

# Diseño y administración de redes

## Práctica 1.2

Diseño y gestión de escenarios IPv4. NAT.

### Informe

Dorian Boleslaw Wozniak - [817570@unizar.es](mailto:817570@unizar.es)

Marcos Pérez Guillén - [820532@unizar.es](mailto:820532@unizar.es)

# Índice

<b>Índice</b>	<b>1</b>
<b>Cuestión 1</b>	<b>2</b>
<b>Cuestión 2</b>	<b>2</b>
Cuestión 2.a	3
Cuestión 2.b	3
<b>Cuestión 3</b>	<b>4</b>
Cuestión 3.a	4
Cuestión 3.b	4
<b>Cuestión 4</b>	<b>5</b>
<b>Cuestión 5</b>	<b>6</b>
<b>Cuestión 6</b>	<b>7</b>

# Cuestión 1

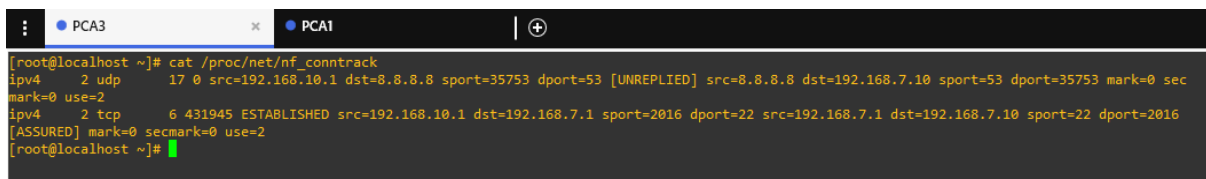
Indica cómo debe configurarse el NAT en los routers PCA3 y PCB3. El NAT que acabas de configurar, ¿es estático o dinámico?

Para configurar el NAT en primer lugar se ha borrado todo el contenido que pudiera haber en la tabla NAT de iptables, para ello se ha ejecutado el comando `iptables -t nat -F`. A continuación se ha introducido la siguiente regla en la tabla para crear el nat:

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

En este caso, eth1 es la interfaz conectada a la red 192.168.7.0/24 (LAN C), de tal manera que todos los paquetes que salgan a través del router se les cambiara su ip de origen por la del router (p.e., en el caso de LAN A, se enmascara el tráfico de salida a la LAN C con la ip 192.168.7.10).

Se trata de una NAT dinámica, al tratarse de una conversión de 2 direcciones locales a una pública. Para distinguir el tráfico de entrada, el sistema se encarga de registrar las asociaciones de dirección y puerto local con la dirección y puertos públicos, la cual se puede consultar en el fichero de proceso `/proc/net/nf_conntrack`.



```
[root@localhost ~]# cat /proc/net/nf_conntrack
ipvs4  2 udp    17 0 src=192.168.10.1 dst=8.8.8.8 sport=35753 dport=53 [UNREPLIED] src=8.8.8.8 dst=192.168.7.10 sport=53 dport=35753 mark=0 sec
mark=0 use=2
ipvs4  2 tcp    6 431945 ESTABLISHED src=192.168.10.1 dst=192.168.7.1 sport=2016 dport=22 src=192.168.7.1 dst=192.168.7.10 sport=22 dport=2016
[ASSURED] mark=0 secmark=0 use=2
[root@localhost ~]#
```

*Figura 1: Extracto de la salida de `nf_conntrack`. Nos interesa la segunda entrada, donde se indica la existencia de una conexión establecida desde 192.168.10.1:2016 a 192.168.7.1:22. La siguiente pareja de puertos y direcciones indica que se modifica la IP origen por 192.168.7.10.*

En esta imagen se puede observar un ejemplo del contenido de dicho fichero tras realizar el apartado 2 de esta práctica.

# Cuestión 2

Una vez configurado el escenario, verifica su funcionamiento en los casos a) y b) descritos a continuación. Captura en las interfaces oportunas e identifica el uso de direccionamiento a la entrada y salida del router NAT y los datos de las cabeceras ICMP y TCP implicados en el NAT.

