

TryHackMe – Pickle Rick CTF Write-Up

Name: Sidhardh D

Platform: TryHackMe

Room: Pickle Rick

Target IP: 10.10.110.194

Difficulty: Easy

Tags: Web, Enumeration, Linux, Privilege Escalation, RCE

Overview

The goal of this CTF is to recover three hidden ingredients that Rick needs to turn back into a human. The box is based on a simple Linux web server with several hidden directories, clues in HTML source code, command injection, and basic privilege escalation via sudo misconfiguration.

Tools Used

Tool	Use Case
nmap	Port and service scanning
gobuster	Directory brute-forcing
Firefox	Manual inspection of webpages
Bash Shell	Interaction with the target shell
sudo, less	Escalation and file access

Step 1: Port Scanning with Nmap

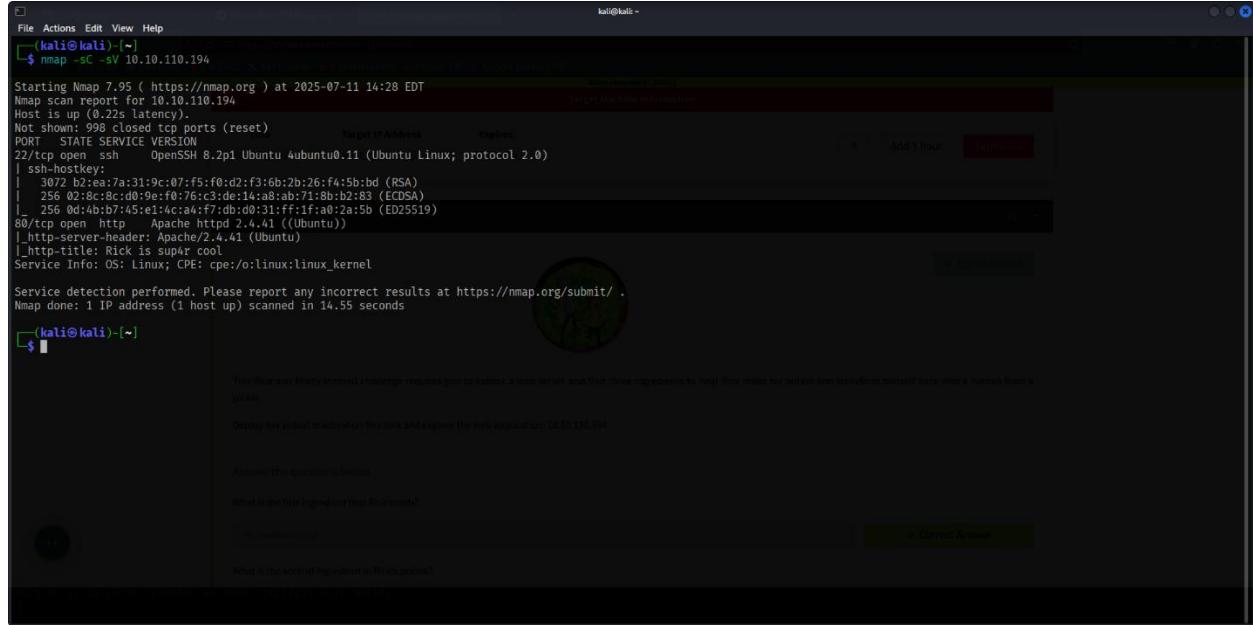
To begin, I scanned the target IP for open ports and running services using Nmap:

```
nmap -sC -sV 10.10.110.194
```

This command performs a default script scan (-sC) and attempts to determine service versions (-sV).

