3.1 Introducción a Wireshark

Una herramienta básica para observar los mensajes intercambiados entre aplicaciones es un **analizador de protocolos** (packet sniffer). Un analizador de protocolos es un elemento pasivo, únicamente observa mensajes que son transmitidos y recibidos desde y hacia un elemento de la red, pero **nunca** envía él mismo mensajes. En su lugar, un analizador de protocolos recibe una copia de los mensajes que están siendo recibidos o enviados en el terminal donde está ejecutándose.

Está compuesto principalmente de dos elementos: una **librería de captura de paquetes**, que recibe una copia de cada trama de enlace de datos que se envía o recibe, y un **analizador de paquetes**, que muestra los campos correspondientes a cada uno de los paquetes capturados. Para realizar esto, el analizador de paquetes ha de conocer los protocolos que está analizando de manera que la información mostrada sea coherente. Es decir, si capturamos un mensaje HTTP, el analizador de paquetes ha de saber que este mensaje se encuentra encapsulado en un segmento TCP, que a su vez se encuentra encapsulado en un paquete IP y éste a su vez en una trama de Ethernet. Un esquema de la integración de Wireshark con el Sistema Operativo del equipo en uso puede verse en la Imagen 1.

Wireshark se trata de un software gratuito disponible para varias plataformas (Unix, Windows y Mac OS). Se puede descargar desde <u>www.wireshark.org</u>., donde además, existe documentación asociada y un manual de usuario.



Imagen 1 - Esquema integración Wireshark con el SO.

3.1.1 Getting Started

En los equipos de laboratorio se encuentra instalado el analizador de paquetes (Wireshark) y las librerías de captura de paquetes (WinPcap). Al ejecutar el acceso directo de Wireshark en el escritorio debería e aparecer una pantalla como la siguiente:



Imagen 2 - Pantalla inicial Wireshark

Con los iconos de arriba a la derecha podemos controlar las capturas. Si pulsamos sobre el de más a la izquierda, veremos las posibles interfaces de red desde donde podemos capturar (Únicamente podremos capturar desde una interfaz de red al mismo tiempo). La pantalla es algo así:

Wireshark: Capture Interfaces						
Description	IP	Packets	Packets/s		Stop	
Intel(R) PRO/1000 MT Network Connection (Microsoft's Packet Scheduler)	130.206.69.106	206	30	<u>S</u> tart	Options	Details lose

Imagen 3 - Interfaces de red disponibles para la captura

En el caso de que existan más interfaces de red, éstas serán listadas. Si pulsamos el botón "Start" comenzaremos con la captura. Veremos cómo Wireshark comienza a mostrar todos los mensajes de red enviados/recididos.

