

Wireshark

Jiménez Bernal, Manuel Castro Guerrero, Juan Miguel
manuasir@correo.ugr.es jcastroguerrero@correo.ugr.es

18 de septiembre de 2016



Índice

1	Introducción	3
2	Funcionalidades de Wireshark	3
3	Guía de instalación en sistemas Windows y Linux	4
3.1	Windows	4
3.2	Linux	5
4	Casos de uso	7
4.1	Análisis de tráfico en un escaneo de puertos	7
4.2	Análisis de tráfico en una sesión telnet	8
4.3	ARP	11
5	Comunidad	12

Índice de figuras

3.1.	Paquetes disponibles para sistemas Windows	4
3.2.	Pantalla del asistente de instalación de Wireshark	5
3.3.	Ejemplo de captura tras realizar un host de la red el comando 'apt-get update'	7
4.1.	Captura de escaneo de puertos	8
4.2.	Escucha de protocolo telnet	9
4.3.	Contenido de la comunicación con telnet	10
4.4.	Protocolo ARP	11
4.5.	Detectando ARP spoofing	11

1. Introducción

En el presente documento se pretenden explicar los diferentes motivos por los cuales la herramienta Wireshark debería ser una herramienta indispensable en el mantenimiento de un entorno de red con servidores y del cual se pretende que se mantenga seguro.[3]

La herramienta Wireshark es un analizador de protocolos open-source cuyo principal objetivo es el análisis de tráfico para solucionar problemas en redes de comunicaciones aunque también es utilizado en el desarrollo de software y protocolos, como herramienta didáctica en entornos de pruebas y laboratorios, y para hallar fallos de seguridad en redes.[3]

En un entorno de granja de servidores, la herramienta tiene un uso muy interesante en la monitorización del tráfico que se maneja para determinar el correcto funcionamiento de los protocolos que en él actúan, mantener la carga bajo parámetros establecidos para determinar sobrecargas y en definitiva asegurar que las conexiones que entran y salen son las esperadas en todos los casos.

2. Funcionalidades de Wireshark

Si por algo es ampliamente utilizado Wireshark es por sus múltiples funcionalidades disponibles y su enorme versatilidad y modularidad, así como por su facilidad en la interacción con el usuario debido a la GUI con la que opera, la que presenta el contenido caótico de los paquetes capturados de forma ordenada y organizada por múltiples parámetros y colores. Entre las funcionalidades más destacables se encuentran las siguientes:

- Captura de paquetes de datos en vivo a partir de una interfaz de red.
- Abrir archivos que contienen datos de paquetes capturados con tcpdump / Win-Dump, Wireshark, y una serie de otros programas de captura de paquetes.
- Importar paquetes de archivos de texto que contienen los códigos hexadecimales de paquetes de datos.
- Visualización de paquetes con información muy detallada de protocolo.
- Guardar los datos de los paquetes capturados.
- Exportar algunos o todos los paquetes en una serie de formatos de archivo de captura.
- Búsqueda de paquetes mediante una gran variedad de filtros.
- Crear varias estadísticas.

3. Guía de instalación en sistemas Windows y Linux

3.1. Windows



Figura 3.1: Paquetes disponibles para sistemas Windows

Desde el sitio web oficial de Wireshark se descargan los instaladores tanto para arquitecturas x86 como AMD64, habiendo que escoger el que se adecue a cada sistema. De la misma manera también es posible descargar los códigos fuente del programa, recompilarlos e incluso modificarlos en caso de que fuese necesario. El código oficial también se encuentra en la comunidad de repositorios GitHub para todo aquel desarrollador que quisiera realizar un fork del mismo o realizar sus propuestas de mejora mediante pull request [4]. El software está desarrollado en lenguaje ANSI C y C++, pero también usa módulos en lenguajes como Perl o Python [1]. Para instalar Wireshark en el sistema hay que comenzar con el proceso de instalación que consiste en seguir las instrucciones que presenta el asistente.

