Redes de computadoras, 9 de enero de 2025

Simulación de protocolos con NS-2

Martínez Buenrostro Jorge Rafael

correo, *molap*96@*gmail.com*
Universidad Autónoma Metropolitana
Unidad Iztapalapa, México

Cuestionario

Describa qué hace cada script ejecutado en la práctica, ilustrando las partes en donde se configura cada una de las capas del Modelo TCP/IP.

Script 1

Ya que este script no incluye nodos, enlaces, o tráfico de paquetes, vamos a describir cada parte del script.
Crea una instancia del simulador NS-2 y la asigna a la variable ns. Este objeto es el núcleo de la simulación, permitiendo condigurar nodos, tráfico, y otros elementos.

```
set ns [new Simulator]
```

Crea un archivo **out.nam** en modo escritura. Este archivo registrará todos los eventos de simulación, como el envío/recepción de paquetes y otros eventos relevantes

```
set nf [open out.nam w]
```

Le indica al simulador que registre todos los eventos de simulación en el archivo **out.nam** previamente creado.

```
$ns namtrace-all $nf
```

Define una función llamada **finish** que:

global ns nf : Declara las variables **ns** y **nf** como globales para usarlas dentro de la función.

$ns flush-trace : Asegura que toda la información de traza se escriba en el archivo antes de cerrarlo.

close $nf : Cierra el archivo de traza **out.nam**

exec nam out.nam & : Ejecuta NAM y abre el archivo **out.nam** para visualizar la simulación.

exit 0 : Finaliza la ejecución del script de manera exitosa.

```
proc finish {} {
  global ns nf
  $ns flush-trace
  close $nf
  exec nam out.nam &
  exit 0
}
```

Programa la ejecución del procedimiento **finish** después de 5 segundos de tiempo simulado. Esto significa, que independientemente de la configuración adicional, la simulación terminará a los 5 segundos.

```
$ns at 5.0 "finish"
```

Inicia la ejecución de la simulación. A partir de este punto, todos los eventos configurados en el simulador se procesan secuencialmente.

```
$ns run
```

Script 2

Como vimos en el punto anterior las siguientes líneas del script son parte de la configuración general y la creación del procedimiento de finalización.

```
set ns [new Simulator]
set nf [open out.nam w]
$ns namtrace-all $nf
proc finish {} {
  global ns nf
  $ns flush-trace
  close $nf
  exec nam out.nam &
  exit 0
}
```

Crea dos nodos (*n0 y n1*) que actuarán como origen y destino en la simulación. Esta parte del script corresponde a la **Capa de Enlace de Datos**, ya que los nodos representan interfaces de red.

```
set n0 [$ns node]
set n1 [$ns node]
```

Crea un enlace dúplex entre los nodos **n0** y **n1** con:

Ancho de banda : 1 Mbps

Retardo : 10 ms

Cola : DropTail (cola FIFO)

La relación con el modelo TCP/IP es con la **Capa de Enlace de Datos** ya que representa el enlace físico y la cola de paquetes. Así como la **Capa de Red** ya que el retardo y el ancho de banda afectan la entrega de paquetes.

```
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
```

Crea un agente UDP (*udp0*) y lo asocia al nodo **n0**. Se relaciona con el modelo TCP/IP ya que representa la **Capa de Transporte**, ya que el protocolo UDP es un protocolo en esta capa.

```
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
```

Crea una fuente de tráfico CBR (*cbr0*) con:

Tamaño de paquete : 1000 bytes

Intervalo entre paquetes : 0.005 segundos

Se conecta al agente UDP *udp0*

Representa la **Capa de Aplicación**, ya que genera tráfico constante para enviar a través de la red.

```
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 1000
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0
```

Crea un agente Null (*null0*) que actúa como receptor de tráfico en el nodo **n1**. Representa la **Capa de Transporte**, ya que recibe y descarta paquetes sin procesarlos.

```
set null0 [new Agent/Null]
$ns attach-agent $n1 $null0
```

Conecta el agente UDP (*udp0*) en el nodo **n0** con el agente Null (*null0*) en el nodo **n1**. Relaciona las **Capas de Transporte y Red**, al definir una ruta lógica entre el origen y el destino.

```
$ns connect $udp0 $null0
```

Programa el inicio del tráfico CBR a los 0.5 segundos y su detención a los 4.5 segundos de tiempo de simulación. Impacta la **Capa de Aplicación**, ya que controla cuándo se envían los datos.

```
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
```

1. Capa de Enlace de Datos
   - Creación de nodos.
   - Configuración del enlace dúplex.
2. Capa de Red
   - Configuración del enlace afecta el enrutamiento y entrega de paquetes.
3. Capa de Transporte
   - Configuración del agente UDP y receptor Null.