🗖 Inte	el(R)	PRO/	1000) MT N	etwo	rk Co	nnectio	n (M	icros	oft's	Packe	t Scl	hedu	ler):(Capturir	ıg - W	iresh	ark				
File	Edit	View	Go	Cap	ture	Analy	ze Sta	atistic	s T	elepho	ny .	Fools	Hel	D								
		-					-		-					_	-		-			~	~	
		1	24				Xá		5	0	4	\$		T	2		*	Œ	5 (2	1	•
Filter:	_													•	Expressi	on	Clea <u>r</u>	. Ap	oply			
No. •		Time							Sou	rce					Destin	ation					Pr	otoc 🔺
	933	4.73	302	31					De	11_4	b:ee	:5e			Broa	dcas	t				AF	P
	934	4.73	353	55					95.	211	.94.	56			130.	206.	69.3	188			H	TP
	935	4.73	355	73					95.	. 211	.94.	56	.c.		130.	206.	69.	188			H	TP
	936	4.73	358.	21					130	0.20	6.69	.18	8		95.2	11.9	4.5	6			T	P
	937	4.78	330.	22					95.	. 211	.94.	56			130.	206.	69.1	188			H	TP
	938	4.70	552:	50 4.4					95.	211	.94. 6.60	10	0		130.	206.	69.	188			H	TP
	939	4.70	2129	+4 20					05	2.20	0.09	56	° -		120	206	60	199				TD
	940	4.0.	21.63	30					95.	211	94.	56			130	200.	69	188			H	TP
	942	4.8	318	74					95.	211	.94.	56			130.	206.	69.	188			HT	TP
1	943	4.8	318	32					130	0.20	6.69	.18	8		95.2	11.9	4.5	6			т	P
	944	4.88	312	70					95.	211	.94.	56			130.	206.	69.3	188			H-	TP
	94.5	4.88	3173	36					95.	211	.94.	56			130.	206.	69.3	188			H	TP 👻
4																						•
E Er.	ame	1 (151.	4 hv	tes	on w	ire.	151	4 h	vtes	car	tur	ed)	8								
	herr	het .	тт	Src	. тп	telc	or 48	.16	•ha	(00	.15.	17.	48.	16·h	a) De	t · I	nell	Con	ın i	18.	ch•5	4 (0)
	torr	net I	Dro	toco	 1	rc.	95 21	1 0	. Da	6 70	5 21	1 9	40 4 5	6) 6)	ost· 1	30 3	206	60	1.89	R (120	206 6
	Derr	iet i	ion	Con	r, ⊃ ⊧nol	Dng	*****	т. э с	4. 5	o (s Dont	J. 21	.1.9	100),)	ost. J			09.	1.0		130.	200.C
E Ir.	arisi	1155	100	Con	trui	Pru		, >	nc i	Purt	: 11	ιp	(80), U	St PUr	.r.: e	euru	ρι	18.	18)	, se	ц: ⊥,
I⊞ Hy	pert	ext	Tr	anst	er P	roto	COL															
वि																		1				
		100	-		101																	لغر
0000	00	b0	dO	08 0	b 54	00	15 3	17 4		6 ba	1 08	00	45	00		.T	<u>.</u> н	•••;	.E.			-
0010	45	hc	00	64 4 50 C	0 00	1 3/	63	55 (71 ;	22 5	d 7	s 58	58	82 50	10	 E B	G. /.	5.	īa [^]	o D			
0030	00	3e	9f	18 (0 00	b9	17	32 7	73 E	e 3	5 2P	a6	9f	12	.>		25	14.				
0040	f6	db	57	3 C 3	e 50	1 11	b8 :	lf 8	39 0)1 ba	a 47	a3	c9	40		>]		G	@	L		Ţ
0050	러도 도구가 드	hd	60	Qh 7	5 SC) 1 F	7~ .		- 9 7	6 0	a do	24	05	00		V I		ø	л			
U Inte	I(R) P	коло	UU M	1 Netw	ork Co	nnect	ion (Micr	F	acke	ts: 94	Displ	ayed:	9451	Marke.	Profi	e: Deh	ault					1.

Imagen 4 - Captura de paquetes.

Una de las principales utilidades de Wireshark es la posibilidad de filtrar los mensajes en tiempo real. Veámoslo con un ejemplo.

- 1. Abrir un navegador de internet.
- 2. Con Wireshark capturando paquetes, filtrar con "dns" (sin comillas).
- 3. Acceder a una página web.

Intentad analizar los mensajes intercambiados:

- 1. ¿Qué tipos de mensajes se intercambian?
- 2. Interpretad la respuesta.

3.1.2 Análisis de una consulta HTTP

A continuación como veremos, podemos utilizar Wireshark para estudiar el intercambio de mensajes que se realiza a la hora de realizar una consulta http a una página web sencilla.

- 1. En primer lugar abrimos un navegador web que esté instalado en la máquina.
- 2. Abrimos Wireshark y seleccionamos la interfaz de red adecuada. Como ya hemos observado anteriormente, se muestra todo el tráfico de red cursado por el PC.
- 3. Le indicamos que únicamente nos filtre los mensajes correspondientes al protocolo http. Escribimos "http" sin comillas en el campo correspondiente al texto y pulsamos el botón "Apply" para que comience a filtrar. En este momento únicamente filtrará mensajes pertenecientes al protocolo http.
- 4. En el navegador, accedemos a la siguiente dirección de red: <u>http://192.168.56.200/part1.html</u>. El navegador deberá mostrar una página web con algún texto.
- 5. Paramos la captura de Wireshark.

