

Práctica 1.2. 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.



Para cada ejercicio, se tienen que proporcionar los **comandos utilizados con sus correspondientes salidas**, las **capturas de pantalla de Wireshark realizadas**, y la **información requerida de manera específica**.

Activar el portapapeles bidireccional en las máquinas (menú Dispositivos) para copiar la salida de los comandos. Las capturas de pantalla se realizarán usando también Virtualbox (menú Ver).

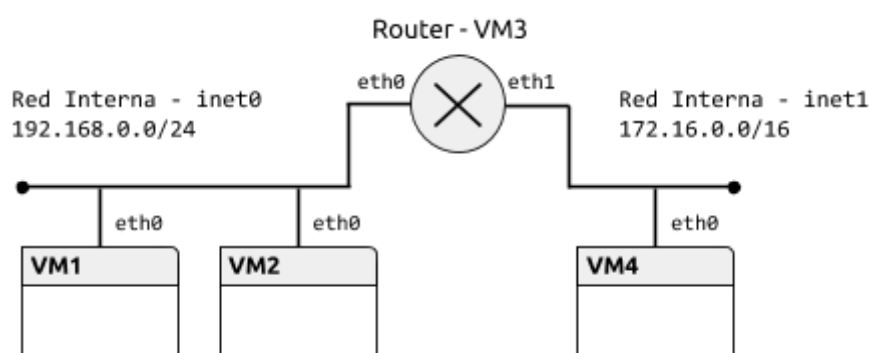
Las **credenciales de la máquina virtual** son: usuario cursoredes, con contraseña cursoredes.

Contenidos

- Preparación del entorno para la práctica
- Estados de una conexión TCP
- Introducción a la seguridad en el protocolo TCP
- Opciones y parámetros TCP
- Traducción de direcciones (NAT) y reenvío de puertos (*port forwarding*)

Preparación del entorno para la práctica

Configuraremos la topología de red que se muestra en la siguiente figura, igual a la empleada en la práctica anterior.



El contenido del fichero de configuración de la topología debe ser el siguiente:

```
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 encaminador por defecto |
| VM2 | 192.168.0.2/24 | Añadir Router como encaminador por defecto |
| Router - VM3 | 192.168.0.3/24 (eth0) 172.16.0.3/16 (eth1) | Activar el <i>forwarding</i> de paquetes |
| VM4 | 172.16.0.4/16 | Añadir Router como encaminador por defecto |

Estados de una conexión TCP

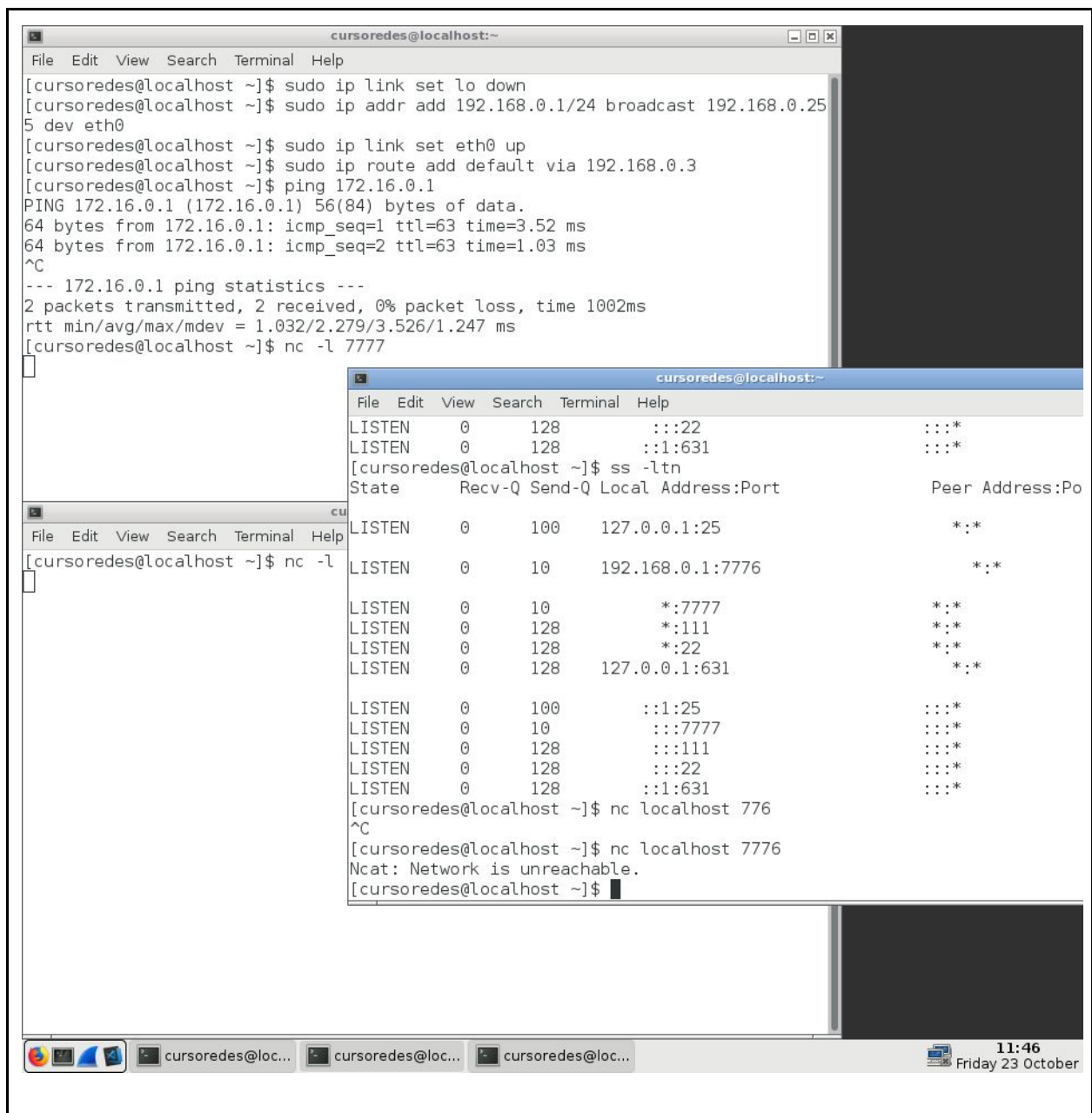
En esta parte usaremos la herramienta Netcat, que permite leer y escribir en conexiones de red. Netcat es muy útil para investigar y depurar el comportamiento de la red en la capa de transporte, ya que permite especificar un gran número de los parámetros de la conexión. Además para ver el estado de las conexiones de red usaremos el comando ss (similar a netstat, pero más moderno y completo).

Ejercicio 1. Consultar las páginas de manual de nc y ss. En particular, consultar las siguientes opciones de ss: -a, -l, -n, -t y -o. 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 VM1 usando el comando nc -l 7777. Comprobar el estado de la conexión en el servidor con el comando ss -ltn. Abrir otro servidor en el puerto 7776 en VM1 usando el comando nc -l 192.168.0.1 7776. Observar la diferencia entre ambos servidores usando ss. Comprobar que no es posible la conexión desde VM1 con localhost como dirección destino usando el comando nc localhost 7776.

Adjuntar la salida del comando ss correspondiente a los servidores