Scan Output:

```
22/tcp open  ssh      OpenSSH 8.2p1 Ubuntu  
80/tcp open  http     Apache httpd 2.4.41 (Ubuntu)
```



```
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-11 14:28 EDT  
Nmap scan report for 10.10.110.194  
Host is up (0.22s latency).  
Not shown: 998 closed tcp ports (reset)  
PORT      STATE SERVICE VERSION  
22/tcp    open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.11 (Ubuntu Linux; protocol 2.0)  
| ssh-hostkey:  
|   3072 b2:ea:a3:31:9c:07:f5:f0:d2:f3:6b:2b:26:f4:5b:bd (RSA)  
|   256 02:8c:8c:d0:9e:f0:76:c3:de:14:a8:ab:71:8b:b2:83 (ECDSA)  
|   256 0d:46:b7:45:e1:c4:a4:f7:db:d0:31:ff:1f:a0:2a:5b (ED25519)  
80/tcp    open  http     Apache httpd 2.4.41 ((Ubuntu))  
| http-server-header: Apache/2.4.41 (Ubuntu)  
| http-title: Rick is super cool  
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 14.55 seconds
```

The SSH port is open, but without credentials it's inaccessible at this point. The HTTP port is of interest and likely holds our initial foothold.

Step 2: Manual Web Exploration

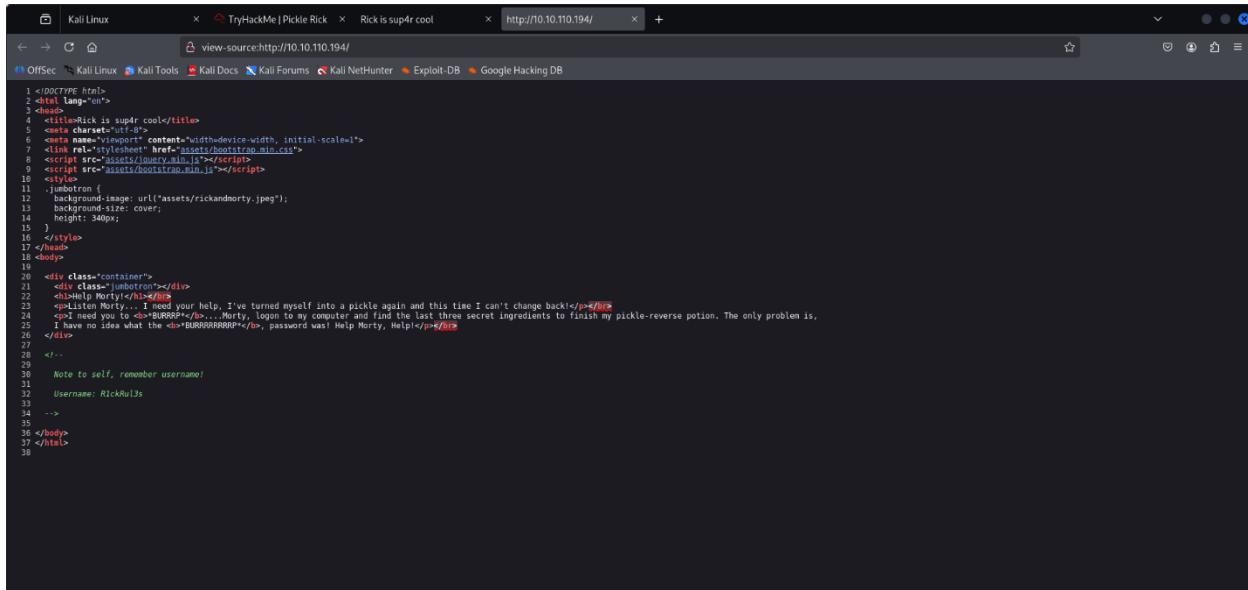
Navigating to the IP in a browser:

```
http://10.10.110.194
```

The homepage shows a Rick and Morty-themed message and nothing interactive. I performed basic checks like hovering over elements and looking for hidden input boxes but found none.

Next, I viewed the page source using the browser (Ctrl + U) and found a hardcoded comment:

Username: RickRul3s --



```
1 <!DOCTYPE html>
2 <html lang="en">
3   <!--
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    </!--
11    <jumbotron>
12      background-image: url("/assets/rickandmorty.jpeg");
13      background-size: cover;
14      height: 30px;
15    </jumbotron>
16  </head>
17 <body>
18
19   <div class="container">
20     <div class="jumbotron">
21       <h1>Help Morty!</h1></div>
22       <p>I need you to <b>BURRRP</b>...Morty, logon to my computer and find the last three secret ingredients to finish my pickle-reverse potion. The only problem is, I have no idea what the <b>BURRRRRRRP</b>, password was! Help Morty, Help!</p><br>
23   </div>
24
25 </div>
26
27 <!--
28 Note to self, remember username!
29
30 Username: RickRul3s
31
32 -->
33
34
35
36 </body>
37 </html>
```

This is a clear hint that a login function exists somewhere and gives us the username upfront.

