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).

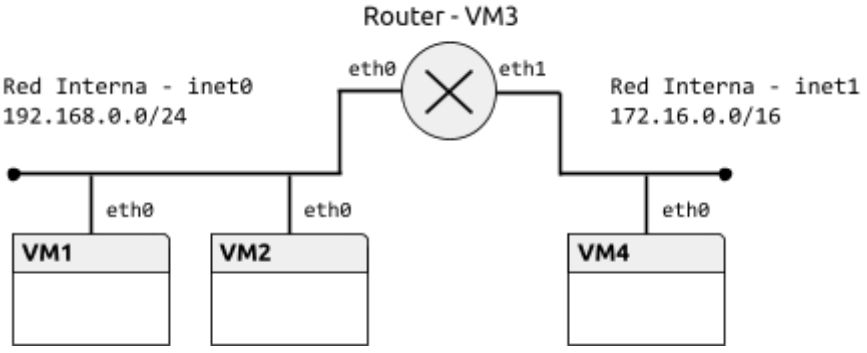
Las **credenciales de la máquina virtual** son: usuario `cursoredes`, con contraseña `cursoredes`.

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 *forwarding* de paquetes
VM4	`172.16.0.4/16`	Añadir Router como encaminador por defecto

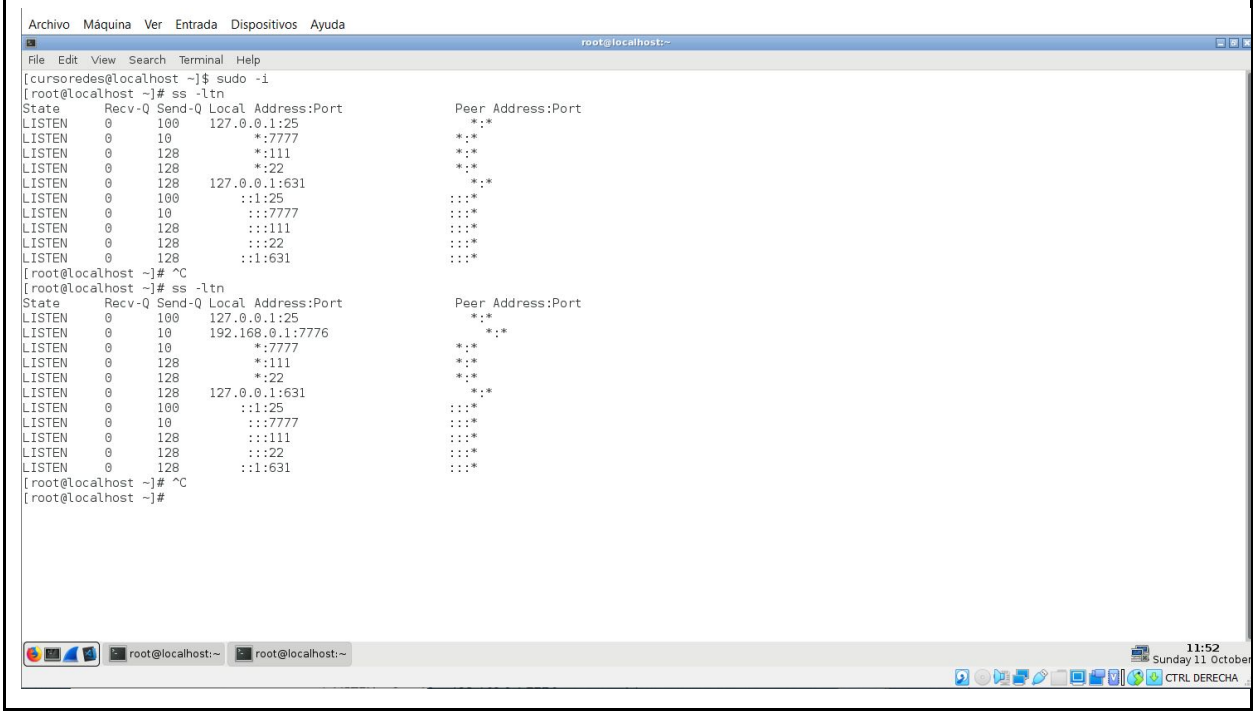
Estados de una conexión TCP

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

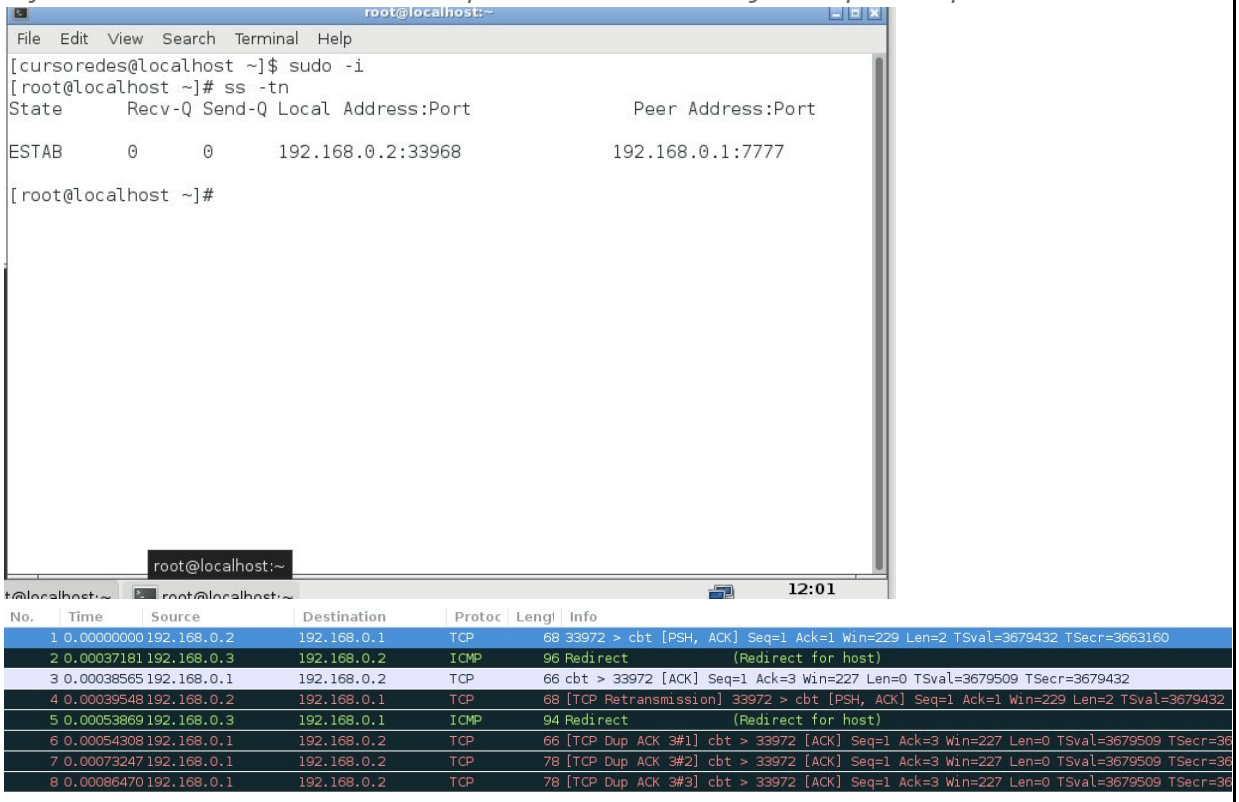


 # nc localhost 7776
Ncat: Connection refused.

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`.

