

Segundo Certamen (Tiempo: 90 min.)

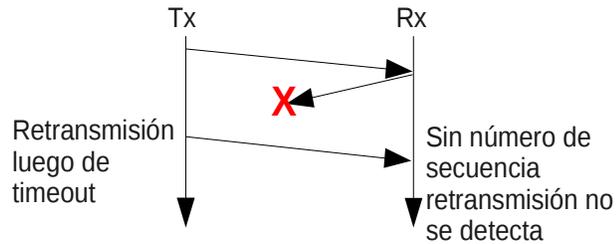
Si algo no está claro, haga una suposición razonable, anótelas y responda conforme a ella.

1.- (50 puntos) Responda en forma breve, con ideas y letra claras:

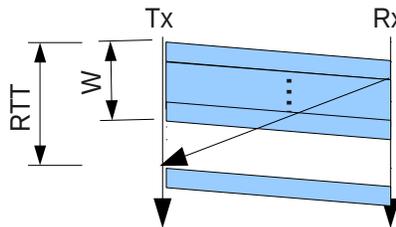
Pauta: Respuestas más cortas son perfectamente correctas. La pauta es más extensa para ayudar el estudio de futuros estudiantes.

- a) Desde el punto de vista del tiempo dedicado a fragmentar y luego re-ensamblar datagramas en la Capa de Red, ¿Qué protocolo de capa de transporte (TCP o UDP) demanda mayor tiempo? Explique.
UDP demanda más tiempo, pues éste puede enviar unidades de información mayor que el MTU y la fragmentación a nivel IP es necesaria. TCP ajusta su tamaño de segmento máximo de modo de evitar fragmentación y conseguir así mayor eficiencia en el uso del canal cuando se presentan pérdidas de fragmentos.
- b) Explique por qué DNS ocupa UDP y no TCP como protocolo de transporte.
Porque así se consigue mayor rapidez al enviar sólo un mensaje de ida y una respuesta. Si usáramos TCP sería necesario establecer conexión primero lo cual ocupa al menos un mensaje de ida y otro de vuelta, luego deberíamos intercambiar los dos mensajes DNS y finalmente se requeriría dos mensajes más para cerrar la conexión. Así es como TCP requiere varios intercambios de mensajes adicionales al uso de UDP. En caso de pérdida de paquetes, usando UDP la aplicación debe ocuparse de hacer las retransmisiones ocupando la misma cantidad de paquetes que debería usar TCP.
- c) Alguien dice: “todos los paquetes que llegan a un socket UDP poseen igual puerto destino e igual puerto origen” ¿Está usted de acuerdo? Justifique.
No. Un socket UDP puede recibir paquetes desde distintos orígenes y puertos. Esto se debe a que UDP no establece conexión con una contra parte específica.
- d) Cuando usted corre dos navegadores en su computador en forma simultánea y con ambos accede a www.google.cl, ¿Qué permite al servidor web enviar la respuesta al navegador desde donde provino la consulta?
**Las dos consultas se hacen vía conexiones TCP las cuales se individualizan con la 4-tupla (IP_origen, IP_destino, Puerto_origen, Puerto_destino); si bien ambos browsers tienen igual IP origen, manejan Puerto Origen distinto y con ello el servidor web sabe a través de qué socket responder al browser específico.
Otra forma de responder sería, el servidor web maneja cada solicitud a través de un socket único creado en el servidor cuando el browser se conecta, es así que el servidor responde a través del mismo socket por el cual llega al consulta permitiendo así responder al proceso correspondiente en el lado cliente.**
- e) Describa una situación que pueda dar origen a la llegada de paquetes UDP fuera de orden.
Cuando un datagrama o fragmento toma una ruta más larga, la llegada de ese fragmento o datagrama puede ser posterior al paquete UDP transmitido después.
- f) En el diseño de un protocolo ¿Cómo propone usted identificar la llegada de paquetes corruptos? ¿Cómo se puede dar cuenta el transmisor que un paquete se perdió?
**Los paquetes dañados o corruptos se pueden identificar agregando un código redundante del tipo suma de chequeo o código de redundancia cíclica (como el dígito verificador en número de rol).
La pérdida de un paquete se puede detectar por la espera excesiva de su acuse de recibo. En protocolos que usan “pipelining” también se puede detectar por la llegada de acuses de recibo duplicados (caso go-back-N), o acuses de recibos de paquetes posteriores (caso Selective-repeat).**
- g) Un aficionado insiste que stop-and-wait está errado pues al enviar un solo paquete por vez, no sería necesario usar números de secuencia. Haga un diagrama que conduzca a falla si no usamos números de secuencia en stop-and-wait.

Cuando se pierde un acuse de recibo, el transmisor hace una retransmisión al cabo de un rato; al no usar números de secuencia el receptor no puede distinguir si se trata de un paquete nuevo o retransmisión.



