Laboratorio de ICMP Redirect

1 Descripción

Una ICMP Redirect es un mensaje de error enviado por un router a quien envía un paquete IP. Las redirecciones son usadas por un router cuando este cree que un paquete está siendo ruteado de forma errónea y le quire informar a quien lo está enviando que debería de usar un router diferente para los paquetes que se envién de ahora en más al destino que se está usando. Un ICMP Redirect puede ser usado por atacantes para cambiar el enrutamiento de una determinada víctima.

El objetivo de este laboratorio es lanzar un ataque de ICMP Redirect sobre un una víctima, tal que los paquetes enviados por la víctima hacia `192.168.60.5`, sea usado por un contenedor router (`10.9.0.111`) malicioso. Dado que el router malicioso es controlado por el atacante, el atacante puede interceptar paquetes, cambiarlos y enviarlos modificados. Este es una forma de ataque Man-In-The-Middle (MITM).

Este laboratorio cubre los siguientes tópicos:

- Protocolos IP y ICMP
- Ataque ICMP Redirect
- Routing

Videos. Para una cobertura más detallada sobre el protocolo IP y ataques en la capa IP puede consultar:

- Sección 4 del curso de SEED en Udemy, *Internet Security: A Hands-on Approach*, by Wenliang Du. Para más detalles `https://www.handsonsecurity.net/video.html`.

Entorno de Laboratorio. Este laboratorio ha sido testeado en nuestra imagen pre-compilada de una VM con Ubuntu 20.04, que puede ser descargada del sitio oficial de SEED. Sin embargo, la mayoría de nuestros laboratorios pueden ser realizados en la nube para esto Ud. puede leer nuestra guía que explica como crear una VM de SEED en la nube.

2 Configuración del entorno usando Contenedores

En este laboratorio usaremos varias máquinas. La configuración del entorno del laboratorio se ilustra en la Figura 1. Para este laboratorio usaremos contenedores.

2.1 Setup del Contenedor y sus Comandos

Para empezar a preparar el contenedor, deberá descargarse el archivo `Labsetup.zip` ubicado en el laboratorio correspondiente dentro del sitio web oficial y copiarlo dentro de la Máquina Virtual prevista por SEED. Una vez descargado deberá descomprimirlo y entrar dentro del directorio `Labsetup` donde encontrará el archivo `docker-compose.yml` que servirá para setear el entorno de laboratorio. Para una

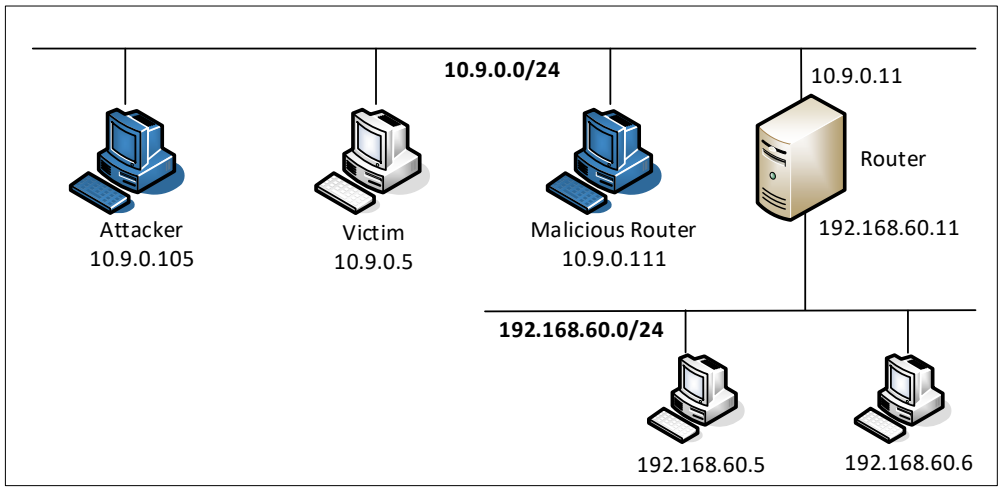

Figure 1: Configuración del entorno

información más detallada sobre el archivo `Dockerfile` y otros archivos relacionados, puede encontrarla dentro del Manual de Usuario del laboratorio en uso, en el sitio web oficial de SEED.