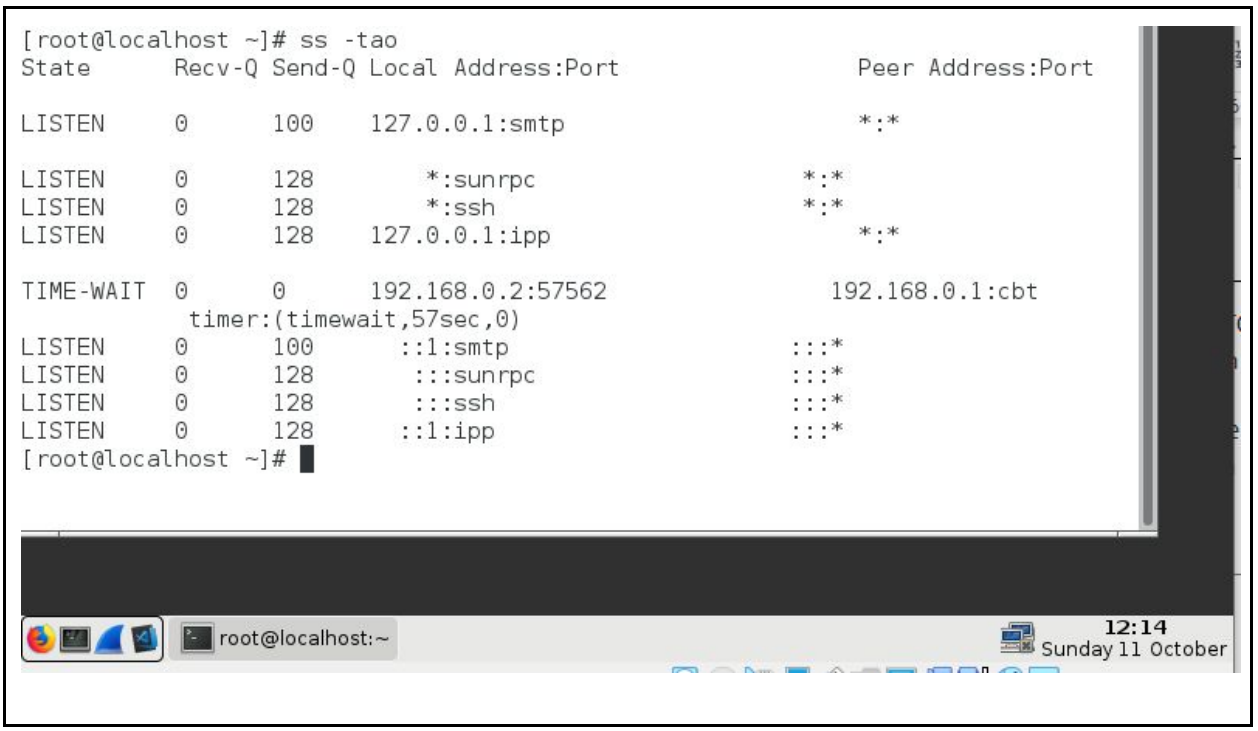
- Comprobar el estado de la conexión e identificar los parámetros (dirección IP y puerto) con el comando `ss -tn`.
- Iniciar una captura con Wireshark. Intercambiar un único carácter con el cliente y observar los mensajes intercambiados (especialmente los números de secuencia, confirmación y flags TCP)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



