Introducción a las Redes de Computador{ae}s y Comunicación de Datos

Examen – 30 de julio de 2011

(ref: eirc1107.odt)

Instrucciones

- **Indique su nombre completo y número de cédula en cada hoja.**
- **Numere todas las hojas e indique la cantidad total de hojas que entrega en la primera.**
- **Escriba las hojas de un solo lado y utilice una caligrafía claramente legible.**
- **Comience cada pregunta teórica y cada ejercicio en una hoja nueva.**
- **Sólo se contestarán dudas de letra. No se aceptarán dudas de ningún tipo los últimos 30 minutos del examen.**
- **El examen es individual y sin material. Apague su celular mientras este en el salón del examen.**
- **Es obligatorio responder correctamente al menos 15 puntos en las preguntas teóricas.**
- **El puntaje mínimo de aprobación es de 60 puntos.**
- **Para todos los ejercicios, si es necesario, puede suponer que dispone de los tipos de datos básicos (p.ej. lista, cola, archivo, string, etc.) y sus funciones asociadas (ej: tail(lista), crear(archivo), concatenar(string, string).**
- **Duración: 3 horas. Culminadas las 3 horas el alumno no podrá modificar las hojas a entregar de ninguna forma.**

Preguntas Teóricas

Pregunta 1 (7 puntos)

a) Describa someramente el principio de funcionamiento del protocolo FTP.
b) Justifique por qué se dice que FTP envía la información de control en modalidad *fuera de banda* (*out-of-band*).

Respuesta Pregunta 1

Capítulo 2 del libro. Capa de Aplicación. Diapos 36..38 del material de referencia del teórico.

a) FTP: File Transfer Protocol. Protocolo que permite la transferencia de archivo/s desde/hacia un host remoto. Se ajusta al modelo cliente/servidor. El cliente es quién inicia la conexión para que ocurra la transferencia. El servidor es el host remoto. Está documentado en la RFC 959. Utiliza TCP como protocolo de transporte. El handshake que ocurre ente el cliente y el servidor permite que primero se establezca una conexión de control (utilizada para autorización y comandos) desde el cliente al puerto 21 del servidor. Cuando quien oficia de servidor recibe un comando de transferencia de archivo, establece otra conexión con el cliente; ésta será con el puerto 20 del servidor.
   No se requiere la descripción del modo pasivo.
b) Porque los comandos son enviados por una conexión separada de la conexión de datos.

Pregunta 2 (10 puntos)

a) ¿Qué es un Sistema Autónomo (*Autonomous System*, AS)?
b) Explique los conceptos *Interior Gateway Protocols* (IGP) y *Exterior Gateway Protocols* (EGP) vinculándolos con el concepto explicado en la parte a) y brinde un ejemplo de cada uno.
c) Describa brevemente el uso que hace BGP (*Border Gateway Protocol*) del atributo AS-PATH y realice un ejemplo sencillo que lo muestre en acción.

Respuesta Pregunta 2

Capítulo 4 del libro. Capa de Aplicación. Diapos 96, 97, 104, 118..122 del material de referencia del teórico.

a) AS es un concepto vinculado al enrutamiento jerárquico. Si podemos decir que Internet es una "red de redes" y se pretende que en la gestión de ellas mantenga un orden, debemos identificar redes con autonomía administrativa para el conjunto de dispositivos que la componen, y en particular para los protocolos de enrutamiento configurados en ella. Por lo tanto podemos decir que un AS es *una agrupación de routers con una administración común*.
b) *Interior Gateway Protocols* (IGP): protocolos de enrutamiento que "corren" dentro de un AS (*routing intra-AS*); ejemplos: RIP y OSPF. *Exterior Gateway Protocols* (EGP): protocolos de enrutamiento que "corren" entre ASs (*routing inter-AS*); ejemplo: BGP.
c) El atributo AS-PATH contiene la lista de ASs que ha atravesado la publicación de un prefijo. Cuando un prefijo es pasado a un AS, éste agrega su *ASNumber* al atributo. Este atributo tiene dos usos

fundamentales: evitado de *loops* en las publicaciones (si un AS se encuentra a sí mismo en el AS-PATH) y, si es tenido en cuenta en el proceso de selección de ruta y cuando se lo considere (existen otros atributos), se seleccionará aquella que tenga el AS-PATH más corto. Ejemplo: Figura 4.41 del libro. Suponiendo que el prefijo 138.16.64/24 es publicado por el AS2 al AS1, si AS1 lo publica al AS3, lo hará con AS-PATH valiendo AS2 AS1.

