Práctica 1: Conceptos Avanzados de TCP

**Objetivos**

En esta práctica estudiaremos el funcionamiento del protocolo TCP. Además veremos algunos parámetros que permiten ajustar el comportamiento de las aplicaciones TCP. Finalmente se consideran algunas aplicaciones del filtrado de paquetes mediante iptables.

**Contenidos**

[Preparación del entorno para la práctica](#h.14a3ftqman5y)

[Estados de una conexión TCP](#h.lkbk11wj05sz)

[Introducción a la seguridad en el protocolo TCP](#h.72odsoledxbm)

[Traducción de direcciones (NAT) y reenvío de puertos (port-forwarding)](#h.pgwskimsb2ry)

# Preparación del entorno para la práctica

Configuraremos la topología de red que se muestra en la Figura 1, igual a la empleada en la práctica 1. En este caso, las direcciones consideradas son IPv4.

|  |
| --- |
| **Figura 1:** Topología y direccionamiento de las redes usadas en la práctica 1 |
|  |

El siguiente fichero muestra el archivo de configuración de la topología:

|  |
| --- |
| $ cat pr1.topol  Netprefix inet machine 1 0 0 machine 2 0 0  machine 3 0 0 1 1 machine 4 0 1 |

Finalmente configurar la red de todas las máquinas de la red según la siguiente tabla. Después de configurar todas las máquinas comprobar la conectividad con la orden ping.

|  |  |  |
| --- | --- | --- |
| **Máquina** | **Dirección IPv4** | **Comentarios** |
| VM1 | 192.168.0.1/24 | Añadir Router como router por defecto |
| VM2 | 192.168.0.2/24 | Añadir Router como router por defecto |
| VM3 - Router | 192.168.0.3/24 (eth0)  172.16.0.3/24 (eth1) | Activar el forwarding de paquetes. |
| VM4 | 172.16.0.1/24 | Añadir Router como router por defecto |

# Estados de una conexión TCP

En esta práctica usaremos la herramienta Netcat, que permite leer y escribir en/de conexiones de red. Netcat es muy útil para investigar y depurar el comportamiento de la red en la capa de transporte, ya que permite especificar un gran número de los parámetros de la conexión. Además para ver el estado de las conexiones de red usaremos la herramienta netstat.

***Ejercicio 1.*** Consultar las páginas de manual para nc y netstat; probar algunas de las opciones para ambos programas para familiarizarse con su comportamiento.

***Ejercicio 2.*** (LISTEN) Abrir un servidor TCP en el puerto 7777 en la VM1 usando el comando nc. Comprobar el estado de la conexión en el servidor con el comando netstat.

***Ejercicio 3.*** (ESTABLISHED) En la VM2 iniciar una conexión cliente al servidor arrancado en el ejercicio anterior. Comprobar el estado de la conexión e identificar los parámetros (dirección IP y puerto) usando el comando netstat:

* Iniciar el servidor usando la opción -s. Comprobar si es posible la conexión desde la VM1 usando como dirección destino localhost. Observar la diferencia con el comando anterior (sin opción -s) usando netstat.
* Iniciar el servidor e intercambiar un único carácter con el cliente. Con ayuda de wireshark, observar los mensajes intercambiados (especialmente los números de secuencia, confirmación y flags TCP) y determinar cuántos bytes (y número de mensajes) han sido necesarios.

***Ejercicio 4.*** (TIMEWAIT) Cerrar la conexión en el servidor Ctl+C y comprobar el estado de la conexión con netstat. Usar la opción -o para determinar el valor del del temporizador TIMEWAIT.

***Ejercicio 5.*** (SYN-SENT y SYN-RCVD) El comando iptables permite filtrar paquetes según los flags TCP del segmento (opción --tcp-flags):

* Fijar una regla que permita filtrar las conexiones en el servidor (VM1) de forma que deje al cliente en el estado SYN-SENT. Comprobar el resultado con el comando netstat en el cliente (VM2).
* Fijar una regla que permita filtrar las conexiones en el cliente (VM2) de forma que deje al servidor en el estado SYN-RCVD. Comprobar que esta regla deja al servidor también en el estado LAST-ACK después de hacer Ctrl+C en el cliente.
* Usando el programa wireshark estudiar los mensajes intercambiados especialmente los números de secuencia , confirmación y flags TCP.
* Con ayuda de netstat (usando la opción -o) determinar cuántas retransmisiones se realizan y con qué frecuencia.