```
cursoredes@localhost:~  
File Edit View Search Terminal Help  
[cursoredes@localhost ~]$ sudo ip link set lo down  
[cursoredes@localhost ~]$ sudo ip addr add 192.168.0.1/24 broadcast 192.168.0.255 dev eth0  
[cursoredes@localhost ~]$ sudo ip link set eth0 up  
[cursoredes@localhost ~]$ sudo ip route add default via 192.168.0.3  
[cursoredes@localhost ~]$ ping 172.16.0.1  
PING 172.16.0.1 (172.16.0.1) 56(84) bytes of data.  
64 bytes from 172.16.0.1: icmp_seq=1 ttl=63 time=3.52 ms  
64 bytes from 172.16.0.1: icmp_seq=2 ttl=63 time=1.03 ms  
^C  
--- 172.16.0.1 ping statistics ---  
2 packets transmitted, 2 received, 0% packet loss, time 1002ms  
rtt min/avg/max/mdev = 1.032/2.279/3.526/1.247 ms  
[cursoredes@localhost ~]$ nc -l 7777  
  
cursoredes@localhost:~  
File Edit View Search Terminal Help  
[cursoredes@localhost ~]$ -ss ltn  
bash: -ss: command not found...  
[cursoredes@localhost ~]$ ss -ltn  
State      Recv-Q Send-Q Local Address:Port      Peer Address:Port  
LISTEN     0      100  127.0.0.1:25           *:*  
LISTEN     0      10   *:7777                 *:*  
LISTEN     0     128   *:111                  *:*  
LISTEN     0     128   *:22                   *:*  
LISTEN     0     128  127.0.0.1:631          *:*  
LISTEN     0     100  :::1:25                :::*  
LISTEN     0      10  :::7777                 :::*  
LISTEN     0     128  :::111                  :::*  
LISTEN     0     128  :::22                   :::*  
LISTEN     0     128  :::1:631                :::*  
[cursoredes@localhost ~]$
```



Ejercicio 3. (ESTABLISHED) En VM2, iniciar una conexión cliente al primer servidor arrancado en el ejercicio anterior usando el comando `nc 192.168.0.1 7777`.

- Comprobar el estado de la conexión e identificar los parámetros (dirección IP y puerto) con el comando `ss -tn`.
- Iniciar una captura con Wireshark. Intercambiar un único carácter con el cliente y observar los mensajes intercambiados (especialmente los números de secuencia, confirmación y flags TCP) y determinar cuántos bytes (y número de mensajes) han sido necesarios.

Adjuntar la salida del comando `ss` correspondiente a la conexión y una captura de pantalla de Wireshark

The screenshot shows a Linux desktop with two main windows. The top window is Wireshark 1.10.14, capturing traffic on the eth0 interface. It displays a list of five packets:

| No. | Time | Source | Destination | Protoc | Length | Info |
|-----|------------|-------------------|-------------------|--------|--------|--|
| 1 | 0.00000000 | 192.168.0.2 | 192.168.0.1 | TCP | 74 | 60718 → cbt [SYN, Seq=0 Win=29200 Len=0 MSS=1460 |
| 2 | 0.00045105 | 192.168.0.1 | 192.168.0.2 | TCP | 74 | cbt → 60718 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len |
| 3 | 0.00046767 | 192.168.0.2 | 192.168.0.1 | TCP | 66 | 60718 → cbt [ACK] Seq=1 Ack=1 Win=29312 Len=0 TS |
| 4 | 5.01253635 | CadmusCo_2a:9b:16 | CadmusCo_79:56:e5 | ARP | 60 | Who has 192.168.0.2? Tell 192.168.0.1 |
| 5 | 5.01255214 | CadmusCo_79:56:e5 | CadmusCo_2a:9b:16 | ARP | 42 | 192.168.0.2 is at 08:00:27:79:56:e5 |

The bottom window is a terminal titled 'cursoredes@localhost:~'. It shows the output of the command 'ss -tn':

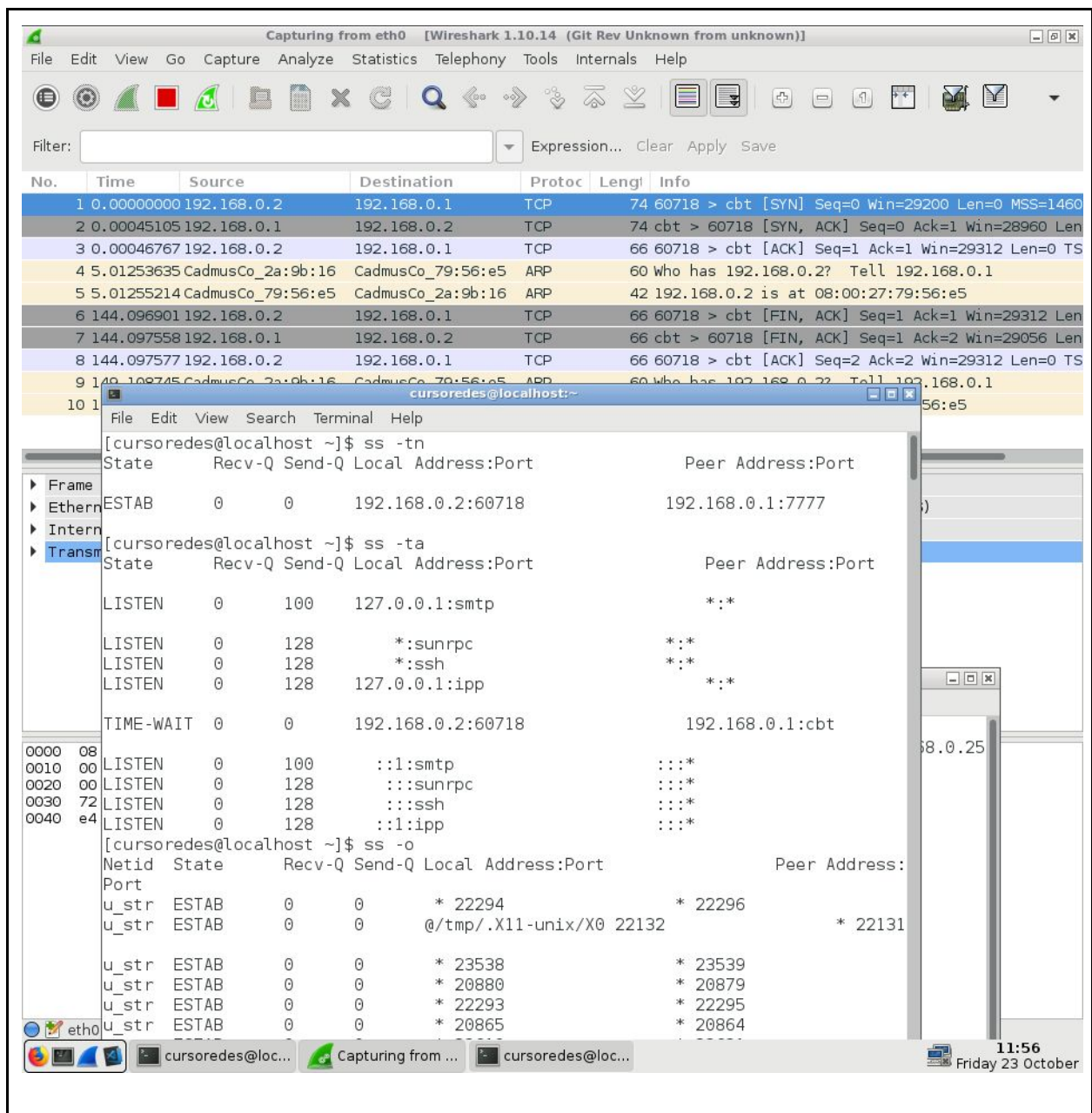
```

