Departamento de Ciencias de la Computación
Facultad de Ciencias Físicas y Matemáticas
Universidad de Chile
CC4303-1 Redes

CLIENTE UDP CON CORRECCIÓN DE ERRORES DE ECO EFICIENTE PARA ARCHIVOS

Tarea 2

Estudiante: Daniel Radrigán Barros

Profesores: José Piquer

Auxiliares: David Miranda

Ayudantes: Agustín Muñoz
Joaquín Salas
Paula Ovalle

Fecha de entrega: 8 de junio de 2024
Santiago, Chile

Preguntas:

1. En base a la experimentación usando selective repeat se pudo llegar a los resultados visualizados en la tabla 1. Estos resultados indican que los mejores valores se alcanzan utilizando una ventana de tamaño 5 y paquetes de tamaño 2000 bytes. La segunda mejor opción tambien es con una ventana de tamaño 5, pero con paquetes de tamaño 1000, estos 2 resultados son significativamente mejores que el resto de opciones.

Tabla 1: Resultado de los experimentos medidos en segundos

	1000	2000	5000	10000
5	10.85	7.67	28.65	94.73
10	24.07	122.18	102.29	243.98
20	32.99	273.23	125.76	-
50	65.77	333.21	-	-

2. Usando *cliente_echo3.py* nos da que la conexión con anakena se demora aproximadamente 10 segundos. Lo cual tiene sentido considerando que usa paquetes de 1500 ya que se demora parecido a los casos de paquetes de tamaños pequeños, además es importante considerar que no revisa si es que llegaron correctamente los paquetes.

3. Para comparar entre los distintos protocolos se va a realizar usando un tamaño de paquete 1000 dado que en general es el que tuvo mejores resultados para los distintos valores de ventana. Con el protocolo Stop and wait obtenemos que se demoro valores cercanos a 49 segundos en realizar el trabajo. Por otra parte con go-back-n usando una ventana 5 obtenemos resultados parecidos a stop and wait, lo que tiene sentido, ya que al no tener problemas con que el ack vuelva para avisar debido a que esta info se pasa entre threads este protocolo no se aprovecha significativamente. Estos resultados son mucho peores a los obtenidos por selective-repeat en la pregunta 1 para ventanas menores a 20.

4. Los timeouts se van a modificar en el protocolo selective-repeat, al igual que en la pregunta anterior se usara paquetes de tamaño 1000 para hacer el analisis de como cambia con estos timeouts. Cambiando el timeout a 0.01 segundos fue posible observar que no hay un cambio significativo, frente a los resultados obtenido en la pregunta 1. Por otra parte, cuando se cambio el timeout a 1 segundo fue posible observar que usando una ventana de 5 el tiempo que toma aumenta a 12.82 en promedio, mientras que para ventanas de tamaño 10 o 50 el tiempo promedio se mantiene en los mismos rangos.

5. En base a la experimentación realizada, se concluye que el protocolo es el factor más relevante. El tamaño del timeout, por otro lado, no fue tan crucial, excepto en los casos donde el tiempo era muy bajo, como con una ventana de 5 y paquetes de 1000 bytes. El tamaño de la ventana también es relevante: si es muy grande, aumenta el tiempo de transferencia. Lo mismo ocurre con el tamaño de los paquetes, ya que un mayor tamaño tiende a incrementar el tiempo de transferencia, probablemente debido a un mayor riesgo de pérdida de datos. Sin embargo, el tamaño de la ventana y el de los paquetes pueden interactuar de manera óptima, creando combinaciones que son más eficientes juntas que por separado, como se observó con la ventana de 5 y paquetes de 2000 bytes.