Si esta es su primera experiencia haciendo el setup del laboratorio usando contenedores es recomendable que lea el manual anteriormente mencionado.

A continuación, se muestran los comandos más usados en Docker y Compose. Debido a que estos comandos serán usados con mucha frecuencia, hemos creados un conjunto de alias para los mismos, ubicados en del archivo `.bashrc` dentro de la Máquina Virtual provista por SEED (Ubuntu 20.04)

```
$ docker-compose build  # Build the container image
$ docker-compose up     # Start the container
$ docker-compose down   # Shut down the container

// Aliases for the Compose commands above
$ dcbuild       # Alias for: docker-compose build
$ dcup          # Alias for: docker-compose up
$ dcdown        # Alias for: docker-compose down
```

Dado que todos los contenedores estarán corriendo en un segundo plano. Necesitamos correr comandos para interactuar con los mismos, una de las operaciones fundamentales es obtener una shell en el contenedor. Para este propósito usaremos "`docker ps`" para encontrar el ID del contenedor deseado y ingresaremos "`docker exec`" para correr una shell en ese contenedor. Hemos creado un alias para ello dentro del archivo `.bashrc`

```
$ dockps        // Alias for: docker ps --format "{{.ID}}  {{.Names}}"
$ docksh <id>   // Alias for: docker exec -it <id> /bin/bash

// The following example shows how to get a shell inside hostC
$ dockps
b1004832e275  hostA-10.9.0.5
0af4ea7a3e2e  hostB-10.9.0.6
9652715c8e0a  hostC-10.9.0.7
```

```
$ docksh 96
root@9652715c8e0a:/#

// Note: If a docker command requires a container ID, you do not need to
//       type the entire ID string. Typing the first few characters will
//       be sufficient, as long as they are unique among all the containers.
```

En caso de problemas configurando el entorno, por favor consulte la sección "Common Problems" en el manual ofrecido por SEED.

2.2 Sobre el Contenedor de Ataque

Para este laboratorio podemos usar tanto una Máquina Virtual como un contenedor como máquina de ataque. Si observa el archivo Docker Compose, verá que el contenedor de ataque está configurado de forma diferente al resto de los contenedores. Las diferencias son las siguientes:

- *Directorio Compartido.* Cuando usemos el contenedor del atacante para realizar los ataques, necesitamos poner el código de ataque dentro del contenedor. La edición del código es más conveniente dentro de la Máquina Virtual que dentro del contenedor, ya que podemos usar nuestro editor de texto preferido. Para que la Máquina Virtual y el contenedor puedan compartir archivos, hemos creado un directorio compartido entre ambos para esto hemos usado `volumes` de Docker. Dentro del archivo de Docker Composer, encontrará que se ha agregado esta entrada en algunos de los contenedores. Esta entrada indica que se montará el diretorio `./volumes` en la Máquina Host (es decir nuestra Máquina Virtual) y se podrá usar dentro del contenedor. Escribiremos nuestro código dentro del directorio `./volumes` (en la Máquina Virtual) y este podrá ser usado en el contenedor.

  ```
  volumes:
        - ./volumes:/volumes
  ```

- *Modo Privilegiado.* Para poder modificar parámetros del kernel en tiempo de ejecución (usando `sysctl`), tal como IP forwarding, el contenedor debe de ser privilegiado. Esto se consigue incluyendo la siguiente entrada dentro del archivo Docker compose del contenedor.

  ```
  privileged: true
  ```

3 Tarea 1: Lanzando el Ataque ICMP Redirect

En los sistemas operativos Ubuntu, existe una protección contra ataques ICMP Redirect. En el archivo Compose, hemos desactivado esta protección, haciendo que el contenedor de la víctima acepte mensajes ICMP Redirect.

```
// In docker-compose.yml
sysctls:
      - net.ipv4.conf.all.accept_redirects=1

// To turn the protection on, set its value to 0
# sysctl net.ipv4.conf.all.accept_redirects=0
```

Para esta tarea, atacaremos el contenedor víctima desde nuestro contenedor de ataque. En la configuración actual la víctima usara el contenedor router (`192.168.60.11`) como router de la red `192.168.60.0/24`. Al ejecutar `ip route` dentro del contenedor de la víctima, observaremos algo como esto:

```
# ip route
default via 10.9.0.1 dev eth0
10.9.0.0/24 dev eth0 proto kernel scope link src 10.9.0.5
192.168.60.0/24 via 10.9.0.11 dev eth0
```

