

Solución Examen – 31 de julio de 2024

(ref: solucionrc202407.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.
- Solo se responderán dudas de letra. No se responderán dudas de ningún tipo los últimos 30 minutos del examen.
- El examen es individual y sin material. Apague su teléfono celular mientras esté en el salón del examen.
- Es obligatorio responder correctamente al menos 15 puntos en las preguntas teóricas y 20 de los problemas prácticos. Los puntos ganados en el curso se suman a los puntos de teórico.
- 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)).
- Justifique todas sus respuestas.**
- Duración: 3 horas. Culminadas las 3 horas el alumno no podrá modificar las hojas a entregar de ninguna forma.

Preguntas Teóricas

Pregunta 1 (10 puntos)

En una red de paquetes:

- Defina los retardos de trasmisión y propagación; plantee ejemplos que permitan comparar los órdenes de magnitud entre ambos tipos de retardos.
- Analice el retardo de cola (*queueing delay*) en función de R (ancho de banda en bps), L (largo del paquete en bits) y a (tasa de arribo de paquetes).
- Defina el mecanismo de almacenamiento y reenvío (*store and forward*).

Solución Pregunta 1

a)

Retardo de trasmisión:

Es el tiempo que demora el nodo en poner todos los bits del paquete a trasmisir en el enlace.

Dados L , longitud del paquete (en bits) y R , ancho de banda del enlace (o velocidad de la interfaz, en bits por segundo, bps o bits/seg), el retardo de trasmisión es $dtrans = L/R$

Retardo de propagación:

Es el tiempo que demora un bit en atravesar el enlace.

Dados d , longitud del enlace físico (en metros) y s , velocidad de propagación en el medio (cobre, fibra, del orden de $\sim 2 \times 10^8$ metros/segundo), el retardo de propagación es $dprop = d/s$

Para un enlace de 1Gbps (1×10^9 bits/segundo) con paquetes de 1000 bits, y un enlace de 2000 kilómetros:

$$dtrans = \frac{1 \times 10^3 \text{ bits}}{1 \times 10^9 \text{ bits/segundo}} = 1 \times 10^{-6} \text{ segundos} = 1 \text{ microsegundo}$$

$$dprop = \frac{2 \times 10^6 \text{ metros}}{2 \times 10^8 \text{ metros/segundo}} = 1 \times 10^{-2} \text{ segundos} = 10 \text{ milisegundos o } 10.000 \text{ microsegundos}$$

Es decir en este ejemplo el retardo de propagación es 4 órdenes de magnitud mayor que el de trasmisión.

b)

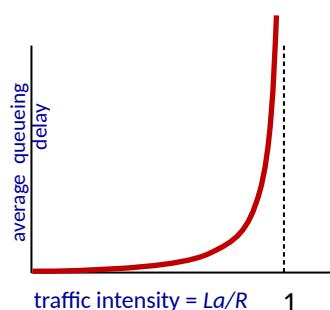
Dados R (ancho de banda en bps), L (largo del paquete en bits) y a (tasa de arribo de paquetes), se define la intensidad del tráfico como el cociente La/R .

Intuitivamente, la relación entre intensidad de tráfico y retardo de cola se puede ver en la figura, que permite comentar:

$La/R \sim 0$: el retardo de cola promedio es pequeño.

$La/R > 1$: el retardo de cola promedio es grande (hay congestión).

$La/R > 1$: llegan más paquetes que los que se pueden procesar, el retardo tiende a



infinito.

c)

La transmisión de almacenamiento y reenvío significa que el conmutador de paquetes debe recibir todo el paquete antes de que pueda comenzar a transmitir el primer bit del paquete al enlace saliente.

Pregunta 2 (10 puntos)

Suponga que tiene un flujo de datos TCP entre dos dispositivos usando un enlace. El nodo transmisor genera datos más rápido que el receptor puede procesar, y el nodo receptor es capaz de consumir los datos de su interfaz de entrada más rápido de lo que en el enlace puede transmitir. Describa el o los mecanismos de capa de transporte que son responsables de regular la velocidad de transmisión en este escenario.

Solución Pregunta 2

En el escenario descrito el transmisor es más rápido que el receptor, y este es más rápido que el enlace. El cuello de botella es por lo tanto el enlace. El mecanismo encargado de ajustar la velocidad de transmisión a la capacidad del enlace es el control de congestión.