[cursoredes@localhost ~]$ ss -tn
State      Recv-Q  Send-Q  Local Address:Port      Peer Address:Port
ESTAB      0        0      192.168.0.2:60718      192.168.0.1:7777
[cursoredes@localhost ~]$
  
```

The terminal also shows a packet capture on eth0 at the bottom of the window, with timestamps and hex data visible.

Ejercicio 4. (TIME-WAIT) Cerrar la conexión en el cliente (con Ctrl+C) y comprobar el estado de la conexión usando `ss -ta`. Usar la opción `-o` de `ss` para observar el valor del temporizador TIME-WAIT.

Adjuntar la salida del comando `ss` correspondiente a la conexión



Ejercicio 5. (SYN-SENT y SYN-RCVD) El comando iptables permite filtrar paquetes según los flags TCP del segmento con la opción `--tcp-flags` (consultar la página de manual iptables-extensions). Usando esta opción:

- Fijar una regla en el servidor (VM1) que bloquee un mensaje del acuerdo TCP de forma que el cliente (VM2) se quede en el estado SYN-SENT. Comprobar el resultado con `ss -ta` en el cliente.
- Borrar la regla anterior y fijar otra en el cliente que bloquee un mensaje del acuerdo TCP de forma que el servidor se quede en el estado SYN-RCVD. Comprobar el resultado con `ss -ta` en el servidor. Además, esta regla debe dejar al servidor también en el estado LAST-ACK después de cerrar la conexión (con Ctrl+C) en el cliente. Usar la opción `-o` de `ss` para determinar cuántas retransmisiones se realizan y con qué frecuencia.

Adjuntar los comandos iptables utilizados y la salida del comando `ss` correspondiente a las conexiones

Comando: `iptables -A INPUT -p tcp --tcp-flags SYN SYN-j DROP`

***eth0 [Wireshark 1.10.14 (Git Rev Unknown from unknown)]**

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: Expression... Clear Apply Save

| No. | Time | Source | Destination | Protoc | Length | Info |
|-----|------------|-------------|-------------|--------|--------|--|
| 1 | 0.00000000 | 192.168.0.2 | 192.168.0.1 | TCP | 74 | 60718 → cbt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 |

cursoredes@localhost:~

File Edit View Search Terminal Help

```
[cursoredes@localhost ~]$ ss -ta
```

| State | Recv-Q | Send-Q | Local Address:Port | Peer Address:Port |
|----------|--------|--------|--------------------|-------------------|
| LISTEN | 0 | 100 | 127.0.0.1:smtp | *:* |
| LISTEN | 0 | 128 | *:sunrpc | *:* |
| LISTEN | 0 | 128 | *:ssh | *:* |
| LISTEN | 0 | 128 | 127.0.0.1:ipp | *:* |
| SYN-SENT | 0 | 1 | 192.168.0.2:60720 | 192.168.0.1:cbt |
| LISTEN | 0 | 100 | :::1:smtp | :::* |
| LISTEN | 0 | 128 | :::sunrpc | :::* |
| LISTEN | 0 | 128 | :::ssh | :::* |
| LISTEN | 0 | 128 | :::1:ipp | :::* |

```
[cursoredes@localhost ~]$
```

0000 08 00
0010 00 3c
0020 00 01
0030 72 10
0040 e4 3a

File: "/tmp/wireshark_pcapng_eth0_..." Packets: 10 · Displayed: 10 (100.0%) · Dropped: 0 (0.0%) | Profile: Default

cursoredes@loc... *eth0 [Wiresha... cursoredes@loc...

12:05
Friday 23 October

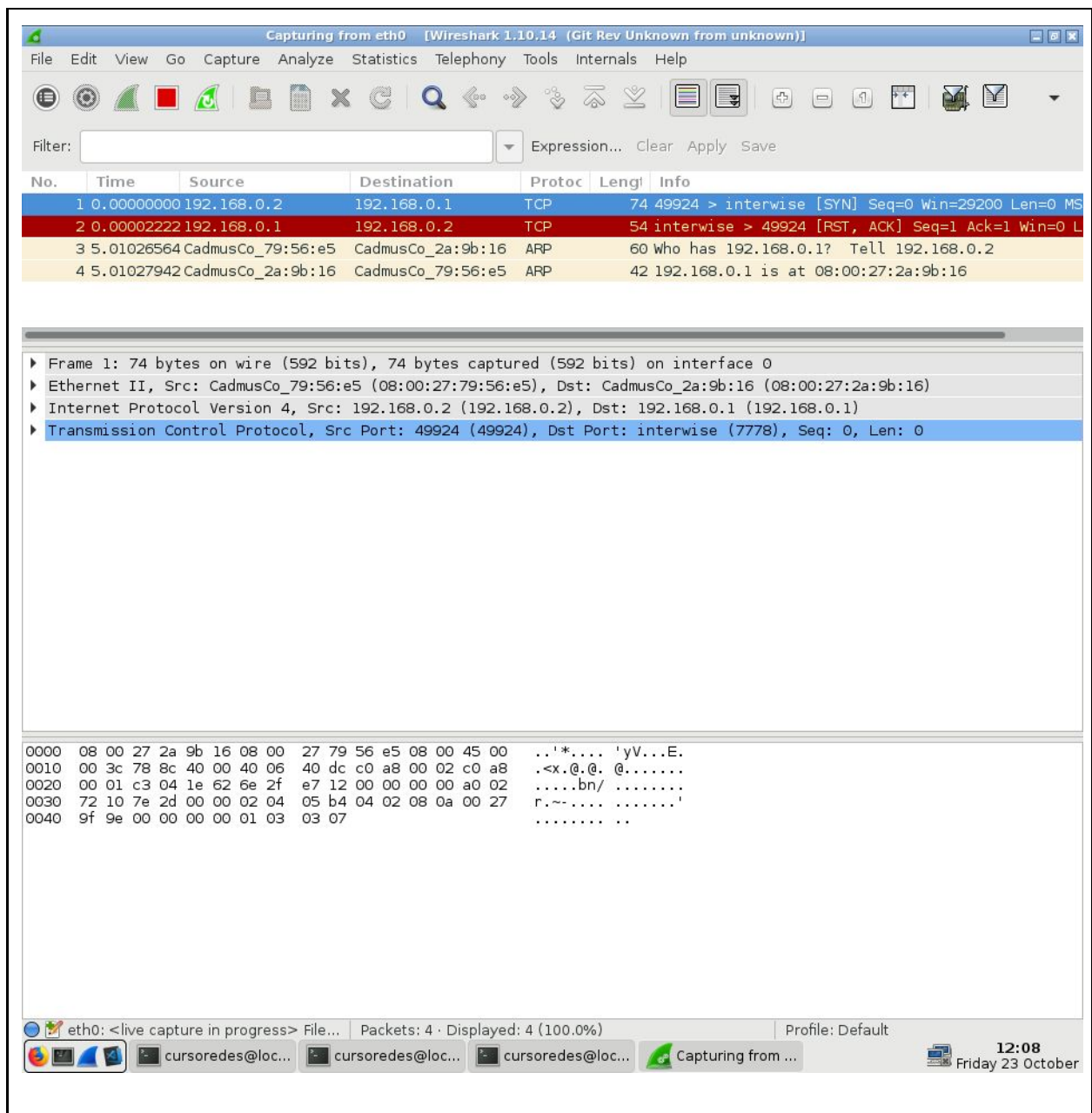
Comando: `iptables -A OUTPUT -p tcp --tcp-flags ALL ACK -j DROP`