Código Base. Se ha provisto un código base que se muestra a continuación con los paramétros esenciales que deberán de ser completados. Los estudiantes deberán de llenarlos con los valores apropiados, estos valores a completar son demarcados por `@@@@`.

```
#!/usr/bin/python3

from scapy.all import *

ip = IP(src = @@@@,  dst = @@@@)
icmp = ICMP(type=@@@@, code=@@@@)
icmp.gw = @@@@

# The enclosed IP packet should be the one that
# triggers the redirect message.
ip2 = IP(src = @@@@, dst = @@@@)
send(ip/icmp/ip2/ICMP());
```

Verificación. Los mensajes ICMP Redirect no afectan la tabla de routing; afectan a la caché del routing. Las entradas en la caché de routing sobreescriben aquellas que se encuentran en la tabla de routing hasta que cada una de estas expire. Para mostrar y limpiar el contenido de esta caché, podemos usar los siguientes comandos:

```
// Display the routing cache
# ip route show cache
192.168.60.5 via 10.9.0.111 dev eth0
    cache <redirected> expires 296sec

// Clean the routing cache
# ip route flush cache
```

Por favor haga un traceroute de la máquina víctima y vea si los paquetes son re-enrutados o no.

```
# mtr -n 192.168.60.5
```

Un hecho extraño. Mientras se fue desarrollando este laboratorio, hemos observado un hecho extraño en los entornos de los contenedores. Este hecho no se presenta si la víctima es la Máquina Virtual, en lugar del contenedor. Si spoofeamos los paquetes que se redireccionan pero la máquina víctima no está enviando paquetes ICMP durante el ataque, el ataque no será exitoso. Más aún, la `ip` dentro de un paquete de redirección debe de coincidir con el tipo y la dirección ip destino de los paquetes que la víctima está enviando (ICMP para ICMP, UDP para UDP, etc.).

Pareciera ser que el kernel del sistema operativo implementa algún tipo de chequeo de seguridad antes de aceptar paquetes ICMP Redirect. No se ha podido determinar con exactitud la causa de esto y porque la Máquina Virtual no tiene estas restricciones. Esta situación es algo que hasta la fecha SEED labs no ha podido solucionar, se alienta a los estudiantes a que nos ayuden a resolver esta cuestión. Recomendamos a los instructores otorgar puntos extras a aquellos estudiantes que hayan podido solucionar este inconveniente.

Antes de encontrar una manera de desactivar este chequeo preventivo y lanzar el ataque, deberíamos de hacer un `ping` al host `192.168.60.5` en la máquina víctima.

Preguntas. Después de que el ataque haya sido exitoso, por favor realice los siguientes experimentos y vea si su ataque sigue funcionando. Por favor explique sus observaciones:

- Pregunta 1: ¿Puede usar ataques ICMP Redirect para redireccionar en una máquina remota? A saber, la dirección IP asignada en `icmp.gw` es una computadora que no está en la LAN local. Muestre el resultado de su experimento y explique su observación.
- Pregunta 2: ¿Puede usar ataques ICMP Redirect para redireccionar en una máquina que no existe en la misma red? A saber, la dirección IP asignada en `icmp.gw` es una computadora que puede estar offline o puede no existir. Muestre el resultado de su experimento y explique su observación.
- Pregunta 3: Si observa en el archivo `docker-compose.yml`, encontrará las siguientes entradas para el contenedor router malicioso. ¿Cual es el propósito de estas entradas? Por favor modifique los valores a `1`, realize el ataque nuevamente. Por favor describa y explique su observación.

```
sysctls:
     - net.ipv4.conf.all.send_redirects=0
     - net.ipv4.conf.default.send_redirects=0
     - net.ipv4.conf.eth0.send_redirects=0
```

4 Tarea 2: Lanzando el Ataque MITM

Usando el ataque de ICMP Redirect, podemos lograr que la víctima use nuestro router malicioso (`10.9.0.111`) como el router para el destino `192.168.60.5`. Todos los paquetes de la máquina de la víctima hacia este destino serán ruteados por nuestro router malicioso. Queremos modificar los paquetes de la víctima.