Este mecanismo se basa en ir incrementando la tasa de transmisión hasta detectar algún evento que sugiera que el canal se saturó. En este caso se procede a reducir dicha tasa para luego continuar el proceso. La tasa de transmisión se acota limitando el número de segmentos que se entrega a la red pendientes de confirmación. Los eventos de pérdida detectados pueden ser la recepción de confirmaciones duplicadas (que indican la pérdida de segmentos, considerado un evento leve) o el timeout a la espera de confirmaciones (que indican la pérdida de segmentos y/o confirmaciones, considerado un evento grave).

Pregunta 3 (10 puntos)

- Describa el algoritmo de Dijkstra y provea un pseudocódigo del mismo. ¿Qué problema resuelve?
- Usando un protocolo de estado de enlace en una red típica con N nodos, ¿cuántas instancias del algoritmo se ejecutan?
- Muestre la ejecución paso a paso del algoritmo en una red sencilla de 4 nodos o más con costos positivos.

Solución

a)

El algoritmo de Dijkstra permite encontrar, en un grafo con costos de aristas no negativos, los caminos más cortos desde un vértice a todos los restantes. Es un algoritmo iterativo que va construyendo los caminos más cortos comparando el costo actual con el costo para llegar a un vecino más la distancia desde ese vecino al nodo objetivo.

Pseudo-código:

Dijkstra's link-state routing algorithm

```

1 Initialization:
2  $N' = \{u\}$                                 /* compute least cost path from u to all other nodes
3 for all nodes v
4   if v adjacent to u                      /* u initially knows direct-path-cost only to direct neighbors
5     then  $D(v) = c_{u,v}$                   /* but may not be minimum cost!
6   else  $D(v) = \infty$ 
7
8 Loop
9   find w not in  $N'$  such that  $D(w)$  is a minimum
10  add w to  $N'$ 
11  update  $D(v)$  for all v adjacent to w and not in  $N'$  :
12     $D(v) = \min(D(v), D(w) + c_{w,v})$ 
13  /* new least-path-cost to v is either old least-cost-path to v or known
14  least-cost-path to w plus direct-cost from w to v */
15 until all nodes in  $N'$ 

```

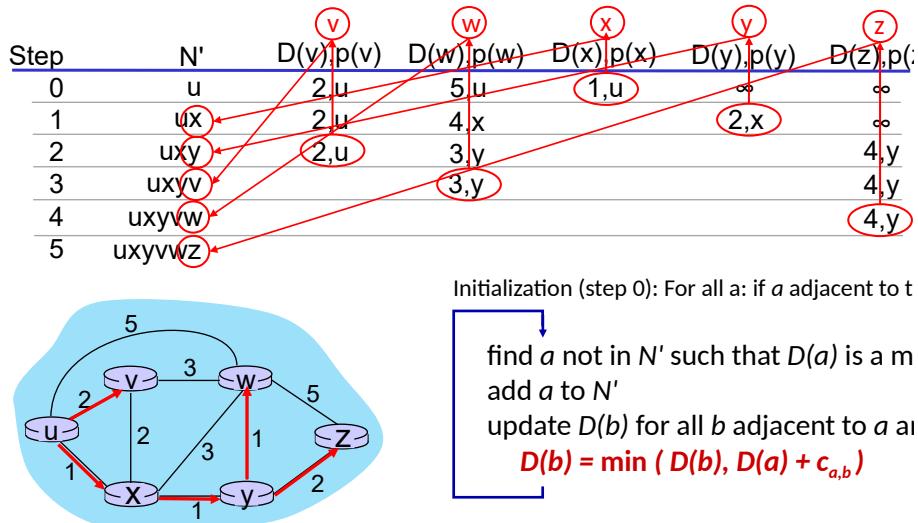


b)

Dijkstra se ejecuta en cada nodo en base a la información topológica (estado del enlace) que se recolecta mediante un mecanismo de inundación de información (flooding). Por lo tanto para una red de N nodos, se ejecutan N instancias del algoritmo.

c) Se muestra un caso de 6 nodos:

Dijkstra's algorithm: an example



Pregunta 4 (10 puntos)

- a) Describa el funcionamiento del protocolo *slotted aloha* (aloha “ranurado”).
 b) Calcule la eficiencia del protocolo, y compárela con la eficiencia del aloha puro (sin sincronización de “ranuras” de tiempo).

Solución

a)

Se asume lo siguiente:

- Todas las tramas constan exactamente de L bits.
- El tiempo se divide en intervalos (slots) de tamaño L/R segundos (es decir, un intervalo equivale al tiempo necesario para transmitir una trama).
- Los nodos comienzan a transmitir tramas sólo al comienzo de los intervalos.
- Los nodos están sincronizados para que cada nodo sepa cuándo comienzan los intervalos.
- Si dos o más tramas colisionan en un intervalo, todos los nodos detectan la colisión antes de que finalice dicho intervalo.