Pregunta 3 (7 puntos)

Suponga que el *host* A envía dos segmentos TCP seguidos al *host* B a través de una conexión TCP. El primer segmento tiene el número de secuencia 90 y el segundo tiene el número de secuencia 110.

a) ¿Cuántos datos hay en el primer segmento?
b) Suponga que el primer segmento se pierde pero el segundo llega a B. En el paquete de reconocimiento (ACK) que el *host* B envía al *host* A, ¿cuál será el número de reconocimiento?

Pregunta 4 (5 puntos)

Si todos los enlaces de Internet tuvieran que proporcionar un servicio de entrega de tramas fiable, ¿sería el servicio de entrega fiable de TCP redundante?¿Por qué?

Respuesta Pregunta 4

No, no seria redundante ya que aun así se pueden perder paquetes por congestión en las colas de los routers o pueden llegar en desorden a causa de diversidad de caminos en un flujo.

Pregunta 5 (11 puntos)

a) Describa en detalle los componentes del cabezal IPv6, comparándolo con el cabezal IPv4
http://eva.fing.edu.uy/mod/resource/view.php?id=6215 págs 10, 11 y 12

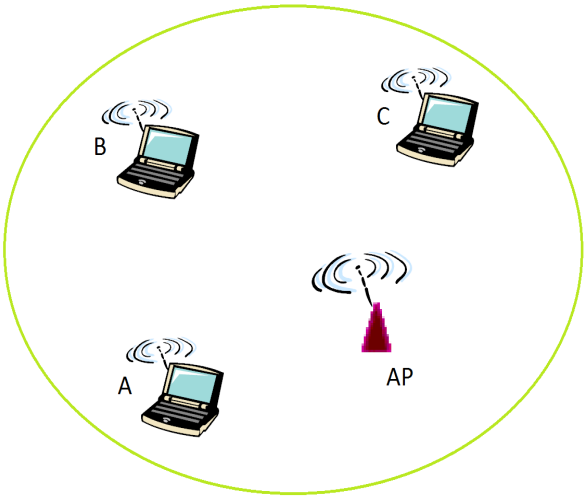
b) Describa por qué, a nivel de routers, el procesamiento de paquetes IPv6 es más simple que el de IPv4.
Los cabezales de largo fijo son más simples de parsear y procesar. La forma en que se incluyen las opciones, en base a extension-headers, hace que su procesamiento sea más eficiente. Al no realizar fragmentación, esta tarea es removida de los routers.

c) Explique qué es y por qué es necesario implementar PathMTU Discovery en IPv6 y no en IPv4
PathMTU Discovery es un algoritmo que sirve para determinar el MTU de un camino. Con ello podemos determinar el valor adecuado para transmitir un mensaje y que no sea fragmentado camino al destino. En IPv6 es obligatorio que los los endpoints de una conexión (por ejemplo, TCP) lo implementen, pues, en IPv6 los routers no realizan fragmentación. Por otra parte, no es necesario que los endpoints en IPv4 lo implementen, ya que los routers pueden resolver la fragmentación.

Problemas Prácticos

Problema 1 (30 puntos)

Considere una red inalámbrica que utiliza un protocolo del tipo 802.11, como la de la figura. Se desea implementar un mecanismo que evite colisiones por reserva de canal (similar al visto en el teórico).



Puede que más de una estación intente reservar el canal a la vez, por lo que puede haber colisiones que provoquen el fracaso de la reserva. Se considera que la reserva fracasó por esta causa luego de **`MAX_WAIT_RESERVE`** *ms* sin recibir respuesta.

Para estimar el tiempo de reserva del canal, utilice el largo de los datos dividido **`BYTES_PER_SEC`**, considere que cada una de las tramas de control involucradas tienen un largo de **`FRAME_CONTROL_BYTES`** y que los tiempos entre tramas toman un valor fijo **`SIFS`**.

