Ataques Man-in-the-Middle

Grupo 1:

David Cabezas Berrido

dxabezas@correo.ugr.es

Patricia Córdoba Hidalgo

patriciacorhid@correo.ugr.es

22 de mayo de 2021

Horas dedicadas al desarrollo del trabajo: 11

Índice

1. Simulación de ataque

A continuación, mostramos una simulación de un ataque Man-in-the-Middle, seguiremos este tutorial.

Creamos una nueva red local: `vboxnet1`. En ella tendremos los siguientes hosts:

- Máquina anfitriona: El atacante. Tiene Ubuntu 20.04.2 LTS.
- Cliente: Una de las víctimas del ataque.
- Servidor: La otra víctima. Tiene el servicio de Apache2 básico, por el que sirve un `index.html` con el mensaje "Hellow World!".

En la siguiente tabla podemos ver las direcciones (IP y MAC) de cada uno de los hosts.

Host	IP	MAC
Atacante	192.168.57.1	0a:00:27:00:00:01
Cliente	192.168.57.3	08:00:27:01:3a:c8
Servidor	192.168.57.4	08:00:27:6c:b7:f2

Tabla 1: Direcciones de las máquinas en la red `vboxnet1`.

Herramientas

Instalamos las siguientes herramientas en la máquina atacante.

Debemos instalar Ettercap, un paquete con utilidades para este tipo de ataques. Podemos hacerlo desde Ubuntu Software.

Para monitorizar el ataque utilizaremos Wireshark. También usaremos `urlsnarf` para husmear el tráfico HTTP entre las víctimas. Esta herramienta muestra las las peticiones HTTP que se realizan. Para descargarla, utilizamos el comando `sudo apt install dsniff`.

Ataque

El primer paso es activar la redirección de paquetes en la máquina atacante. De otro modo, las víctimas no podrían comunicarse entre sí, ya que los paquetes serían interceptados por el atacante, y se percatarían rápidamente del ataque. Podemos lograr esto ejecutando la orden `sysctl -w net.ipv4.ip_forward=1`, o escribiendo el valor 1 en el fichero `/proc/sys/net/ipv4/ip_forward`.

```
root@Lenovo:/proc/sys/net/ipv4# cat ip_forward
0
root@Lenovo:/proc/sys/net/ipv4# echo '1' > ip_forward
root@Lenovo:/proc/sys/net/ipv4# cat ip_forward
1
root@Lenovo:/proc/sys/net/ipv4# sysctl -p
```

Figura 1: Activamos la redirección de paquetes escribiendo un 1 en el fichero `/proc/sys/net/ipv4/ip_forward`.

Tras esto ejecutamos el comando `sysctl -p` para hacer efectivos los cambios recién realizados.

Seguidamente, lanzamos Ettercap en el atacante, especificando el nombre de la interfaz de red que queremos atacar: `vboxnet1`.

Figura 2: Pantalla de inicio de Ettercap. Seleccionamos la red `vboxnet1`.

Al entrar, recibimos los siguientes mensajes en la parte inferior de la ventana.

```
Listening on:
vboxnet1 -> 0A:00:27:00:00:01
    192.168.57.1/255.255.255.0
    fe80::800:27ff:fe00:1/64

SSL dissection needs a valid 'redir_command_on' script in the etter.conf file
Ettercap might not work correctly. /proc/sys/net/ipv6/conf/all/use_tempaddr is not set to 0.
Ettercap might not work correctly. /proc/sys/net/ipv6/conf/vboxnet1/use_tempaddr is not set to 0.
Privileges dropped to EUID 65534 EGID 65534...

34 plugins
42 protocol dissectors
57 ports monitored
24609 mac vendor fingerprint
1766 tcp OS fingerprint
2182 known services
Lua: no scripts were specified, not starting up!
Starting Unified sniffing...
```

Podemos detener y reanudar el sniffing (husmeo) pulsando el botón de stop/play en la esquina superior izquierda. Este comienza activado.

Pulsamos en el botón *Scan for hosts*, la lupa que se encuentra en la esquina superior izquierda, para buscar los host conectados a dicha interfaz. Al hacer esto nos aparece el siguiente mensaje en la parte inferior de la ventana:

```
Randomizing 255 hosts for scanning...
Scanning the whole netmask for 255 hosts...
3 hosts added to the hosts list...
```

Podemos ver los hosts encontrados pulsando en el botón que aparece a la derecha de la lupa, *Hosts List*.

