3.2.1 IP/ICMP

En esta sesión investigaremos el protocolo IP. Analizaremos paquetes IP enviados y recibidos y su fragmentación. Entenderemos cómo funciona el protocolo ICMP y algunas de sus aplicaciones mediante los procesos *ping* y *traceroute*.

3.2.1.1 **ICMP**

ICMP (Internet Control Message Protocol) es un subprotocolo de control y notificación de errores de IP. Su funcionalidad básica es la de comunicar errores en la red, como el de no ser posible llegar a un nodo y host determinado.

En principio, ICMP no es usado por una aplicación de nivel superior (como ocurre con TCP y UDP). No obstante existen un par de procesos que están desarrollados sobre ICMP y que nos servirán para ver los pasos que se llevan a cabo al enrutar sobre IP.

El formato del mensaje ICMP es el siguiente:

0		8	16	31		
	Tipo	Código	CRC			
	Identi	ficador	N° Secuencia			
	Datos (Opcional)					

A modo de ejemplo, haremos uso de los mensajes Echo Request/Reply, que corresponden a los tipos 0 y 8 respectivamente y código 0.

Más información acerca del protocolo ICMP en <u>http://www.rfc-es.org/rfc/rfc0792-es.txt</u>.

3.2.1.2 **PING**

Utilizaremos el proceso ping para estudiar los mensajes ICMP.

La aplicacion ping comprueba el estado de la conexión con uno o varios equipos remotos por medio de los paquetes de solicitud de eco y de respuesta de eco (ambos definidos en el protocolo de red ICMP) para determinar si un sistema IP específico es accesible en una red. Es útil para diagnosticar los errores en redes o enrutadores IP.

- 1. Abrir una consola de MS-DOS.
- 2. Abrir Wireshark y comenzar a capturar filtrando por ICMP.
- 3. En la consola de MS-DOS, introducir el comando "*ping –n 2 ip_destino*" sin comillas y siendo *ip_destino* la IP de algún equipo de la sala.
- 4. Detener la captura de Wireshark.

La captura debería mostrar algo parecido a lo siguiente:



Imagen 1 - Captura comando ping Wireshark

- 1. Explica la opción utilizada en el comando ping. Busca en la ayuda (ping –h).
- 2. ¿Cuántos mensajes ICMP se han intercambiado? ¿Por qué?
- 3. Copia los valores de los campos "Type", "Code", "Identifier" y "Sequence Number" del primer par de mensajes Echo request/reply y explica los valores.

3.2.1.3 TraceRoute

Otro proceso que hace uso del protocolo ICMP es el traceroute (tracert para window). Este programa envía en primer lugar uno (o varios) mensajes ICMP (Echo Request) encapsulados en un paquete IP con TTL=1 (Time To Live). Tras recibir la correspondiente respuesta, envía otro mensaje ICMP (Echo Request) con TTL=2. Continúa así sucesivamente.

Cada vez que un paquete alcanza un router, éste chequea el valor del TTL y le descuenta uno. En el caso de que el valor del TTL sea 0, el router contesta al originante del mensaje con un mensaje ICMP (tipo 11 - TTL-exceeded).

Como resultado de este comportamiento, un paquete con TTL=1 dará lugar a que el primer router que se encuentre le mensaje ICMP lo devuelva al originante. El paquete con TTL=2 dará lugar a que el segundo router por el que pase el mensaje lo de vuelva por la razón explicada anteriormente y así sucesivamente.

Analicémoslo con Wireshark:

- 1. Abrir Wireshark
- 2. Abrir una consola de MS-DOS e introducir el comando "tracert 192.168.56.200".
- 3. Detener la captura de Wireshark una vez traceroute haya finalizado.

En la consola de MS-DOS podremos ver los saltos que se van realizando entre los nodos de la red. En este caso únicamente tenemos un router entre cada equipo del laboratorio y 192.168.56.200.

Observad cómo en la captura de Wireshark en valor del TTL en los diferentes mensajes ICMP se va incrementando.

3.2.1.4 Fragmentación IP

A continuación analizaremos la fragmentación sobre IP.

- 1. Abrir una consola de MS-DOS.
- 2. Abrir Wireshark y comenzar a capturar filtrando por IP.
- 3. En la consola de MS-DOS, introducir el comando "*ping –l 3000 ip_destino*" sin comillas y siendo *ip_destino* la IP de algún equipo de la sala.
- 4. Detener la captura de Wireshark.

Tras la captura de Wireshark podemos ver cómo se forman los mensajes ICMP a base de paquetes IP.

