

pecl1: emulación de redes con netgui

Redes de Computadores / Grado: Ingeniería Informática



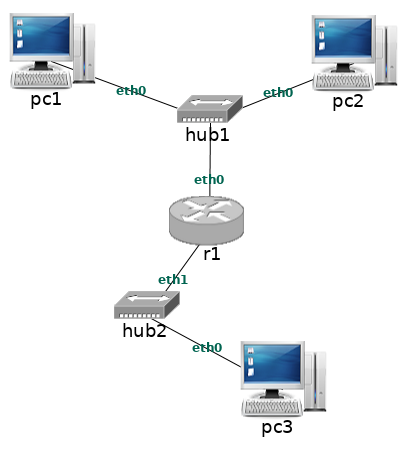
20 de febrero de 2019

rOBERT PETRISOR Y DAVID MÁRQUEZ

LABORATORIO 12-14H

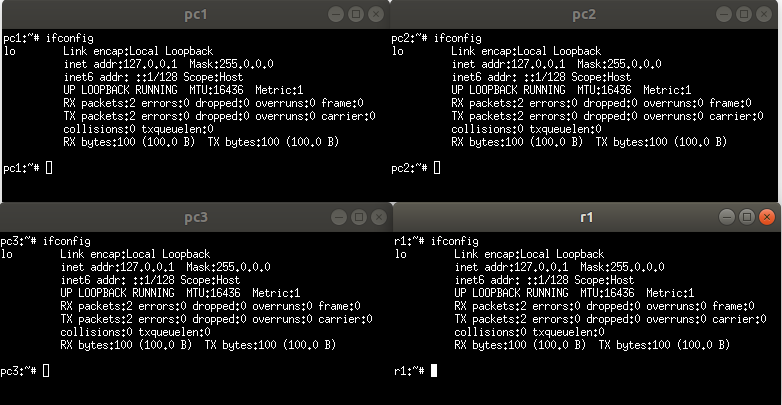
Ejercicio 3

Paso 1-. El esquema de la red que vamos a usar en este ejercicio consta de la siguiente forma:



Paso 2-. Arrancamos los ordenadores y el router de uno en uno, y comprobamos la configuración de la red en cada uno de ellos mediante la orden *ifconfig*.

Una vez que, los PCs y el router están arrancados, ejecutamos la orden ifconfig en cada uno de ellos, a continuación, mostramos las capturas de cada uno de los PCs:



Paso 3-. Utilizando las órdenes ifconfig o ip, asigna las siguientes direcciones IP a las interfaces de red de las máquinas de la siguiente forma:

• pc1 (eth0): 192.168.1.10

• pc2 (eth0): 192.168.1.11

• pc3 (eth0): 192.168.2.10

• La máscara de subred debe ser en todos los casos la 255.255.255.0

Paso 4-. El router r1 tiene dos interfaces de red: eth0 en la red 192.168.1.0 y eth1 en la red 192.168.2.0. Por norma, el router de una red suele tener asignada la primera dirección IP del rango. Configura las interfaces de red de r1:

• eth0 192.168.1.1

• eth1 192.168.2.1

Con los datos obtenidos, configuramos las direcciones IP de cada máquina con los siguientes comandos:

PC1: ~$ ***ifconfig eth0 192.168.1.10 netmask 255.255.255.0***

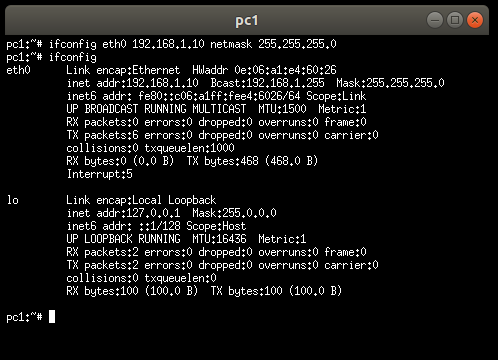
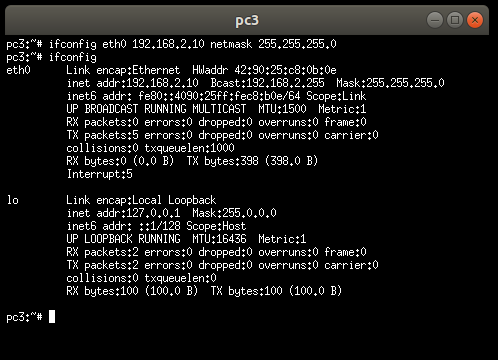
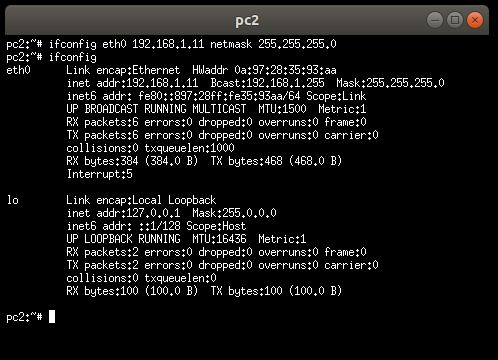
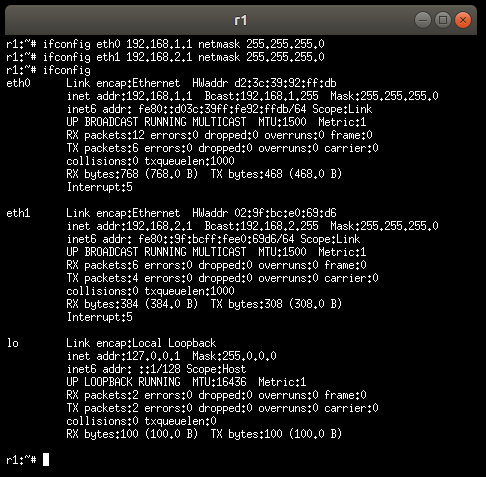
PC2: ~$ ***ifconfig eth0 192.168.1.11 netmask 255.255.255.0***