4. Capa de Aplicación

    - Generación de tráfico constante (*CBR*).
    - Programación del inicio y finalización del tráfico.

Script 3

Crea un simulador NS-2 **Simulator** y asigna colores para distinguir flujos de datos en NAM.

```
set ns [new Simulator]
$ns color 1 Blue
$ns color 2 Red
```

Abre archivos para registrar trazas generales **out.tr** y específicas para NAM **out.nam**.

```
set file1 [open out.tr w]
set winfile [open WinFile w]
$ns trace-all $file1
set file2 [open out.nam w]
$ns namtrace-all $file2
```

Finaliza la simulación, cierra los archivos de traza y lanza NAM para visualización.

```
proc finish {} {
  global ns file1 file2
  $ns flush-trace
  close $file1
  close $file2
  exec nam out.nam &
  exit 0
}
```

Crea nodos **n0** a **n5** y define enlaces entre ellos con las siguientes características:

Ancho de banda : 2 Mbps

Retraso : 10 ms

colas : DropTail con tamaño de 20

Además orienta visualmente los enlaces en NAM. Relaciona la **Capa de Enlace de Datos** y la **Capa de Red**. Ya que configura enlaces físicos y colas además de modelar el retardo y ancho de banda.

```
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]

$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns simplex-link $n2 $n3 0.3Mb 100ms DropTail
$ns simplex-link $n3 $n2 0.3Mb 100ms DropTail
$ns duplex-link $n3 $n4 0.5Mb 40ms DropTail
$ns duplex-link $n3 $n5 0.5Mb 30ms DropTail
```

Configura un agente TCP NewReno en el nodo **n0** y un receptor (TCPSink) en el nodo **n4**. Ajusta parámetros como tamaño de ventana y tamaño de paquetes. Relaciona la **Capa de Transporte** con un protocolo confiable (TCP).

```
set tcp [new Agent/TCP/Newreno]
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink/DelAck]
$ns attach-agent $n4 $sink
$ns connect $tcp $sink
$tcp set fid_ 1
$tcp set window_ 8000
$tcp set packetSize_ 552
```

Configura una fuente FTP sobre TCP para generar tráfico. Relaciona la **Capa de Aplicación** emulando una transferencia de archivos.

```
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTP
```

Configura un agente UDP en el nodo **n1** y un receptor (Null) en el nodo **n5**. Relaciona la **Capa de Transporte** con un protocolo no confiable (UDP).

```
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set null [new Agent/Null]
$ns attach-agent $n5 $null
$ns connect $udp $null
$udp set fid_ 2
```

Configura una fuente CBR sobre UDP para generar tráfico constante. Relaciona la **Capa de Aplicación** simulando tráfico constante en la red.

```
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set type_ CBR
$cbr set packet_size_ 1000
$cbr set rate_ 0.01mb
$cbr set random_ false
```

Inicia y detiene las aplicaciones FTP y CBR en momentos específicos. Controla la **Capa de Aplicación**, definiendo tiempos de inicio y fin.

```
$ns at 0.1 "$cbr start"
$ns at 1.0 "$ftp start"
$ns at 124.0 "$ftp stop"
$ns at 124.5 "$cbr stop"
```

Monitorea el tamaño de ventana de congestión **cwnd** de TCP y registra los valores en un archivo para análisis posterior. Relacionado con la **Capa de Transporte**, ya que mide el comportamiento de TCP.

```
proc plotWindow {tcpSource file} {
  global ns
  set time 0.1
  set now [$ns now]
  set cwnd [$tcpSource set cwnd_]
  puts $file "$now $cwnd"
  $ns at [expr $now+$time] "plotWindow $tcpSource $file"
}
$ns at 0.1 "plotWindow $tcp $winfile"
```

¿Cómo podría usarse NS-2 para explicar la relación entre el retardo de propagación y el de transmisión en un enlace?¿Qué parte de un script TCL debería modificar para ejemplificar esto?