- h) En ausencia de errores y cuando el buffer de recepción es muy grande, la tasa promedio de transferencia de TCP durante un RTT se puede aproximar por (Tamaño de Ventana de congestión)/RTT. Muestre un diagrama que explique la deducción de esta expresión. ¿Es esta expresión válida para todo valor de “Ventana de Congestión”? Explique.



Luego de enviar la ventana W, el Tx debe esperar la llegada del acuse de recibo más antiguo para retomar la transmisión.

No es válida siempre, pues conforme la ventana de congestión aumenta, aumenta la utilización del canal y cuando ésta alcanza 100% no es posible seguir aumentando la tasa pues la tasa de transmisión del enlace pone una cota máxima.

- i) TCP corriendo sobre IP espera hasta recibir tres paquetes ACK duplicados para iniciar una retransmisión rápida. ¿Por qué piensa usted que los diseñadores de TCP no escogieron iniciar la retransmisión rápida tan pronto llega el primer ACK duplicado? ¿Se justifica la espera por tres ACKs duplicados en la implementación de un protocolo confiable en una red ATM? Explique.

La retransmisión no se efectúa ante la llegada del primer duplicado pues es posible que el primer paquete haya tomado una ruta más larga y llegue con posterioridad al segundo. En este caso, un cambio de orden de llegada genera acuses de recibo duplicados pero no constituye pérdida de paquete.

No se justifica esperar tres ACKs en ATM pues ésta establece una conexión entre fuente y destino luego todos los paquetes deben usar la misma ruta y con ello el orden de salida y llegada se mantiene cuando no hay pérdidas.

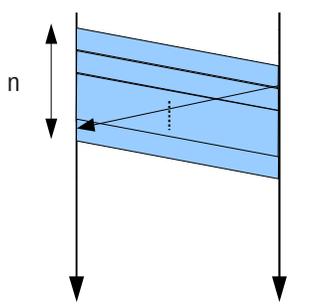
- j) En presencia de errores esporádicos regulares TCP reduce su ventana (W) a la mitad (W/2) por cada error y luego la incrementa linealmente desde W/2 hasta llegar a W. En presencia de este ciclo la tasa promedio conseguida es de $3*W/4$. Si TCP redujera su tasa a $3/4$ de su valor cada vez que se presenta un error esporádico, ¿qué tasa promedio conseguiría?

NOTA: La pregunta mezcla la ventana promedio con la idea de tasa promedio. La tasa, como se muestra en pregunta h, es Tamaño_de_Ventana/RTT. $3*W/4$ es la ventan promedio, la tasa promedio sería $3*W/(4RTT)$.

Cuando el incremento es lineal, la ventan promedio es el promedio entre los valores extremos. Si la ventana se reduce a $3/4$ W, la ventan promedio sería $(1+ 3/4)W/2 = 7*W/8$. (y la tasa promedio sería $7W/8RTT$.

2.- (25 puntos) En esta pregunta usted explorará la capacidad de transferencia máxima usando “Selective Repeat” en enlaces de altas tasas de transmisión. Suponga un RTT de 150 [ms] entre dos sitios A y B, paquetes de 1500 bytes, y números de secuencia de 16 bits.

a) Si la tasa de transmisión del enlace es de 100 Mbps ¿cuántos paquetes logran ser transmitidos hasta la llegada del primer ACK?



$$n * 1500 [\text{bytes/paquete}] * 8 [\text{bit/byte}] = 100 * 10^6 [\text{bit/s}] * 150 * 10^{-3} [s]$$

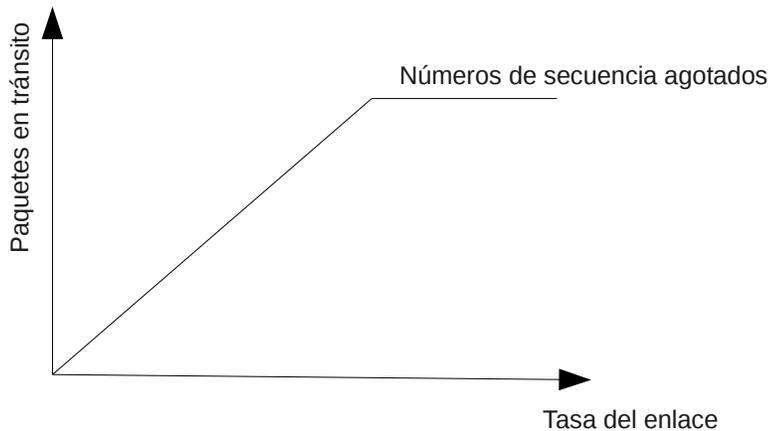
$$n = \text{tasa}_{\text{enlace}} * \text{RTT} / \text{Tamaño}_{\text{paqueteEnBits}}$$

$$n = 1250 [\text{paquetes}]$$

Se logra transmitir 1250 paquetes.
Como $n < 2^{16}/2$, entonces los números de secuencias no se agotan antes de alcanzar utilización 100%.

b) Haga un gráfico cualitativo para el número de paquetes en tránsito (aquellos sin ACK recibido) para distintas tasa de transmisión del enlace.

