Tarea 6

INET Framework

Aplicación TCP

Maria Jose Monterrosa Mérida,999014670

Mario Rolando Calderon España, 999014184

Oscar Enrique Escalante Luna, 999013658

Universidad San Carlos de Guatemala, Facultad de Ingeniería

Escuela de Postgrado

Fundamentos de Modelado y Simulación

Sección: A

Catedrático: Mgtr. Carlos Leonel Ramírez

23 de marzo de 2024

Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Post Grados

Aplicación TCP

Preguntas y Actividades Adicionales

- ¿Cuál es la secuencia de mensajes para el inicio y finalización de las sesiones TCP?

Los primeros eventos observados son los relacionados con ARP (Protocolo de Resolución de Direcciones), el cual es utilizado por los dispositivos en una red para mapear direcciones IP a direcciones MAC.

La secuencia de inicio es:

1. El cliente envía un paquete de solicitud de conexión (SYN) al servidor.
2. El servidor responde con un paquete para confirmar la conexión (SYN-ACK).
3. Por último, el cliente envía un paquete de confirmación de conexión (ACK) hacia el servidor.

Event#	Time	Relevant Hops	Name	TxUpdate? / Sourc	Length / Destinat	Info / Protocol	Type
#6	1.000	client	⟶ ethernetSwitch			arpREQ 0A-AA-00-00-00-01 FF-FF-FF-FF-FF-FF	ARP REQUEST
#13	1.000 001 576	ethernetSwitch	⟶ server			arpREQ 0A-AA-00-00-00-01 FF-FF-FF-FF-FF-FF	ARP REQUEST
#20	1.000 003 152	server	⟶ ethernetSwitch			arpREPLY 0A-AA-00-00-00-02 0A-AA-00-00-00-01	ARP REPLY
#27	1.000 004 728	ethernetSwitch	⟶ client			arpREPLY 0A-AA-00-00-00-02 0A-AA-00-00-00-01	ARP REPLY
#34	1.000 006 304	client	⟶ ethernetSwitch			SYN 10.0.0.1:1025 10.0.0.2:10021	TCP
#41	1.000 007 880	ethernetSwitch	⟶ server			SYN 10.0.0.1:1025 10.0.0.2:10021	TCP
#50	1.000 009 456	server	⟶ ethernetSwitch			SYN+ACK 10.0.0.2:10021 10.0.0.1:1025	TCP
#57	1.000 011 032	ethernetSwitch	⟶ client			SYN+ACK 10.0.0.2:10021 10.0.0.1:1025	TCP

Para finalizar la sesión la serie de eventos observados son:

1. El cliente envía un paquete de finalización de conexión (FIN).
2. El servidor responde con un paquete de confirmación de finalización (FIN-ACK).
3. El cliente envía un ACK final.

Aplicación TCP

```
#6653        580.000'839'616 client ⟶ ethernetSwitch FIN 10.0.0.1:1049 10.0.0.2:10021 TCP
#6660        580.000'840'520 ethernetSwitch ⟶ server TcpAck 10.0.0.1:1049 10.0.0.2:10021 TCP
#6667        580.000'841'192 ethernetSwitch ⟶ server FIN 10.0.0.1:1049 10.0.0.2:10021 TCP
#6681        580.000'842'768 server ⟶ ethernetSwitch TcpAck 10.0.0.2:10021 10.0.0.1:1049 TCP
#6688        580.000'844'344 ethernetSwitch ⟶ client TcpAck 10.0.0.2:10021 10.0.0.1:1049 TCP
#6700        581.000'842'768 server ⟶ ethernetSwitch FIN 10.0.0.2:10021 10.0.0.1:1049 TCP
#6707        581.000'844'344 ethernetSwitch ⟶ client FIN 10.0.0.2:10021 10.0.0.1:1049 TCP
#6717        581.000'845'920 client ⟶ ethernetSwitch TcpAck 10.0.0.1:1049 10.0.0.2:10021 TCP
#6724        581.000'847'496 ethernetSwitch ⟶ server TcpAck 10.0.0.1:1049 10.0.0.2:10021 TCP
```

- ¿Qué host inicia y termina las sesiones en esta simulación?

En esta simulación el cliente inicia las sesiones TCP con el servidor al igual que inicia el proceso de finalización.

```
#34          1.000'006'304 client ⟶ ethernetSwitch SYN 10.0.0.1:1025 10.0.0.2:10021 TCP

#6653        580.000'839'616 client ⟶ ethernetSwitch FIN 10.0.0.1:1049 10.0.0.2:10021 TCP
```

- ¿Cuánto segmentos TCP se envían entre el cliente y el servidor en cada sesión?

Se envían 4 segmentos por sesión.

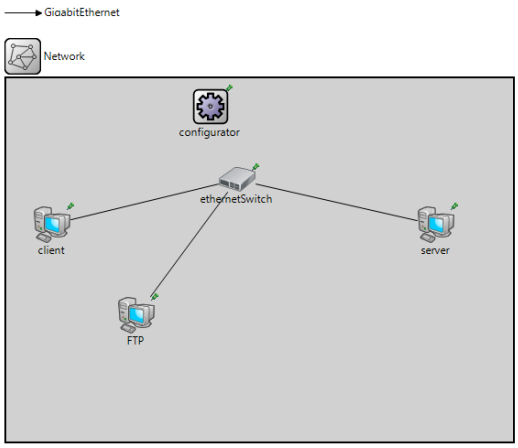
- Aumente el valor del requestLength a 1024 bytes. Con esta nueva configuración ¿Cuántos segmentos TCP se envían entre el cliente y el servidor en cada sesión? explique su respuesta

En este caso se envían 6 segmentos entre cliente y servidor.