En NS-2 se puede explicar esta relación al configurar las propiedades del enlace entre nodos, específicamente el ancho de banda y el retardo del enlace.

Retarde de propagación : Se configura directamente en la línea donde se define el enlace entre nodos, como el tercer argumento. Repreenta el tiempo que tarda una señal en propagarse desde el origen al destino.

Retarde de transmisión : Depende del tamaño del paquete y del ancho de banda configurado en el enlace.

Modificando el **ancho de banda**, se puede observar cómo afecta el tiempo necesario para transmitir paquetes másicas grandes. Mientras que cambiando el **retardo del enlace**, se puede observar cómo afecta el tiempo total para que un paquete llegue al destino, independientemente de su tamaño.

```
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
```

Figura 1: Modificación del script para ejemplificar la relación

Para el segundo ejemplo de la práctica, modifique el código para que el tamaño del paquete sea igual a 1,500 bytes. ¿Qué diferencias observa en la animación con NAM?

Cambiar el tamaño del paquete a 1,500 bytes afectará el tiempo de transmisión en el enlace. Si el tamaño del paquete aumenta, el tiempo de transmisión también aumentará, lo que resultará en una animación más lenta en NAM.

- Se observa que los paquetes tardan más tiempo en recorrer los enlaces, especialmente si hay varios paquetes en tránsito.
- El enlace puede congestionarse más rápido, lo que genera pérdida de paquetes en colas configuradas con DropTail.

```
$cbr0 set packetSize_ 1500
```

Figura 2: Código modificado

Investigue la disponibilidad de simuladores de red y realice una tabla con al menos cinco simuladores que compare sus características básicas.

Simulador	Lenguaje principal	Modelos soportados	Interfaz	Licencia
NS-2	C++, Tcl	TCP, UDP, Multicast, Routing	Basada en Scritps	Open Source GPL
NS'3	C++, Python	TCP, UDP, WiFi, IPv4/IPv6, LTE	Basada en Scripts	Open Source GPLv2
OMNeT++	C++	Redes cableadas, inálambricas, IoT	Gráfica y Scripts	Open Source Academic
Mininet	Python	Redes definidas por Software (SDN)	Línea de comandos	Open Source MIT
Packet Tracer	Propietario	Modelos de Cisco (RIP, OSPF, STP)	Gráfica	Propietario Cisco
GNS3	Python	Redes virtualizadas y físicas	Gráfica	Open Source GNU GPLv3

Detalles destacados

NS-2
- Enfoque: Simulación de protocolos de red
- Ventaja: Una gran cantidad de documentación y ejemplos debido a su uso histórico.
- Desventaja: Es más antiguo y menos actualizado que NS-3.

NS-3
- Enfoque: Simulación de redes modernas
- Ventaja: Mejor soporte para redes modernas como WiFi y LTE.
- Desventaja: Es más complejo que NS-2.

OMNeT++
- Enfoque: Modelado de redes y sistemas distribuidos
- Ventaja: Perfecto para simulaciones gráficas detalladas.
- Desventaja: No es tan popular para simulaciones prácticas como NS-3

Mininet
- Enfoque: Redes definidas por software (SDN)
- Ventaja: Ideal para probar redes definidas por software (SDN).
- Desventaja: No es tan adecuado para redes físicas tradicionales.

Packet Tracer
- Enfoque: Simulación de redes Cisco.
- Ventaja: Excelente herramienta educativa para estudiantes de redes.
- Desventaja: No es adecuado para simulaciones avanzadas fuera del ecosistema Cisco.

GNS3
- Enfoque: Redes reales con simulaciones de hardware.
- Ventaja: Soporta redes virtuales y físicas.
- Desventaja: Requiere más recursos de hardware que otros simuladores.