PC3: ~$ ***ifconfig eth0 192.168.2.10 netmask 255.255.255.0***

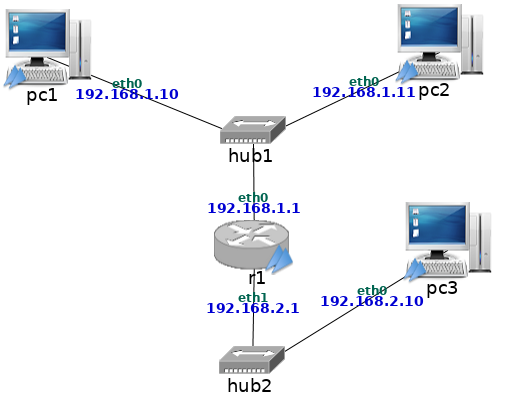
R1: ~$ ***ifconfig eth0 192.168.1.1 netmask 255.255.255.0***

R1: ~$ ***ifconfig eth1 192.168.2.1 netmask 255.255.255.0***

A continuación, ejecutamos la orden ifconfig, para comprobar que tanto los PCs como el router se han configurado correctamente. Mostramos capturas del resultado obtenido:

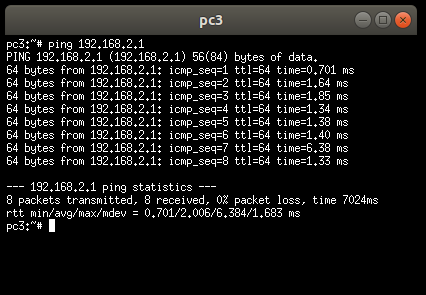
  

Como resultado final, la red nos queda de esta forma:



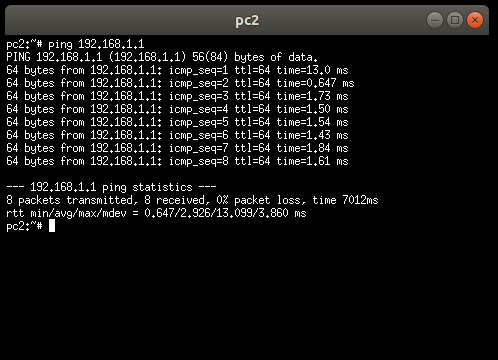
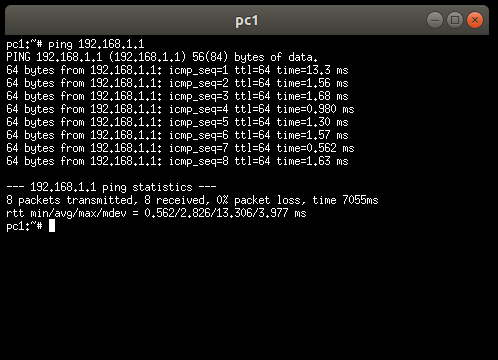
Paso 5-. Comprueba que desde pc3 se alcanza la dirección 192.168.2.1.

Para comprobar que desde el pc3 alcanzamos dicha dirección, que pertenece a r1, ejecutamos la orden ping. Como podemos observar en la imagen adjunta, la orden se ejecuta correctamente puesto que las dos máquinas se encuentran en la misma subred.



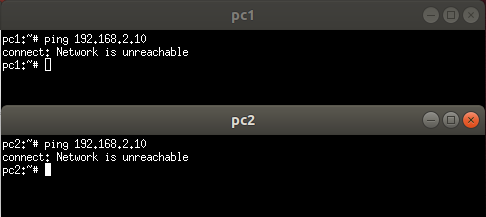
Paso 6-. Comprueba que desde pc1 y pc2 se alcanza la dirección 192.168.1.1.

En este caso, vamos a comprobar que desde pc1 y pc2 podemos llegar al r1. Al igual que antes, ejecutamos la orden ping en cada PC indicando la dirección IP de destino. Ejecutamos la orden se ejecuta correctamente puesto que las maquias se encuentran dentro de la misma subred.



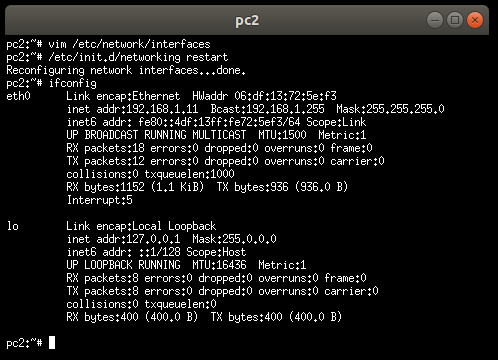
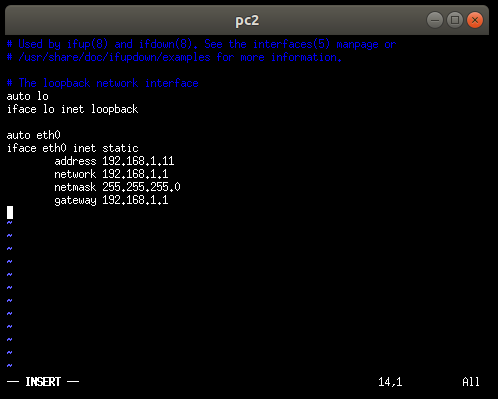
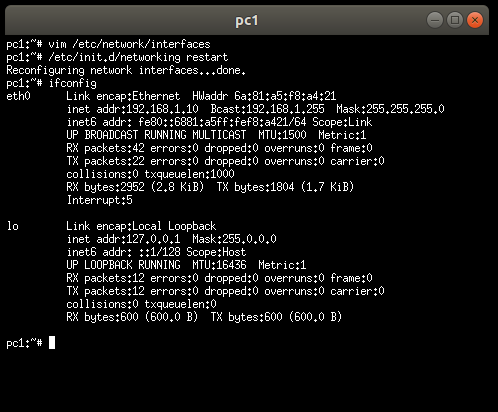
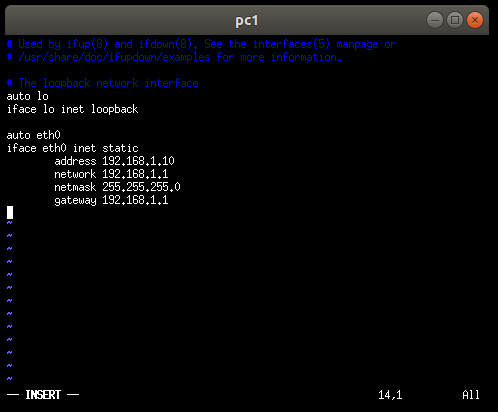
Paso 7-. Comprueba si desde pc1 y pc2 se alcanza la dirección 192.168.2.10.