Figura 3: Lista de hosts conectados a la red `vboxnet1`.

Entre los hosts encontrados se encuentran nuestras dos víctimas. Seleccionamos el cliente, aquel con la dirección IP 192.168.57.3, y hacemos click en el botón *Add to Target 1*. Análogamente seleccionamos el host con dirección IP 192.168.57.4 y hacemos click en el botón *Add to Target 2*. Al hacer esto nos aparecen los siguientes mensages en la ventana inferior.

```
Host 192.168.57.3 added to TARGET1
Host 192.168.57.4 added to TARGET2
```

La otra dirección que aparece, 192.168.57.2, corresponde al router.

Una vez seleccionadas nuestras víctimas procedemos a envenenar la caché de ARP, es decir, desde el atacante se enviarán paquetes a ambas víctimas para que estas lo identifiquen como la víctima contraria. Para ello pulsamos el botón de *MITM Menu*, aquel con el logo del globo terráqueo situado en la esquina superior derecha, y seleccionamos la opción *ARP poisoning*.

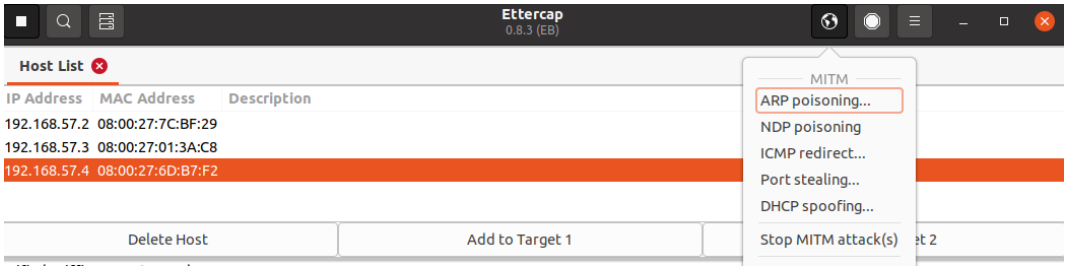


Figura 4: Pulsamos en el botón *MITM Menu*.

Nos aparece en una pantalla emergente con parámetros opcionales de esta funcionalidad. Marcamos el parámetro *Sniff remote connections*.

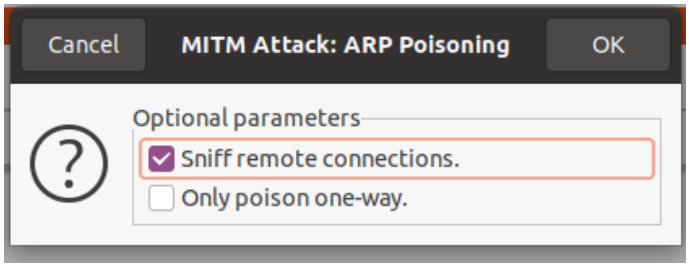


Figura 5: Ventana emergente con las opciones de *ARP poisoning*.

Obtenemos el siguiente mensage en la parte inferior de la ventana.

```
ARP poisoning victims:

GROUP 1 : 192.168.57.3 08:00:27:01:3A:C8

GROUP 2 : 192.168.57.4 08:00:27:6D:B7:F2
```

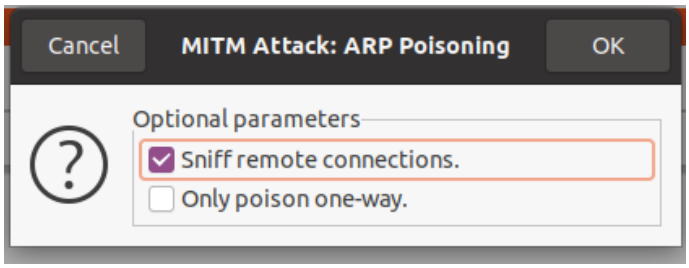


Figura 6: Ventana emergente con las opciones de *ARP poisoning*.

