Curso 24-25. Examen semana 1

1.-(2 ptos) Considere el envío de voz en tiempo real desde el Host A al Host B a través de una red conmutada por paquetes (VoIP). El Host A convierte la voz analógica en una transmisión digital de 64 kbps en tiempo real. Luego, el Host A agrupa los bits en paquetes de 56 bytes. Hay un enlace entre los Hosts A y B; su tasa de transmisión es de 10 Mbps y su retraso de propagación es de 10 ms. Tan pronto como el Host A recopila un paquete, lo envía al Host B. Tan pronto como el Host B recibe un paquete completo, convierte los bits del paquete en una señal analógica. ¿Cuánto tiempo transcurre desde que se genera un bit (a partir de la señal analógica original en el Host A) hasta que el bit es decodificado (como parte de la señal analógica en el Host B)?

Problema 7 pag 60

Problem 7

Consider the first bit in a packet. Before this bit can be transmitted, all of the bits in the packet must be generated. This requires

$$\frac{56 \cdot 8}{64 \times 10^3} \text{ sec} = 7 \text{msec}.$$

The time required to transmit the packet is

$$\frac{56 \cdot 8}{2 \times 10^6} \text{ sec} = 224 \mu \text{ sec}.$$

Propagation delay = 10 msec.
The delay until decoding is

$$7\text{msec} + 224\mu \text{ sec} + 10\text{msec} = 17.224\text{msec}$$

A similar analysis shows that all bits experience a delay of 17.224 msec.

2.- (2 ptos) Un cliente envía un mensaje HTTP GET a un servidor web, *example.com*. Suponga que dicho mensaje es el siguiente, línea a línea:

```
GET /images/photos/vacation2023.jpg HTTP/1.1
Host: example.com
Accept: image/png, image/jpeg, image/webp
Accept-Language: es-es, es;q=0.8, en-us;q=0.7, fr;q=0.5
If-Modified-Since: Mon, 01 Jan 2024 10:30:00 -0700
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36
```

Responda razonadamente las siguientes preguntas, indicando explícitamente en qué parte del mensaje ha encontrado la respuesta:

a. Identifique y explique los campos de la *línea de solicitud*.

b. ¿Qué tipo de navegador inicia este mensaje? ¿Por qué es necesario indicar el tipo de navegador en un mensaje de solicitud HTTP?

c. ¿Qué sucedería si el servidor sólo tiene una versión del documento solicitado en francés?

d. Indique razonadamente si el cliente tiene ya una copia en caché del archivo o no.

Pag 86: a, b, c. Pag 95: d

a) La primera línea es la línea de solicitud y consta de 3 campos: el campo del método, el campo URL y el campo de la versión de HTTP. El primero podría ser. GET, POST, HEAD, PUT, DELETE. En este caso se utiliza GET, se pide un archivo imagen y el navegador utiliza la versión de HTTP 1.1
b) El navegador utilizado (agente de usuario/User Agent) es Mozila 5.0, un navegador Firefox. Es importante porque el servidor puede enviar diferentes versiones del mismo objeto a diferentes agentes de usuario.
c) Accept-Language: es-es, es;q=0.8, en-us;q=0.7, fr;q=0.5 (Prefiere español de España, luego español general, luego inglés de EEUU, luego francés general).
Por lo tanto si solo tiene en francés se la enviará en francés.
d) la línea If-Modified-Since de 1 de enero de 2024 indica que ya tiene una copia en cache de este doc y de esa fecha. Solo hay que volver a enviarlo si se ha modificado a posteriori.

3.- (2 ptos) Dos paquetes llegan a dos puertos de entrada distintos de un *router* exactamente al mismo tiempo. No hay ningún otro paquete en ninguna parte del *router*.

a. Suponga que los dos paquetes deben ser reenviados a dos puertos de salida diferentes. ¿Es posible reenviar los dos paquetes a través del entramado de conmutación al mismo tiempo, cuando el entramado utiliza un *bus compartido*? ¿Y si el entramado utiliza la *conmutación vía memoria*?

b. Suponga que los dos paquetes deben ser reenviados al mismo puerto de salida. ¿Es posible reenviar los dos paquetes a través del entramado de conmutación al mismo tiempo, cuando el entramado utiliza una *malla*?