Para probar que realmente hemos configurado correctamente el NAT vamos a hacer un ping desde PCA1 a PCC1 (para el NAT de la red B se haría lo mismo pero con la IP

correspondiente). Nos interesa observar el cambio de dirección de origen que se produce al pasar por PCA3.

## Cuestión 2.a

Se puede observar este comportamiento en los paquetes ICMP del primer par de capturas de la pregunta 2. El paquete 1 de la LAN A, se puede observar un Echo Request, id 0x9705 secuencia 1, con dirección 192.168.10.1 (PCA1) a 192.168.7.1 (PCC1). En la LAN C, se puede observar que la trama 1 es esta misma trama, pero la dirección de origen ha cambiado a la dirección pública 192.168.7.10 (PCA3). En la siguiente Echo Reply, se puede observar como se deshace el cambio: en la LAN C, la trama 2 tiene destino 192.168.7.1, y al llegar a la LAN A su destino ha sido cambiado por el router al original, 192.168.10.1.

## Cuestión 2.b

A continuación, añadiremos una regla para forzar que tanto PCA1 y PCA2 abran el mismo puerto origen para las conexiones SSH para ver qué ocurre:

```
iptables -t nat -A POSTROUTING -o eth0 -p tcp --dport 22 -j SNAT --to 192.168.10.1:2016
```

```
iptables -t nat -A POSTROUTING -o eth0 -p tcp --dport 22 -j SNAT --to 192.168.10.2:2016
```

En el segundo par de capturas podemos observar cómo resuelve la existencia de dos conexiones SSH simultáneas en este caso:

El arranque de la sesión SSH por parte de PCA1 se produce entre las tramas 23 y 42 en LAN A, y 25 a 45 en PCC1. En este caso observaremos específicamente las tramas TCP SYN y su respuesta (23 y 24 LAN A, 25 y 26 LAN C) al ser las primeras. Se puede observar que se produce la traducción de IP de forma esperada. En este caso, en la LAN A se realiza un envío de tráfico SSH desde el puerto 2016 de PCA1 al puerto 22 de PCC1. En el caso de LAN C, se produce un envío desde el puerto 2016 de PCC1 (que ha sustituido la IP origen a la pública) al 22 de PCC1.

El cliente PCA2 se une entre las tramas 47 y 71 de la LAN A, y 48 y 78 en la LAN C, usando otra vez las tramas TCP SYN (45 y 46 A, 48 y 49 C). En este caso se observa que en la LAN A se sigue realizando un envío desde el puerto 2016, pero en la LAN C el router cambia adicionalmente el puerto de salida al puerto 1024. Esto es, el tráfico sale del puerto 2016 de la LAN A y la LAN C, pero al pasar por el router PCA3, se reenvía el tráfico de LAN A por el puerto 2016 y el de B por el puerto 1024.

## Cuestión 3

Conéctate con el servidor ftp de PCC1:

```
$ ftp 192.168.7.??
```

```
name = anonymous
```

```
password = <una dirección de e-mail>
```

(la dirección de correo puede ser inexistente)

Trata de establecer una conexión ftp de datos mediante el comando ls dentro del ftp (ya que este comando usa una conexión de datos), tanto en modo activo como pasivo. Para los casos a) y b) descritos a continuación, verifica el correcto funcionamiento en base a la captura en las interfaces oportunas y analiza el resultado de dichas capturas.

Este escenario permite mostrar algunos de los problemas que puede ocasionar la NAT, en casos donde necesita realizar modificaciones a los datos contenidos en los paquetes durante el proceso de traducción.

El tráfico de ambas sesiones FTP se puede observar en el mismo par de capturas.

### Cuestión 3.a

La primera sesión abarca desde la trama 3 a la 43 en la LAN A, y desde la 3 a la 41 en la LAN C.