Todos las estaciones cuentan con las siguientes primitivas:

- **`send_frame(char* data)`** – Envía una trama de datos al canal.
- **`receive_frame(int timeout): char*`** – Bloquea la estación hasta que se reciban datos del canal, o hasta que transcurran **`timeout`** milisegundos, en este caso devuelve **`NULL`**. El tiempo remanente de espera queda en la variable global **`timeout_remaining_time`**.
- **`wait(int time)`** – Bloquea la estación por una cantidad **`time`** de milisegundos.
- **`rand(): int`** – Retorna un entero entre 0 y **`RAND_MAX`**. Considerar que lo calcula de manera randómica.

Los emisores cuentan además con las siguientes primitivas:

- **`get_data_to_send(): char*`** - Bloquea al emisor hasta que hayan datos para enviar desde la capa superior. En ese momento se desbloquea y retorna un **`char*`** con los datos.
- **`get_my_mac(): MAC`** – Obtiene la dirección MAC del emisor.

El receptor cuenta además con la siguiente primitiva:

- **`process_data(char* data)`** – Envía los datos recibidos de la estación a la capa superior para que sean procesados. Si estos datos estaban encapsulados en una trama, será necesario sacarlos de la trama primero, para que lo que se envíe a esta función sea lo mismo que el emisor obtuvo por medio de **`get_data_to_send`**.

Suponga que:

- Las tramas que se obtengan con la función **`receive_frame`** estarán completas y vendrán de solo una de las estaciones.
- En caso de colisión, el tiempo de espera antes de volver a intentar transmitir es aleatorio.
- Las estaciones A, B, y C solo ejecutan el programa emisor del protocolo, mientras que el *Access Point* solo ejecuta el programa receptor.

Se pide:

a) Proponga un formato de trama para el intercambio entre los nodos y el AP que permita cumplir con el protocolo.
b) Implemente en un lenguaje de alto nivel el programa que debe estar ejecutándose en el emisor.
c) Implemente en un lenguaje de alto nivel el programa que debe estar ejecutándose en el receptor.

Solución Problema 1

a)

```
int TYPE_RTS 1;
int TYPE_CTS 2;
int TYPE_DATA 3;
int TYPE_ACK 4;

struct frame {
   int type;
   int duration; // en milisegundos
   MAC node;
   char* data;
}
```

b)

Emisor:

```
void sender() {
   while (true) {
      char* data = get_data_to_send();
      int data_send_time = length(data) / BYTES_PER_SEC;
      int control_send_time =  FRAME_CONTROL_BYTES / BYTES_PER_SEC;
      bool end = false;
      while (!end) {
         frame request_frame = new frame();
         request_frame.type = TYPE_RTS;
         request_frame.duration = data_send_time + 2 x control_send_time + 3 x SIFS;
         request_frame.node = get_my_mac();
         send_frame((char*)request_frame);
         char* result = receive_frame(MAX_WAIT_RESERVE);
         if (result != NULL) {
            frame result_frame = (frame)result;
            if (result.type == TYPE_CTS && result.node == get_my_mac()) {
               // Tengo el canal para mí
               frame data_frame = new frame();
               data_frame.type = TYPE_DATA;
               request_frame.duration = data_send_time + control_send_time + SIFS;
               data_frame.node = get_my_mac();
               data_frame.data = data;
               send_frame((char*)data_frame);
               result = receive_frame(data_send_time + SIFS);
               if (result != null) {
                  result_frame = (frame)result;
                  if (result.type == TYPE_ACK && result.node == get_my_mac()) {
                     // Salió todo bien
                     end = true;
                  }
                  // Si no manda ACK o no me lo manda a mí, es que algo salió mal
                  // Así que volvemos a intentar desde el principio
               }
            }
         }
         if (!end) {
            // No pudimos enviarlo, probablemente por colisiones
            // Así que esperamos un tiempo (acotado por data_send_time) antes de
            // intentar de nuevo
            wait(rand()%data_send_time);
         }
      }
   }
}
```