🗹 fragmentacion_ip. pcap - Wireshark							
<u>File Edit View Go Capture Analyze Statistics Telephony Tools H</u> elp							
	🔍 🗢 🖨 🌍 🚡	业 ■		1 🍢 💥 »			
Filter: ip		► Expressi	on Clea <u>r</u> App <u>l</u> y				
No Time Source	Destination	Protocol	Info				
1 0.000000 172.20.200.7	172.20.200.255	BROWSER	Become Backup Browser_				
2 1.992265 172.20.200.4	209.85.229.103	IP	Fragmented IP protocol	(proto=IC			
4 1.992374 172.20.200.4	209.85.229.103	TP	Fragmented IP protocol	(proto=IC) (proto=IC)			
5 1.992404 172.20.200.4	209.85.229.103	ICMP	Echo (ping) request	(), 000-10,			
6 1.995617 172.20.200.1	172.20.200.4	ICMP	Time-to-live exceeded (Time to l			
	209.85.229.103	IP TD	Fragmented IP protocol	(proto=IC)			
9 2.032107 172.20.200.4	209.85.229.103	IP	Fragmented IP protocol	(proto=IC)			
10 2.032137 172.20.200.4	209.85.229.103	ICMP	Echo (ping) request				
11 2.038342 130.206.68.254	172.20.200.4	ICMP	Time-to-live exceeded (rime to 🖬 M			
	1111						
Reader Tength: 20 bytes		Dofault.	FCN: 0×00)				
Total Length: 207	1. UXUU (DSCP UXUU.	peraurt,	ECN. 0X00)				
Identification: 0x3027 (1232)	'n						
■ Flags: 0x00	/						
Fragment offset: 0							
Time to live: 128	Time to live: 128						
Protocol: UDP (0x11)	Protocol: UDP (0x11)						
🕀 Header checksum: 0x20c7 [corr	ect]						
Source: 172.20.200.7 (172.20.	Source: 172.20.200.7 (172.20.200.7)						
Destination: 172.20.200.255 (172.20.200.255)							
🕀 User Datagram Protocol, Src Por	t: netbios-dgm (13	8), Dst P	ort: netbios-dgm (138)				
🗈 NetBIOS Datagram Service 🔽							
0000 Ff ff ff ff ff ff <u>00 23 4</u> d	0b 4a d4 08 0 <u>0 45</u>	00	# M.JE.				
0010 00 cf 30 27 00 00 80 11 20	c7 ac 14 c8 07 ac	140					
0030 C8 07 00 8a 00 a5 00 00 20	45 4f 45 42 45 4e	45	EOEBENE				
	43 41 43 41 43 41	43 JECE	EJEBD CCACACAC				
Frame (frame), 221 bytes	Packets: 147 Displayed: 147	Marked: 0	Profile: Default	.:			

Imagen 2 - Captura Wireshark fragmentación IP

- 4. Describir la opción usada en el comando ping.
- 5. ¿Qué información en la cabecera de IP indica que se trata de un paquete fragmentado?¿Cómo puedes identificar el paquete con el primer fragmento?
- 6. ¿Qué tamaño tiene el paquete IP?

- 7. ¿Cómo podemos saber si aún quedan más paquetes fragmentos por venir?
- 8. ¿Cómo identificaremos el paquete portador del último fragmento?
- 9. ¿Cuánta información extra es enviada, debida a la fragmentación, por cada paquete original?(Se pide calcular y justificar la diferencia entre los bytes transferidos y los que se transferirían si no hubiera fragmentación.)

3.2.2 Ethernet

En esta sesión, investigaremos el protocolo Ethernet y ARP.

3.2.2.1 Captura de trazas Ethernet

El formato de una traza Ethernet es el siguiente:

0	6	12	14 6	50-1514
Dest. MAC	Source MAC	Type/length	Payload	CRC(4bytes)

Para verlo de forma práctica con Wireshark, hagamos lo siguiente:

- 1. Abrir un navegador web y limpiar los archivos de caché.
- 2. Abrir Wireshark y comenzar a capturar filtrando por http.
- 3. Acceder a la dirección : 192.168.56.200
- 4. Parar la captura de Wireshark.

Ya que esta práctica trata sobre Ethernet y ARP, no estamos interesados en IP o en capas de protocolos de más alto nivel. De manera que, para contestar a las siguientes cuestiones hemos de fijarnos en la traza Ethernet que encapsula el mensaje GET HTTP.

10. ¿Cuál es la dirección Ethernet de vuestro equipo?

11.¿Cuál es la dirección Ethernet destino de la trama?¿Es la misma que la dirección Ethernet de 192.168.56.200? (Pista: la respuesta es no) ¿Quién tiene esa dirección?

12.¿Qué indican los bytes correspondientes al campo "Type"?¿Qué valor tienen?

3.2.2.2 ARP INTRANET

Veamos ahora como actúa el protocolo ARP (Arddress Resolution Protocol) sobre Ethernet.

Es un protocolo de nivel de red responsable de encontrar la dirección hardware (Ethernet MAC) que corresponde a una determinada dirección IP. Para ello se envía un paquete (ARP request) a la dirección de difusión de la red (broadcast (MAC = xx xx xx xx)) que contiene la dirección IP por la que se pregunta, y se espera a que esa máquina (u otra) responda (ARP reply) con la dirección Ethernet que le corresponde. Cada máquina mantiene una caché con las direcciones traducidas para reducir el retardo y la carga.