Antes de ejecutar el ataque MITM, debeos de ejecutar un cliente y un servidor TCP usando `netcat`. Vea los siguientes comandos:

```
On the destination container 192.168.60.5, start the netcat server:
# nc -lp 9090

On the victim container, connect to the server:
# nc 192.168.60.5 9090
```

Una vez que se establece la conexión, puede escribir mensajes en la máquina víctima. Cada linea de mensajes será colocada en un paquete TCP que será enviado al destino, quién mostrará el mensaje. Su tarea es reemplazar cada ocurrencia de su nombre dentro del mensaje con una secuencia de letras A. La longitud de la secuencia debería de ser la misma que la longitud de su nombre de lo contrario el número de secuencia TCP se verá corrompido y por ende toda la conexión TCP. Necesita usar su nombre real, de esta forma sabremos que el trabajo fue hecho por ud.

Desactivando IP Forwarding. En la configuración que se usa, el IP Forwarding para el router está activado, lo que le permite funcionar como un router y forwardear paquetes. Cuando lanzamos un ataque MITM, debemos de detener el forwardeo de paquetes IP; en vez de esto, interceptaremos el paquete, lo modificaremos y enviaremos un nuevo paquete. Para hacer esto, necesitamos desactivar el IP Forwarding en el router malicioso.

```
# sysctl net.ipv4.ip_forward=0
```

Código MITM. Una vez desactivado el IP Forwarding, nuestro programa necesita ser capaz de hacer el forwardeo de los paquetes desde la víctima hacia el objetivo, desde ya todo esto se hace después de modificar los paquetes. Dado que el destino de un paquete no somos nosotros, el kernel no nos lo entregará; simplemente descartará el paquete. Sin embarg, si nuestro programa es un sniffer, podemos obtener el paquete del kernel. Para implementar este ataque MITM usaremos la técnica de sniff-and-spoof. A continuación brindamos un ejemplo de un programa de sniff-and-spoof que captura paquetes TCP, modifica algunos de sus valores antes de que sean enviados. Puede encontrar el código dentro de los archivos del laboratorio.

Listing 1: Sample code: `mitm_sample.py`

```
#!/usr/bin/env python3
from scapy.all import *

def spoof_pkt(pkt):
   newpkt = IP(bytes(pkt[IP]))
   del(newpkt.chksum)
   del(newpkt[TCP].payload)
   del(newpkt[TCP].chksum)

   if pkt[TCP].payload:
       data = pkt[TCP].payload.load
       print("*** %s, length: %d" % (data, len(data)))

       # Replace a pattern
       newdata = data.replace(b'seedlabs', b'AAAAAAAA')

       send(newpkt/newdata)
   else:
       send(newpkt)

f = 'tcp'
pkt = sniff(iface='eth0', filter=f, prn=spoof_pkt)
```

Cabe señalar que el código anterior, captura todo los paquetes TCP, incluído el que es generado por el mismo programa. Eso no es conveniente, y afectará a la performance. Los estudiantes necesitan cambiar el filtro para que el programa no capture sus propios paquetes.

Preguntas. Después de lograr un ataque exitoso, por favor conteste las siguientes preguntas:

- Pregunta 4: En el programa de MITM, ud. sólo necesita captura el tráfico en una sola dirección. Por favor indique cual es esta dirección y explique el porque.
- Pregunta 5: En el programa de MITM, cuando ud. captura el tráfico de`nc` desde A (`10.9.0.5`), puede usar la dirección IP de A o la dirección MAC en el filtro. Una de estas dos opciones no es la

correcta y va a crear un inconveniente, aunque ambas pueden funcionar. Por favor intente ambas y use los resultados de su experimentos para demostrar cual de estas dos opciones en la correcta y explique su conclusión.

5 Informe del Laboratorio

Debe enviar un informe de laboratorio detallado, con capturas de pantalla, para describir lo que ha hecho y lo que ha observado. También debe proporcionar una explicación a las observaciones que sean interesantes o sorprendentes. Enumere también los fragmentos de código más importantes seguidos de una explicación. No recibirán créditos aquellos fragmentos de códigos que no sean explicados.

Agradecimientos

Este documento ha sido traducido al Español por Facundo Fontana