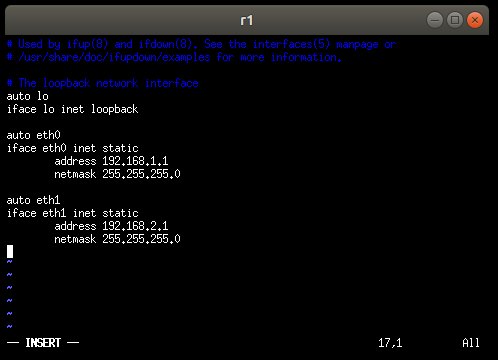
En este caso, estamos comprobando que desde el pc1 y el pc2 podemos alcanzar al pc3. Como podemos observar en las imágenes adjuntas, en este caso no es posible alcanzar el pc3. El motivo es porque a diferencia de los pasos anteriores, dichos PCs no se encuentran dentro de la misma subred.



Paso 8-. Hay que indicar a cada equipo cuál es su router por defecto. Configura el fichero interfaces de pc1, pc2 y pc3 para que utilicen r1 como su router por defecto. Entrega un pantallazo de la configuración.

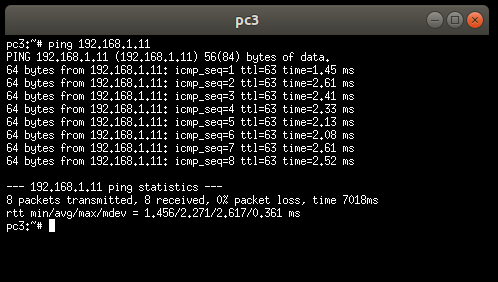
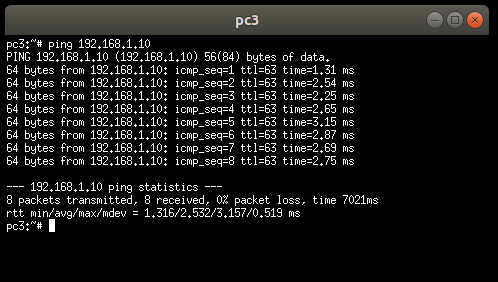
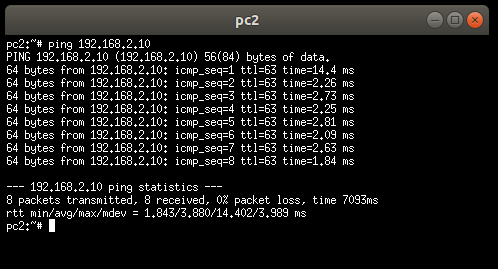
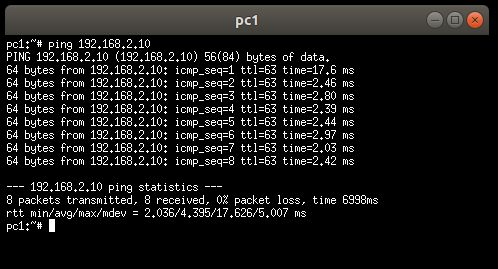
Para editar el fichero interfaces, usaremos el comando: ***vim /etc/network/interfaces***. Una vez rellenado a cada máquina con los datos que aparece en las capturas, lo guardamos mediante el comando: ***/etc/init.d/networking restart***. A continuación, mostramos la configuración de cada máquina en las imágenes adjuntas.



Paso 9-. Comprueba que ahora es posible desde pc1 (o pc2) alcanzar a pc3 (y viceversa).

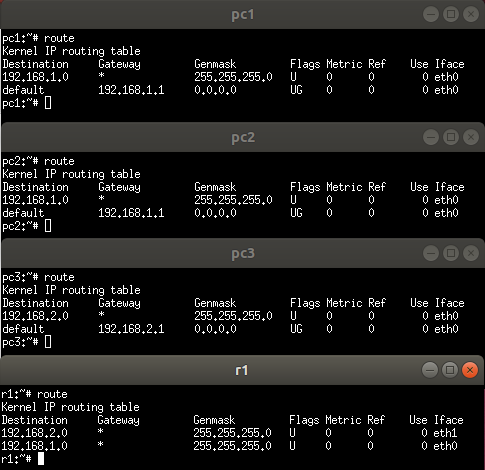
Para comprobar que podemos conectar las máquinas entre sí, ejecutamos la orden ping. Como podemos observar la orden se ejecuta correctamente, esto es porque hemos configurado cada maquina con la ruta por defecto Gateway.



Paso 10-. Comprueba cómo queda la tabla de enrutamiento de pc1, pc3 y r1, con el comando route. (También es posible modificar la tabla de enrutamiento con el comando route, aunque para que el cambio sea permanente es preferible utilizar el fichero interfaces).

Para comprobar como queda la tabla de enrutamiento de pc1, pc2 y pc3 y r1, usaremos el siguiente comando, en pc1, pc2 y pc3: ~ $ ***route***.

A continuación, mostramos en las imágenes adjuntas como queda la tabla de enrutamiento en cada pc.