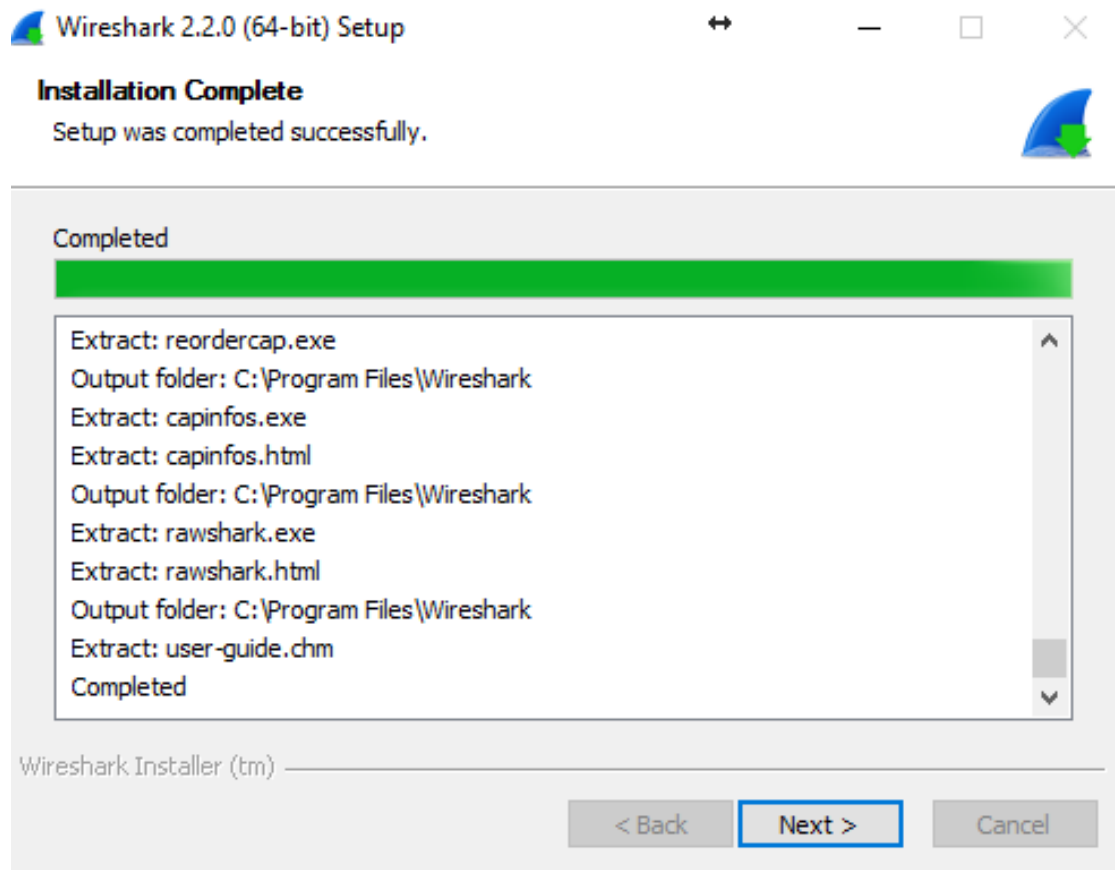


Figura 3.2: Pantalla del asistente de instalación de Wireshark

3.2. Linux

Para instalar Wireshark en los sistemas Linux basados en Debian y que usen el sistema de paquetes dpkg desde el shell basta con ejecutar las siguientes secuencias de comandos, ya que por defecto suele venir indexado en los repositorios oficiales que traen las distintas distribuciones:

```
$ sudo apt-get update  
$ sudo apt-get install wireshark
```

Para sistemas basados en Red Hat con el gestor de paquetes yum:

```
# yum check-update  
  
# yum instal wireshark
```

Una vez descargado se abrirá una ventana para configurar el archivo wireshark-common en la que debemos permitir a los no superusuarios capturar paquetes. En caso de que no apareciera esta ventana debemos ejecutar el comando:

```
$ sudo dpkg-reconfigure wireshark-common
```

Una vez finalizada la configuración inicial y la instalación, podemos proceder a ejecutar wireshark con el siguiente comando:

```
$ sudo wireshark
```