Sea p una probabilidad, es decir, un número entre 0 y 1. La operación de Aloha ranurado en cada nodo es la siguiente:

- Cuando el nodo tiene una nueva trama para enviar, espera hasta el comienzo del siguiente intervalo y transmite la trama completa en dicho intervalo.
- Si no hay colisión, el nodo ha transmitido exitosamente su trama y por lo tanto no es necesario considerar la retransmisión de la misma (el nodo puede preparar una nueva trama para transmisión, si la tiene).

- Si hay una colisión, el nodo la detecta antes del final del intervalo. El nodo retransmite su trama en cada intervalo subsiguiente con probabilidad p hasta que la trama se transmite sin colisión.

b)

Aloha ranurado:

En una red con N nodos, la probabilidad de que un slot sea exitoso (o sea, sin colisiones), es la probabilidad de que uno de los nodos transmita y que los $N-1$ nodos restantes no transmiten. La probabilidad de que un nodo determinado transmita es p , mientras que la probabilidad de que los nodos restantes no transmitan es $(1-p)^{N-1}$. Por lo tanto, la probabilidad de que un nodo determinado tenga éxito es:

$$p(1-p)^{N-1}$$

Debido a que hay N nodos, la probabilidad de que cualquiera de los N nodos tenga éxito es:

$$Np(1-p)^{N-1}$$

Esta es pues la expresión de la eficiencia de Aloha ranurado. Para obtener la máxima eficiencia debemos maximizar esta expresión, tomando el límite para $N \rightarrow \infty$. Se puede calcular que dicho límite es:

$$1/e \text{ (donde } e \text{ es el número neperiano, que vale } \approx 2.72)$$

Esto da una eficiencia de alrededor del 37%.

Aloha puro:

En este caso no existe la sincronización de intervalos, y por lo tanto en cada nodo las colisiones se pueden dar con el "intervalo anterior" y el "intervalo posterior". Intuitivamente esto implica que puede haber el doble de colisiones que en el caso ranurado, luego la eficiencia baja a la mitad, alrededor del 18%.

Problemas Prácticos

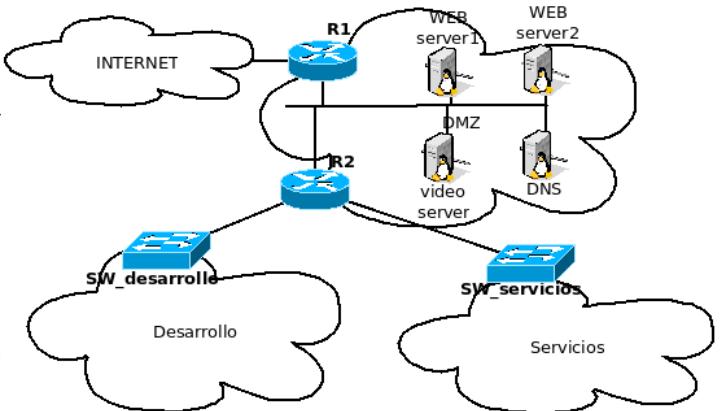
Problema 1 (30 puntos)

Una empresa que desarrolla y provee servicios web para Internet, tiene contratada su conexión a Internet a través de un proveedor, quien suministra el router (R1) y le asignó la red pública 191.6.1.0/29 para su uso.

La empresa cuenta con tres subredes: *DMZ*, donde se encuentran los equipos que brindan servicios al público en Internet, *Servicios* donde se alojan los recursos que permiten resolver servicios brindados en la *DMZ* y *Desarrollo*, subred utilizada para el desarrollo de los servicios brindados.

La red debe cumplir las siguientes condiciones:

- *Servicios* tiene 33 equipos (contiene los equipos que proveen funcionalidades a los servicios web brindados), y solo se conecta con la *DMZ*.
- *Desarrollo* tiene 62 puestos de trabajo, debe tener acceso a Internet, pero no deben ser accesibles desde el mundo
- *DMZ* con 4 equipos (2 servidores web, un equipo servidor de video y un equipo para DNS). Estos equipos deben tener conectividad con la red de *Servicios*.