El número de paquetes crece linealmente conforme sube la tasa del enlace; sin embargo, éste no puede superar el tamaño de la ventana (rango de números de secuencia), luego el número de paquetes en transito se satura al llegar a 2^{15} .



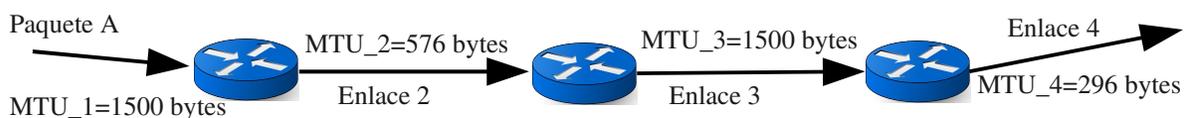
b) ¿Para qué valor de tasa de transmisión del enlace se logra la tasa de transferencia máxima alcanzable con este protocolo?

$$n = \text{tasa}_{\text{enlace}} * \text{RTT} / \text{Tamaño}_{\text{paqueteEnBits}}$$

$$\text{tasa}_{\text{enlace}} = n * \text{Tamaño}_{\text{paqueteEnBits}} / \text{RTT}$$

$$\text{tasa}_{\text{enlace}} = 2^{15} * 1500 * 8 [\text{bit}] / 150 [\text{ms}] = 2^{10} * 2^5 * 80 * 10^3 [\text{bps}] \approx 10^3 * 2^8 * 10^4 = 2,55 * 10^9 [\text{bps}] \approx 2,6 [\text{Gbps}]$$

3.- (25 puntos) Para la ruta adjunta y los enlaces 2, 3 y 4, muestre el contenido de los campos Largo, Identificador de datagrama, flag y offset de fragmentación y Time to live (TTL) del encabezado IP de los paquetes generados a partir del paquete A. Use misma notación usada para paquete A.



Paquete A:

	largo:1120	ID=X	frag.flag=0	offset:0	TTL=23	//
						//

Obs: El encabezado del paquete IP ocupa 20 bytes.

En enlace 2 el MTU es 576, luego el datagrama de 1120 debe fragmentarse. El largo de datos es de 1100 bytes, los cuales deben fragmentarse en 2 fragmentos, el primero transporta 552 bytes (notar que debe ser múltiplo de 8 porque el offset lleva el factor que multiplicado por 8 da la posición en bytes) y el segundo 548 bytes. Como el primer fragmento lleva 552 bytes, al agregar los 20 del encabezado se obtiene un tamaño de 572.

	Largo: 572	ID=X	frag.flag=1	offset:0	TTL=22	//
						//
	Largo: 568	ID=X	frag.flag=0	offset:69	TTL=22	//
						//

En enlace 3 el MTU es de 1500 bytes, luego los fragmentos del enlace 2 pasan sin fragmentaciones nuevas, pero con menor TTL.

	Largo: 572	ID=X	frag.flag=1	offset:0	TTL=21	//
						//
	Largo: 568	ID=X	frag.flag=0	offset:69	TTL=21	//
						//

En enlace 4 el MTU es de 296, luego cada fragmento debe ser nuevamente fragmentado.

El primer fragmento original lleva 552 bytes, luego es fragmentado en $272 + 272 + 8 = 552$. Los primeros fragmentos llevan 272 datos porque $292 - 20 = 272 = 34 * 8$ es el múltiplo de 8 más grande transportable. Notar que los primeros fragmentos terminan siendo de 292 bytes. El último lleva sólo 8 bytes.

	Largo: 292	ID=X	frag.flag=1	offset:0	TTL=20	//
						//
	Largo: 292	ID=X	frag.flag=1	offset:34	TTL=20	//
						//
	Largo: 28	ID=X	frag.flag=1	offset:68	TTL=20	//
						//

El segundo fragmento lleva 548 bytes = $272 + 276$, luego se fragmenta en dos paquetes. El primero con igual offset del fragmento previo y el segundo con offset $69 + 34 = 103$. Se verifica que $103 * 8 + 276 = 1100$, que es el tamaño de los datos iniciales.

	Largo: 292	ID=X	frag.flag=1	offset:69	TTL=20	//
						//
	Largo: 296	ID=X	frag.flag=0	offset:103	TTL=20	//
						//