```
cursoredes@localhost:~  
File Edit View Search Terminal Help  
[cursoredes@localhost ~]$ sudo ip link set lo down  
[cursoredes@localhost ~]$ sudo ip addr add 192.168.0.1/24 broadcast 192.168.0.255 dev eth0  
[cursoredes@localhost ~]$ sudo ip link set eth0 up  
[cursoredes@localhost ~]$ sudo ip route add default via 192.168.0.3  
[cursoredes@localhost ~]$ ping 172.16.0.1  
PING 172.16.0.1 (172.16.0.1) 56(84) bytes of data:  
64 bytes from 172.16.0.1: icmp_seq=1 ttl=64 time=0.054 ms  
64 bytes from 172.16.0.1: icmp_seq=2 ttl=64 time=0.054 ms  
^C  
[cursoredes@localhost ~]$ nc localhost 7776  
Ncat: Network is unreachable.  
[cursoredes@localhost ~]$ sudo iptables -A INPUT -p tcp --tcp-flags SYN SYN -j DROP  
[cursoredes@localhost ~]$ sudo iptables -D INPUT -p tcp --tcp-flags SYN SYN -j DROP  
[cursoredes@localhost ~]$ ss -ta  
^C  
[cursoredes@localhost ~]$ ss -ta  
State      Recv-Q Send-Q Local Address:Port Peer Address:Port  
LISTEN     0      100  127.0.0.1:smtp    *:*  
LISTEN     0      10  *:*:cbt           *:*  
[cursoredes@localhost ~]$ ss -ta  
[cursoredes@localhost ~]$ ss -ta  
State      Recv-Q Send-Q Local Address:Port Peer Address:Port  
LISTEN     0      128  192.168.0.1:cbt   192.168.0.2:60722  
LISTEN     0      128  *:*:sunrpc          *:*  
LISTEN     0      128  *:*:ssh             *:*  
LISTEN     0      128  127.0.0.1:ipp     *:*  
LISTEN     0      100  :::1:smtp         :::*  
LISTEN     0      10  :::cbt            :::*  
LISTEN     0      128  :::sunrpc         :::*  
LISTEN     0      128  :::ssh            :::*  
LISTEN     0      128  :::1:ipp          :::*  
[cursoredes@localhost ~]$ ss -ta  
[cursoredes@localhost ~]$ ss -ta  
State      Recv-Q Send-Q Local Address:Port Peer Address:Port  
LISTEN     0      100  127.0.0.1:smtp    *:*  
LISTEN     0      128  *:*:sunrpc          *:*  
LISTEN     0      128  *:*:ssh             *:*  
LISTEN     0      128  127.0.0.1:ipp     *:*  
LAST-ACK   0      1  192.168.0.1:cbt   192.168.0.2:60722  
LISTEN     0      100  :::1:smtp         :::*  
LISTEN     0      128  :::sunrpc         :::*  
LISTEN     0      128  :::ssh            :::*  
LISTEN     0      128  :::1:ipp          :::*  
[cursoredes@localhost ~]$
```

Ejercicio 6. Iniciar una captura con Wireshark. Intentar una conexión a un puerto cerrado del servidor (ej. 7778) y observar los mensajes TCP intercambiados, especialmente los flags TCP.

Adjuntar una captura de pantalla de Wireshark

Comando en VM2: nc 192.168.0.1 7778



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 TCP SYN *flood* y técnicas de exploración de puertos.

Ejercicio 7. El ataque TCP SYN *flood* consiste en saturar un servidor mediante el envío masivo de mensajes SYN.

- (Cliente VM2) Para evitar que el atacante responda al mensaje SYN+ACK del servidor con un mensaje RST que liberaría los recursos, bloquear los mensajes SYN+ACK en el atacante con iptables.
- (Cliente VM2) Para enviar paquetes TCP con los datos de interés usaremos el comando hping3 (estudiar la página de manual). En este caso, enviar mensajes SYN al puerto 22 del servidor (ssh) lo más rápido posible (*flood*).

- (Servidor VM1) Estudiar el comportamiento de la máquina, en términos del número de paquetes recibidos. Comprobar si es posible la conexión al servicio ssh.

Repetir el ejercicio desactivando el mecanismo SYN cookies en el servidor con el comando sysctl (parámetro net.ipv4.tcp_syncookies).

Adjuntar los comandos iptables y hping3 utilizados. Describir el comportamiento de la máquina con y sin el mecanismo SYN cookies

Comando: `sudo iptables -A INPUT -p tcp --tcp-flags ALL SYN, ACK -j DROP`

Comando: `sudo hping3 --flood --syn -p 22 192.168.0.1`

Ejercicio 8. (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 un proceso escuchando.

- (Servidor VM1) Abrir un servidor en el puerto 7777.
- (Cliente VM2) Explorar, de uno en uno, el rango de puertos 7775-7780 usando nc, en este caso usar las opciones de exploración (-z) y de salida detallada (-v). **Nota:** La versión de nc instalada no soporta rangos de puertos.
- Con ayuda de Wireshark, observar los paquetes intercambiados.

Adjuntar los comandos nc utilizados y su salida

Comando: `nc -l -p 7777`

Comando: `nc -z -v 192.168.0.1 7775 ----- repetir este comando con el resto de puertos`

Capturing from eth0 [Wireshark 1.10.14 (Git Rev Unknown from unknown)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: Expression... Clear Apply Save

| No. | Time | Source | Destination | Protoc | Lengt | Info |
|-----|------------|-------------|-------------|--------|-------|---|
| 1 | 0.00000000 | 192.168.0.2 | 192.168.0.1 | TCP | 74 | 41874 → 7776 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 |
| 2 | 0.00038334 | 192.168.0.1 | 192.168.0.2 | TCP | 60 | 7776 → 41874 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0 |

Frame 1: 134 bytes on wire (1072 bits) captured (1072 bits) on interface eth0
 Ethernet II, Src: VirtualBox__08:00:00:00:00:00, Dst: VirtualBox__08:00:00:00:00:00, Type: 0x0000
 Internet Protocol Version 4, Src: 192.168.0.2, Dst: 192.168.0.1
 Transmission Control Protocol, Src Port: 41874, Dst Port: 7776, Seq=0, Win=29200, Len=0

Transmissions: 2 (100.0%)

eth0: <live capture in progress> File... Packets: 2 · Displayed: 2 (100.0%) Profile: Default

13:07 Friday 23 October