Se pide:

- a) Asigne direcciones IP a todas las redes de la empresa para que representen la realidad planteada.
- b) Indique las tablas de forwarding de los routers R1 y R2, el equipo DNS, un equipo de la red *Servicios* y uno de *Desarrollo*, para que se cumplan las condiciones de la letra.
- c) Especifique los paquetes que se transmiten en la red al realizar un *ping* desde un equipo de la red de *Servicios* a la IP del equipo *DNS*. Enumere mensajes, direcciones y payload de capa 2, 3 y 4. Asuma que las caches están vacías previo al ejecutar el comando.

Solución

a)

LAN pública disponible 191.6.1.0/29, que equivale a 6 direcciones IP

R1: 191.6.1.1

DNS: 191.6.1.2

web_server1: 191.6.1.3

web_server2: 191.6.1.4

video_server: 191.6.1.5

R2: 191.6.1.6

LAN Servicio: 33 hosts.

Se necesitan además: 1 dir IP para el R2, dirección de red y broadcast, 36 direcciones IP en total → prefijo /26 ($2^6 = 64$).

LAN Desarrollo: 62 hosts.

Se necesitan además: 1 dir IP para el R2, dirección de red y broadcast, 65 direcciones IP en total → prefijo /25 ($2^7 = 128$).

La subred Desarrollo se comunica con Internet pero no al revés, se puede resolver con direcciones privadas y conectividad mediante NAT para ese rango de direcciones

Redes de Computadoras

en la interfaz de R2 correspondiente (eth2).

La red de Servicios solo se comunica con la DMZ, y por lo tanto puede utilizar numeración privada y NO necesita NAT para ese rango de direcciones en la interfaz correspondiente de R2 (eth1).

LAN Servicios: 192.168.1.0/26

Se propone la IP 192.168.1.1 para el router R2 (eth1)

LAN Desarrollo: 192.168.2.0/25

Se propone la IP 192.168.2.1 para el router R2 (eth2)

b)

R1:

se supone eth0 conectada a Internet y eth1 a la red DMZ.

Prefijo de Destino	Interfaz de salida	Next-Hop
191.6.1.0/29	eth1	Directamente conectado
0.0.0.0/0	eth0	IP_Proveedor

R2:

se supone eth0 conectada a DMZ y eth1 a la red Sevicios y eth2 conectada a Desarrollo.

Prefijo de Destino	Interfaz de salida	Next-Hop
191.6.1.0/29	eth0	Directamente conectado
0.0.0.0/0	eth0	191.6.1.1
192.168.1.0/26	eth1	Directamente conectado
192.168.2.0/25	eth2	Directamente conectado

Para el tráfico con destino las redes públicas, cursado en las interfaces eth1 y eth2 se implementa NAT, para permitir la salida a Internet. La dirección IP pública de salida para ambas redes es 190.1.1.6.

DNS_server:

Prefijo de Destino	Interfaz de salida	Next-Hop
191.6.1.0/29	eth0	Directamente conectado
0.0.0.0/0	eth0	191.6.1.1
192.168.1.0/26	eth0	191.6.1.6

El equipo DNS debe tener conectividad con la red se Servicios. No se pide conectividad con la de desarrollo.

equipo_red_servicios:

Prefijo de Destino	Interfaz de salida	Next-Hop
192.168.1.0/26	eth0	Directamente conectado
191.6.1.0/29	eth0	192.168.1.1

equipo_red_desarrollo:

Prefijo de Destino	Interfaz de salida	Next-Hop
192.168.2.0/25	eth0	Directamente conectado
0.0.0.0/0	eth0	192.168.2.1

c)

No se presenta información de capa 4, dado que en el envío de un ping (ICMP echo request y reply) no interviene capa 4.

Es importante aclarar que como WEB_server1 no tiene ruta a desarrollo para que el ping funcione se debe realizar a través del NAT, de lo contrario nunca llegará la respuesta.