La razón del cambio en la cantidad de segmentos TCP enviados entre el cliente y el servidor en cada sesión al aumentar el valor del requestLength a 1024 bytes es que los paquetes de datos son más grandes y pueden fragmentarse en un mayor número de segmentos TCP para

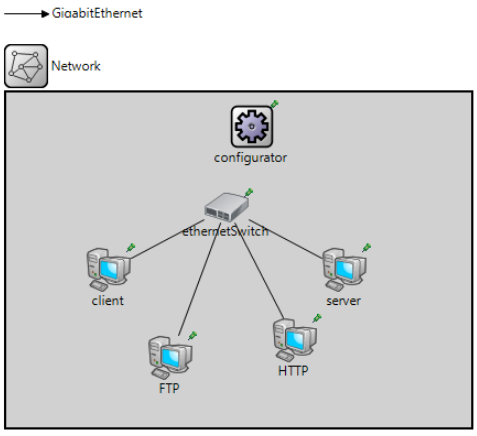
ser enviados a través de la red. Esto resulta en un aumento de la cantidad de segmentos TCP enviados entre el cliente y el servidor en cada sesión.

- Agregue un cliente y modele la generación de tráfico FTP hacia el servidor.




```
# Configuración del cliente FTP
*.FTP.numApps = 1
*.FTP.app[0].typename = "TcpBasicClientApp"
*.FTP.app[0].connectAddress = "server"
*.FTP.app[0].connectPort = 21
*.FTP.app[0].requestLength = 1B*int(truncnormal(20,5))
*.FTP.app[0].replyLength = 1B*int(exponential(1000000))
*.FTP.app[0].numRequestsPerSession = int(exponential(3))
*.FTP.app[0].thinkTime = 1s
*.FTP.app[0].idleInterval = 20s
```

- Agregue un cliente y modele la generación de tráfico HTTP 1.1 hacia el servidor




```
# Configuración del cliente HTTP
*.HTTP.numApps = 1
*.HTTP.app[0].typename = "TcpBasicClientApp"
*.HTTP.app[0].connectAddress = "server"
*.HTTP.app[0].connectPort = 21
*.HTTP.app[0].requestLength = 1B*int(truncnormal(350,20))
*.HTTP.app[0].replyLength = 1B*int(exponential(2000))
*.HTTP.app[0].numRequestsPerSession = int(exponential(5))
*.HTTP.app[0].thinkTime = 1s
*.HTTP.app[0].idleInterval = 20s
```

.

Aplicación TCP

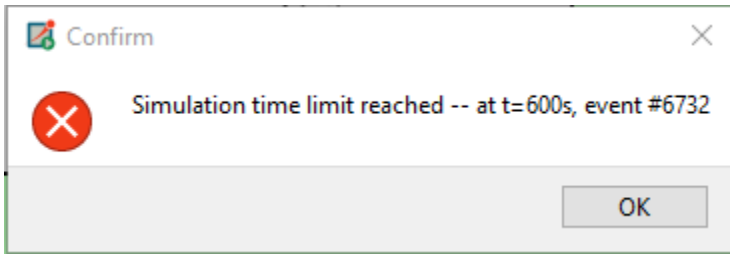
- ¿Cuál es el troughtput del tráfico recibido por el cliente para cada aplicación?

  Sugerencia, Utilice la estadística packetReceived.

TCP 100B

Utilizando las gráficas del cliente de la simulación original, se obtienen estos resultados:

packetReceived:sum(packetBytes) (scalar)	5,000
Module name	Network.client.app[0]
Type	double
Value	5,000
Interpolationmode	none
Recordingmode	sum(packetBytes)



$$Throughput = \frac{5{,}000\ bytes}{600\ segundos}$$

En este caso el troughtput es 8.33 bytes/segundo

TCP 1024B

$$Throughput = \frac{51{,}200\ bytes}{600\ segundos}$$

En este caso el troughtput es 85.33 bytes/segundo

Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Post Grados

Aplicación TCP

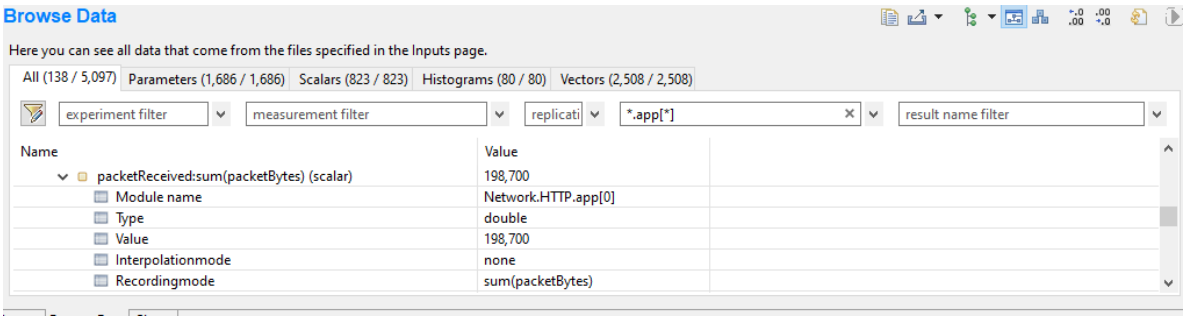
FTP

Name	Value
packetReceived:sum(packetBytes) (scalar)	77,814,525
Module name	Network.FTP.app[0]
Type	double
Value	77,814,525
Interpolationmode	none
Recordingmode	sum(packetBytes)

$$Throughput = \frac{77{,}814{,}525\ bytes}{600\ segundos}$$

En este caso el troughtput es 129,690.88 bytes/segundo

HTTP



$$Throughput = \frac{198{,}700\ bytes}{600\ segundos}$$

En este caso el troughtput es 331.17 bytes/segundo