Una vez instalado se procede a realizar una primera escucha de pruebas, tras ejecutar el programa y configurar la tarjeta de red (NIC) por la que se va a analizar el contenido de los paquetes, se procede a ejecutar el Start y el programa comienza a capturar paquetes de cualquier protocolo, al no habersele establecido un filtro. Un ejemplo de filtro podría ser el siguiente

```
"ud.port == 53 || tcp.port == 80"
```

Para generar tráfico accedemos a una terminal desde la que se realiza algún comando que implique tráfico TCP y resolución de nombres. En el ejemplo se usó el siguiente comando:

```
$ sudo apt-get update
```

Tras lo cual se comprueba que en color azul se aprecia con facilidad las conexiones DNS para resolver la Ip que direcciona hacia el repositorio oficial de Ubuntu. Esto puede resultar de utilidad en caso de querer comprobar que los servidores a los cuales se accede son legítimos y no han sido modificados maliciosamente para que se apunte a otros no lícitos.

Length	Info
90	NTP Version 4, client
90	NTP Version 4, server
90	NTP Version 4, client
90	NTP Version 4, server
90	NTP Version 4, client
90	NTP Version 4, server
81	Standard query 0x1dd3 A gb.archive.ubuntu.com
81	Standard query 0x9371 AAAA gb.archive.ubuntu.com
145	Standard query response 0x1dd3 A gb.archive.ubuntu.com A 91.189.88.161 A 91.189.88.162 A 91.189.88.152 A 91.189.88.149
193	Standard query response 0x9371 AAAA gb.archive.ubuntu.com AAAA 2001:67c:1360:8001::17 AAAA 2001:67c:1360:8001::21 AAAA 20...
74	48362→80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2415869 TSecr=0 WS=128
74	80→48362 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=1592910935 TSecr=2415869 WS=128
66	48362→80 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=2415881 TSecr=1592910935
229	GET /ubuntu/pool/main/libl/liblinear/liblinear1_1.8%2bdfsg-5_amd64.deb HTTP/1.1
66	80→48362 [ACK] Seq=1 Ack=164 Win=30336 Len=0 TSval=1592910983 TSecr=2415881
1514	[TCP segment of a reassembled PDU]
66	48362→80 [ACK] Seq=164 Ack=1449 Win=32128 Len=0 TSval=2415895 TSecr=1592910983
1514	[TCP segment of a reassembled PDU]
1514	[TCP segment of a reassembled PDU]
1514	[TCP segment of a reassembled PDU]
1514	[TCP segment of a reassembled PDU]
1514	[TCP segment of a reassembled PDU]
1514	[TCP segment of a reassembled PDU]
66	48362→80 [ACK] Seq=164 Ack=10137 Win=49536 Len=0 TSval=2415900 TSecr=1592910983
1514	[TCP segment of a reassembled PDU]
1514	[TCP segment of a reassembled PDU]
66	48362→80 [ACK] Seq=164 Ack=13033 Win=55296 Len=0 TSval=2415902 TSecr=1592910983
1514	[TCP segment of a reassembled PDU]
66	48362→80 [ACK] Seq=164 Ack=14481 Win=58240 Len=0 TSval=2415903 TSecr=1592910983
1514	[TCP segment of a reassembled PDU]
66	48362→80 [ACK] Seq=164 Ack=15929 Win=61056 Len=0 TSval=2415908 TSecr=1592910996

Figura 3.3: Ejemplo de captura tras realizar un host de la red el comando 'apt-get update'

4. Casos de uso

A continuación se presentan distintos escenarios de relevancia en el ámbito de los servidores o granjas web en los cuales se va a analizar tráfico con el fin de ilustrar prácticas útiles y habituales para un administrador de sistemas desde el punto de vista de la seguridad e integridad de los datos.