```

[cursoredes@localhost ~]$ sudo ip link set lo down
[cursoredes@localhost ~]$ sudo ip addr add 192.168.0.2/24 broadcast 192.168.0.255 dev eth0
[cursoredes@localhost ~]$ sudo ip link set eth0 up
[cursoredes@localhost ~]$ sudo ip route add default via 192.168.0.3
[cursoredes@localhost ~]$ nc 192.168.0.1 7777
^C
[cursoredes@localhost ~]$ nc 192.168.0.1 7777
Ncat: Connection timed out.
[cursoredes@localhost ~]$ nc 192.168.0.1 7777
^C
[cursoredes@localhost ~]$ nc 192.168.0.1 7778
Ncat: Connection refused.
[cursoredes@localhost ~]$ nc -z -v 192.168.0.1 7775
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connection refused.
[cursoredes@localhost ~]$ nc -z -v 192.168.0.1 7776
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connection refused.
[cursoredes@localhost ~]$
  
```


Capturing from eth0 [Wireshark 1.10.14 (Git Rev Unknown from unknown)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: Expression... Clear Apply Save

| No. | Time | Source | Destination | Protoc | Lengt | Info |
|-----|------------|-------------|-------------|--------|-------|--|
| 1 | 0.00000000 | 192.168.0.2 | 192.168.0.1 | TCP | 74 | 60730 > cbt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 |
| 2 | 0.00056504 | 192.168.0.1 | 192.168.0.2 | TCP | 74 | cbt > 60730 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 |
| 3 | 0.00059527 | 192.168.0.2 | 192.168.0.1 | TCP | 66 | 60730 > cbt [ACK] Seq=1 Ack=1 Win=29312 Len=0 TS |
| 4 | 0.00079971 | 192.168.0.2 | 192.168.0.1 | TCP | 66 | 60730 > cbt [FIN, ACK] Seq=1 Ack=1 Win=29312 Len=0 |
| 5 | 0.00112324 | 192.168.0.1 | 192.168.0.2 | TCP | 66 | cbt > 60730 [FIN, ACK] Seq=1 Ack=2 Win=29056 Len=0 |
| 6 | 0.00113173 | 192.168.0.2 | 192.168.0.1 | TCP | 66 | 60730 > cbt [ACK] Seq=2 Ack=2 Win=29312 Len=0 TS |

Frame 1: Ethernet II, Internet Protocol Version 4, Transmission Control Protocol

Transmissions

```

[cursoredes@localhost ~]$ sudo ip link set lo down
[cursoredes@localhost ~]$ sudo ip addr add 192.168.0.2/24 broadcast 192.168.0.255 dev eth0
[cursoredes@localhost ~]$ sudo ip link set eth0 up
[cursoredes@localhost ~]$ sudo ip route add default via 192.168.0.3
[cursoredes@localhost ~]$ nc 192.168.0.1 7777
^C
[cursoredes@localhost ~]$ nc 192.168.0.1 7777
Ncat: Connection timed out.
[cursoredes@localhost ~]$ nc 192.168.0.1 7777
^C
[cursoredes@localhost ~]$ nc 192.168.0.1 7778
Ncat: Connection refused.
[cursoredes@localhost ~]$ nc -z -v 192.168.0.1 7775
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connection refused.
[cursoredes@localhost ~]$ nc -z -v 192.168.0.1 7776
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connection refused.
[cursoredes@localhost ~]$ nc -z -v 192.168.0.1 7777
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connected to 192.168.0.1:7777.
Ncat: 0 bytes sent, 0 bytes received in 0.01 seconds.
[cursoredes@localhost ~]$

```

eth0: <live capture in progress> File... Packets: 6 · Displayed: 6 (100.0%) Profile: Default

cursoredes@loc... Capturing from ... cursoredes@loc...

13:07 Friday 23 October

Capturing from eth0 [Wireshark 1.10.14 (Git Rev Unknown from unknown)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: Expression... Clear Apply Save

| No. | Time | Source | Destination | Protoc | Lengt | Info |
|-----|------------|-------------|-------------|--------|-------|--|
| 1 | 0.00000000 | 192.168.0.2 | 192.168.0.1 | TCP | 74 | 49932 → interwise [SYN] Seq=0 Win=29200 Len=0 MS |
| 2 | 0.00045286 | 192.168.0.1 | 192.168.0.2 | TCP | 60 | interwise → 49932 [RST, ACK] Seq=1 Ack=1 Win=0 L |

Frame 1: Ethernet Internet Transmis