**Nota:** La regla debe ser lo suficientemente restrictiva para afectar sólo a las conexiones al servidor netcat. Después de cada ejercicio eliminar las reglas de filtrado.

***Ejercicio 6.*** Finalmente, intentar una conexión a un puerto cerrado del servidor (ej. 7778) y con ayuda de la herramienta wireshark observar los mensajes TCP intercambiados, especialmente los flags TCP.

SYN Cliente a servidor

RST Servidor a cliente

ACK Cliente a servidor

# Introducción a la seguridad en el protocolo TCP

Diferentes aspectos del protocolo TCP pueden aprovecharse para comprometer la seguridad del sistema. En este apartado vamos a estudiar dos: ataques DoS basados en SYN\_FLOOD y técnicas de exploración de puertos.

***Ejercicio 1.*** El ataque SYN\_FLOOD consiste en saturar un servidor por el envío masivo de mensajes con el flag SYN.

* (Cliente VM2) Para envíar paquetes TCP con los datos de interés usaremos el comando hping3 (estudiar la página de manual). En este caso queremos enviar mensajes con el flag SYN y lo más rápido posible al puerto 23 del servidor (servidor telnet). Opcionalmente podría enmascararse el origen usando una dirección IP arbitraria.
* (Servidor VM1) Estudiar el comportamiento de la máquina, en términos del número de paquetes recibidos. Comprobar si es posible la conexión al servicio telnet (23)
* (Servidor VM1) Activar la defensa syncookies en el servidor con el comando sysctl (net.ipv4.tcp\_syncookies).

***Ejercicio 2.*** (Técnica CONNECT) Netcat permite explorar puertos usando la técnica CONNECT que intenta establecer una conexión a un puerto determinado. En función de la respuesta (SYN+ACK o RST) es posible determinar si hay proceso escuchando.

* (Servidor VM1) Abrir un servidor en el puerto 7777
* (Cliente VM2) Explorar el rango de puertos 7775-7780 usando nc, en este caso usar las opciones de exploración (-z) y de salida detallada (-v). **Opcional:** Incluir en un bucle la sentencia de exploración de un puerto para automatizar el proceso.
* Con ayuda de wireshark observar los paquetes intercambiados.

**Opcional.** La herramienta nmap permite realizar diferentes tipos de exploración de puertos, que emplean estrategias más eficientes. Estas estrategias (SYN stealth, ACK stealth, FIN-ACK stealth…) son más rápidas que la anterior y se basan en el funcionamiento del protocolo TCP. Estudiar la página de manual de nmap (PORT SCANNING TECHNIQUES) y emplearlas para explorar los puertos del servidor. Comprobar con wireshark los mensajes intercambiados.

**Opciones TCP**

El comportamiento de la conexión TCP se puede controlar con varias opciones que se incluyen en la cabecera en los mensajes SYN y que son configurables en el sistema operativo. La siguiente tabla incluye alguna de las opciones y las variables asociadas del kernel:

|  |  |  |  |
| --- | --- | --- | --- |
| **Opción TCP** | **Parámetro del Kernel** | **Propósito** | **Valor por defecto** |
| Escalado de la ventana | net.ipv4.tcp\_window\_scaling | is an option to increase the [receive window](https://en.wikipedia.org/wiki/Receive_window) size allowed in [Transmission Control Protocol](https://en.wikipedia.org/wiki/Transmission_Control_Protocol) above its former maximum value of 65,535 bytes |  |
| Temporizador keepalive | net.ipv4.tcp\_keepalive\_time  net.ipv4.tcp\_keepalive\_probes  net.ipv4.tcp\_keepalive\_intvl | Tiempo que permanece la conexión activa, nº de sondas que se envía para ver si la conexión es activa, intervalo de tiempo en el que se envían |  |
| Marcas de Tiempo | net.ipv4.tcp\_timestamps | Ayuda el tiempo total de ida y vuelta desde que se envía un paquete TCP al servidor #remoto y se recibe su correspondiente ACK. |  |
| ACK selectivos | net.ipv4.tcp\_sack | Evitar largas retransmisiones e identifica selectivamente los paquetes que se deben #retransmitir. |  |
| Tamaño mínimo de segmento base | net.ipv4.tcp\_base\_mss |  |  |

***Ejercicio 1.*** Con ayuda del comando sysctl y la bibliografía recomendada completar la tabla anterior.

***Ejercicio 2.*** Abrir el servidor en el puerto 7777 y realizar una conexión desde la VM cliente. Con ayuda de wireshark estudiar el valor de las opciones que se intercambian durante la conexión. Variar algunos de los parámetros anteriores (ej. no usar SACKS) y observar el resultado en una nueva conexión.