Tipo de mensaje	Cabezal capa 2	Cabezal capa 3	payload
ARP_request (realizados por equipo en desarrollo)	MAC_orig: MAC_des MAC_dest: ff:ff:ff:ff:ff:ff	-	sender_mac: MAC_des sender_IP: 192.168.2.X target_mac: 00:00:00:00:00:00 target_IP: 192.168.2.1
ARP_response (realizados por equipo R2)	MAC_orig: MAC_R2_eth2 MAC_dest: MAC_des	-	sender_mac: MAC_R2_eth2 sender_IP: 192.168.2.1 target_mac: MAC_des target_IP: 192.168.2.X
ICMP echo request	MAC_orig: MAC_des MAC_dest: MAC_R2_eth2	IP origen: 192.168.2.X IP destino: 190.1.1.3	Echo request Tipo 8, Código 0
ARP_request (realizados por equipo R2)	MAC_orig: MAC_R2_eth0 MAC_dest: ff:ff:ff:ff:ff:ff	-	sender_mac: MAC_R2_eth0 sender_IP: 191.6.1.6 target_mac: 00:00:00:00:00:00 target_IP: 191.6.1.3
ARP_response (realizados por equipo WEB_server1)	MAC_orig: MAC_WSer1 MAC_dest: MAC_R2_eth0	-	sender_mac: MAC_WSer1 sender_IP: 190.1.1.3 target_mac: MAC_R2_eth0 target_IP: 191.6.1.6
ICMP echo request	MAC_orig: MAC_R2_eth0 MAC_dest: MAC_WSer1	IP origen: 191.6.1.6 IP destino: 191.6.1.3 (cambia IP por NAT)	Echo request Tipo 8, Código 0
ICMP echo response	MAC_orig: MAC_WSer1 MAC_dest: MAC_R2_eth0	IP origen: 191.6.1.3 IP destino: 191.6.1.6 (cambia IP por NAT)	Echo response Tipo 0, Código 0
ICMP echo response	MAC_orig: MAC_R2_eth2 MAC_dest: MAC_des	IP origen: 191.6.1.3 IP destino: 192.168.2.X	Echo response Tipo 0, Código 0

Problema 2 (30 puntos)

Se desea desarrollar el firmware (software de control) de un switch (comutador de capa 2). El switch debe implementar, conmutación de frames, self-learning y adicionalmente detección de MACs repetidas en la subred.

Dispone de las siguientes llamadas y constantes:

- **N_PORTS**
número de bocas del switch.
- **frame, port = read_frame()**
Llamada bloqueante, queda a la espera de la llegada del siguiente frame. Devuelve el frame recibido y en qué boca se recibió
- **write_frame(frame, port)**
Llamada bloqueante que escribe el frame en la boca indicada.
- **alarm_mac(address, port1, port2)**
Deberá llamarse cada vez que se detecta una MAC repetida en diferentes bocas. El procesamiento deberá continuarse asumiendo que la ubicación nueva es la correcta.
- **src, dest = get_addresses(frame)**
Devuelve las direcciones origen y destino de un frame.
- **is_broadcast(mac)**
Devuelve verdadero si la dirección es FF:...:FF

Además dispone de estructuras de datos típicas (listas, diccionarios, etc.)

Se pide:

- Implemente el switch en pseudocódigo o un lenguaje de alto nivel usando las primitivas brindadas en la letra.
- En el caso de soportar VLANs, ¿qué modificaciones deben realizarse al sistema planteado en a)? Describa, sin implementar.

Solución

a)

```

FoundOn: MAP( MAC → number ); // para registrar el puerto donde fue vista la MAC
LastSeen: MAP( MAC → time ); // no se pide obsolescer entradas, pero se puede hacer

while true do
    frame, inport = read_frame();

    // mantenimiento (opcional)
    for addr, time in LastSeen do
        if getTime()-time > LIFETIME then
            FoundOn.remove(addr);
            LastSeen.remove(addr);
        end
    end

    src, dest = get_addresses(frame);

    // registramos ubicación de MAC origen, chequeando si ya está en otro lado
    if FoundOn[src]<>inport then
        alarm_mac(src, inport, FoundOn[src])
        FoundOn.remove(src);
    end
    FoundOn[src] = inport;
    LastSeen[src] = getTime();

    // transmitimos a donde corresponda
    if (is_broadcast(dst)) then
        for outport=1, N_PORTS do
            if outport<> inport then write_frame(frame, outport) end
    end

```

```
    end
else
    outport = FoundOn(dst);
    if (outport==nil)           // destino desconocido, se hace flooding.
        for outport=1, N_PORTS do
            if outport<> inport then write_frame(frame, outport) end
        end
    else
        write_frame(frame, outport)
    end
end
end
```

b)

Hace falta introducir las siguientes modificaciones:

1. Agregar funciones para parsear la trama 802.1Q para poder obtener la etiqueta VLAN ademas de la direcciones.
2. Agregar una configuración que permita asociar una VLAN a cada puerto, o marcarlo como Trunk.
3. Al recibir una trama por un puerto no-trunk, conmutarlo solo considerando los puertos de igual VLAN configurada. Además, se deberá armar una trama 802.1Q etiquetada con esa VLAN y enviarla por todos los puertos TRUNK.
4. Al recibir una trama por un puerto TRUNK, extraer la VLAN de la trama 802.1Q y conmutar una trama Ethernet en los puertos correspondientes.