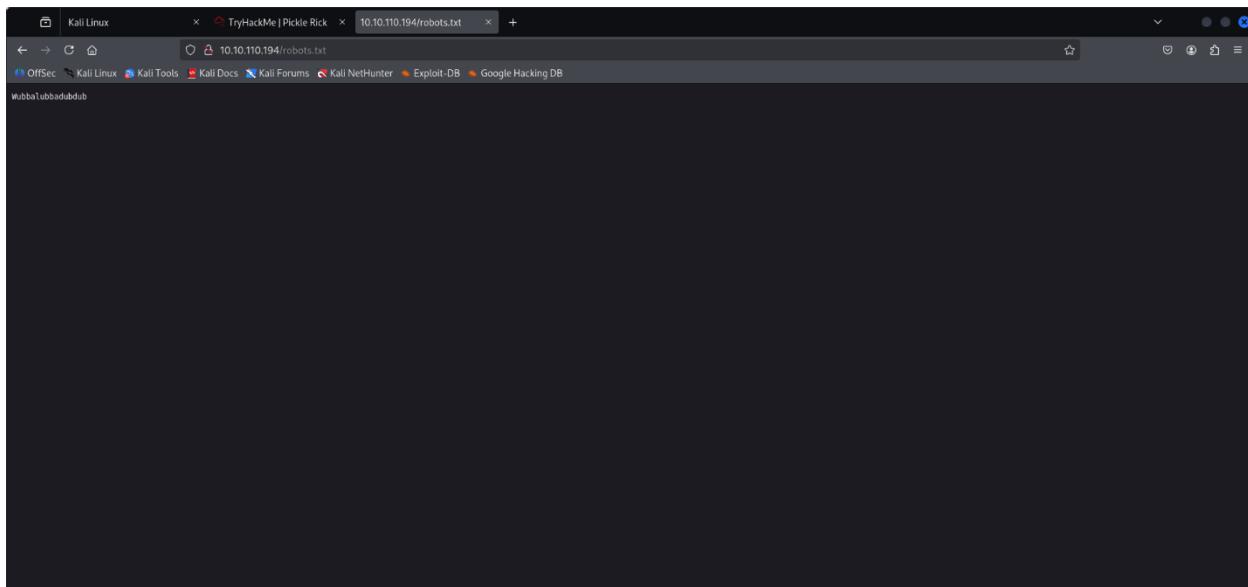
Step 3: Finding Password via robots.txt

A common misconfiguration is leaving sensitive information in `robots.txt`. I navigated to:

`http://10.10.110.194/robots.txt`

It contained the line:

`Wubbalubbadubdub`



Given the format, it's likely the password corresponding to the username found earlier.

So far, we have:

- **Username:** RickRul3s
 - **Password:** Wubbalubbadubdub

Step 4: Discovering Hidden Pages with Gobuster

To locate hidden directories or files, I ran Gobuster:

```
sudo gobuster dir -u http://10.10.110.194/ \ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt \ -x php,txt,html
```

Results:

- /login.php – likely the login page
 - /portal.php – potentially a restricted or post-login page
 - /robots.txt – already accessed
 - /assets/ – usually static files like CSS or images