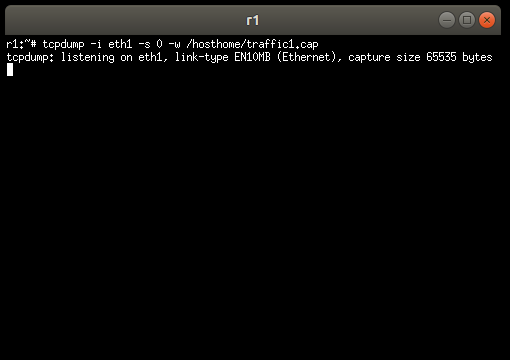


ANÁLISIS DE TRÁFICO

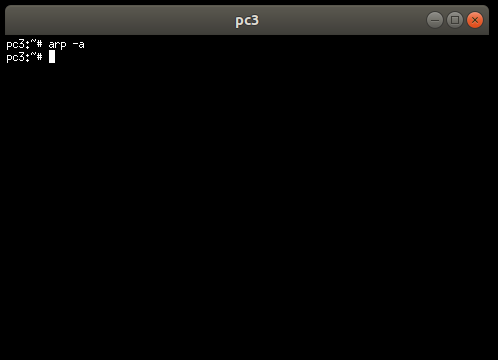
Paso 11-. En primer lugar, vamos a capturar y visualizar el tráfico que circula a través de r1 por la red 192.168.2.0, con el comando tcpdump. (podemos también realizar las modificaciones de ejecución oportunas que nos permita visualizar y estudiar los resultados a través de wireshark

• Ejecuta: ~$> ***tcpdump -i eth1 –s 0 –w /hosthome/traffic1.cap***

Lo que estamos haciendo es capturar y guardar todo el trafico que pasa por pc1 en una captura, para así despues poder analizarlo con whireshark.



Paso 12-. En pc3, vamos a comprobar que la cache ARP está vacía con el comando arp. En caso de no estar vacía, con la opción -d se pueden eliminar las entradas existentes.

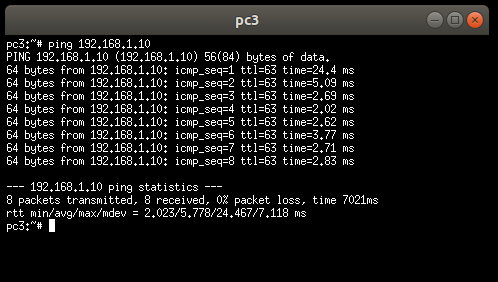


Efectivamente la caché ARP (Protocolo de Resolución de Direcciones), tiene que estar vacía. Ya que es un protocolo que se encarga de tener a su disposición las direcciones MAC de cada máquina de una red. Y por eso es importante, tenerla vacía.

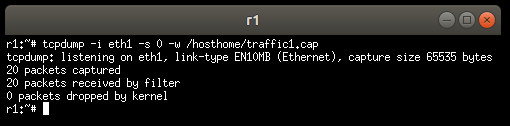
Paso 13-. Desde pc3 realiza lo siguiente:

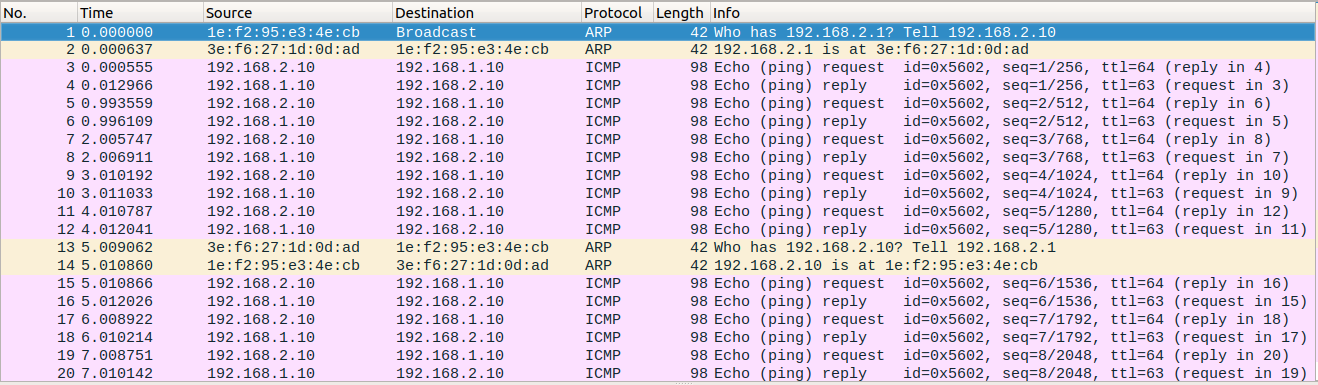
• Ejecuta: $> ***ping 192.168.1.10.***

• Intenta razonar: ¿De qué tipo es el primer mensaje que captura r1? ¿Y los siguientes?



Una vez que paramos el tcpdump con ctrl + C, obtenemos esto:



Y lo obtenido con el wireshark es lo siguiente:

En ella podemos observar que los dos primeros mensajes que captura r1 son de tipo ARP, que es un protocolo que obtiene la dirección hardware del equipo (Ethernet MAC) al que se quiere conectar mediante la dirección IP dada. En el paso 13 borramos la caché ARP del pc3 para asegurarnos de que al capturar con wireshark se va a realizar este paso del protocolo ARP. En el paso 14, hizo lo mismo que en los dos primeros. En el resto, los siguientes mensajes son de tipo ICMP (protocolo de mensajes de control de internet), es decir, envío de mensajes.

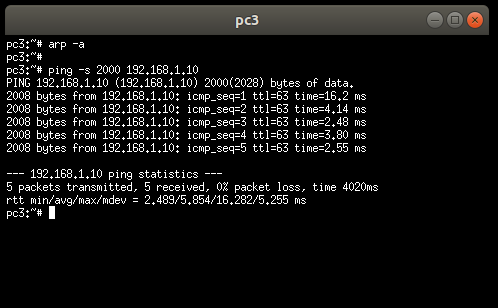
Paso 14-. Seguimos en pc3, y vamos a ampliar el campo de datos en los mensajes ping con la opción -s:

• Ejecuta: $> ping -s 2000 192.168.1.10

• Intenta razonar: ¿Cuántos mensajes se capturan en r1 por cada ICMP echo request?