Al iniciar sesión, el servidor se encuentra en modo pasivo por defecto, lo cual se puede observar en las tramas 18 y 19 (A y C). En la trama 19, se puede observar cómo se producirá la comunicación pasiva (donde el servidor FTP indica un puerto desde donde solicitar datos): el tráfico de control al servidor se produce desde el puerto 21, mientras que el de datos indica que se realiza sobre el puerto 31598. El cliente envía órdenes desde 53318, y envía y recibe datos desde el 53104. El cliente solicita un listado y el servidor lo devuelve en las tramas 24 y 31 (A y C). Se produce la traducción de direcciones de la NAT de forma esperada.

Ahora se cambia a modo activo en las tramas 33 y 34 (A y B). Ahora, el cliente se vuelve en el “servidor” (esto es, esperará que el servidor FTP le envíe datos a un puerto concreto), anunciando que enviará y recibirá datos en la dirección 192.168.10.1:49495. El servidor, al recibir la trama, desconoce quien es 192.168.10.1 al no ser una IP pública para él, y avisa de que el puerto es incorrecto si se trata de realizar una operación.

### Cuestión 3.b

Ahora, activaremos el módulo del *kernel* `ip_nat_ftp` para permitir al router modificar el contenido del tráfico FTP.

Esta sesión abarca las tramas 44 a 87 de la LAN A, y hasta la 85 de la LAN C. El tráfico en modo pasivo se produce de forma esperada a la vez anterior.

Al pasar a modo activo en las tramas 76 y 77 (A), 74 y 75 (C), se produce la siguiente situación: en la LAN A, PCA1 avisa que estará escuchando en modo activo datos en 192.168.10.1:58748. La misma trama en PCC1 informa que estará escuchando de forma activa en la dirección 192.168.7.10:58758. Ahora el servidor responde con un código 200 al poder alcanzar dicha dirección y puerto.

Entre las tramas 76 y 84 (A), 82 (C), se observa que se produce un listado exitoso: el cliente solicita desde el puerto 53220 el comando al puerto 22 del servidor, y el servidor contesta enviando datos desde el puerto 20 al puerto 58748.

## Cuestión 4

Indica cómo has configurado el NAT y qué comandos has utilizado. Tal y como has configurado el NAT, ¿es estático o dinámico?

Se puede asociar el tráfico SSH dirigido a la IP pública con la máquina PCB1 con la siguiente regla NAT:

```
iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 22 -j DNAT --to 192.168.20.1
```

De esta forma, todo el tráfico TCP al puerto 22 (reservado para SSH) que entra por la interfaz eth1 (192.168.7.20, en LAN C) cambia de destinatario a la máquina PCB1. En este caso, es una asociación de IP:puerto uno a uno entre PCB1 y el router PCB3.

## Cuestión 5

Tomando como base las capturas, verifica y demuestra el funcionamiento del servicio ssh.

En la siguiente imagen se puede observar como se ha logrado el funcionamiento del servicio ssh entre las máquinas PCA1 (cliente) y PCB1 (servidor):

```
[root@localhost ~]# ssh alumno@192.168.7.20
alumno@192.168.7.20's password:
Last login: Tue Oct 17 17:26:13 2023 from 192.168.20.3
[alumno@localhost ~]$ ifconfig
eth0      Link encap:Ethernet  HWaddr 0C:5D:90:73:00:00
          inet addr:192.168.20.1  Bcast:192.168.20.255  Mask:255.255.255.0
          inet6 addr: fe80::e5d:90ff:fe73:0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:124 errors:0 dropped:0 overruns:0 frame:0
          TX packets:105 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:14312 (13.9 KiB)  TX bytes:15412 (15.0 KiB)

eth1      Link encap:Ethernet  HWaddr 0C:5D:90:73:00:01
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:224 (224.0 b)  TX bytes:224 (224.0 b)
```

*Figura 2: Ejecución del comando ifconfig en la máquina SSH asociada a 192.168.7.20. Se puede observar que la IP de la única interfaz de la máquina es 192.168.20.1, correspondiente a PCB1.*

Podemos verificar el correcto funcionamiento de la conexión SSH examinando las siguientes capturas de Wireshark:

En la trama 12 de LAN A, observamos cómo se envía un paquete SSH desde 192.168.10.1:2016 (PCA1) a la dirección destino 192.168.7.20:22, que corresponde a PCB3 en LAN C.