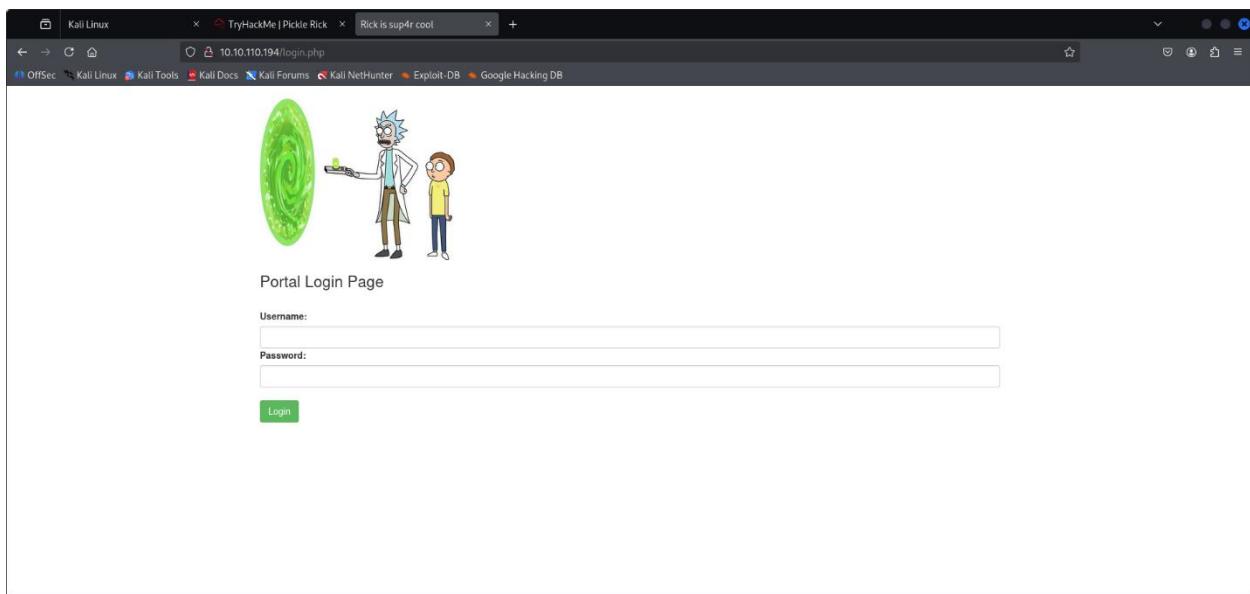
```
[kali㉿kali: ~] $ gobuster dir -u http://10.10.110.194/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,txt,html,js,py
[+] Url: http://10.10.110.194/ [Status: 403] [Size: 278]
[+] Threads: 10 [Status: 200] [Size: 1063]
[+] Threads: 10 [Status: 301] [Size: 315] [→ http://10.10.110.194/assets/]
[+] Threads: 10 [Status: 302] [Size: 0] [→ /login.php]
[+] Threads: 10 [Status: 200] [Size: 17]
Progress: 15750 / 1323366 (1.19%)
```

This confirmed that /login.php is where we can try the credentials.

Step 5: Logging In

I visited:

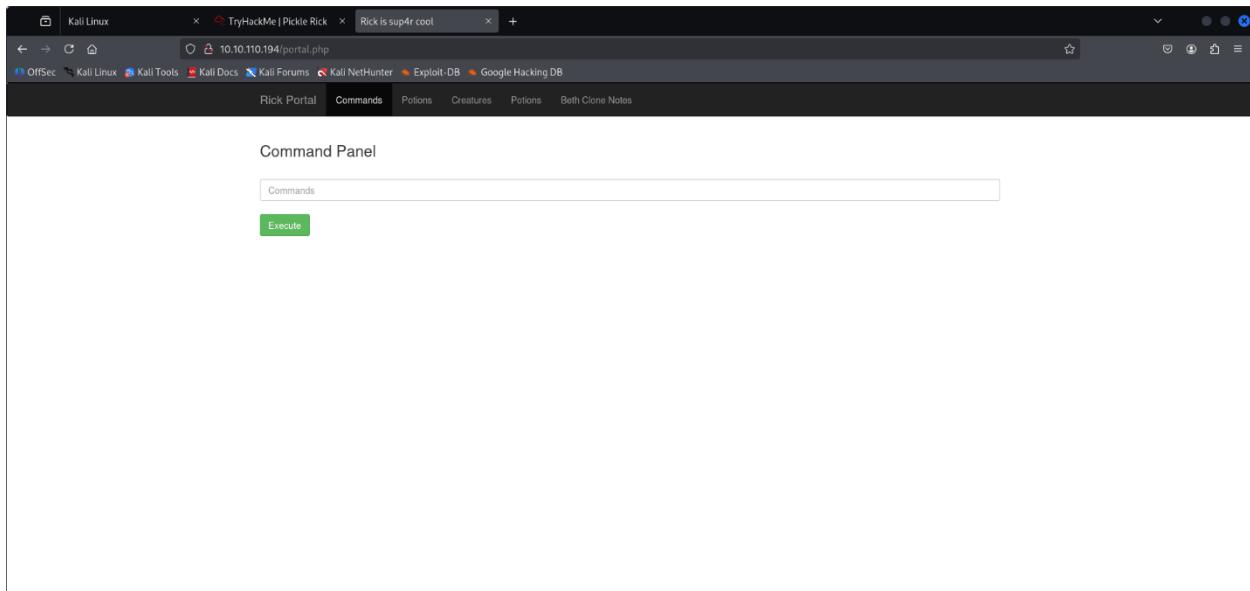
<http://10.10.110.194/login.php>



and entered the known credentials:

- **Username:** RickRul3s
- **Password:** Wubbalubbadubdub

The login succeeded and redirected me to /portal.php, which contains a simple form allowing command input.



Step 6: Exploring the Portal Command Interface

This page has a single-line text field which appears to execute commands on the server. I verified functionality with:

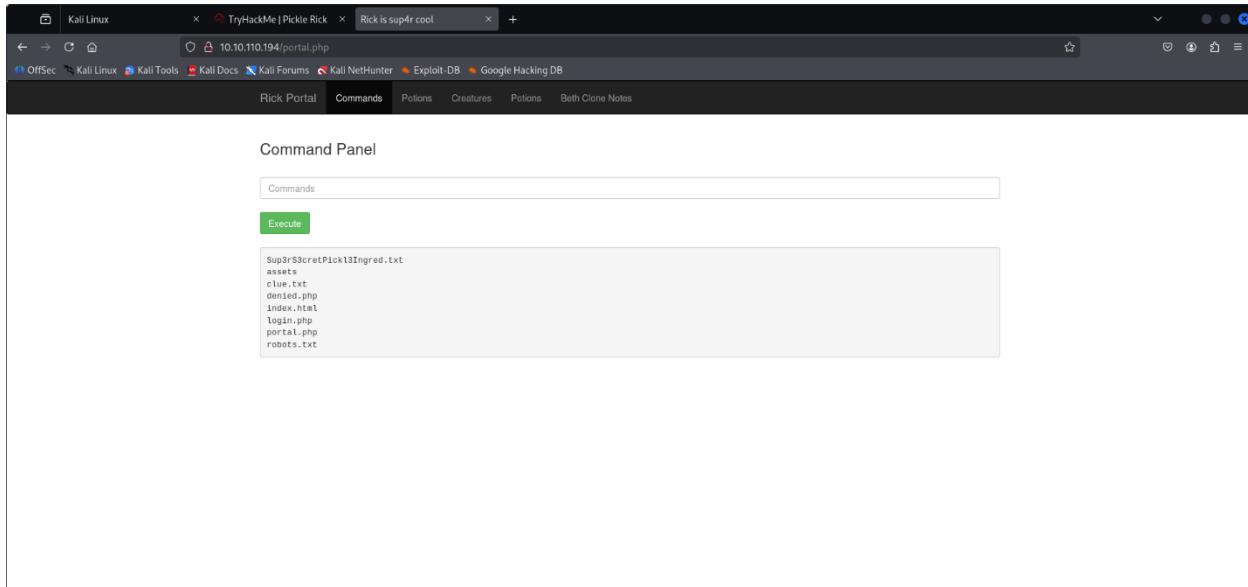
```
whoami
```

Result:

```
www-data
```

Then I listed directory contents:

```
ls
```



I saw the file Sup3rS3cretPickl3Ingred.txt.