4.1. Análisis de tráfico en un escaneado de puertos

Para este escenario se ha instalado el paquete nmap en una de las máquinas de nuestra red. Esta herramienta realiza un escaneo de puertos en los hosts que se le indiquen por parámetro y devuelve un feedback con aquellos que se encuentren en modo 'listening', con el fin de monitorizar actividad que pueda llegar a comprometer vulnerabilidades en esos puertos. Por otra parte este tipo de monitorización puede ser útil al administrador del sistema para comprobar si algún host remoto está intentando escanear el estado de los puertos sin permiso, ya que ese caso indicaría que un atacante pretende vulnerar el sistema y para ello está tratando de descubrir agujeros de seguridad que pudiera haber en él. Para la simulación se atacó la IP de un servidor con los siguientes resultados:

71	15.322192	192.168.0.156	192.168.0.155	TCP	74	47940-199	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2897996 TSecr=0 WS=128
72	15.420953	192.168.0.156	192.168.0.155	TCP	74	35360-21	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898020 TSecr=0 WS=128
73	15.421199	192.168.0.156	192.168.0.155	TCP	74	59780-3986	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898020 TSecr=0 WS=128
74	15.421386	192.168.0.156	192.168.0.155	TCP	74	33812-554	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
75	15.421553	192.168.0.156	192.168.0.155	TCP	74	49666-113	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
76	15.421690	192.168.0.156	192.168.0.155	TCP	74	68902-445	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
77	15.421953	192.168.0.156	192.168.0.155	TCP	74	37518-139	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
78	15.422144	192.168.0.156	192.168.0.155	TCP	74	39312-80	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
79	15.422280	192.168.0.156	192.168.0.155	TCP	74	59290-8888	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
80	15.422441	192.168.0.156	192.168.0.155	TCP	74	41886-443	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
81	15.422684	192.168.0.155	192.168.0.156	TCP	74	443-41886	[SYN, ACK]	Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 TSval=49258388 TSecr=2898021
82	15.422750	192.168.0.156	192.168.0.155	TCP	66	41886-443	[ACK]	Seq=1 Ack=1 Win=29312 Len=0 TSval=2898021 TSecr=49258388
83	15.422975	192.168.0.156	192.168.0.155	TCP	74	46422-135	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
84	15.423171	192.168.0.156	192.168.0.155	TCP	66	41886-443	[RST, ACK]	Seq=1 Ack=1 Win=29312 Len=0 TSval=2898021 TSecr=49258388
85	15.423344	192.168.0.156	192.168.0.155	TCP	74	51686-53	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
86	15.423485	192.168.0.156	192.168.0.155	TCP	74	36819-1025	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
87	15.423621	192.168.0.156	192.168.0.155	TCP	74	37224-22	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
88	15.423728	192.168.0.155	192.168.0.156	TCP	74	22-37224	[SYN, ACK]	Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 TSval=49258389 TSecr=2898021
89	15.423850	192.168.0.156	192.168.0.155	TCP	66	37224-22	[ACK]	Seq=1 Ack=1 Win=29312 Len=0 TSval=2898021 TSecr=49258389
90	15.424052	192.168.0.156	192.168.0.155	TCP	74	55894-256	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
91	15.424190	192.168.0.156	192.168.0.155	TCP	74	38456-143	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
92	15.424347	192.168.0.156	192.168.0.155	TCP	74	56156-9900	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
93	15.424484	192.168.0.156	192.168.0.155	TCP	74	57534-993	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
94	15.424618	192.168.0.156	192.168.0.155	TCP	74	48590-8000	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128
95	15.424751	192.168.0.156	192.168.0.155	TCP	74	34076-6001	[SYN]	Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2898021 TSecr=0 WS=128

Figura 4.1: Captura de escaneo de puertos

Como se comprueba en la figura, se reafirma que Wireshark es una herramienta excepcional en cuanto a las facilidades que proporciona al usuario con su interfaz gráfica resaltando con colores los diferentes tipos de mensajes que se intercambian en los protocolos. De esta forma se comprueba cómo una máquina con ip 192.168.0.156 está realizando un escaneo de puertos contra la máquina de IP 192.168.0.155. En éste intercambio de mensajes se aprecian cómo los paquetes que son respondidos con un ACK (acuse de recibo) son aquellos que se envían a los puertos que efectivamente tiene escuchando la máquina objetivo, en la figura el puerto 443 y 80, puertos bien conocidos por ser dedicados a usos de servidores Web y el puerto 22 dedicado por defecto a las conexiones del protocolo SSH. De esta forma descubrimos cómo opera la herramienta NMAP por debajo, realizando un barrido de peticiones a todos los puertos conocidos y señalando como abiertos los cuales envíen un ACK.