En la pantalla de Wireshark debemos de tener algo parecido a lo siguiente:

🗖 pract.cap	🗖 prac1.cap - Wireshark							
<u>Eile E</u> dit <u>V</u> ie	Eile Edit View Go Capture Analyze Statistics Telephony Tools Help							
		🖻 🔀 🗶 🛃	🔍 🗢 👄 🖓 孩	⊻ 🔳] 📓 🖻 畅 💥 💢		
Filter: http								
No Tin	ne	Source	Destination	Protocol	Info			
47 1.	713964	130.206.66.113	130.206.69.46	НТТР	GET /uri/index.	html HTTP/1.1		
481.	716027	130.206.69.46	130.206.66.113	HTTP	GET /favicon.ic	o HTTP/1.1		
50 1.	742575	130.206.69.46	130.206.66.113	HTTP	HTTP/1.1 404 No	t Found (text/html)		
< [IIII					
⊕ Frame 47	7 (504 by	/tes on wire, 504	bytes captured)				~	
	t II, Sro	: Uniwill⊂_44:8d	:af (00:03:0d:44:	8d:af), C	st: Cisco_37:a1:	80 (00:1a:e3:37:a1:80)		
Internet	t Protoco	ol, Src: 130.206.	66.113 (130.206.6 na Dont, bp 2000 (6.113), [toloot (1	St: 130.206.69.4	6 (130.206.69.46)	1	
Hyperte	ct Transf	Fer Protocol	re Pore. np-3000-	tennet (2	JO4), DSC POIC.	http (80), seq. 1, Ack.	⊥, ∎	
□ GET /u	uri/index	.html HTTP/1.1∖r	\n					
⊕ [E×p	pert Info) (Chat/Sequence)	: GET /uri/index.	html HTTF	/1.1\r\n]			
Requ	uest Metł	nod: GET						
Requ	uest URI	: /uri/index.html						
Requ	Jest Vers	510n: HTTP/1.1						
HUSC:	130.206. -+:	.09.40\r\n 						
<							$\mathbf{\Sigma}$	
0000 00 1	a e3 37	al 80 00 03 0d 4	4 8d af 08 00 45	00	7DE.		~	
0010 01 e 0020 45 2	a 4† 46 e 0a 04	40 00 80 06 1c 8 00 50 5a eb b7 3	3C 82 CE 42 71 82 3b d7 1d 89 8C 50	ce0 18 E	F®Bq		-	
0030 80 0	0 8f 18	00 00 47 45 54 2	20 2f 75 72 69 2f	69	GE Ţ /uri/i			
0040 6e 6	4 65 78 1 od op	2e 68 74 6d 6d 2 <u>49 af 72 74 - 25 7</u>	20 48 54 54 50 2f	31 nde 20 1	X.htm HTTP/1			
🔵 File: "D:\Javi	er\ICAI\ComI	ndustrialesAvanzadas F	ackets: 88 Displayed: 4 Mari	ked: 0		Profile: Default	.:	

Imagen 5 – Pantalla Wireshark tras haber capturado una consulta http sencilla.

En el ejemplo de la Imagen 5 podemos ver ágilmente el encapsulamiento que se ha llevado a cabo para enrutar el mensaje http. Wireshark muestra el mensaje http que se

encapsuló en un segmento TCP, que se encapsuló en un paquete IP y que se encapsuló en una trama de ethernet. Ya que únicamente estamos interesados en el mensaje http, comprimiremos el resto de información.

Observamos cómo se ha enviado una petición ("GET") a la url solicitada al navegador. Tras ésta, el servidor web donde se encuentra alojada la página ha contestado satisfactoriamente (200 OK) encapsulando en un mensaje http el código html que contiene la ruta requerida. Es el navegador (aplicación) quien desencapsula el código y lo interpreta.

Se observará también la existencia de una petición contigua en la uri: favicon.ico. Esto se debe a que el navegador hace una consulta acerca de si el servidor tiene algún icono que pueda mostrar junto a la ruta. Ya que nuestro servidor no tiene dicho icono, se contesta con el código correspondiente (404 Not Found).

La cabecera del mensaje http contiene alguna información que puede ayudar al navegador a interpretar el mensaje encapsulado. Intentad averiguar lo siguiente:

- 3. ¿Qué idioma tiene definido el navegador? ¿Para qué puede querer el servidor web conocer esta información?
- 4. ¿Cuándo fue la última vez que se realizó algún cambio en la página solicitada? ¿Para quién es útil saber esta información (Navegador o servidor)? ¿Por qué?
- 5. ¿Cuál es el tamaño del "payload" encapsulado en http?
- 6. Sin mirar en una tabla ASCII, ¿qué codificación hexadecimal tiene el carácter "G"?(Pista: hay una "G" en "GET HTTP").
- 7. ¿Qué valor tiene el CRC(Hexadecimal)?