I first attempted to use cat:

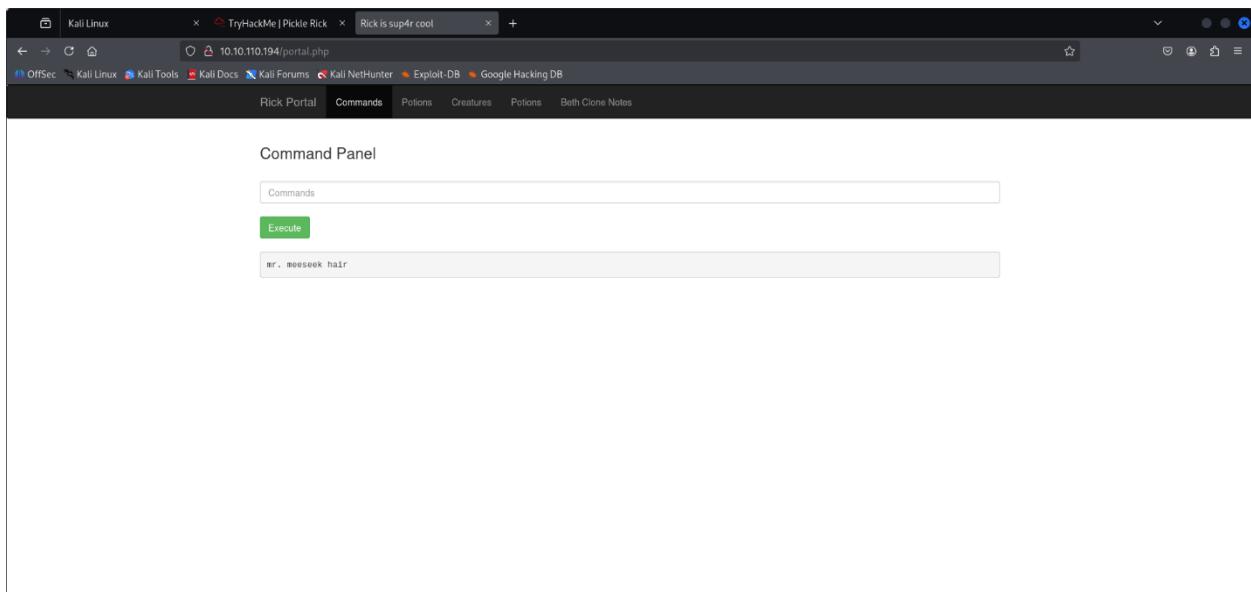
```
cat Sup3rS3cretPickl3Ingred.txt
```

It returned:

Command disabled

Then I tested less, which worked:

```
less Sup3rS3cretPickl3Ingred.txt
```



🔑 **Ingredient 1:** mr. meeseek hair

Step 7: Reading the Clue

Listing files again, I found:

```
clue.txt
```

Using the working `less` command:

```
less clue.txt
```

Content:

“Look around the file system for the other ingredients.”

This was a useful hint. It told me that the remaining ingredients aren’t here in the web root but somewhere else in the file system. Based on this, I tried navigating to common system directories.

Step 8: Exploring User Home Directory

Since user data often resides in `/home`, I tried:

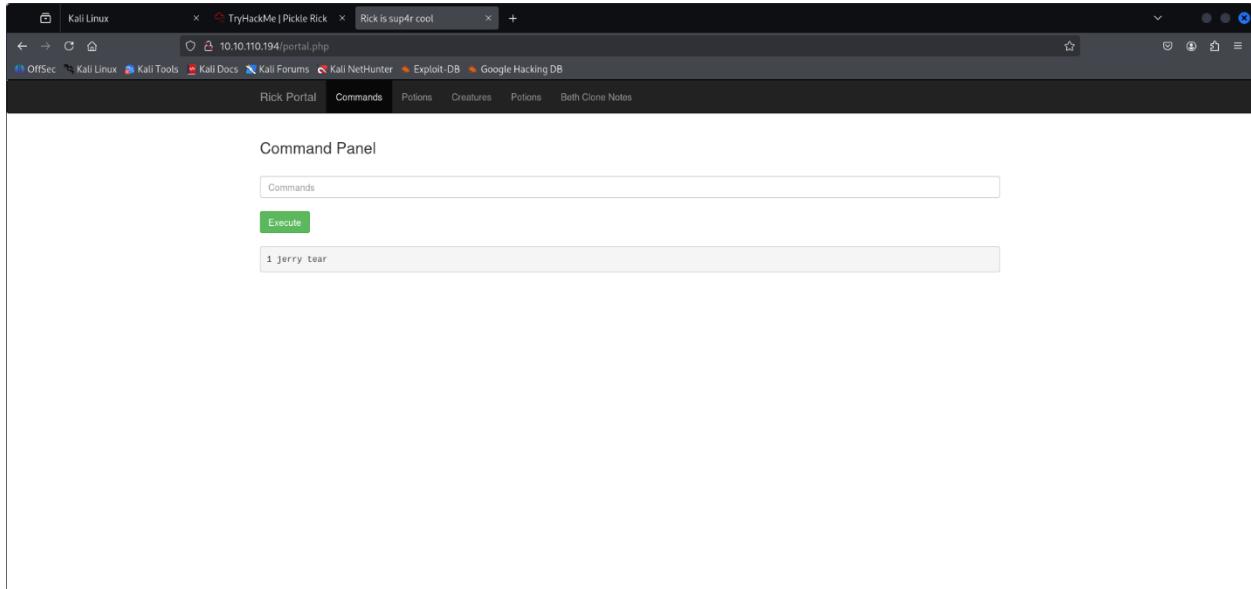
```
cd /home  
ls  
cd rick  
ls
```

I found a file called:

second ingredients

Accessing it with:

```
less "second ingredients"
```



🔑 **Ingredient 2: 1 jerry tear**

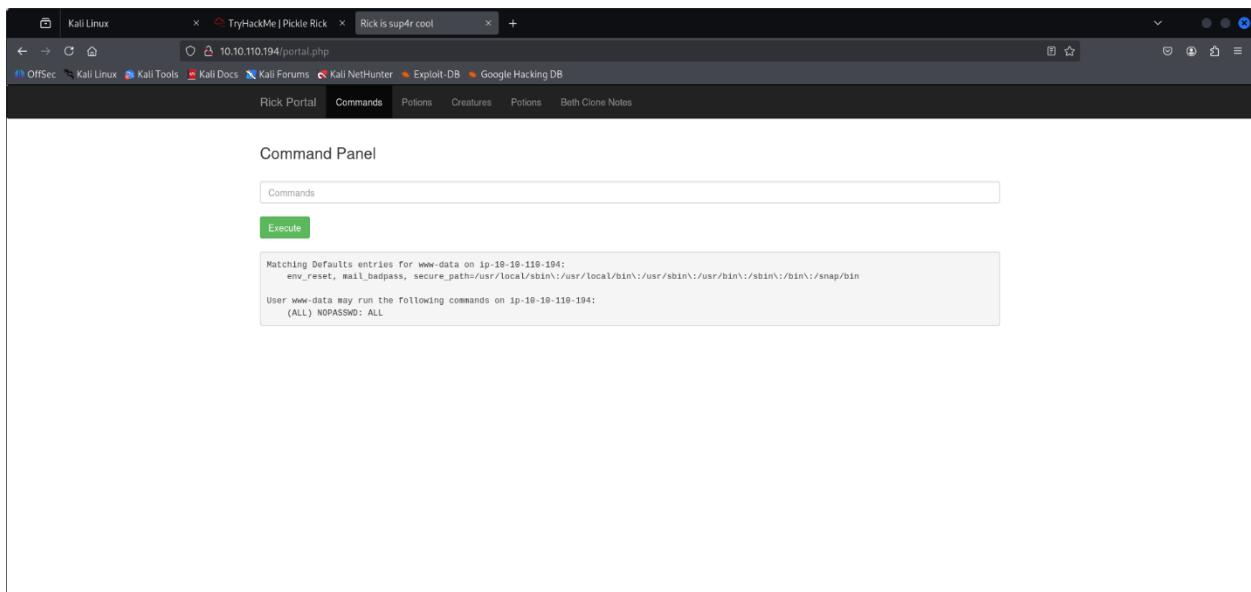
Step 9: Privilege Escalation

Next, I tested for elevated permissions:

```
sudo -l
```

It returned:

```
(ALL) NOPASSWD: ALL
```

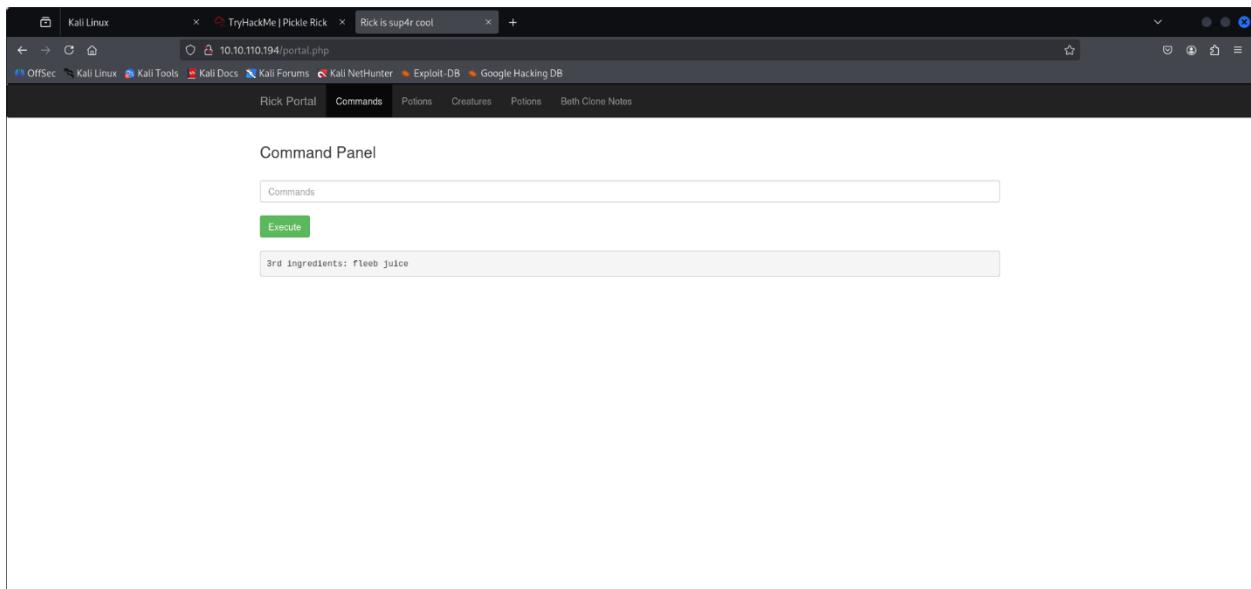


This confirmed I could run any command as root without needing a password. I escalated to root using:

```
sudo -i
```

Once as root, I checked:

```
cd /root  
ls  
less 3rd.txt
```



🔑 **Ingredient 3: fleeb juice**

Final Summary

Ingredient	Value
1	mr. meeseek hair
2	1 jerry tear
3	fleeb juice

The screenshot shows a web browser window for the TryHackMe platform. The title bar reads "Kali Linux" and "TryHackMe | Pickle Rick". The URL is "https://tryhackme.com/room/picklerick". The page content is titled "Rick is sup4r cool" and features a circular icon with two cartoon characters. Below the icon, text says: "This Rick and Morty-themed challenge requires you to exploit a web server and find three ingredients to help Rick make his potion and transform himself back into a human from a pickle. Deploy the virtual machine on this task and explore the web application: 10.10.110.194". There are three input fields for answers: "mr. meeseek hair", "1 jerry tear", and "fleeb juice", each with a "Correct Answer" button. A progress bar at the bottom indicates "Room completed (100%)".

This concludes the Pickle Rick CTF. The room was simple but instructive in web enumeration, dealing with restricted shell environments, and basic privilege escalation via misconfigured sudo rules.
