il valore di ssthresh non cambia in congestion avoidance, cerchiamo il valore precedente di cwnd: 13/6=cwnd+1/cwnd e quindi $cwnd^2$-13/6cwnd+1=0 le cui radici sono 3/2 e 2/3. 2/3 non è possibile perché cwnd non è mai minore di 1. Consideriamo cwnd=3/2. Se si ripete il ragionamento sopra, si ottiene un numero immaginario. Quindi, quello è il valore di ingresso in congestion avoidance. Non si può essere passati in congestion avoidance da slow start, perché in slow start cwnd è sempre intero (inizia da 1 ed è incrementato di 1). Pertanto, si è passati da slow start in fast recovery quando cwnd era uguale a 3 (e quindi si è posto ssthresh=cwnd/2), e poi da qui a congestion avoidance, ponendo cwnd=ssthresh: la risposta è sstresh=3/2.

Q3. Il numero massimo di connessioni TCP relative agli host può essere limitato dagli host o dai router NAT. I numeri di porta utilizzabili da ogni entità (host o router) è al massimo 65536 (tutti i numeri di porta) - 1024 (i numeri noti), cioè 64512. Dato che gli host sono 8000 (con almeno 8000 indirizzi IP), il limite è dato dai router NAT. In particolare, ogni indirizzo IP pubblico dei router NAT deve avere assegnata una porta. Gli indirizzi pubblici sono 8, ed il numero delle porte per ciascuno di essi è 65536 (tutti i numeri di porta) - 1024 (i numeri noti), cioè 64512. Pertanto, in totale si possono avere al massimo 64512x8=516096 connessioni TCP uscenti da AS e relative agli host.

Q4. All'inizio del tunnel (cioè in R_2), viene premessa a D una intestazione IPv4. All'atto della frammentazione, i dati per poter riassemblare D vengono inseriti in tale intestazione. Questa intestazione viene tolta da R_3 alla fine del tunnel, e nel resto del percorso, D viaggia con la sola intestazione IPv6. Quindi, S non potrebbe riassemblare D perché gli mancherebbero i dati per tale azione. Pertanto, il riassemblaggio deve avvenire in R_3, che è la destinazione del tunnel, e cioè all'uscita del tunnel.

Q5. Per avere il massimo livello di sicurezza che PGP permette, il mittente deve usare la sua chiave privata (per la firma digitale), poi la chiave di sessione condivisa per la riservatezza, chiave di sessione che viene inviata dal mittente crittografata con la chiave pubblica del destinatario, e la chiave di sessione, che viene usata per crittografare il tutto. Il destinatario, a sua volta, usa la sua chiave privata per decrittare la chiave di sessione, poi decritta il messaggio usando la chiave di sessione, ed infine verifica la firma digitale del mittente usando la chiave pubblica di quest'ultimo.

E1.

```
while (true)
{
receive (F); //per ricevere il frame F
if ((F.tipo==RTS)&&(F.destinatario==IO)&&(check(F)) //se è un RTS destinato ad IO ed è corretto,  allora deve eseguire il protocollo,
                                                    altrimenti lo ignora
        {  settimer (SIFS);
           CTS.destinatario==F.mittente;
           CTS.lunghezza==F.lunghezza; // per permettere a tutti i nodi che ricevono il CTS di settare il NAV
           wait (timeout);
           send(CTS);
           receive(F);
           if ((check(F))&&(F.destinatario==IO))  // se è il frame atteso ed è corretto (se corrotto ignora il tutto)
                 {  settimer (SIFS);
                    ACK.destinatario==F.mittente; // prepara l'ACK e lo invia
                    wait (timeout);
                    send(ACK);
                 }
        }
}
```

E2. La struttura dell'automa rimane quella del materiale didattico. Di seguito, si indicano solamente le azioni relative ad ogni evento: le modifiche rispetto all'automa standard sono evidenziate in grassetto. Bisogna distinguere tra pacchetti in volo e pacchetti nuovi da spedire, come in TCP. La variabile lastpack punta all'ultimo pacchetto inserito nella finestra, mentre nexseqnum-1 indica l'ultimo pacchetto inviato (a causa del valore di rwnd).

Inizializzazione

---

base=1
nextseqnum=1
lastpack=0
rwnd=vi

RDT_send(data)

---

if (**lastpack**<base+N-**1**)
{
sndSgm[**lastpack**]=makesegment(**lastpack**,data)
lastpack++
if (nexseqnum-base)<rwnd
{ UDT_send(sndSgm[nextseqnum])
if base == nextseqnum {start_timer}
nextseqnum++
}
}
else refuse_data(data)

timeout()

UDT_send(sndSgm[base])
UDT_send(sndSgm[base+1])
….
UDT_send(sndSgm[nextseqnum-1])
start_timer()

rcvSgm=UDT_rcv()&&!corrupted(rcvSgm)&&isACKinWindow(rcvSgm)

---

rwnd=getrwnd(rcvSgm)
base=getacknum(rcvSgm)
while (((nextseqnum-base) < rwnd)&&(lastpack >= nextseqnum))
{ UDT_send(sndSgm[nextseqnum])
nextseqnum++
}
if (base==nextseqnum) stop_timer() else start_timer()

rcvSgm=UDT_rcv()&&(corrupted(rcvSgm)|| ! isACKinWindow(rcvSgm))

Λ