# Traducción de direcciones (NAT) y reenvío de puertos (port-forwarding)

En esta sección supondremos que la red que conecta el router (VM3) con la VM4 es pública y que no puede encaminar el tráfico 192.168.0.0/24. Además asumiremos que la IP del router es dinámica.

***Ejercicio 1.*** Configurar la traducción de direcciones dinámica en Router :

* (VM3 - Router) Configurar el router para que haga NAT (masquerade) sobre la interfaz eth1 usando el comando iptables.
* (VM1) Comprobar la conexión entre el VM1 y VM4 con la orden ping.
* (VM4 y VM1) Usando wireshark determinar la IP origen y destino de los ICMP de echo request y reply en ambas redes.

***Ejercicio 2.*** Acceso a un servidor en la red privada:

* (VM1) Arrancar el servidor netcat en el puerto 7777
* (VM3 - Router) Usando el comando iptables reenviar las conexiones al router al puerto 80 a la máquina VM1 al puerto 7777
* (VM4) Conectarse al puerto 80 de router con netcat y comprobar el resultado en VM1. Analizar el tráfico intercambiado con wireshark, especialmente los puertos y direcciones IP origen y destino en ambas redes.

ASOR\_1:

ip link set eth0 up

ip address add 192.168.0.1/24 dev eth0

ip route add default via 192.168.0.3 dev eth0

ASOR\_2:

ip link set eth0 up

ip address add 192.168.0.2/24 dev eth0

ip route add default via 192.168.0.3 dev eth0

ASOR\_3:

ip link set eth0 up

ip address add 192.168.0.3/24 dev eth0

ip link set eth1 up

ip address add 172.16.0.2/24 dev eth1

sysctl -w net.ipv4.ip\_forward=1

ip route add default via 172.16.0.1 dev eth1

ASOR\_4:

ip link set eth0 up

ip address add 172.16.0.1/24 dev eth0

Estados de una conexión TCP

Ejercicio 1

man nc

man netstat

Ejercicio 2

En ASOR\_1:

- nc -l -p 7777

- netstat -n -t -l

Ejercicio 3:

Enunciado:

ASOR\_2:

nc 192.168.0.1 7777

En ASOR\_1 ó ASOR\_2:

netstat -n -t -p 7777

Estado de la conexión: ESTABLISHED

Dirección y puerto del servidor: 192.168.0.1:7777

Dirección y puerto del cliente: 192.168.0.2:40286

Apartado 1:

nc sin -s:

ASOR\_1:

- nc -l -p 7777

- nc localhost 7777

nc con -s:

En ASOR\_1:

- nc -l -p 7777 -s 192.168.0.1

- nc localhost 7777

Usando la opción -s con direccion de origen 192.168.0.1 no te deja iniciar

conexión desde la misma máquina usando localhost. En cambio, si no

especificas el -s si te deja. Esto es porque con -s automáticamente crea una conexión donde el puerto 7777

de la máquina esté siendo usado para que no te puedas conectar con tu misma máquina.

Sin -s

Active Internet connections (w/o servers)

Proto Recv-Q Send-Q Local Address Foreign Address State PID/Program name

Con -s

Active Internet connections (w/o servers)

Proto Recv-Q Send-Q Local Address Foreign Address State PID/Program name

tcp 0 0 localhost:52715 localhost:7777 TIME\_WAIT

Apartado 2:

Desde ASOR\_1:

nc -l -p 7777

Desde ASOR\_2:

nc 192.168.0.1 7777

Sin contar los ARP, han sido necesarios 8 mensajes TCP ocupando un total de 2\*74 + 66\*5 + 68 = 546 bytes

Ejercicio 4

netstat -n -o -t -p 7777

Active Internet connections (w/o servers)

Proto Recv-Q Send-Q Local Address Foreign Address State PID/Program name Timer

tcp 0 0 192.168.0.2:57395 192.168.0.1:7777 TIME\_WAIT - timewait (32,14/0/0)