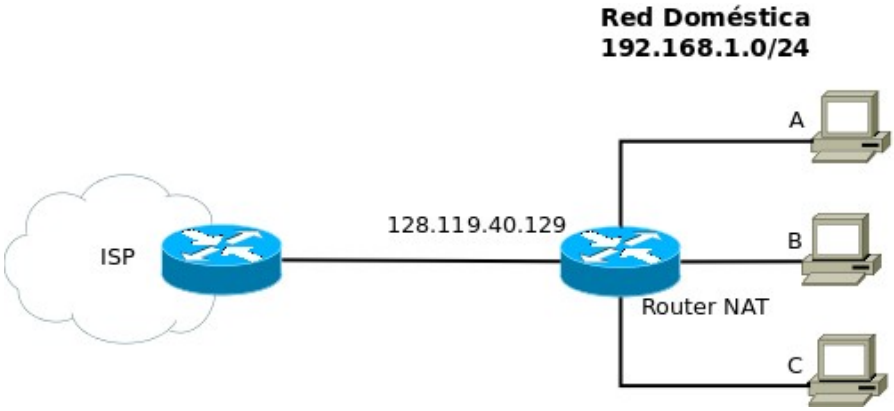
Receptor:

```
int MAX_WAIT_RECEIVER 2000;

void receiver() {
   MAC reserved_node;
   int reserved_time;
   while (true) {
      char* data = receive_frame(MAX_WAIT_RECEIVER);
      if (data != null) {
         // Llegó una trama válida
         frame f = (frame)data;
         if (f.type == TYPE_RTS){
            // Estamos libres, así que reservamos para este nodo
            reserved_node = f.node;
            reserved_time = f.duration;
            frame clear_frame = new frame();
            clear_frame.type = TYPE_CTS;
            clear_frame.duration = reserved_time - control_send_time - SIFS;
            clear_frame.node = reserved_node;
            send_frame((char*)clear_frame);

            int remaining_time = clear_frame.duration;
            bool data_en_AP = false;
            while (remaining_time > 0) {
               data = receive_frame(remaining_time);
               remaining_time = timeout_remaining_time;
               if (data != NULL) {
                  f = (frame)data;
                  if (f.type == TYPE_DATA && f.node = reserved_node) {
                        data_en_AP == true;
                        process_data(f.data);
                  }
                  // Si no es un frame datos del nodo que espero, lo descarto
               }
            }
            if (data_en_AP) {
                  frame ack_frame = new frame();
                  ack_frame.type = TYPE_ACK;
                  ack_frame.node = reserved_node;
                  ack_frame.duration = 0;
                  send_frame((char*)ack_frame);
            }
      }
   }
}
```

Problema 2 (15 puntos)

Considere la red presentada en la figura, dónde tenemos una red doméstica conectada a través de un *router* NAT a Internet. El ISP le asigna la IP 128.119.40.129.

1. Asigne direcciones a todas las interfaces de los tres equipos de la red doméstica.
2. Suponga que cada *host* tiene dos conexiones TCP activas, todas ellas en el puerto 80 del *host* 128.119.40.86. Proporcione las entradas correspondientes de la tabla de traducciones NAT.
3. Para alguna de dichas conexiones TCP activas (en algún equipo a su elección), indique los **valores** de IP y puerto origen e IP y puerto destino tanto para un paquete enviado como para un recibido en el *host*. Indique dichos valores para los segmentos entre:
    1. El *router* NAT y el *host*.
    2. El *router* NAT y el ISP.



Solución Problema 2

1) A = 192.168.1.1, B=192.168.1.2, C=192.168.1.3
2) Los puertos internos pueden repetirse para diferentes IP. Los puertos externos no. Se asignan valores al azar.

IP Int.	Puerto Int.	IP Ext.	Puerto Ext.
192.168.1.1	30100	128.119.40.129	40001
192.168.1.1	30101	128.119.40.129	40002
192.168.1.2	29800	128.119.40.129	40003
192.168.1.2	29801	128.119.40.129	40004
192.168.1.3	29999	128.119.40.129	40005
192.168.1.3	30100	128.119.40.129	40006

3)

Paquete entre Router NAT y Equipo A

Ip origen	Puerto origen	Ip destino	Puerto destino
128.119.40.86	80	192.168.1.1	30100

Paquete entre Equipo A y Router NAT

Ip origen	Puerto origen	Ip destino	Puerto destino
192.168.1.1	30100	128.119.40.86	80

Paquete entre Router NAT e ISP

Ip origen	Puerto origen	Ip destino	Puerto destino
128.119.40.129	40001	128.119.40.86	80

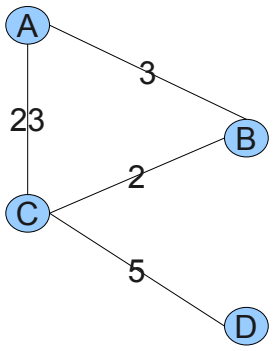
Paquete entre ISP y Router NAT

Ip origen	Puerto origen	Ip destino	Puerto destino
128.119.40.86	80	128.119.40.129	40001

Problema 3 (15 puntos)

Considere la red mostrada en la figura y asuma que cada nodo inicialmente conoce el costo a cada uno de sus vecinos.

a) Suponiendo que se utiliza un algoritmo de vector distancia, muestre la evolución de las tablas de distancias de todos los nodos hasta que se estabilicen.
b) Describa el problema que se genera a causa de la caída del enlace B-C e indique en cuantas iteraciones se resuelve. Justifique su respuesta.