No olvide completar y justificar su exposición con los diagramas correspondientes.

Respuesta A

¿Pueden reenviarse simultáneamente si el entramado usa un bus compartido?

No. En una arquitectura de bus compartido, solo un paquete puede usar el bus a la vez. Aunque los paquetes vayan a puertos de salida distintos, el bus es un recurso único y compartido; por tanto, uno de los dos paquetes tendrá que esperar.

¿Y si el entramado utiliza conmutación vía memoria?

No. En la conmutación vía memoria, los paquetes deben ser escritos y leídos uno por vez, porque la CPU y el bus interno hacia la memoria son recursos compartidos. Aunque haya múltiples puertos, el camino interno hacia la memoria es único → no se permite acceso simultáneo para dos paquetes. De nuevo, un paquete debe esperar al otro.

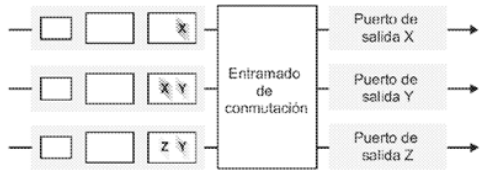
Respuesta B

Una malla (crossbar) permite múltiples transferencias simultáneas solo si las rutas no comparten entradas ni salidas:

- Mismo input a distintas salidas: ❌ (conflicto en entrada)
- Distintos inputs a distintas salidas: ✔️ (paralelismo)
- Distintos inputs a la misma salida: ❌ (conflicto en salida)

En este caso, ambos paquetes quieren usar el mismo puerto de salida, por lo que habría una colisión en el mismo output y **no es posible** reenviar simultáneamente los dos paquetes

4.- (2 ptos) Considere el dispositivo de conmutación mostrado en la figura. Suponga que todos los datagramas tienen la misma longitud fija, que el dispositivo opera de forma síncrona y particionada y que en una partición temporal se puede transferir un datagrama desde un puerto de entrada a un puerto de salida. El entramado de conmutación emplea una *estructura de malla*, por lo que, como máximo, se puede transferir un datagrama a un puerto de salida determinado en una partición de tiempo, pero distintos puertos de salida pueden recibir datagramas procedentes de distintos puertos de entrada en una misma partición de tiempo. Conteste razonadamente a las siguientes preguntas:



a. ¿Cuál es el número mínimo de particiones de tiempo necesarias para transferir los paquetes mostrados desde los puertos de entrada a sus puertos de salida, suponiendo cualquier orden de planificación de la cola de entrada que desee (es decir, no tiene por qué existir el *bloqueo de cabecera*, HOL)?

b. ¿Cuál es el número máximo necesario de particiones de tiempo, suponiendo el orden de planificación de caso peor que pueda imaginar y suponiendo que una cola de entrada no vacía nunca está inactiva?

c. Repita el apartado anterior, suponiendo ahora que el primer datagrama de la cola de entrada inferior es X.

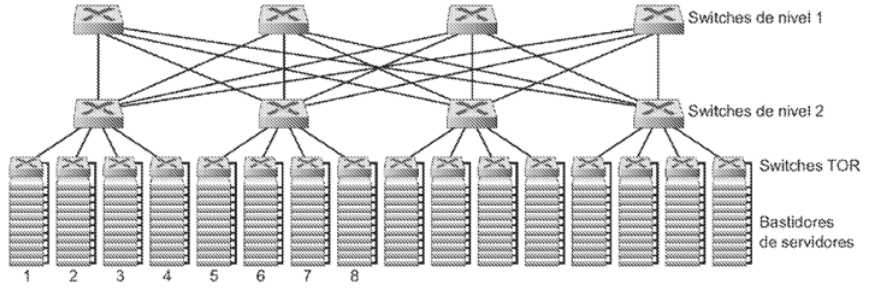
Problem 4

The minimal number of time slots needed is 3. The scheduling is as follows.
Slot 1: send X in top input queue, send Y in middle input queue.
Slot 2: send X in middle input queue, send Y in bottom input queue
Slot 3: send Z in bottom input queue.