Ejercicio 5

5.1 (ASOR\_1)iptables -A INPUT -p tcp --dport 7777 -m state --state NEW -j DROP

Desde ASOR\_2:

netstat -n -t -o -p 7777

Active Internet connections (w/o servers)

Proto Recv-Q Send-Q Local Address Foreign Address State PID/Program name Timer

tcp 0 1 192.168.0.2:57396 192.168.0.1:7777 SYN\_SENT 1430/nc on (9,46/2/0)

5.2 (ASOR\_2)iptables -A OUTPUT -p tcp --tcp-flags ALL ACK -j DROP

Al establecer conexión con el cliente, el servidor queda así:

netstat -o -t -p 7777

Active Internet connections (w/o servers)

Proto Recv-Q Send-Q Local Address Foreign Address State PID/Program name Timer

tcp 0 0 192.168.0.1:7777 192.168.0.2:57398 SYN\_RECV - on (10,13/2/0)

Después de cerrar la conexión en el cliente:

Active Internet connections (w/o servers)

Proto Recv-Q Send-Q Local Address Foreign Address State PID/Program name Timer

tcp 0 1 192.168.0.1:7777 192.168.0.2:57399 LAST\_ACK - on (16,86/0/0)

Ejercicio 6

Desde ASOR\_1:

- wireshark &

Desde ASOR\_2:

- nc 192.168.0.1 7778

Observar wireshark: Cliente envía SYN y Servidor responde con RST, ACK.

Introducción a la seguridad en el protocolo TCP Falta testearla

Ejercicio 1: SYN FLOOD

Desde ASOR\_2:

man hping3

hping3 --flood --rand-source -p 23 --syn 192.168.0.1

telnet 192.168.0.1

No responde el telnet

Si usas netstat en ASOR\_1 y salen muchos SYN\_RECV. Es que te están haciendo un ataque SYN\_FLOOD

Para que evitar este ataque hay que hacer lo siguiente:

En ASOR\_1:

sysctl net.ipv4.tcp\_syncookies=1

En ASOR\_2:

telnet 192.168.0.1

Ejercicio 2:

En ASOR\_1:

nc -l -p 7777

En ASOR\_2:

nc -vz 192.168.0.1 7775-7780

Usamos nmap ya que nc no funciona en la máquina:

nmap -p 7775-7780 192.168.0.1

Opciones TCP

-- 1

tcp\_window\_scaling = 1 [http://tools.ietf.org/html/rfc1323#section-2] Permite aumentar el tamaño de la RW a mas de 64K, algo necesario para altas velocidades de transmision con alta latencia

tcp\_keepalive\_time = 7200 [] Tiempo necesario (en silencio) para revisar si una conexion sigue viva

tcp\_keepalive\_probes = 9 [] Numero de pruebas tras el tiempo anterior superado, para ver si sigue viva la conexion

tcp\_keepalive\_intvl = 75 [] Intervalo entre pruebas

tcp\_timestamps = 1 [http://tools.ietf.org/html/rfc1323#section-4] Sirven para ayudar a TCP a determinar en que orden se enviaron los paquetes

tcp\_sack = 1 [] Sirve para activar los SACK: Selective-ACK

tcp\_base\_mss = 512 [] Indica el tamaño maximo de segmento base

-- 2

sysctl -w net.ipv4.tcp\_sack = 0 -- VM1

[Al hacer el netcat la VM2, rechaza usar SACK la VM1]

Traducción de direcciones (NAT) y reenvío de puertos (port-forwarding) Funcionando

Ejercicio 1

(ASOR\_3) iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE. Esto cambia

la dirección origen del paquete para encaminar el tráfico.

Se hace el ping y se ve que el ping se hace con de ASOR\_3 con ASOR\_4.

Se hace el ping y se ve que el ping se hace con de ASOR\_3 con ASOR\_4.

Se hace el ping y se ve que el ping se hace con de ASOR\_3 con ASOR\_4.

Se hace el ping y se ve que el ping se hace con de ASOR\_3 con ASOR\_4.

Esto se ve mejor cuando se hace telnet con nat y sin nat.

Con nat, el telnet se hace de ASOR\_3 con ASOR\_4.

Sin nat, el telnet se hace de ASOR\_1 con ASOR\_4.

(ASOR\_3) iptables -t nat -A PREROUTING -d 172.16.0.2 -p tcp --dport 80 -j DNAT --to 192.168.0.1:7777