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

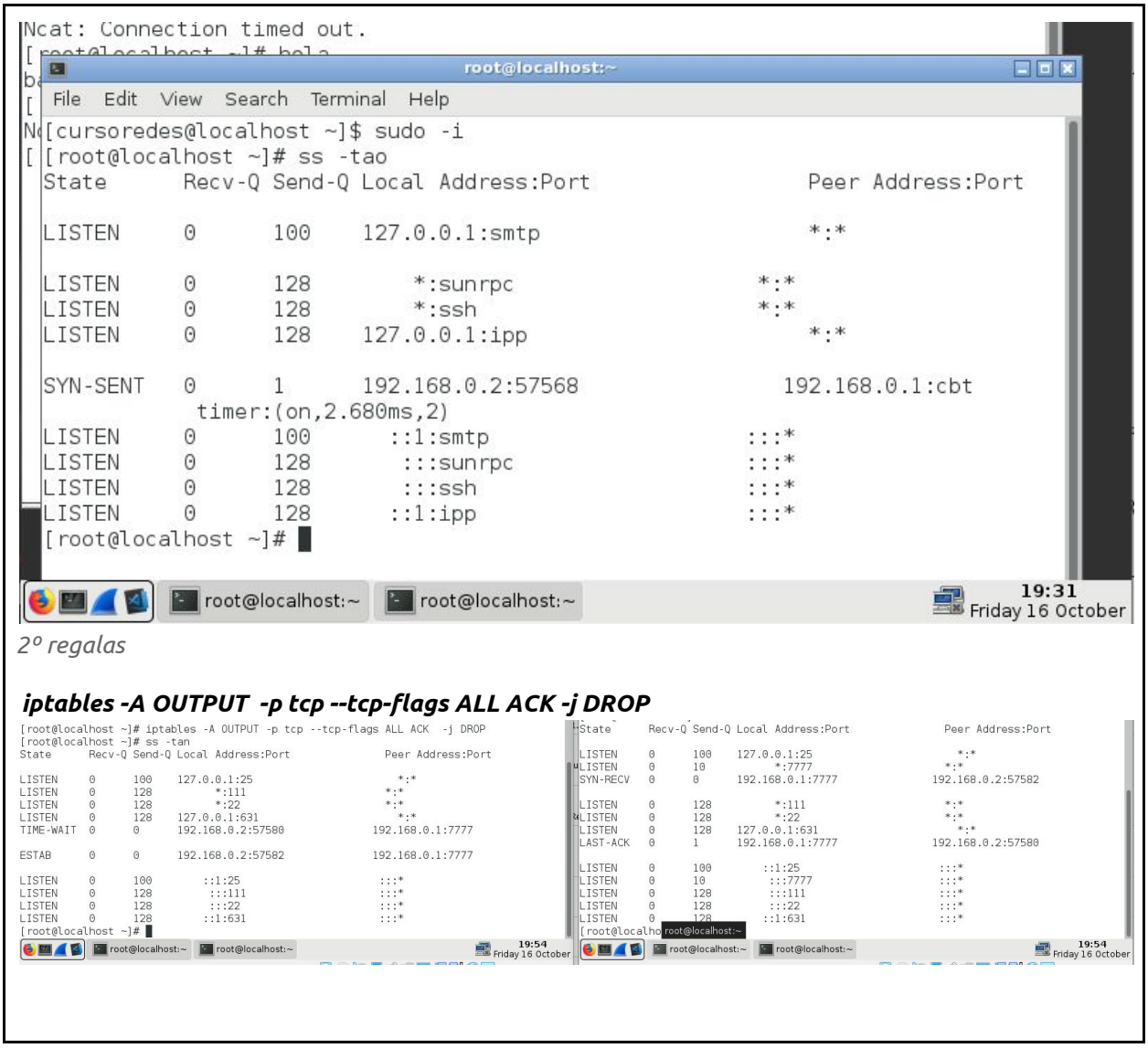


Ejercicio 5. (SYN-SENT y SYN-RCVD) El comando `iptables` permite filtrar paquetes según los flags TCP del segmento con la opción `--tcp-flags` (consultar la página de manual `iptables-extensions`). Usando esta opción:

- 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

1 regla ***iptables -A OUTPUT -p tcp --tcp-flags ALL SYN,ACK -j DROP***

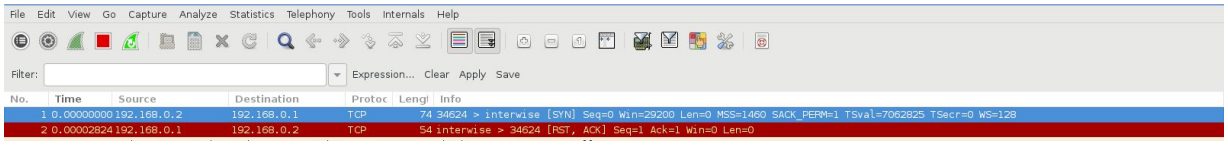


2º regalas

iptables -A OUTPUT -p tcp --tcp-flags ALL ACK -j DROP

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



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
/ptables -F en V1 y V2
Iptables -A INPUT-p tcp --tcp-flags ALL SYN,ACK -j DROP
Hping3 -S --flood 192.168.0.1 -p 22
ssh cursoredes@192.168.0.1

net.ipv4.tcp_syncookies = 0
[root@localhost ~]# ssh cursoredes@192.168.0.1
ssh: connect to host 192.168.0.1 port 22: Connection timed out