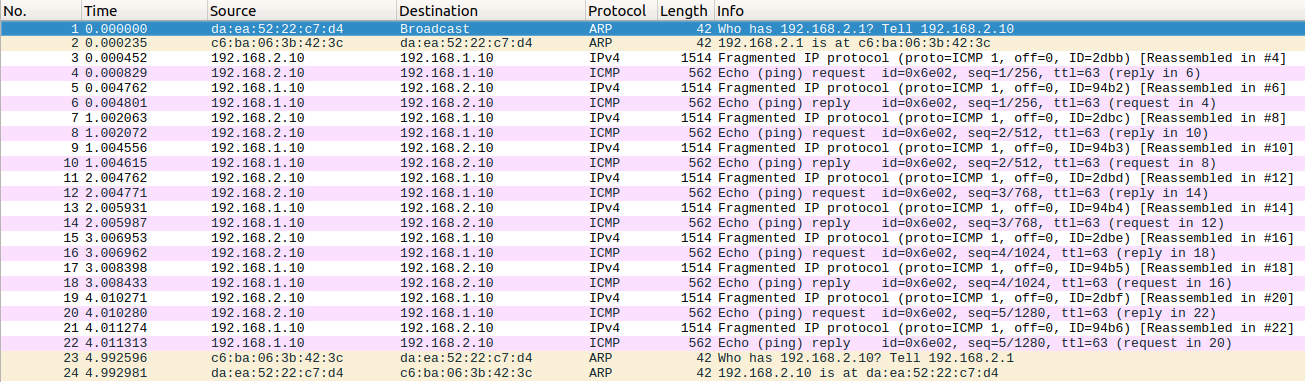
Si repetimos los pasos anteriormente, nos queda lo siguiente:





Una vez que paramos el tcpdump con ctrl + C, obtenemos esto:



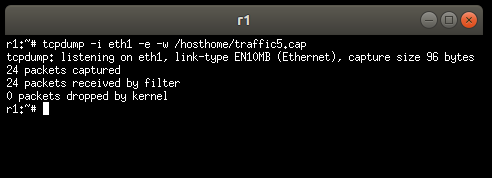
Y con wireshark esto:

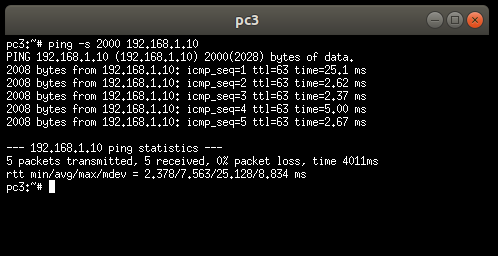
Como vemos en la imagen, con la opción –s aumentamos la información recibida por el comando ping. Por cada Echo Request tenemos el propio paquete Echo Request (request y reply) y un paquete llamado Fragmented IP Protocol. Está fragmentado en dos paquetes debido a que lo que queremos hacer es enviar un paquete de 2000 bytes, y no se puede enviar todo ese tamaño a la vez. Así que, el MTU como deja pasar tamaños entorno a 1500, resulta que lo fragmenta en dos partes, saliendo que el otro paquete es el restante de la diferencia total del paquete.

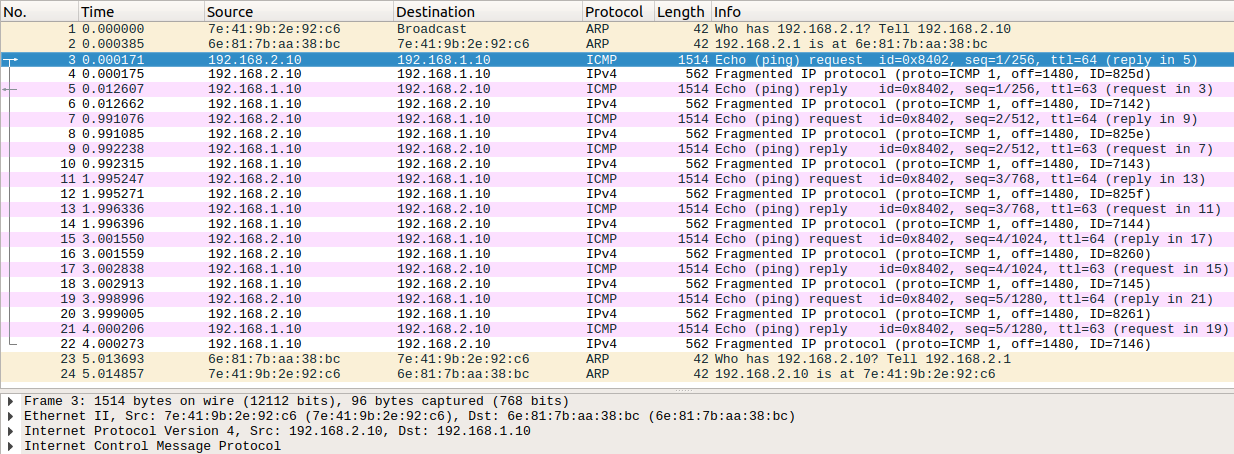
Paso 15-. Para comprobar el tamaño de los mensajes que se envían es necesario capturarlos (en r1) con el siguiente comando:

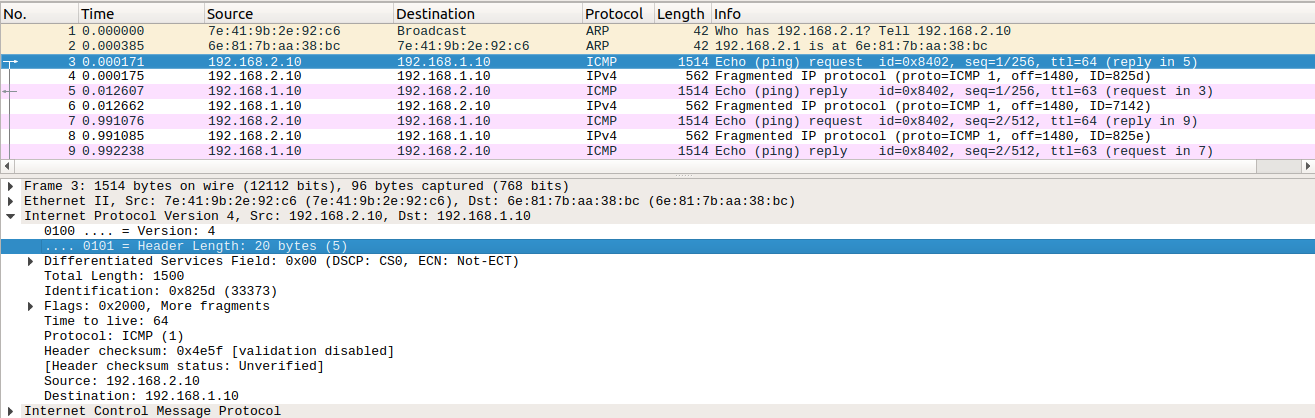
• Ejecuta: ***$> tcpdump -i eth1 –e –w /hosthome/traffic5.cap***