Abrimos Wireshark y capturamos los paquetes de la interfaz `vboxnet1`. Podemos observar que se envían varios paquetes desde 0a:00:27:00:00:01, el atacante, a ambas víctimas. Por ejemplo, si nos fijamos en el primer paquete capturado podemos comprobar que es un paquete del atacante al cliente. Este paquete informa al cliente de que la dirección IP 192.168.57.4 se encuentra en la dirección MAC del atacante, cuando realmente esa dirección IP pertenece al servidor. Análogamente en el segundo paquete observamos como el atacante informa al servidor de que la dirección IP del cliente, 192.168.57.3, se encuentra en la dirección física del atacante.

Por tanto, el atacante ha conseguido que ambas víctimas le identifiquen como la contraria, por lo que todo el tráfico que circule entre ambas máquinas pasará por este y será capaz de husmearlo. Ha envenenado la caché de ARP.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	0a:00:27:00:00:01	PcsCompu_01:3a:c8	ARP	42	192.168.57.4 is at 0a:00:27:00:00:01
2	0.000055607	0a:00:27:00:00:01	PcsCompu_6d:b7:f2	ARP	42	192.168.57.3 is at 0a:00:27:00:00:01
3	10.010320904	0a:00:27:00:00:01	PcsCompu_01:3a:c8	ARP	42	192.168.57.4 is at 0a:00:27:00:00:01
4	10.010376009	0a:00:27:00:00:01	PcsCompu_6d:b7:f2	ARP	42	192.168.57.3 is at 0a:00:27:00:00:01

Figura 7: ARP poisoning. El atacante envía paquetes a sus víctimas con una identificación falsa.

Seguidamente, lanzamos `urlsnarf` en el atacante, indicando la interfaz de red con la opción `-i`. Esto nos permitirá conocer las peticiones HTTP que se realicen entre el cliente y el servidor.

Análisis del tráfico

El ataque ya está montado. Ahora hacemos CURL desde el cliente al servidor (`curl http://192.168.57.4`). Observamos la siguiente secuencia de intercambios en Wireshark.

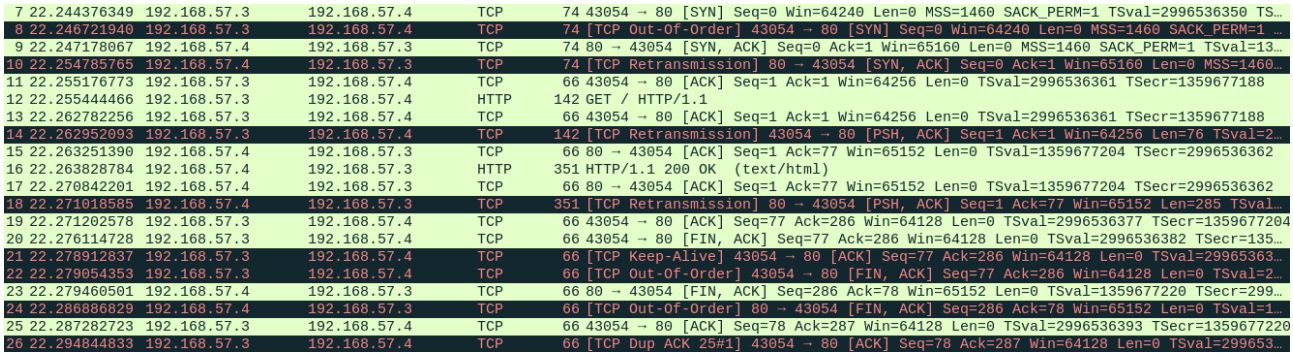


Figura 8: Comunicaciones capturadas mientras se realiza la petición.

Lo primero que nos llama la atención es la cantidad de paquetes *Bad TCP*, que aparecen en negro y najanja. Esto no es ningún fallo. Debido a que el atacante está actuando de intermediario, se realizan más intercambios de los esperados entre las máquinas. A la hora de comprobar que la comunicación es correcta, Wireshark también piensa que el atacante es una de las víctimas, de modo que la secuencia de paquetes que registra le confunde y piensa que se deben a errores en TCP. En seguida mostraremos ejemplos que reflejen esto.

Al analizar cualquiera de estos paquetes, tanto los relativos a la comunicación TCP (3-Way Handshake y cierre) como las peticiones y respuestas HTTP, descubrimos que en ningún momento el cliente y el servidor consiguen comunicar directamente. A pesar de lo que aparece en los campos IP origen e IP destino (tercera y cuarta columna), la dirección física de la máquina atacante aparece implicada en cada transmisión. Observamos ésto analizando los paquetes 7 y 8, correspondientes a SYN. Podemos consultar las direcciones físicas en la tabla 1.