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
Nc -l 7777

```
#!/bin/bash
for i in {7775..7780}
do
nc 192.168.0.1 -zv
echo "nc 192.168.0.1 -zv $i"
done
```

```
[root@localhost ~]# sh for.s
Ncat: Version 7.50 ( https:/
Ncat: Connection refused.
nc 192.168.0.1 -zv 7775
Ncat: Version 7.50 ( https:/
Ncat: Connection refused.
nc 192.168.0.1 -zv 7776
Ncat: Version 7.50 ( https:/
Ncat: Connection refused.
nc 192.168.0.1 -zv 7777
Ncat: Version 7.50 ( https:/
Ncat: Connection refused.
nc 192.168.0.1 -zv 7778
Ncat: Version 7.50 ( https:/
Ncat: Connection refused.
nc 192.168.0.1 -zv 7779
Ncat: Version 7.50 ( https:/
Ncat: Connection refused.
nc 192.168.0.1 -zv 7780
```

1	0.00000000	192.168.0.2	192.168.0.1	TCP	74 39398 > 31337 [SYN] Seq=0 Win=29200 Len=0 MSS=1
2	0.00045522	192.168.0.1	192.168.0.2	TCP	60 31337 > 39398 [RST, ACK] Seq=1 Ack=1 Win=0 Len=
3	0.03942383	192.168.0.2	192.168.0.1	TCP	74 39400 > 31337 [SYN] Seq=0 Win=29200 Len=0 MSS=1
4	0.03977625	192.168.0.1	192.168.0.2	TCP	60 31337 > 39400 [RST, ACK] Seq=1 Ack=1 Win=0 Len=
5	0.07885440	192.168.0.2	192.168.0.1	TCP	74 39402 > 31337 [SYN] Seq=0 Win=29200 Len=0 MSS=1
6	0.07926457	192.168.0.1	192.168.0.2	TCP	60 31337 > 39402 [RST, ACK] Seq=1 Ack=1 Win=0 Len=
7	0.12131636	192.168.0.2	192.168.0.1	TCP	74 39404 > 31337 [SYN] Seq=0 Win=29200 Len=0 MSS=1
8	0.12167903	192.168.0.1	192.168.0.2	TCP	60 31337 > 39404 [RST, ACK] Seq=1 Ack=1 Win=0 Len=
9	0.16108740	192.168.0.2	192.168.0.1	TCP	74 39406 > 31337 [SYN] Seq=0 Win=29200 Len=0 MSS=1
10	0.16141948	192.168.0.1	192.168.0.2	TCP	60 31337 > 39406 [RST, ACK] Seq=1 Ack=1 Win=0 Len=
11	0.20097848	192.168.0.2	192.168.0.1	TCP	74 39408 > 31337 [SYN] Seq=0 Win=29200 Len=0 MSS=1
12	0.20135511	192.168.0.1	192.168.0.2	TCP	60 31337 > 39408 [RST, ACK] Seq=1 Ack=1 Win=0 Len=

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:

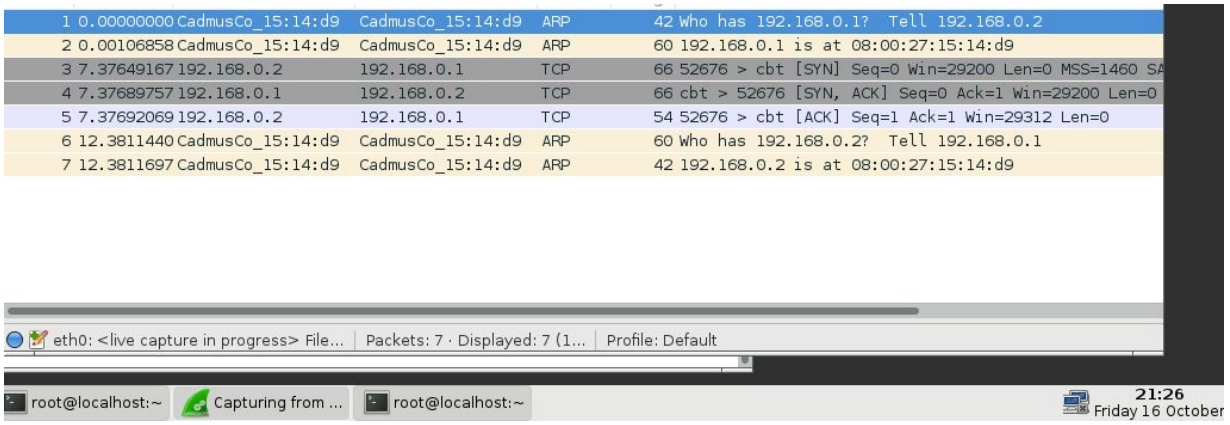
https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt

Parámetro del kernel	Propósito	Valor por defecto

`net.ipv4.tcp_window_scaling`	Habilita la ventana de escalada	1
`net.ipv4.tcp_timestamps`	Habilita las marcas de tiempo	1
`net.ipv4.tcp_sack`	Activa los acks selectivos	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



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

https://tldp.org/HOWTO/TCP-Keepalive-HOWTO/usingkeepalive.html

Parámetro del kernel	**Propósito**	**Valor por defecto**
`net.ipv4.tcp_keepalive_time`	el intervalo entre el último paquete de datos enviado y la primera sonda keepalive; después de que la conexión esté marcada para necesitar keepalive, este contador no se usa más	7200
`net.ipv4.tcp_keepalive_probes`	el intervalo entre las sondas de keepalive subsecuentes, independientemente de lo que la conexión haya intercambiado mientras tanto	9
`net.ipv4.tcp_keepalive_intvl`	a cantidad de sondas no reconocidas para enviar antes de considerar la conexión finalizada y notificar a la capa de aplicación	75

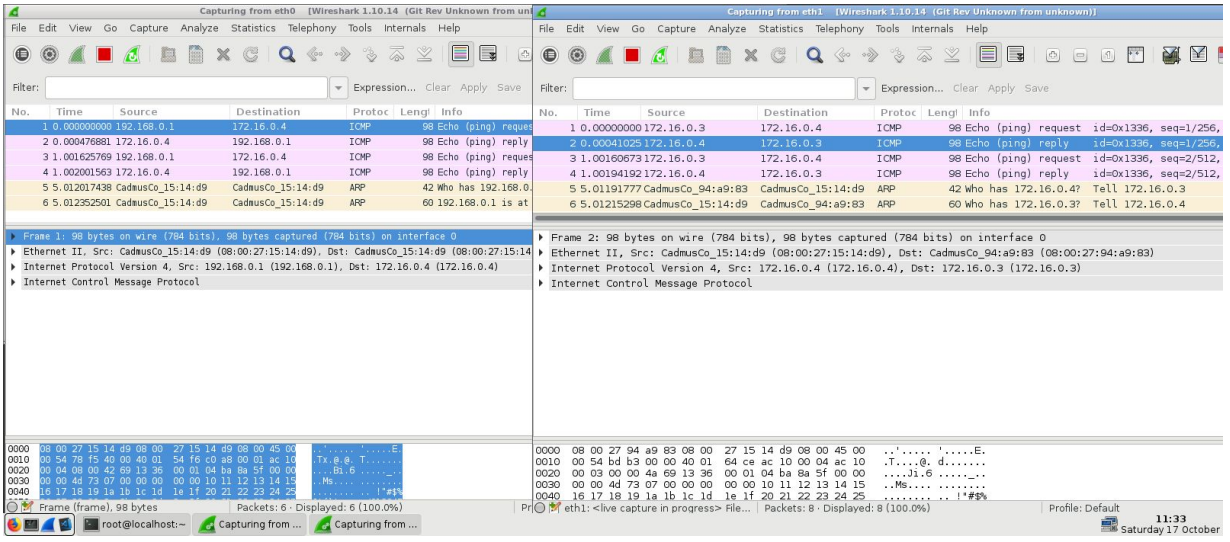
Traducción de direcciones (NAT) y reenvío de puertos (*port forwarding*)

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
iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE



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

-m state --state ESTABLISHED
conntrack -L
conntrack v1.4.4 (conntrack-tools): 0 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
Activado el iptables del ejercicio anterior
iptables -t nat -A PREROUTING -p tcp --dport 80 -j DNAT --to 192.168.0.1:7777