3.1.3 Consulta HTTP condicional

En este apartado veremos cómo se realiza el diálogo de una consulta http condicional.

En primer lugar es necesario limpiar la caché del navegador que estemos utilizando (Para Netscape 7.0, Edicion->Preferencias->Avanzado->Cache, limpiar caché. Para Firefox, Herramientas->Limpiar datos privados. Para IE, Herramientas->Opciones de Internet->Eliminar archivo. Para Chrome, Eliminar Datos de Navegación.)

A continuación realizamos los siguientes pasos:

- 1. Abrimos un navegador web con la caché limpia.
- 2. Abrimos Wireshark y filtramos el tráfico http.
- 3. Usando el navegador, accedemos a la dirección anteriormente usada (<u>http://192.168.56.200/part1.html</u>).
- 4. Una vez se haya cargado la página, pulsamos el botón "Refrescar" en el navegador (F5).
- 5. Paramos la captura de Wireshark.

Habremos obtenido algo similar a lo siguiente:



Imagen 6 - Pantalla Wireshark tras haber capturado una consulta http condicional

Ignorando de nuevo las peticiones favicon.ico, a primera vista parece que los mensajes GET son idénticos, pero observando más detenidamente el encabezado de http, veremos que no es así. Además, en ambos casos la respuesta del servidor web es completamente diferente.

Intentad averiguar lo siguiente:

- 8. ¿Se encuentra la cabecera de "If-Modified-Since" en ambos mensajes GET http? ¿A qué crees que de debe?
- 9. ¿Cuál es el Status-Code devuelto por el servidor tras la segunda consulta?
- 10.; Se devolvió el contenido de la página en tras la primera consulta?; Y tras la segunda?; Cómo puedes verlo?

3.1.4 Consultas de archivos con gran longitud

Hasta ahora hemos estudiado el intercambio de mensajes cortos en un entorno Cliente-Servidor. Veremos ahora una potente ayuda que Wireshark proporciona a la hora de analizar relaciones entre paquetes.

- 1. Abrir un navegador web.
- 2. Ejecutar Wireshark sin aplicar ningún tipo de filtrado.
- 3. Acceder a la siguiente dirección utilizando el navegador: http://192.168.56.200/part2.html).
- 4. Parar la captura de Wireshark.

Si filtramos ahora por http, veremos como no existe diferencia alguna con la consulta/respuesta realizada en el apartado 1. No obstante, mientras en el apartado 1, el archivo que nos devolvía el servidor tenía un tamaño aproximado de 165 bytes, en este caso el tamaño del archivo es de 30KBytes. Este detalle es irrelevante en la capa correspondiente a http, pero en capas inferiores es necesario gestionarlo de alguna manera.

Filtremos ahora la	captura con "	tcp". Veremos a	lgo parecido a l	lo siguiente:
--------------------	---------------	-----------------	------------------	---------------