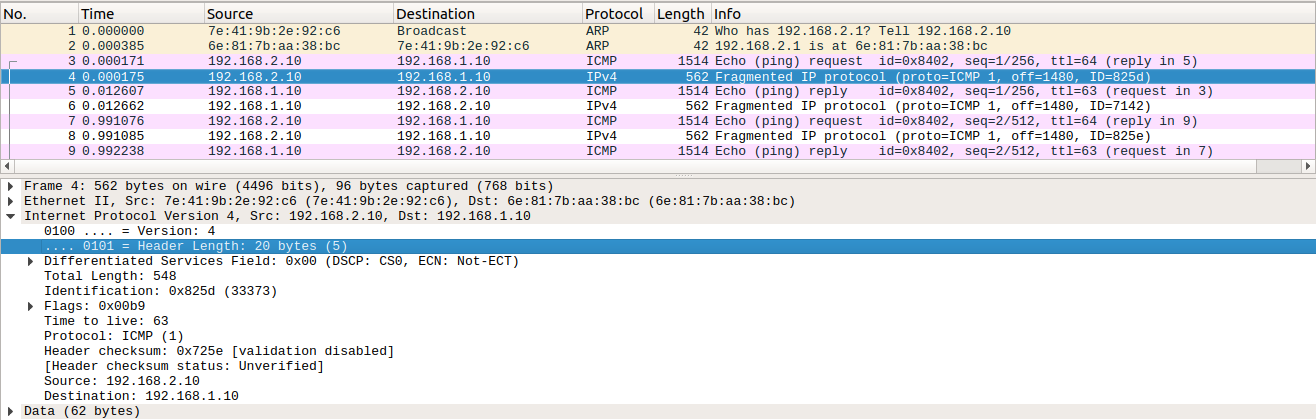
• Repite el paso 14 y comprueba el tamaño de los mensajes enviados e intenta razonar: ¿Cuántos bytes de datos se envían en el primer datagrama? ¿Y en el segundo?





Y en wireshark:

Para calcular el tamaño de los dos paquetes, se hace lo siguiente:



Dentro de la sección Internet Protocol Versión 4, accedemos a dos cosas:

* Si seleccionamos el fragmento de 1514 bytes:
  + La primera donde pone Header Length: 20 bytes. Esa información nos proporciona que hay 20 bytes de cabecera.
  + La otra donde pone Total Length: 1500 bytes. De ahí significa que, de esos 1500 bytes, 20 son de cabecera. Por lo tanto, el resto que es el paquete en sí es de 1480 bytes.
* Si seleccionamos el fragmento de 562 bytes:
  + La primera donde pone Header Length: 20 bytes. Esa información nos proporciona que hay 20 bytes de cabecera.
  + La otra donde pone Total Length: 548 bytes. De ahí significa que, de esos 548 bytes, 20 bytes son de cabecera. Por lo tanto, el resto que es el paquete en sí es de 528 bytes.

Y si realizamos la suma de 1480 + 528 = 2008, sale efectivamente la cifra que está en la orden del ping en el pc3.

Paso 16-. Para enviar datos mediante UDP y TCP se utiliza el comando nc.

• Lanzar un servidor en pc1 en el puerto 1111:

Ejecuta: $> nc –l –p 1111

• Lanzar un cliente en pc3 para que se conecte con el servidor:

Ejecuta: $> nc 192.168.1.10 1111

• Intenta razonar:

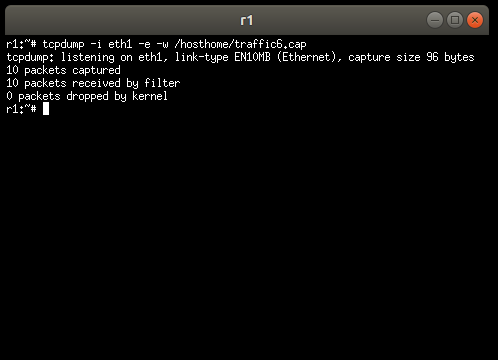
o ¿Qué mensajes se han capturado en r1?

o Teclea algo en el cliente y pulsa ENTER. ¿Qué circula por la red?

o Para cerrar la conexión pulsa Ctrl-C en el cliente. ¿Cuántos mensajes se intercambian para cerrar la conexión?

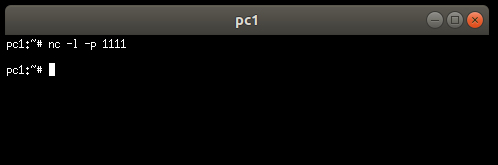
Lo primero de todo es utilizar el comando de tcpdump, es decir, el siguiente:

***~$ tcpdump -i eth1 -e -w /hosthome/traffic6.cap*** . Y cuando terminemos los siguientes dos pasos, ya pulsaremos ctrl + c, para pararlo.