Largest number of slots is still 3. Actually, based on the assumption that a non-empty input queue is never idle, we see that the first time slot always consists of sending X in the top input queue and Y in either middle or bottom input queue, and in the second time slot, we can always send two more datagram, and the last datagram can be sent in third time slot.

NOTE: Actually, if the first datagram in the bottom input queue is X, then the worst case would require 4 time slots.

5.- (2 ptos) Considere un centro de datos cuya red presenta la *topología altamente interconectada* que se muestra en la figura. Suponga que hay 80 flujos: diez flujos entre el primer y el noveno bastidores, diez flujos entre el segundo y el décimo, y así sucesivamente. Suponga también que todos los enlaces de la red son de 10 Gbps, excepto los enlaces entre los *hosts* (servidores) y los *switches* TOR, que son de 1 Gbps.



a. Suponga que todos los flujos tienen la misma velocidad de datos. Determine la velocidad máxima de un flujo.

b. Suponga ahora que hay un patrón de tráfico similar, pero con 20 hosts en cada bastidor y 160 pares de flujos. Determine las velocidades máximas de flujo para las dos topologías.

Teoría en pag 412 del libro de texto

identificar varias tendencias importantes.

Una de esas tendencias es la de implantar nuevas arquitecturas de interconexión y protocolos de red que resuelvan los problemas de los diseños jerárquicos tradicionales. Una de dichas soluciones consiste en sustituir la jerarquía de switches y routers por una **topología completamente conectada** [Facebook 2014; Al-Fares 2008; Greenberg 2009b; Guo 2009], como la mostrada en la Figura 6.31. En este diseño, cada switch de nivel 1 se conecta a todos los switches de nivel 2, de modo que (1) el tráfico de host a host nunca necesita subir por encima de los niveles de switches y (2) con n switches de nivel 1, entre cualesquiera dos switches de nivel 2 existirán n rutas disjuntas. Un diseño de este tipo puede mejorar significativamente la capacidad host a host. Para comprobarlo, considere de nuevo nuestro ejemplo de los 40 flujos. La topología de la Figura 6.31 puede gestionar perfectamente ese patrón de flujo, ya que existen cuatro rutas diferentes entre el primer switch de nivel 2 y el segundo, las cuales proporcionan una capacidad agregada de 40 Gbps entre los dos primeros switches de nivel 2. Este tipo de diseño no solo alivia la limitación de la capacidad host a host, sino que también crea un entorno más flexible de cómputo y de prestación de servicios, en el que la comunicación entre cualesquiera dos bastidores que no estén conectados al mismo switch es lógicamente equivalente, independientemente de dónde estén ubicados en el centro de datos.

modulares (MDC, *Modular Data*

Hace referencia al problema 32 página 425, pero con 80 flujos.

Problem 32

a) Each flow evenly shares a link's capacity with other flows traversing that link, then the 80 flows crossing the B to access-router 10 Gbps links (as well as the access router to border router links) will each only receive 10 Gbps / 80 = 125 Mbps

b) In Topology of Figure 5.31, there are four distinct paths between the first and third tier-2 switches, together providing 40 Gbps for the traffic from racks 1-4 to racks 9-12. Similarly, there are four links between second and fourth tier-2 switches, together providing 40 Gbps for the traffic from racks 5-8 to 13-16. Thus the total aggregate bandwidth is 80 Gbps, and the value per flow rate is 1 Gbps.

c) Now 20 flows will need to share each 1 Gbps bandwidth between pairs of TOR switches. So the host-to-host bit rate will be 0.5 Gbps.

El apartado b) de este problema responde a la pregunta a). Se utilizan dos pares de switches para poder dotar de 40Gbps cada uno, haciendo una capacidad agregada de 80Gbps. En vez de 40 hay 80 flujos, por lo que al dividir por 80 Gbps por los 80 flujos sale 1 Gbps.

El apartado c) responde a la pregunta b). En los bastidores como hay 20 flujos, se divide 1 Gbps por 20 y sale 0,5 Gbps.