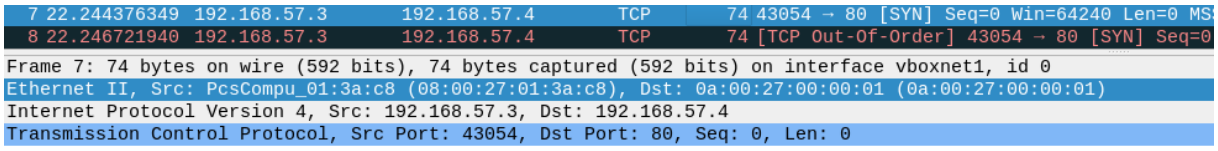


Figura 9: Las direcciones IP y física de origen corresponden al cliente; y la dirección IP de destino es la del servidor. Sin embargo, la dirección física de destino realmente corresponde a la máquina atacante, que es la que realmente recibe el paquete. Esto se debe a un fallo en la caché de ARP del cliente, creado por el atacante. En nuestro caso, el paquete se reenvía a su verdadero destino, ya que activamos la redirección de paquetes, pero el atacante tiene la oportunidad de conocer el contenido.

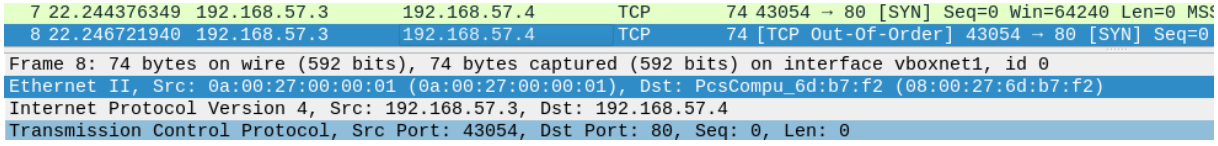


Figura 10: El atacante reenvía el SYN al servidor y éste se piensa que procede del cliente debido a un error en la caché de ARP del servidor creado por el atacante. La dirección IP de origen es la del cliente y tanto la dirección IP como la dirección física de destino son las del servidor. La dirección física de origen es la del atacante. Puesto que Wiresharck interpreta que tanto el mensaje anterior como este son SYN que proceden de la misma IP y con la misma IP de destino piensa que se ha podido producir algún error durante el 3-Way Handshake. Por esto, señala este segundo paquete como *Bad TCP* (Out-Of-Order).

Analizamos ahora los paquetes correspondientes a la petición HTTP que se realiza de la máquina cliente

al servidor.

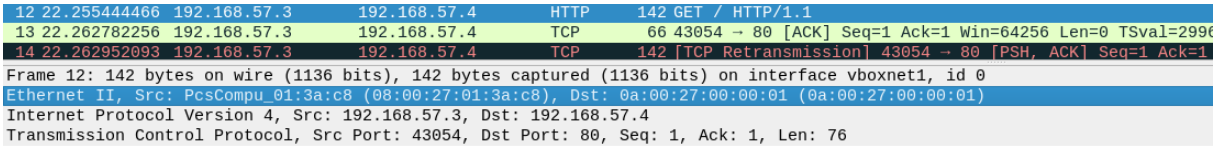


Figura 11: El cliente pretende mandar una petición HTTP al servidor. Podemos observar que la dirección IP y física de origen corresponden al cliente. La dirección IP de destino es la del servidor, pero la dirección física es la del atacante, luego realmente es al atacante al que se le manda la petición HTTP.

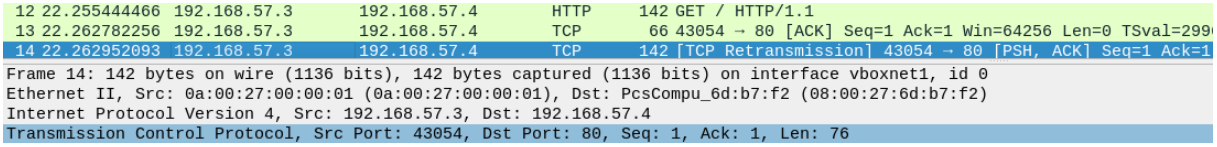
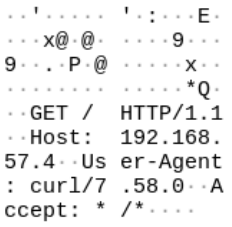
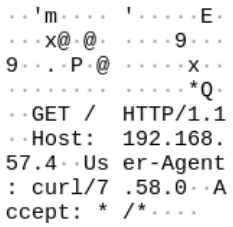


Figura 12: El atacante redirecciona la petición HTTP recibida al servidor.