Solución Problema 3

Parte a)

Estado inicial:

T=0		Costo a						Costo a						Costo a						Costo a						Costo a			
Nodo A		A	B	C	D	Nodo B		A	B	C	D	Nodo C		A	B	C	D	Nodo D		A	B	C	D						
Desde	A	0	3	23	INF	Desde	A					Desde	A					Desde	A										
	B						B	3	0	2	INF		B						B										
	C						C						C	23	2	0	5		C										
	D						D						D						D	INF	INF	5	0						

En la 1a. iteración se intercambian los vectores de distancias, y se producen los cambios marcados en amarillo:

T=1		Costo a						Costo a						Costo a						Costo a						Costo a			
Nodo A		A	B	C	D	Nodo B		A	B	C	D	Nodo C		A	B	C	D	Nodo D		A	B	C	D						
Desde	A	0	3	5	28	Desde	A	0	3	23	INF	Desde	A	0	3	23	INF	Desde	A										
	B	3	2	0	INF		B	3	0	2	7		B	3	0	2	INF		B										
	C	23	2	0	5		C	23	2	0	5		C	5	2	0	5		C	23	2	0	5						
	D						D						D	INF	INF	5	0		D	28	7	5	0						

En la 2a. iteración se intercambian los vectores de distancias, y se producen los cambios marcados en amarillo:

T=2		Costo a						Costo a						Costo a						Costo a						Costo a			
Nodo A		A	B	C	D	Nodo B		A	B	C	D	Nodo C		A	B	C	D	Nodo D		A	B	C	D						
Desd	A	0	3	5	10	Desd	A	0	3	5	28	Desd	A	0	3	5	28	Desd	A										
	B	3	0	2	7		B	3	0	2	7		B	3	0	2	7		B										
	C	5	2	0	5		C	5	2	0	5		C	5	2	0	5		C	5	2	0	5						

e	D					e	D					e	D	28	7	5	0	e	D	10	7	5	0

En la 3a. iteración solamente envían sus vectores de distancias los nodos A y D, que son los únicos que sufrieron cambios:

T=3		Costo a						Costo a						Costo a						Costo a			
Nodo A		A	B	C	D	Nodo B		A	B	C	D	Nodo C		A	B	C	D	Nodo D		A	B	C	D
Desde	A	0	3	5	10	Desde	A	0	3	5	10	Desde	A	0	3	5	10	Desde	A				
	B	3	0	2	7		B	3	0	2	7		B	3	0	2	7		B				
	C	5	2	0	5		C	5	2	0	5		C	5	2	0	5		C	5	2	0	5
	D						D						D	10	7	5	0		D	10	7	5	0

Dado que no se producen nuevos cambios, no hay más intercambios y el algoritmo converge.
Nota: dadas las características del grafo, A y B no se enteran del vector de distancias de D (porque no son vecinos). Por el mismo motivo, D no conoce los vectores de distancias de A y B.

Parte b)

Cuando se cae el enlace B-C se produce el problema de conteo a infinito, debido que B, al enterarse que su costo a C sube a infinito, avisa a A, quien dice tener un camino de costo 5, tal como quedaron las tablas de distancias en T=3. B calcula entonces un camino de costo 8 a través de A y lo propaga, A rehace las cuentas y encuentra un camino de 11, y así sucesivemente hasta que el camino ficticio de A via B supere en costo al camino directo A-C (23).
Cuántas iteraciones se necesitan para alcanzar el estado estable?

i=1, B=INF
i=2, A=5
i=3, B=8
i=4, A=11
i=5, B=14
i=6, A=17
i=7, B=20
i=8, A=23
i=9, B=26
i=10, A=29 > 23 (camino directo) -> A propaga el camino directo y no hay más oscilaciones.

Durante el conteo a infinito el tráfico hace un ping-pong entre B y A.