4.2. Análisis de tráfico en una sesión telnet

Telnet es un protocolo de red cuyo propósito es proveer de una comunicación bidireccional, principalmente proporcionando conexión remota con las terminales de hosts remotos. La comunicación entre estos terminales se realiza por la red sin usar encriptación, por lo que es un protocolo que se considera no seguro [7]. Pese a que hace años que el uso de este se vio reducido frente al protocolo SSH, que proporciona comunicación cifrada, telnet sigue usándose en dispositivos que se usan hoy en día como dispositivos de red del tipo router o switches. El fabricante Cisco es uno de los distribuidores que opera globalmente, y en sus manuales de configuración de los dispositivos se indica que la comunicación con los dispositivos se realiza mediante telnet, enviándose tanto los comandos como las credenciales en texto plano [2]. Un atacante puede hacer un uso malintencionado de Wireshark para interceptar estas comunicaciones y conseguir las credenciales de acceso a los router. En la siguiente simulación se pretende mostrar la captura como resultado de una escucha al protocolo telnet:

192.168.0.1	TCP	66 1550→23 [ACK] Seq=1 Ack=1 Win=32120 Len=0 TSval=10233636 TSecr=2467372
192.168.0.1	TELNET	93 Telnet Data ...
192.168.0.2	TELNET	69 Telnet Data ...
192.168.0.1	TCP	66 1550→23 [ACK] Seq=28 Ack=4 Win=32120 Len=0 TSval=10233651 TSecr=2467372
192.168.0.1	TELNET	69 Telnet Data ...
192.168.0.2	TCP	66 23→1550 [ACK] Seq=4 Ack=31 Win=17376 Len=0 TSval=2467372 TSecr=10233651
192.168.0.2	TELNET	91 Telnet Data ...
192.168.0.1	TELNET	130 Telnet Data ...
192.168.0.2	TCP	66 23→1550 [ACK] Seq=29 Ack=95 Win=17312 Len=0 TSval=2467372 TSecr=10233651
192.168.0.2	TELNET	84 Telnet Data ...
192.168.0.1	TELNET	75 Telnet Data ...
192.168.0.2	TCP	66 23→1550 [ACK] Seq=47 Ack=104 Win=17367 Len=0 TSval=2467372 TSecr=10233651
192.168.0.2	TELNET	90 Telnet Data ...
192.168.0.1	TCP	66 1550→23 [ACK] Seq=104 Ack=71 Win=32120 Len=0 TSval=10233652 TSecr=2467372
192.168.0.1	TELNET	151 Telnet Data ...
192.168.0.1	TCP	66 1550→23 [PSH, ACK] Seq=104 Ack=71 Win=32120 Len=0 TSval=10233652 TSecr=2467372
192.168.0.2	TELNET	69 Telnet Data ...
192.168.0.1	TELNET	69 Telnet Data ...
192.168.0.2	TCP	66 23→1550 [ACK] Seq=74 Ack=192 Win=17373 Len=0 TSval=2467372 TSecr=10233654
192.168.0.2	TELNET	78 Telnet Data ...
192.168.0.1	TELNET	72 Telnet Data ...
192.168.0.2	TCP	66 23→1550 [ACK] Seq=86 Ack=198 Win=17370 Len=0 TSval=2467372 TSecr=10233655
192.168.0.2	TELNET	81 Telnet Data ...
192.168.0.1	TCP	66 1550→23 [ACK] Seq=198 Ack=101 Win=32120 Len=0 TSval=10233657 TSecr=2467372
192.168.0.2	TELNET	98 Telnet Data ...
192.168.0.1	TCP	66 1550→23 [ACK] Seq=198 Ack=133 Win=32120 Len=0 TSval=10233659 TSecr=2467372
192.168.0.2	TELNET	73 Telnet Data ...
192.168.0.1	TCP	66 1550→23 [ACK] Seq=198 Ack=140 Win=32120 Len=0 TSval=10233769 TSecr=2467374
192.168.0.1	TELNET	72 Telnet Data ...
192.168.0.1	TCP	66 1550→23 [PSH, ACK] Seq=198 Ack=140 Win=32120 Len=0 TSval=10233892 TSecr=2467374