En la trama 6 de LAN C, vemos cómo el NAT de PCA3 realiza la traducción de direcciones IP cuando el paquete sale hacia la red externa. El tráfico se envía ahora desde 192.168.7.10:2016 (PCA3).

En la trama 6 de LAN B, podemos observar cómo el paquete SSH entra con el destinatario cambiado: ahora se dirige a 192.168.20.1:22 (PCB1) desde 192.168.20.3:2016 (PCB3).

El proceso inverso se puede observar en la siguiente trama de cada captura:

En la trama 7 de LAN B, se contesta a PCA3; en la 7 de LAN C a PCA3 enmascarando el origen con la IP pública de PCA3. Finalmente, en la LAN A, el router reenvía la respuesta a PCA1 sin modificar el destinatario.

Estas capturas de Wireshark muestran la secuencia de eventos que permiten que la conexión SSH se enrute adecuadamente desde PCA1 hasta PCB1 a través de la red externa C, demostrando el funcionamiento correcto de la configuración de red.

## Cuestión 6

En base a las capturas del tráfico, calcula el throughput (en pps) y el ancho de banda (en bps) total, útil a nivel IP y útil a nivel de aplicación, que se consigue en el escenario propuesto para cada uno de los casos de generación de tráfico y compáralos con el dato de ancho de banda dado por la herramienta *iperf*. Analiza si la limitación aparece en el valor del throughput o el de ancho de banda y si es a causa del generador de tráfico o del router.

Para poder realizar estimaciones del ancho de banda y throughput se usa el comando *iperf*. En PCC1 se establece un servidor a la escucha de tráfico TCP/UDP, según el escenario, mientras que desde PCA1 se realizan las pruebas de ancho de banda. E

En un primer lugar se realizará una transferencia de paquetes tipo *tcp* desde PCA1 (cliente) hacia PCC1 (servidor) durante 30 segundos sin ningún límite:

```
iperf -c 192.168.7.1 -t 30
```

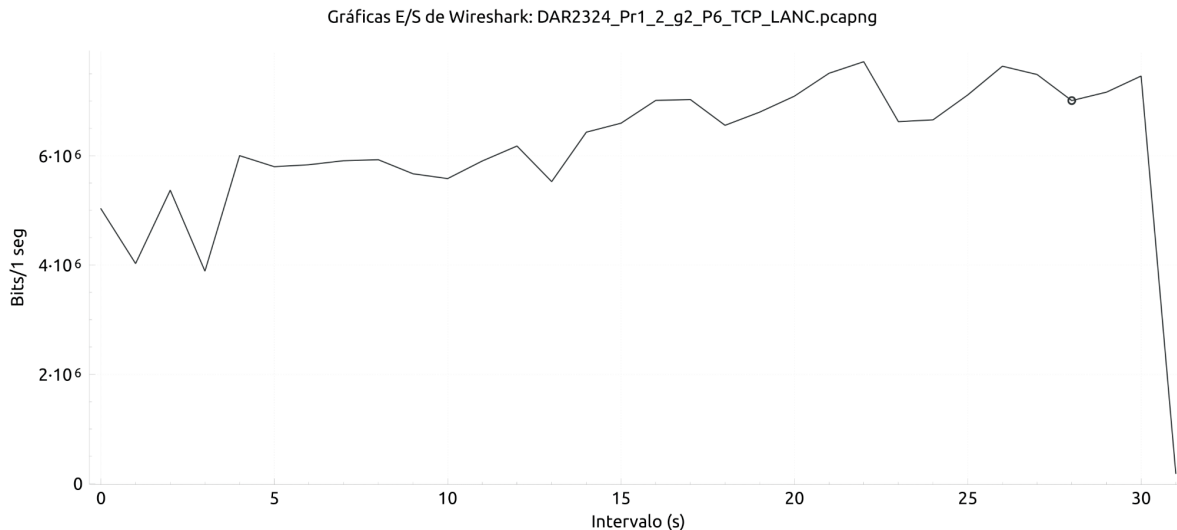
Obteniendo el resultado que se puede ver en la imagen, en el cual se indica que el ancho de banda medio obtenido es de 6.18Mbps:

```
[root@localhost ~]# iperf -c 192.168.7.1 -t 30
-----
Client connecting to 192.168.7.1, TCP port 5001
TCP window size: 18.9 KByte (default)
-----
[ 3] local 192.168.10.1 port 54486 connected with 192.168.7.1 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3]  0.0-30.4 sec  22.4 MBytes  6.18 Mbits/sec
```