```

[cursoredes@localhost ~]$ sudo ip link set eth0 up
[cursoredes@localhost ~]$ sudo ip route add default via 192.168.0.3
[cursoredes@localhost ~]$ nc 192.168.0.1 7777
^C
[cursoredes@localhost ~]$ nc 192.168.0.1 7777
Ncat: Connection timed out.
[cursoredes@localhost ~]$ nc 192.168.0.1 7777
^C
[cursoredes@localhost ~]$ nc 192.168.0.1 7778
Ncat: Connection refused.
[cursoredes@localhost ~]$ nc -z -v 192.168.0.1 7775
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connection refused.
[cursoredes@localhost ~]$ nc -z -v 192.168.0.1 7776
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connection refused.
[cursoredes@localhost ~]$ nc -z -v 192.168.0.1 7777
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connected to 192.168.0.1:7777.
Ncat: 0 bytes sent, 0 bytes received in 0.01 seconds.
[cursoredes@localhost ~]$ nc -z -v 192.168.0.1 7778
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connection refused.
[cursoredes@localhost ~]$

```

eth0: <live capture in progress> File... Packets: 2 · Displayed: 2 (100.0%) Profile: Default

13:07 Friday 23 October

Capturing from eth0 [Wireshark 1.10.14 (Git Rev Unknown from unknown)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: Expression... Clear Apply Save

| No. | Time | Source | Destination | Protoc | Lengt | Info |
|-----|------------|-------------|-------------|--------|-------|--|
| 1 | 0.00000000 | 192.168.0.2 | 192.168.0.1 | TCP | 74 | 60762 → vstat [SYN] Seq=0 Win=29200 Len=0 MSS=14 |
| 2 | 0.00048030 | 192.168.0.1 | 192.168.0.2 | TCP | 60 | vstat → 60762 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0 |

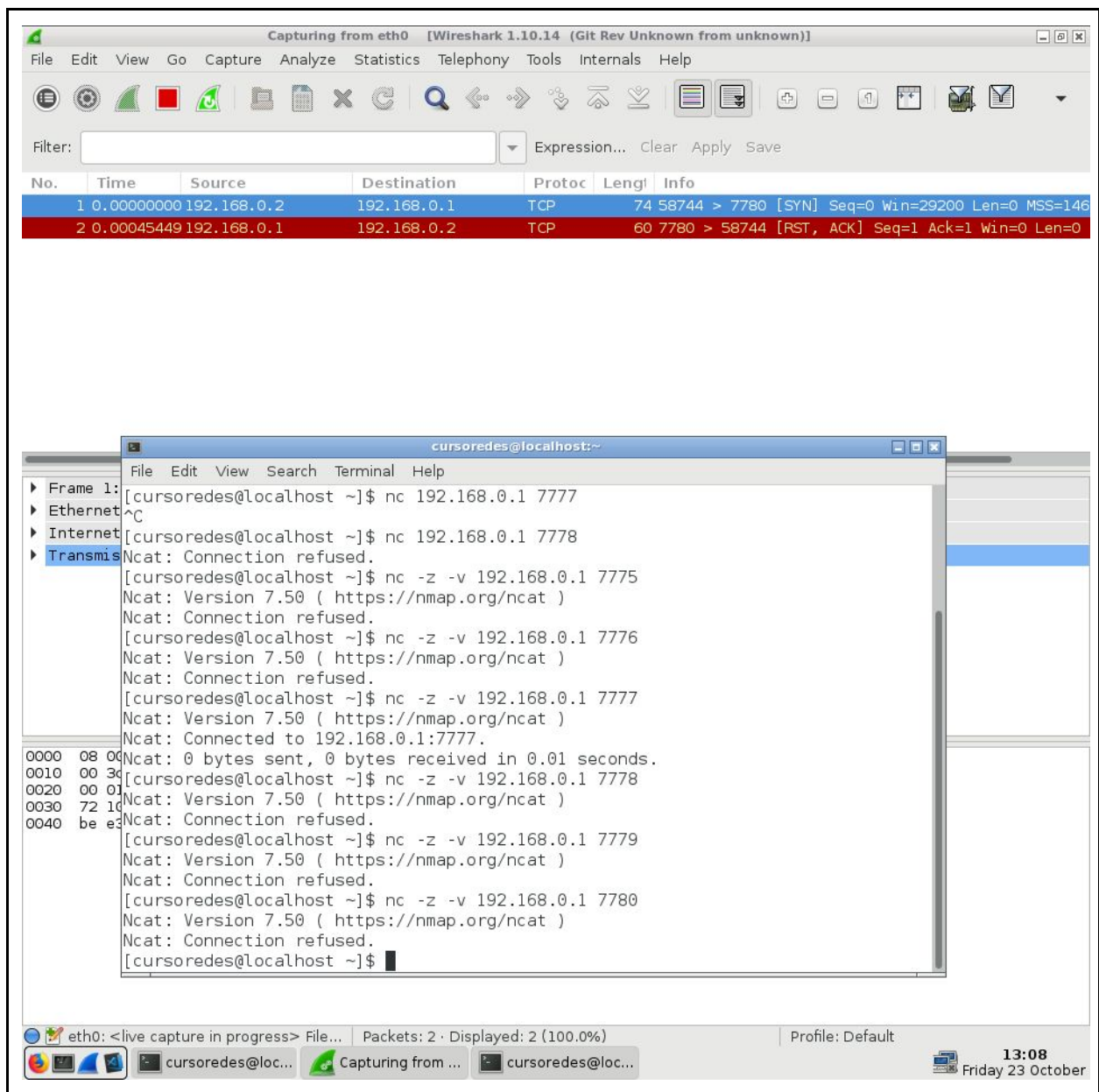
Frame 1: Ethernet
Internet
Transmission

```

[cursoredes@localhost ~]$ nc 192.168.0.1 7777
Ncat: Connection timed out.
[cursoredes@localhost ~]$ nc 192.168.0.1 7777
^C
[cursoredes@localhost ~]$ nc 192.168.0.1 7778
Ncat: Connection refused.
[cursoredes@localhost ~]$ nc -z -v 192.168.0.1 7775
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connection refused.
[cursoredes@localhost ~]$ nc -z -v 192.168.0.1 7776
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connection refused.
[cursoredes@localhost ~]$ nc -z -v 192.168.0.1 7777
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connected to 192.168.0.1:7777.
Ncat: 0 bytes sent, 0 bytes received in 0.01 seconds.
[cursoredes@localhost ~]$ nc -z -v 192.168.0.1 7778
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connection refused.
[cursoredes@localhost ~]$ nc -z -v 192.168.0.1 7779
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connection refused.
[cursoredes@localhost ~]$
  
```

eth0: <live capture in progress> File... Packets: 2 · Displayed: 2 (100.0%) Profile: Default

13:08 Friday 23 October



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*...) 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 y parámetros de 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 por medio de parámetros del kernel.

Ejercicio 9. Con ayuda del comando `sysctl` y la bibliografía recomendada, completar la siguiente tabla con parámetros que permiten modificar algunas opciones de TCP:

| Parámetro del kernel | Propósito | Valor por defecto |
|-----------------------------|---|-------------------|
| net.ipv4.tcp_window_scaling | Aumentar tamaño de la ventana si ambos extremos lo soportan | 1 |
| net.ipv4.tcp_timestamps | Permiten identificar con bajo coste cuándo se generó el mensaje | 1 |
| net.ipv4.tcp_sack | Si se pierden paquetes, solo se retransmiten estos y no toda la secuencia | 1 |

Ejercicio 10. Iniciar una captura de Wireshark. Abrir el servidor en el puerto 7777 y realizar una conexión desde la VM cliente. Estudiar el valor de las opciones que se intercambian durante la conexión. Variar algunos de los parámetros anteriores (ej. no usar ACKs selectivos) y observar el resultado en una nueva conexión.

Adjuntar una captura de pantalla de Wireshark donde se muestren las opciones TCP

The screenshot displays the Wireshark interface with a capture on eth0. The packet list shows the following packets:

| No. | Time | Source | Destination | Protoc | Length | Info |
|-----|------------|-------------------|-------------------|--------|--------|--|
| 1 | 0.00000000 | 192.168.0.2 | 192.168.0.1 | TCP | 74 | 60738 > cbt [SYN Seq=0 Win=29200 Len=0 MSS=1460 |
| 2 | 0.00003621 | 192.168.0.1 | 192.168.0.2 | TCP | 74 | cbt > 60738 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 |
| 3 | 0.00042635 | 192.168.0.2 | 192.168.0.1 | TCP | 66 | 60738 > cbt [ACK] Seq=1 Ack=1 Win=29312 Len=0 TS |
| 4 | 0.00080029 | 192.168.0.2 | 192.168.0.1 | TCP | 66 | 60738 > cbt [FIN, ACK] Seq=1 Ack=1 Win=29312 Len=0 |
| 5 | 0.00084336 | 192.168.0.1 | 192.168.0.2 | TCP | 66 | cbt > 60738 [FIN, ACK] Seq=1 Ack=2 Win=29056 Len=0 |
| 6 | 0.00110127 | 192.168.0.2 | 192.168.0.1 | TCP | 66 | 60738 > cbt [ACK] Seq=2 Ack=2 Win=29312 Len=0 TS |
| 7 | 5.01209239 | CadmusCo_2a:9b:16 | CadmusCo_79:56:e5 | ARP | 42 | Who has 192.168.0.2? Tell 192.168.0.1 |
| 8 | 5.01259046 | CadmusCo_79:56:e5 | CadmusCo_2a:9b:16 | ARP | 60 | 192.168.0.2 is at 08:00:27:79:56:e5 |

The packet details pane for packet 6 shows:

- Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
- Ethernet II, Src: CadmusCo_79:56:e5 (08:00:27:79:56:e5), Dst: CadmusCo_2a:9b:16 (08:00:27:2a:9b:16)
- Internet Protocol Version 4, Src: 192.168.0.2 (192.168.0.2), Dst: 192.168.0.1 (192.168.0.1)
- Transmission Control Protocol, Src Port: 60738 (60738), Dst Port: cbt (7777), Seq: 0, Len: 0

The packet bytes pane shows the raw data in hexadecimal and ASCII:

```

0000  08 00 27 2a 9b 16 08 00 27 79 56 e5 08 00 45 00  ..'*.... 'yV...E.
0010  00 3c 5a 9f 40 00 40 06 5e c9 c0 a8 00 02 c0 a8  .<Z.@.^.....
0020  00 01 ed 42 1e 61 48 37 f5 d6 00 00 00 00 a0 02  ...B.aH7 .....
0030  72 10 5b 30 00 00 02 04 05 b4 04 02 08 0a 00 70  r.[0.... .....p
0040  af 49 00 00 00 00 01 03 03 07                    .I.....

```

Ejercicio 11. Con ayuda del comando `sysctl` y la bibliografía recomendada, completar la siguiente tabla con parámetros que permiten configurar el temporizador *keepalive*:

| Parámetro del kernel | Propósito | Valor por defecto |
|--|--|-------------------|
| <code>net.ipv4.tcp_keepalive_time</code> | Tiempo a partir del cual se comienzan a enviar mensajes <i>keepalive</i> | 7200s |
| <code>net.ipv4.tcp_keepalive_probes</code> | Núm. máx. De mensajes <i>keepalive</i> que se enviarán antes de terminar la conexión | 9 |
| <code>net.ipv4.tcp_keepalive_intvl</code> | Intervalo de tiempo entre señales <i>keepalive</i> | 75 |

Traducción de direcciones (NAT) y reenvío de puertos (*port forwarding*)

En esta sección supondremos que la red que conecta Router con VM4 es pública y que no puede encaminar el tráfico `192.168.0.0/24`. Además, asumiremos que la dirección IP de Router es dinámica.

Ejercicio 12. Configurar la traducción de direcciones dinámica en Router:

- (Router) Usando `iptables`, configurar Router para que haga SNAT (*masquerade*) sobre la interfaz `eth1`. Iniciar una captura de Wireshark en los dos interfaces.
- (VM1) Comprobar la conexión entre VM1 y VM4 con la orden `ping`.
- (Router) Analizar con Wireshark el tráfico intercambiado, especialmente los puertos y direcciones IP origen y destino en ambas redes

Adjuntar el comando `iptables` utilizado y una captura de pantalla de Wireshark

Comando Router: `sudo iptables -t nat -A POSTROUTING -s 192.168.0.0/24 -o eth1 -j MASQUERADE`

Comando VM1: `sudo ping 172.16.0.1`

Wireshark 1.10.14 (Git Rev Unknown from unknown)

Filter: Expression... Clear Apply Save

| No. | Time | Source | Destination | Protoc | Lengt | Info |
|-----|------------|-------------------|-------------------|--------|-------|---|
| 1 | 0.00000000 | 192.168.0.1 | 172.16.0.1 | ICMP | 98 | Echo (ping) request id=0x0ddb, seq=1/256, ttl=6 |
| 2 | 0.00005521 | 172.16.0.2 | 172.16.0.1 | ICMP | 98 | Echo (ping) request id=0x0ddb, seq=1/256, ttl=6 |
| 3 | 0.00061323 | 172.16.0.1 | 172.16.0.2 | ICMP | 98 | Echo (ping) reply id=0x0ddb, seq=1/256, ttl=6 |
| 4 | 0.00063156 | 172.16.0.1 | 192.168.0.1 | ICMP | 98 | Echo (ping) reply id=0x0ddb, seq=1/256, ttl=6 |
| 5 | 1.00140710 | 192.168.0.1 | 172.16.0.1 | ICMP | 98 | Echo (ping) request id=0x0ddb, seq=2/512, ttl=6 |
| 6 | 1.00146813 | 172.16.0.2 | 172.16.0.1 | ICMP | 98 | Echo (ping) request id=0x0ddb, seq=2/512, ttl=6 |
| 7 | 1.00263486 | 172.16.0.1 | 172.16.0.2 | ICMP | 98 | Echo (ping) reply id=0x0ddb, seq=2/512, ttl=6 |
| 8 | 1.00267138 | 172.16.0.1 | 192.168.0.1 | ICMP | 98 | Echo (ping) reply id=0x0ddb, seq=2/512, ttl=6 |
| 9 | 2.00376729 | 192.168.0.1 | 172.16.0.1 | ICMP | 98 | Echo (ping) request id=0x0ddb, seq=3/768, ttl=6 |
| 10 | 2.00381807 | 172.16.0.2 | 172.16.0.1 | ICMP | 98 | Echo (ping) request id=0x0ddb, seq=3/768, ttl=6 |
| 11 | 2.00500697 | 172.16.0.1 | 172.16.0.2 | ICMP | 98 | Echo (ping) reply id=0x0ddb, seq=3/768, ttl=6 |
| 12 | 2.00504353 | 172.16.0.1 | 192.168.0.1 | ICMP | 98 | Echo (ping) reply id=0x0ddb, seq=3/768, ttl=6 |
| 13 | 3.00590813 | 192.168.0.1 | 172.16.0.1 | ICMP | 98 | Echo (ping) request id=0x0ddb, seq=4/1024, ttl= |
| 14 | 3.00686284 | 172.16.0.1 | 192.168.0.1 | ICMP | 98 | Echo (ping) reply id=0x0ddb, seq=4/1024, ttl= |
| 15 | 3.00598115 | 172.16.0.2 | 172.16.0.1 | ICMP | 98 | Echo (ping) request id=0x0ddb, seq=4/1024, ttl= |
| 16 | 3.00683135 | 172.16.0.1 | 172.16.0.2 | ICMP | 98 | Echo (ping) reply id=0x0ddb, seq=4/1024, ttl= |
| 17 | 5.00709682 | CadmusCo_d4:4a:6a | CadmusCo_9c:52:23 | ARP | 42 | Who has 172.16.0.1? Tell 172.16.0.2 |
| 18 | 5.00712820 | CadmusCo_1a:ae:3d | CadmusCo_2a:9b:16 | ARP | 42 | Who has 192.168.0.1? Tell 192.168.0.3 |
| 19 | 5.00742169 | CadmusCo_2a:9b:16 | CadmusCo_1a:ae:3d | ARP | 60 | 192.168.0.1 is at 08:00:27:2a:9b:16 |
| 20 | 5.00739940 | CadmusCo_9c:52:23 | CadmusCo_d4:4a:6a | ARP | 60 | 172.16.0.1 is at 08:00:27:9c:52:23 |

Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0

- Ethernet II, Src: CadmusCo_2a:9b:16 (08:00:27:2a:9b:16), Dst: CadmusCo_1a:ae:3d (08:00:27:1a:ae:3d)
- Internet Protocol Version 4, Src: 192.168.0.1 (192.168.0.1), Dst: 172.16.0.1 (172.16.0.1)
- Internet Control Message Protocol