(a) Contenido del paquete del cliente al atacante.



(b) Contenido del paquete del atacante al servidor.

Figura 13: En el contenido del paquete podemos ver la petición HTTP mandada.

Del mismo modo, podemos analizar los paquetes de respuesta de la petición del servidor al cliente.

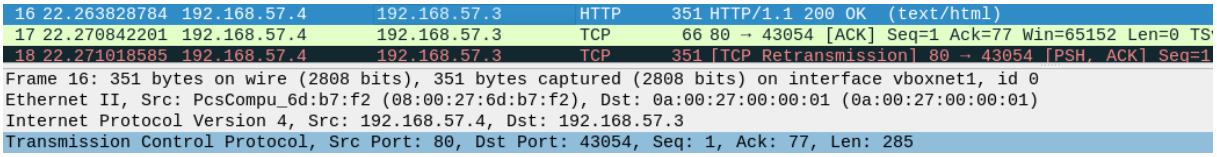


Figura 14: Como ocurre en el resto de paquetes, el servidor manda los paquetes cuya dirección IP de destino es el cliente a la dirección física del atacante. Así, el atacante puede husmear el contenido de los paquetes.

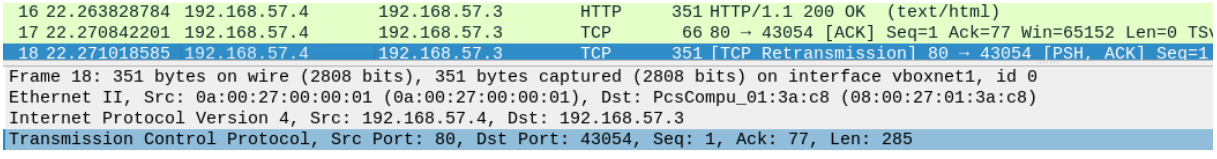
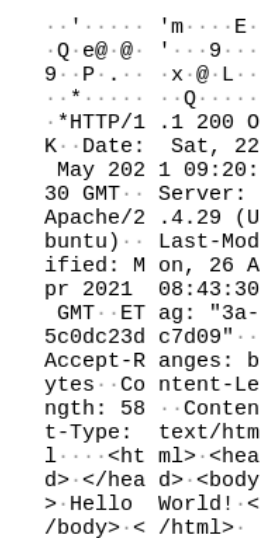
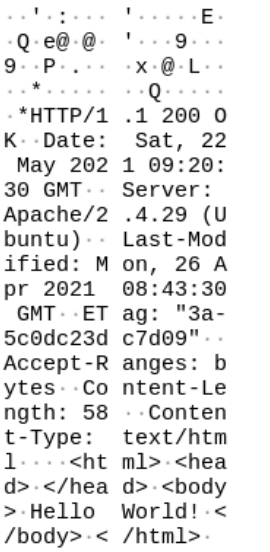


Figura 15: El atacante redirecciona el paquete recibido (la respuesta de la petición HTTP) al cliente.



(a) Contenido del paquete del servidor al atacante.



(b) Contenido del paquete del atacante al cliente.

Figura 16: En el contenido del paquete podemos ver la respuesta de la petición HTTP mandada.

Además, `urlsnarf` registra cada petición HTTP que se haga entre las víctimas. Aparece una entrada por cada petición.

```
dcabezas@Lenovo:~$ sudo urlsnarf -i vboxnet1
urlsnarf: listening on vboxnet1 [tcp port 80 or port 8080 or port 3128]
192.168.57.3 - - [22/May/2021:10:54:02 +0200] "GET http://192.168.57.4/ HTTP/1.1" - - "-" "curl/7.58.0"
192.168.57.3 - - [22/May/2021:10:55:09 +0200] "GET http://192.168.57.4/ HTTP/1.1" - - "-" "curl/7.58.0"
```

Figura 17: Aparece una entrada por cada petición HTTP entre las víctimas.

2. Bibliografía