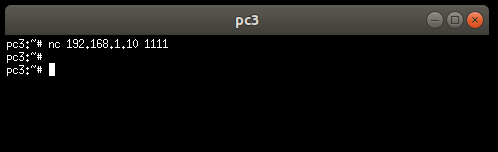


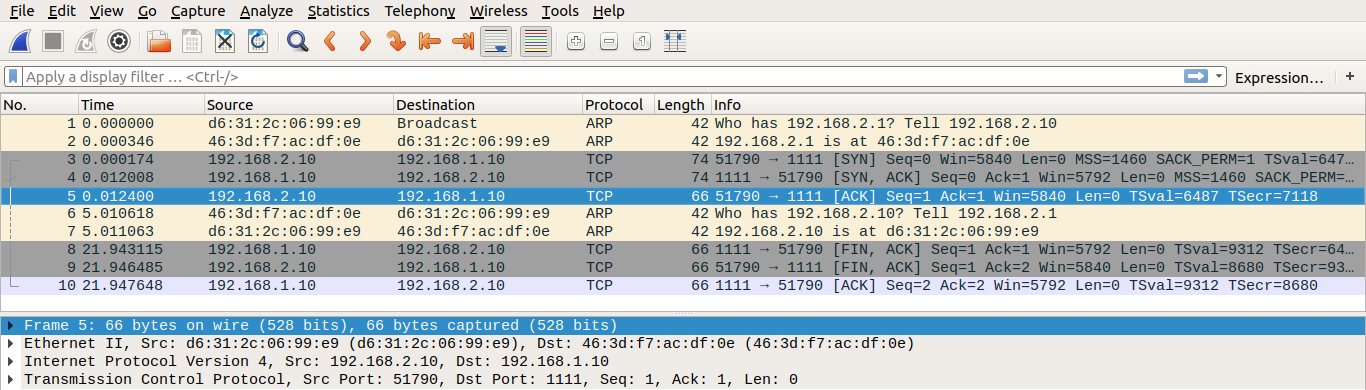
Ahora lanzaremos un servidor en pc1 desde el puerto 1111 a través de este comando:

***~$ nc -l -p 1111.***

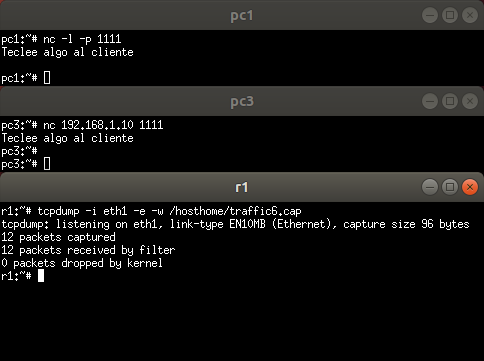


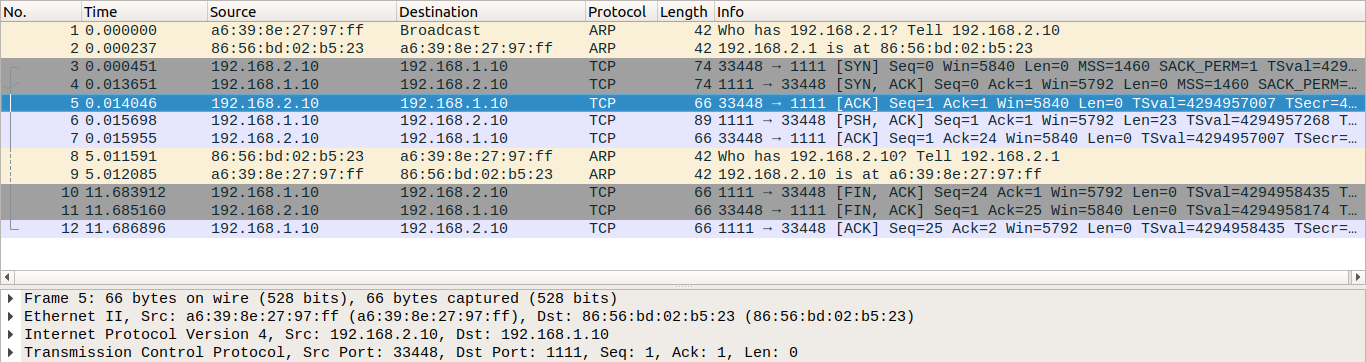
Ahora lanzaremos un cliente en pc3 para que se conecte con el servidor, que usaremos el siguiente comando: ***~$ nc 192.168.1.10 1111.***



Una vez realizado estos pasos, veamos el resultado en wireshark:

Se capturan primero mensajes ARP de reconocimiento entre las máquinas, posteriormente se ejecutan mensajes TCP (establecimiento de conexión, intercambio de datos y fin de conexión).



Y con el wireshark, obtenemos lo siguiente:

Los paquetes 1, 2, 8 y 9 corresponden al reconocimiento de máquinas. Los paquetes 3, 4 y 5 es establecimiento de conexión TCP entre cliente y servidor. El paquete 6 es el paquete de envío de datos (“Teclee algo al cliente”) entre el cliente y el servidor y el 7 es el ACK que reconoce que el mensaje ha sido recibido. Los paquetes 10, 11 y 12 establecen el final de conexión TCP ([FIN, ACK], [FIN, ACK], [ACK]).

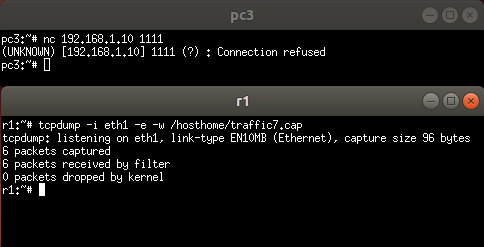
Como hemos descrito en el párrafo anterior, son 3 los mensajes enviados para finalizar la conexión TCP entre el cliente y el servidor [FIN, ACK], [FIN, ACK], [ACK]), en el primero el cliente solicita el cierre de conexión, en el segundo el servidor recibe el mensaje anterior y solicita el fin de conexión y en el último el cliente reconoce el paquete recibido y finaliza la conexión.