Figura 3: Salida de *iperf* tras realizar la prueba con el protocolo TCP. Indica que se han transferido en 30 segundos unos 22.4MB de datos (unos 180Mb) a una tasa de 6.18 Mbps.



En la siguiente gráfica tomada a partir de la captura de tráfico en Wireshark, se puede observar que, efectivamente, la velocidad oscila alrededor de los 6 Mbps la mayor parte del tiempo.



*Figura 4: Gráfica temporal del ancho de banda a lo largo de la prueba TCP.*

A continuación se realizará una transferencia de paquetes tipo *udp* con diferentes tamaños desde PCA1 (cliente) hacia PCC1 (servidor) en intervalos de 10 segundos:

```
iperf -c 192.168.7.1 -t 10 -u -b 20000000 -l 100
```

Los tamaños de paquete que se van a utilizar son 100, 200, 300, 600, 1200 y 1472, y para cambiar el tamaño del paquete se ajusta el valor del parámetro *-l*. Adicionalmente, el parámetro *-b* permite limitar el ancho de banda del tráfico UDP. Se establece a 20Mbps para que no limite nuestras pruebas.

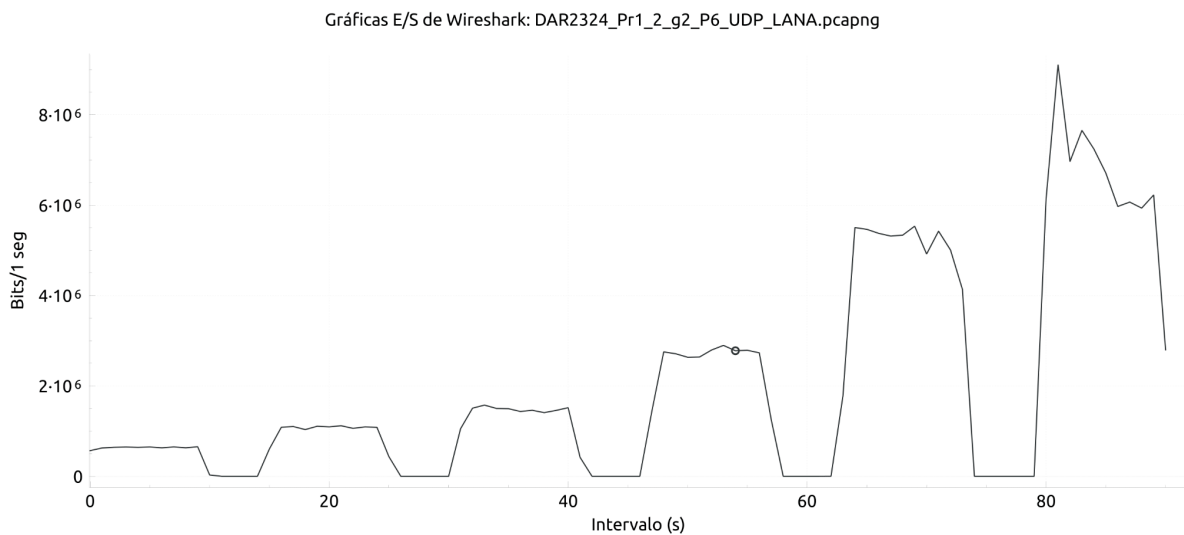
En estas pruebas nos interesa observar las dos potenciales limitaciones: el ancho de banda y los paquetes que se pueden enviar cada segundo, según la longitud de la cabecera.