Figura 4.2: Escucha de protocolo telnet

Una vez se encuentre Wireshark capturando los paquetes, ofrece la funcionalidad de concatenar y ordenar los paquetes para reconstruir el flujo de comunicación con la herramienta Follow TCP stream, de esta manera se puede apreciar con detalle el contenido o payload de las tramas, al que podemos acceder y el cual es completamente legible al no estar cifrada la comunicación:

```

.....!.."..'.....#..%..%.....!.."..".....P. ....
.....".....'.....#..&..&..$..&..&..$.....#.....
0.0.....'..DISPLAY.bam.zing.org:0.0.....xterm-color.....!....
OpenBSD/i386 (oof) (ttyp2)

login: fake
.....Password:user

.....Last login: Sat Nov 27 20:11:43 on ttyp2 from bam.zing.org
Warning: no Kerberos tickets issued.
OpenBSD 2.6-beta (OOF) #4: Tue Oct 12 20:42:32 CDT 1999

Welcome to OpenBSD: The proactively secure Unix-like operating system

Please use the sendbug(1) utility to report bugs in the system.
Before reporting a bug, please try to reproduce it with the latest
version of the code. With bug reports, please try to ensure that
enough information to reproduce the problem is enclosed, and if a
known fix for it exists, include that as well.

$ /sbin/ping www.yahoo.com
PING www.yahoo.com (204.71.200.67): 56 data bytes
64 bytes from 204.71.200.67: icmp_seq=0 ttl=241 time=69.885 ms
64 bytes from 204.71.200.67: icmp_seq=1 ttl=241 time=73.591 ms
64 bytes from 204.71.200.67: icmp_seq=2 ttl=241 time=72.302 ms
64 bytes from 204.71.200.67: icmp_seq=3 ttl=241 time=73.493 ms
64 bytes from 204.71.200.67: icmp_seq=4 ttl=241 time=75.068 ms
64 bytes from 204.71.200.67: icmp_seq=5 ttl=241 time=70.239 ms
.....
.--- www.yahoo.com ping statistics ---
6 packets transmitted, 6 packets received, 0% packet loss
round-trip min/avg/max = 69.885/72.429/75.068 ms
$ ls
$ ls -a
.      ..      .cshrc  .login  .mailrc  .profile  .rhosts
$ exit

```

Figura 4.3: Contenido de la comunicación con telnet

Con esto se reafirma la necesidad de no usar protocolos que no conlleven encriptación o cifrado del contenido de los paquetes puesto que la integridad, confidencialidad y disponibilidad de la información se ven completamente comprometidas.

4.3. ARP

El principio de ARP Spoofing es enviar mensajes ARP modificados por la red Ethernet para conseguir asociar la dirección MAC del atacante con la dirección IP de otro host.[8] Para ello se modifican las tramas Ethernet para forzarle una dirección MAC de origen falsa y engañar a los dispositivos que estén en la red. Normalmente se hace creer que la MAC del atacante corresponde al router o default gateway [?]. Cualquier tráfico dirigido a la IP de ese nodo será enviado al atacante, el cual puede elegir reenviar ese tráfico a la puerta de enlace legítima, o modificar esta información antes de reenviarla.

Una forma de averiguar si la red está siendo comprometida mediante esta técnica, se puede monitorizar el tráfico ARP con Wireshark:

6	5.956722	AsustekC_7f:0d:2a	Broadcast	ARP	42 Who has 192.168.0.1? Tell 192.168.0.155
7	5.957593	SercommC_8a:45:e0	AsustekC_7f:0d:2a	ARP	60 192.168.0.1 is at d4:21:22:8a:45:e0
31	16.039900	AsustekC_7f:0d:2a	Broadcast	ARP	42 Who has 192.168.0.1? Tell 192.168.0.155
32	16.040800	SercommC_8a:45:e0	AsustekC_7f:0d:2a	ARP	60 192.168.0.1 is at d4:21:22:8a:45:e0
102	26.104412	AsustekC_7f:0d:2a	Broadcast	ARP	42 Who has 192.168.0.1? Tell 192.168.0.155
103	26.105278	SercommC_8a:45:e0	AsustekC_7f:0d:2a	ARP	60 192.168.0.1 is at d4:21:22:8a:45:e0
140	36.149841	AsustekC_7f:0d:2a	Broadcast	ARP	42 Who has 192.168.0.1? Tell 192.168.0.155
141	36.150638	SercommC_8a:45:e0	AsustekC_7f:0d:2a	ARP	60 192.168.0.1 is at d4:21:22:8a:45:e0
207	46.246340	AsustekC_7f:0d:2a	Broadcast	ARP	42 Who has 192.168.0.1? Tell 192.168.0.155
208	46.247147	SercommC_8a:45:e0	AsustekC_7f:0d:2a	ARP	60 192.168.0.1 is at d4:21:22:8a:45:e0
596	56.323697	AsustekC_7f:0d:2a	Broadcast	ARP	42 Who has 192.168.0.1? Tell 192.168.0.155
597	56.324576	SercommC_8a:45:e0	AsustekC_7f:0d:2a	ARP	60 192.168.0.1 is at d4:21:22:8a:45:e0
626	66.412067	AsustekC_7f:0d:2a	Broadcast	ARP	42 Who has 192.168.0.1? Tell 192.168.0.155
627	66.412941	SercommC_8a:45:e0	AsustekC_7f:0d:2a	ARP	60 192.168.0.1 is at d4:21:22:8a:45:e0
680	76.500621	AsustekC_7f:0d:2a	Broadcast	ARP	42 Who has 192.168.0.1? Tell 192.168.0.155
681	76.501433	SercommC_8a:45:e0	AsustekC_7f:0d:2a	ARP	60 192.168.0.1 is at d4:21:22:8a:45:e0
700	86.564009	AsustekC_7f:0d:2a	Broadcast	ARP	42 Who has 192.168.0.1? Tell 192.168.0.155
701	86.564955	SercommC_8a:45:e0	AsustekC_7f:0d:2a	ARP	60 192.168.0.1 is at d4:21:22:8a:45:e0
725	96.653463	AsustekC_7f:0d:2a	Broadcast	ARP	42 Who has 192.168.0.1? Tell 192.168.0.155
726	96.654379	SercommC_8a:45:e0	AsustekC_7f:0d:2a	ARP	60 192.168.0.1 is at d4:21:22:8a:45:e0
761	106.720902	AsustekC_7f:0d:2a	Broadcast	ARP	42 Who has 192.168.0.1? Tell 192.168.0.155
762	106.721775	SercommC_8a:45:e0	AsustekC_7f:0d:2a	ARP	60 192.168.0.1 is at d4:21:22:8a:45:e0
790	116.819415	AsustekC_7f:0d:2a	Broadcast	ARP	42 Who has 192.168.0.1? Tell 192.168.0.155
791	116.820308	SercommC_8a:45:e0	AsustekC_7f:0d:2a	ARP	60 192.168.0.1 is at d4:21:22:8a:45:e0
800	126.889857	AsustekC_7f:0d:2a	Broadcast	ARP	42 Who has 192.168.0.1? Tell 192.168.0.155

Figura 4.4: Protocolo ARP

Monitorizando el contenido de las comunicaciones se usa el filtro 'arp' para se vean los mensajes que intercambia el protocolo ARP. La forma de detectar este tipo de ataque pasa por vigilar que no existan IP duplicadas, y conociendo la MAC del router, se detecta cuándo estamos bajo un ataque de estas características:

1	1010.994450	universa_8f:92:3a	Broadcast	ARP	42 who has 192.168.2.253? Tell 192.168.2.102
2	1011.294443	universa_8f:92:3a	Broadcast	ARP	42 who has 192.168.2.254? Tell 192.168.2.102
3	1012.101040	universa_8f:92:3a	Broadcast	ARP	42 who has 192.168.2.254? Tell 192.168.2.102
4	1013.317261	universa_8f:92:3a	Azurewav_31:ff:65	ARP	42 192.168.2.1 is at 70:f3:95:8f:92:3a (duplicate use of 192.168.2.1 detecte
5	1015.318993	universa_8f:92:3a	Azurewav_31:ff:65	ARP	42 192.168.2.1 is at 70:f3:95:8f:92:3a (duplicate use of 192.168.2.1 detecte
6	1017.320592	universa_8f:92:3a	Azurewav_31:ff:65	ARP	42 192.168.2.1 is at 70:f3:95:8f:92:3a (duplicate use of 192.168.2.1 detecte
7	1019.322744	universa_8f:92:3a	Azurewav_31:ff:65	ARP	42 192.168.2.1 is at 70:f3:95:8f:92:3a (duplicate use of 192.168.2.1 detecte
8	1021.324403	universa_8f:92:3a	Azurewav_31:ff:65	ARP	42 192.168.2.1 is at 70:f3:95:8f:92:3a (duplicate use of 192.168.2.1 detecte
9	1023.325258	universa_8f:92:3a	Azurewav_31:ff:65	ARP	42 192.168.2.1 is at 70:f3:95:8f:92:3a (duplicate use of 192.168.2.1 detecte
10	1025.327395	universa_8f:92:3a	Azurewav_31:ff:65	ARP	42 192.168.2.1 is at 70:f3:95:8f:92:3a (duplicate use of 192.168.2.1 detecte
11	1027.329117	universa_8f:92:3a	Azurewav_31:ff:65	ARP	42 192.168.2.1 is at 70:f3:95:8f:92:3a (duplicate use of 192.168.2.1 detecte
12	1029.330937	universa_8f:92:3a	Azurewav_31:ff:65	ARP	42 192.168.2.1 is at 70:f3:95:8f:92:3a (duplicate use of 192.168.2.1 detecte
13	1031.331765	universa_8f:92:3a	Azurewav_31:ff:65	ARP	42 192.168.2.1 is at 70:f3:95:8f:92:3a (duplicate use of 192.168.2.1 detecte
14	1033.334261	universa_8f:92:3a	Azurewav_31:ff:65	ARP	42 192.168.2.1 is at 70:f3:95:8f:92:3a (duplicate use of 192.168.2.1 detecte
15	1035.335990	universa_8f:92:3a	Azurewav_31:ff:65	ARP	42 192.168.2.1 is at 70:f3:95:8f:92:3a (duplicate use of 192.168.2.1 detecte
16	1037.337842	universa_8f:92:3a	Azurewav_31:ff:65	ARP	42 192.168.2.1 is at 70:f3:95:8f:92:3a (duplicate use of 192.168.2.1 detecte

Figura 4.5: Detectando ARP spoofing

5. Comunidad

Wireshark cuenta con una amplia comunidad de desarrolladores que le da soporte, al ser éste de código abierto se presta a que cualquier interesado se preste a mejorar o añadir funcionalidades o módulos.

Un reflejo del interés que este software genera es el evento SHARKFEST, una conferencia anual de educación centrada en el intercambio de conocimientos, experiencias y mejores prácticas entre los miembros de las comunidades de usuarios y desarrolladores globales de Wireshark.

Los participantes se reúnen para perfeccionar sus habilidades en el arte del análisis de paquetes, asistiendo a una programación rica y variada de sesiones impartidas por los expertos más experimentados en la industria.[5]

Por otra parte Wireshark monopoliza el uso de este tipo de software, y para acreditar a los profesionales que lo usan ofrece formación y certificaciones mediante entidades colaboradoras. [6]

Referencias

- [1] Development and maintenance of wireshark. https://www.wireshark.org/docs/wsdg_html_chunked/ChIntroDevelopment.html/.
- [2] Set up your cisco router. http://www.cisco.com/public/technotes/smba/en/us/internet/proc_setup_router.pdf, 2006.
- [3] Documentación/manual de usuario. https://www.wireshark.org/docs/wsug_html/, 2016.
- [4] Github. <https://github.com/wireshark/wireshark>, 2016.
- [5] Sharkfest. <https://sharkfest.wireshark.org/>, 2016.
- [6] Wireshark university. <http://www.wiresharktraining.com/>, 2016.
- [7] J. Reynolds J. Poster. Rfc 854. <https://tools.ietf.org/html/rfc854>, 1983.
- [8] Sukumar Ramachandran, Vivek. Nandi. Information systems security: first international conference. ISBN 978-3-540-30706-8., 2005.