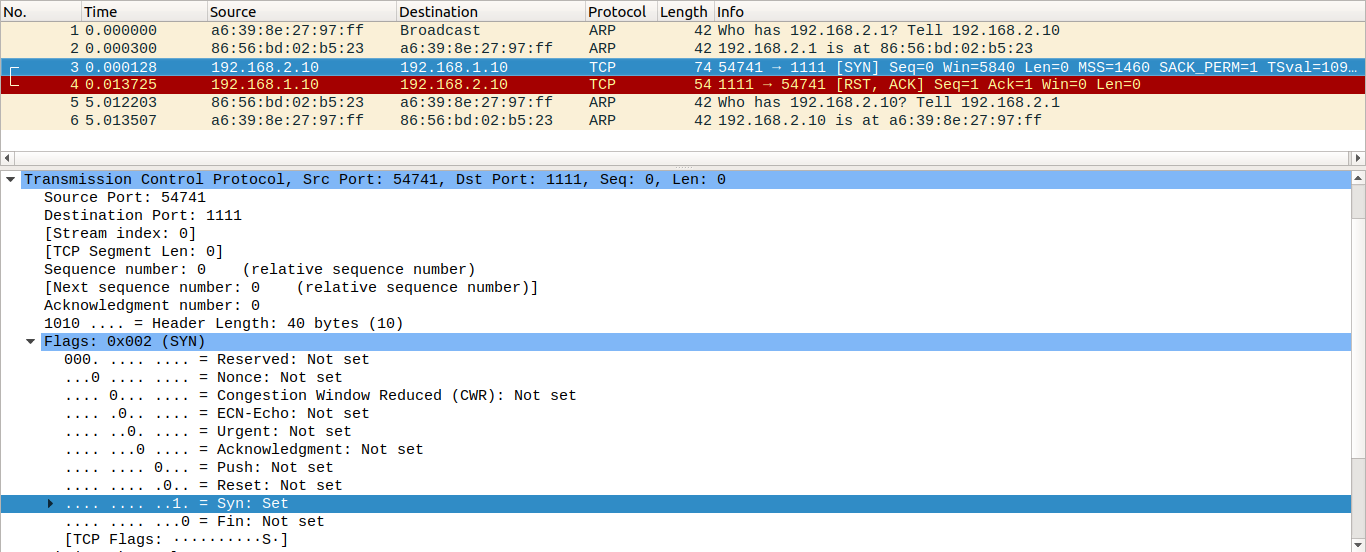
Paso 17-. Vamos a comprobar qué sucede cuando intentamos establecer una conexión a un puerto no existente.

• Ejecuta: $> nc 192.168.1.10 1111

• Intenta razonar: ¿Qué tipo de mensaje envía el servidor?

Realizando estos pasos, nos sale lo siguiente en wireshark:



Sabiendo que el cliente trata de establecer una conexión TCP con el servidor a través de un puerto no existente, resulta que éste responde con el bit RST (Reset) activado del protocolo TCP. Este bit se utiliza para reiniciar la conexión. En estos casos, en la imagen nos muestra por defecto en color azul, el Set (Reset) y en rojo (que el puerto sea no existente).

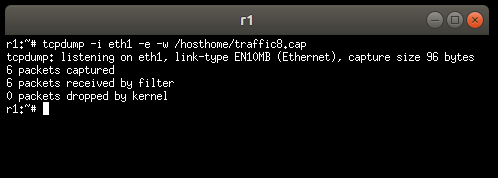
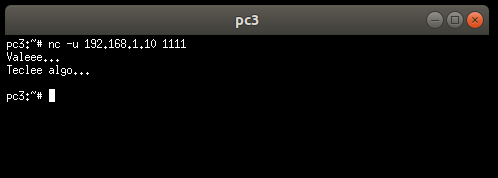
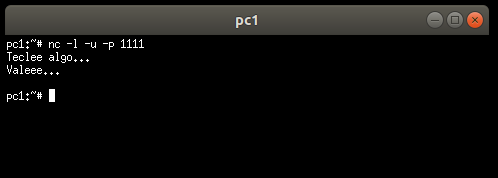
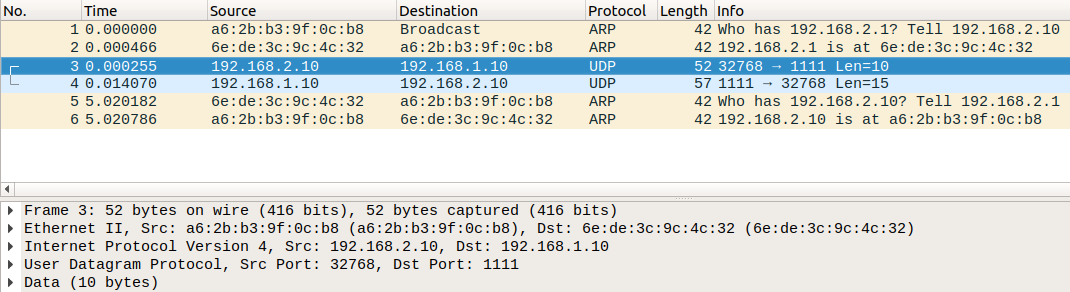
Paso 18-. Repetimos el paso 6 pero ahora con UDP:

• Lanzar un servidor en pc1 en el puerto 1111:

Ejecuta: ***$> nc –l –u –p 1111***

• Lanzar un cliente en pc3 para que se conecte con el servidor:

Ejecuta: ***$> nc –u 192.168.1.10 1111*** (pulsa enter, teclea unos datos y vuelve a pulsar enter).



• Intenta razonar: ¿Qué respuesta obtenemos del servidor? ¿Cuántos mensajes se intercambian para cerrar la conexión?

Al tratarse de un segmento UDP, no obtenemos ninguna respuesta del servidor al enviar el mensaje “Teclea algo…”. Con respecto al cierre de conexión, tampoco. Esto se debe a que, a diferencia de TCP, no existen fases de establecimiento y cierre de conexión.

• ¿Qué diferencias encuentras entre UDP y TCP?

UDP es un protocolo que no requiere establecimiento de conexión, de manera que en la captura de Wireshark solo se encuentran 3 paquetes, el primero y el segundo corresponden al reconocimiento de las máquinas (ARP) y el tercer paquete es el mensaje enviado entre el cliente y el servidor. Al ser un protocolo que no requiere autentificación (protocolo no orientado a conexión) no requiere establecimiento de conexión entre el cliente y el servidor ni finalización de conexión, por ello solo se registra un paquete UDP (el del mensaje).