Hagamos lo siguiente:

- 1. Limpiad la caché ARP del equipo (consultad la ayuda tecleando el comando arp en una consola de MS-DOS).
- 2. Limpiad la caché del navegador web que vayáis a utilizar.
- 3. Abrid Wireshark y comenzar a capturar filtrando por HTTP.
- 4. Acceded a la página web del servidor web que se encuentre en vuestra subred.
- 5. Parad la captura de Wireshark, limpiar el campo de filtrado.

La captura mostrará algo parecido a la Imagen 3.

Observamos como, antes de lanzar la traza Ethernet encapsulando el mensaje HTTP (sobre TCP/IP), existen dos mensajes ARP. Analicémoslos con las siguientes cuestiones:

- 13.¿Cuáles son los valores hexadecimales de las direcciones origen y destino del mensaje ARP?
- 14.¿Cómo podemos saber que la trama Ethernet encapsula un mensaje ARP?
- 15.; Cuál es la dirección Ethernet destino en el mensaje ARPrequest?; Por qué tiene ese valor?
- 16.¿A qué corresponden los valores hexadecimales de las direcciones Ethernet en el mensaje ARP-reply?

Broadcom 440x 10	/100 Integrated Controller -	Wireshark		-	
File Edit View G	io <u>C</u> apture <u>A</u> nalyze <u>S</u> ta	tistics Telephony <u>T</u> ools <u>H</u> el	p		
	K 🖻 🖬 🗙 😂 🖴	🔍 🗇 🔿 春 👱		🖉 🗹 🛚	8 % 😫
Filter:		-	Expression Clea <u>r</u> App <u>l</u> y		
No Time	6472	Source	Destination	Protocol	Info
1 0.0000	175	192.100.30.150	224.U.U.ZOI	MDNS	Stanuar u quer y PTRuaaptcp. Tota
2 12.003	L/ 3 RE3	VaudTach 06:52:47	Broaucast	ARP	102 168 56 1 is at 00:01:38:06:52:
4 12 865	850	102 168 56 2	102 168 56 1	ARP	Standard query A occo thauto com
5 12 860	062	102 168 56 2	102 168 56 1	DNS	Standard query A ocsp. thawte.com
6 14 883	851	192 168 56 2	192.168.56.1	DNS	Standard query A ocsp. thawte.com
7 16 896	283	192 168 56 2	192 168 56 1	DNS	Standard query A ocsp. thawte.com
8 20, 906	483	192, 168, 56, 2	192,168,56,1	DNS	Standard query A ocsp. thawte.com
9 24, 915	182	192,168,56,2	192,168,56,127	NBNS	Name query NB OCSP. THAWTE, COM<00>
10 25.217	198	192.168.56.2	192.168.56.130	TCP	49562 > http [SYN] Seg=0 Win=8192
11 25.217	805	192.168.56.130	192.168.56.2	TCP	http > 49562 [SYN, ACK] Seg=0 Ack=
12 25.217	853	192.168.56.2	192.168.56.130	TCP	49562 > http [ACK] Seg=1 Ack=1 Wir
13 25.218	996	192.168.56.2	192.168.56.130	HTTP	GET / HTTP/1.1
14 25.219	671	192.168.56.130	192.168.56.2	TCP	http > 49562 [ACK] Seq=1 Ack=373 W
15 25.220	729	192.168.56.130	192.168.56.2	HTTP	HTTP/1.1 304 Not Modified
16 25.233	452	192.168.56.2	192.168.56.1	DNS	Standard query A urs.microsoft.com
17 25.248	427	192.168.56.2	192.168.56.130	HTTP	GET /favicon.ico HTTP/1.1
< [
⊞ Frame 13 (42	6 bytes on wire, 42	6 bytes captured)			*
🖃 Ethernet II,	Src: Dell_9b:46:19	(00:1c:23:9b:46:19),	Dst: XaviTech_96:f2:47	(00:01:3	8:96:f2:47)
🗄 Destinatio	n: XaviTech_96:f2:4	7 (00:01:38:96:f2:47)			
∃ Source: De	11_9b:46:19 (00:1c:	23:9b:46:19)			-
0000 00 01 38	96 f2 47 00 1c 23	9b 46 19 08 00 45 00	8 G # F F		
0010 01 9c 05	76 40 00 80 06 02	11 c0 a8 38 02 c0 a8	V@		*
0020 38 82 c1	9a 00 50 16 dd c9	d1 eb 41 d4 b9 50 18	8PAP.		
0030 41 2d 9b	f7 00 00 47 45 54	20 2f 20 48 54 54 50	AGE T / HTTP		
0040 2f 31 2e	31 Od Oa 41 63 63	65 70 74 3a 20 2a 2f	/1.1 Ac cept: */		-
	AT 63 63 65 70 74	Deduce 01 00 67 75 61	* Accon t Langua) Els Defect
File: C:\Users\alici	us\AppData\Local\Temp	Packets: 24 Displayed: 24 Marked	i: 0 Dropped: 0		/ file: Default

Imagen 3 - Captura ARP con Wireshark