El ancho de banda cambia para cada tamaño de paquete (datos obtenidos de la salida del comando *iperf*):

tamaño del paquete	Ancho de banda (Mbps)
100	0.436
200	0.886
300	1.33
600	2.64
1200	5.16

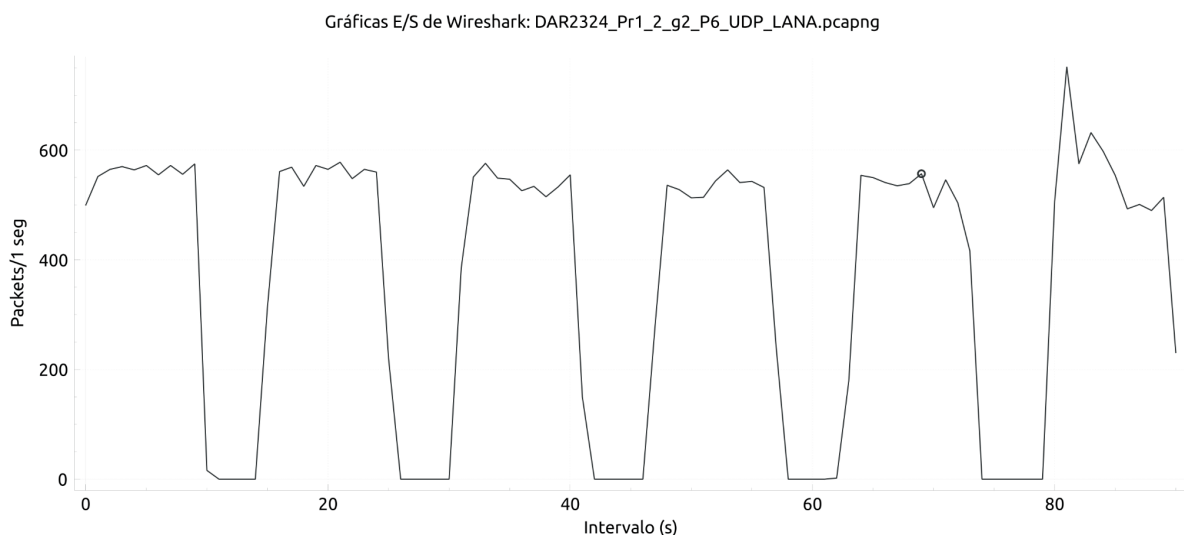
1472	5.92
------	------

Los datos se han obtenido a partir de la información otorgada por *iperf*, así como las gráficas E/S de la captura de las pruebas en Wireshark (nota: los datos de *iperf* y los reportados por Wireshark difieren porque *iperf* mide el ancho de banda a nivel de transporte, mientras que Wireshark lo hace a nivel de enlace, por lo que siempre será mayor):



*Figura 5: Ancho de banda para cada prueba desde tramas UDP de longitud 100 a 1472, en orden.*

Por otro lado, también se ha obtenido los paquetes enviados cada segundo:



*Figura 6: Gráfica de número de paquetes por segundo para cada prueba.*

Cabe destacar que el ancho de banda va creciendo conforme aumenta el tamaño de la trama UDP. Observando la gráfica de paquetes se puede observar la razón: el número de paquetes que se puede enviar en un segundo está limitado en torno a los 600 por el

generador de tráfico, bajando ligeramente a medida que aumenta hasta llegar al entorno de los 500 en las tramas de tamaño 1472, que a su vez alcanza el ancho de banda visto en la conexión TCP. Por tanto, enviar tramas mayores permite aprovechar mejor dicha limitación al enviar más datos por paquete.

Otra cuestión importante es que con paquetes más pequeños, la proporción entre los datos útiles y las cabeceras de cada protocolo disminuye en contra de los datos: las cabeceras siempre ocupan un tamaño determinado, sin importar cuantos datos haya, Esto también reduce el ancho de banda útil a nivel de transporte.