```

0000  08 00 27 1a ae 3d 08 00 27 2a 9b 16 08 00 45 00  ..'..'. '*...E.
0010  00 54 fd 01 40 00 40 01 d0 ec c0 a8 00 01 ac 10  .T..@.@. ....
0020  00 01 08 00 69 18 0d db 00 01 a1 c0 92 5f 00 00  ....i... ..
0030  00 00 8d 18 01 00 00 00 00 00 10 11 12 13 14 15  ....
0040  16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25  .... !"#%&
0050  26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35  &'()*+,-./012345
0060  36 37                                     67
  
```

Terminal:

```

cursoredes@localhost:~$ sudo conntrack -L
icmp 1 3 src=192.168.0.1 dst=172.16.0.1 type=8 code=0 id=3547 src=172.16.0.1 dst=172.16.0.2
type=0 code=0 id=3547 mark=0 use=1
conntrack v1.4.4 (conntrack-tools): 1 flow entries have been shown.
  
```

System Tray: 13:38 Friday 23 October

Ejercicio 13. ¿Qué parámetro se utiliza, en lugar del puerto origen, para relacionar las solicitudes con las respuestas? Comprueba la salida del comando `conntrack -L` o, alternativamente, el fichero `/proc/net/nf_conntrack`.

Adjuntar la salida del comando `conntrack` y responder a la pregunta

```

[cursoredes@localhost ~]$ sudo conntrack -L
icmp 1 3 src=192.168.0.1 dst=172.16.0.1 type=8 code=0 id=3547 src=172.16.0.1 dst=172.16.0.2
type=0 code=0 id=3547 mark=0 use=1
conntrack v1.4.4 (conntrack-tools): 1 flow entries have been shown.
  
```

Ejercicio 14. Acceso a un servidor en la red privada:

- (Router) Usando iptables, reenviar las conexiones (DNAT) del puerto 80 de Router al puerto 7777 de VM1. Iniciar una captura de Wireshark en los dos interfaces.
- (VM1) Arrancar el servidor en el puerto 7777 con nc.
- (VM4) Conectarse al puerto 80 de Router con nc y comprobar el resultado en VM1.
- (Router) Analizar con Wireshark el tráfico intercambiado, especialmente los puertos y direcciones IP origen y destino en ambas redes.

Adjuntar el comando iptables utilizado y una captura de pantalla de Wireshark

Comando: `sudo iptables -t nat -A PREROUTING -d 172.16.0.2 -p tcp --dport 80 -j DNAT --to 192.168.0.1:7777`

Filter: Expression... Clear Apply Save

| No. | Time | Source | Destination | Protoc | Lengt | Info |
|-----|------------|-------------------|-------------------|--------|-------|--|
| 1 | 0.00000000 | 172.16.0.1 | 172.16.0.2 | TCP | 74 | 59422 > http [SYN] Seq=0 Win=29200 Len=0 MSS=1460 |
| 2 | 0.00003814 | 172.16.0.1 | 192.168.0.1 | TCP | 74 | 59422 > cbt [SYN] Seq=0 Win=29200 Len=0 MSS=1460 |
| 3 | 0.00048484 | 192.168.0.1 | 172.16.0.1 | TCP | 74 | cbt > 59422 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 |
| 4 | 0.00050263 | 172.16.0.2 | 172.16.0.1 | TCP | 74 | http > 59422 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 |
| 5 | 0.00101176 | 172.16.0.1 | 192.168.0.1 | TCP | 66 | 59422 > cbt [ACK] Seq=1 Ack=1 Win=29312 Len=0 TS=0 |
| 6 | 0.00110411 | 172.16.0.1 | 192.168.0.1 | TCP | 66 | 59422 > cbt [FIN, ACK] Seq=1 Ack=1 Win=29312 Len=0 TS=0 |
| 7 | 0.00100089 | 172.16.0.1 | 172.16.0.2 | TCP | 66 | 59422 > http [ACK] Seq=1 Ack=1 Win=29312 Len=0 TS=0 |
| 8 | 0.00109679 | 172.16.0.1 | 172.16.0.2 | TCP | 66 | 59422 > http [FIN, ACK] Seq=1 Ack=1 Win=29312 Len=0 TS=0 |
| 9 | 0.00156547 | 192.168.0.1 | 172.16.0.1 | TCP | 66 | cbt > 59422 [FIN, ACK] Seq=1 Ack=2 Win=29056 Len=0 TS=0 |
| 10 | 0.00157759 | 172.16.0.2 | 172.16.0.1 | TCP | 66 | http > 59422 [FIN, ACK] Seq=1 Ack=2 Win=29056 Len=0 TS=0 |
| 11 | 0.00193262 | 172.16.0.1 | 172.16.0.2 | TCP | 66 | 59422 > http [ACK] Seq=2 Ack=2 Win=29312 Len=0 TS=0 |
| 12 | 0.00194291 | 172.16.0.1 | 192.168.0.1 | TCP | 66 | 59422 > cbt [ACK] Seq=2 Ack=2 Win=29312 Len=0 TS=0 |
| 13 | 5.00550810 | CadmusCo_1a:ae:3d | CadmusCo_2a:9b:16 | ARP | 42 | Who has 192.168.0.1? Tell 192.168.0.3 |
| 14 | 5.00571826 | CadmusCo_2a:9b:16 | CadmusCo_1a:ae:3d | ARP | 60 | 192.168.0.1 is at 08:00:27:2a:9b:16 |
| 15 | 5.00553936 | CadmusCo_d4:4a:6a | CadmusCo_9c:52:23 | ARP | 42 | Who has 172.16.0.1? Tell 172.16.0.2 |
| 16 | 5.00562959 | CadmusCo_9c:52:23 | CadmusCo_d4:4a:6a | ARP | 60 | Who has 172.16.0.2? Tell 172.16.0.1 |
| 17 | 5.00564165 | CadmusCo_d4:4a:6a | CadmusCo_9c:52:23 | ARP | 42 | 172.16.0.2 is at 08:00:27:d4:4a:6a |
| 18 | 5.00573516 | CadmusCo_9c:52:23 | CadmusCo_d4:4a:6a | ARP | 60 | 172.16.0.1 is at 08:00:27:9c:52:23 |

Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 1

Ethernet II, Src: CadmusCo_9c:52:23 (08:00:27:9c:52:23), Dst: CadmusCo_d4:4a:6a (08:00:27:d4:4a:6a)

Internet Protocol Version 4, Src: 172.16.0.1 (172.16.0.1), Dst: 172.16.0.2 (172.16.0.2)

Transmission Control Protocol, Src Port: 59422 (59422), Dst Port: http (80), Seq: 0, Len: 0

0000 08 00 27 d4 4a 6a 08 00 27 9c 52 23 08 00 45 00 ..'.Jj..'.R#..E.
 0010 00 3c 38 84 40 00 40 06 aa 14 ac 10 00 01 ac 10 .<8.@.@.
 0020 00 02 e8 1e 00 50 0d 09 e8 87 00 00 00 00 a0 02 ...I..
 0030 72 10 cf 49 00 00 02 04 05 b4 04 02 08 0a 00 7f r..I....
 0040 d0 03 00 00 00 00 01 03 03 07

eth0 and eth1: <live capture in prog... Packets: 18 · Displayed: 18 (100.0%) Profile: Default

cursoredes@loc... Capturing from ... 13:45 Friday 23 October