🗖 p	rac1_3.cap - Wires	shark							
Eile	<u>E</u> dit <u>V</u> iew <u>G</u> o <u>C</u> a	pture <u>A</u> nalyze <u>S</u> tatistics	; Telephon <u>y T</u> ools <u>H</u> elp	0					
		e 🖪 🗙 😂 🖴	🔍 🔅 🔿 🕢 🕢		🕾 😹 🖻 畅 💥 💢				
Filter	Filter: tcp Expression Clear Apply								
No	- Time	Source	Destination	Protocol Info					
	42 1.203753	130.206.66.113	130.206.69.46	TCP ncdToadbaTance	> http [SYN] Seq=0 Win=65535				
	43 1.204775	130.206.69.46	130.206.66.113	TCP http > ncdload	balance [SYN, ACK] Seq=0 Ack=1				
	44 1.204798	120 206 66 112	120,200,09,40	HTTP CET (uni(part2	<pre>> nttp [ACK] Seq=1 ACK=1 Win= html HTTp/1 1</pre>				
	46 1.207648	130.206.69.46	130,206,66,113	TCP TCP segment o	f a reassembled PDUl				
	47 1.207741	130.206.69.46	130.206.66.113	TCP [TCP segment o	f a reassembled PDU]				
	48 1.207766	130.206.66.113	130.206.69.46	TCP ncdloadbalance	> http [ACK] Seq=425 Ack=2921				
	49 1.208748	130.206.69.46	130.206.66.113	TCP [TCP segment o	f a reassembled PDU]				
	50 1.208/80	130.206.66.113	130.206.69.46	TCP ncdloadbalance	> http [ACK] Seq=425 ACK=4383				
	52 1 208807	130.206.69.46	130.206.66.113	TCP [TCP segment o	f a reassembled PDU]				
	53 1.209012	130.206.66.113	130.206.69.46	TCP ncdloadbalance	> http [ACK] Seg=425 Ack=7301				
	54 1.209719	130.206.69.46	130.206.66.113	TCP [TCP segment o	f a reassembled PDU]				
	55 1.209759	130.206.66.113	130.206.69.46	TCP ncdloadbalance	> http [ACK] Seq=425 Ack=8761				
	56 1.209970	130.206.69.46	130.206.66.113	TCP [TCP segment o	f a reassembled PDU]				
	57 1.210718	130.206.69.46	130.206.66.113	ICP LICP segment o	T a reassembled PDUj				
<			III						
⊕F	'rame 42 (66 by1	tes on wire, 66 b	ytes captured)		<u>^</u>				
ΞE	thernet II, Sro	c: Uniwill⊂_44:8d	:af (00:03:0d:44:	8d:af), Dst: Cisco_37:a1	.:80 (00:1a:e3:37:a1:80) 🔤				
ΞI	nternet Protoco	ol, src: 130.206.	66.113 (130.206.6)	6.113), Dst: 130.206.69.	46 (130.206.69.46) =				
🖯 🖓	ransmission Com	ntrol Protocol, S	rc Port: ncdloadba	alance (2683), Dst Port:	http (80), Seq: 0, Len: 0				
	Source port: r	ncdloadbalance (2	683)						
	Destination po	ort: http (80)							
	Ectnorm Andrew	. 67		1					
				U					
0000	0 00 1a e3 37	al 80 00 03 0d 4	4 8d af 08 00 45	007 DE.	~				
0010	D 00 34 5f 6a	40 00 80 06 0e 1	.e 82 ce 42 71 82	се .4_ј@Вq					
0020	0 40 20 0a 70 0 ff ff 55 88	00 00 02 04 05 k	a 00 00 00 00 80 a 01 03 03 01 01	01					
0040	04 02								
🔵 Fil	le: "D:\Javier\ICAI\ComI	IndustrialesAvanzadas	ackets: 128 Displayed: 42 M	arked: 0	Profile: Default .:				

Imagen 7 - - Pantalla Wireshark tras haber capturado una consulta http "larga"

TCP entre el cliente y el servidor.

Efectivamente, la respuesta http del servidor es descompuesto en varios segmentos TCP que se envían de manera secuencial. Wireshark nos muestra todos los paquetes y nos indica el lugar en el que se reconstruirá el mensaje que encapsulan:

 ■ Flags. 0410 (ACR) Window size: 65111 ⊕ Checksum: 0x4185 [validation of [SEQ/ACK analysis] [Number of bytes in flight: [Reassembled PDU in frame: 81] TCP segment data (1460 bytes) 	disabled] 2920] 1	
	IIII	
0020 42 71 00 50 0a 7b 9b 7d 9b 0030 free 57 41 85 00 00 20 62 79 0040 3c 61 20 68 72 65 66 3d 22 0050 47 50 50 32 22 20 74 69 74 0060 50 32 22 20 74 69 74 0070 64 69 73 65 62 74 73 73	09 8f cd f3 63 50 10 20 33 47 50 50 2c 20 .wA b y 3GPP, 2f 77 69 6b 69 2f 33 ca href= '/wiki/3 6c 65 3d 22 33 47 50 GPP2" ti tle="3GP 3d 22 6d 77 2d 72 65 P2" clas s="mw-re 47 50 62 75 91 clas s="mw-re	
Transmission Control Protocol (tcp), 20 bytes	Packets: 128 Displayed: 42 Marked: 0	Profile: Default .:

Imagen 8 - Detalle información del paquete TCP indicando cuando se reconstruirá el mensaje fragmentado en la PDU.

11.¿En cuántos segmentos TCP se ha fragmentado la información?
12.¿Por qué hay segmentos TCP en sentido Cliente->Servidor?
13.¿Qué función correspondiente un protocolo de Enlace de Datos realizan estos paquetes?. Explica las funciones de los campos "Acknowledgement number" "Y "Sequence number".