**Phase 1**

**Advanced Exploitation Lab**

**Executive Summary**

The Objective of this phase is to simulate a chained attack on Mr. Robot VM, develop custom PoC, bypassing modern defense systems. This lab aims to improve exploitation skills using tools such as Metasploit, python, Ghidra while performing full attack life cycle. Additional goals include using public exploits, modify them to our needs and documenting all the steps for replay ability.

**Tools & Environment**

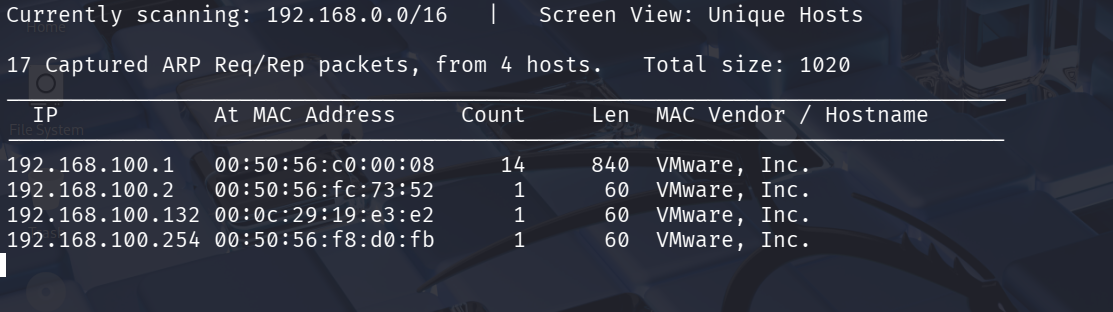
1. Kali Linux Environment
2. Mr Robot VM
3. Metasploit
4. Hydra
5. Netdiscover
6. Nmap

**Methodology**

1. **Reconnaissance**

First step is to discover the Mr. Robot VM On the Network. We can do this by switch to Sudo user and using netdiscover command.

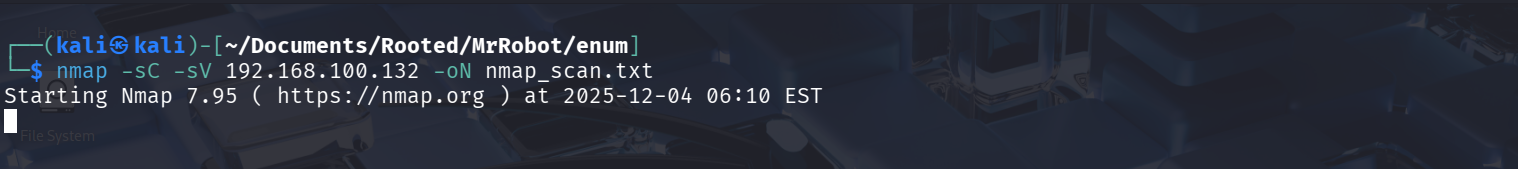
****

****

By the Above snapshot/output we can determine the IP address of the **Mr. Robot VM = 192.168.100.132**

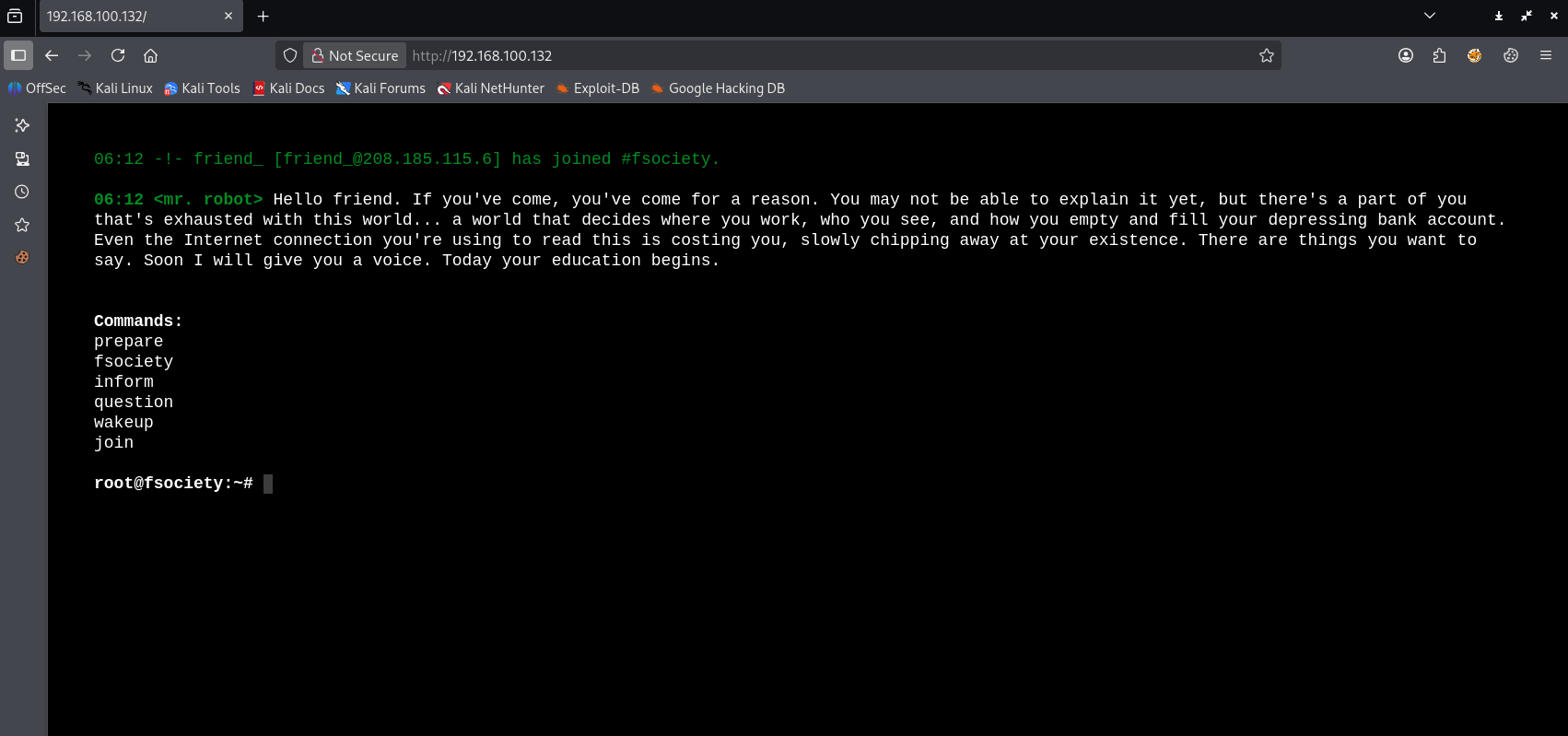
1. **Nmap Scanning of the Kioptrix**

Now we perform Nmap Enumeration/Scan on the Mr Robot VM machine.



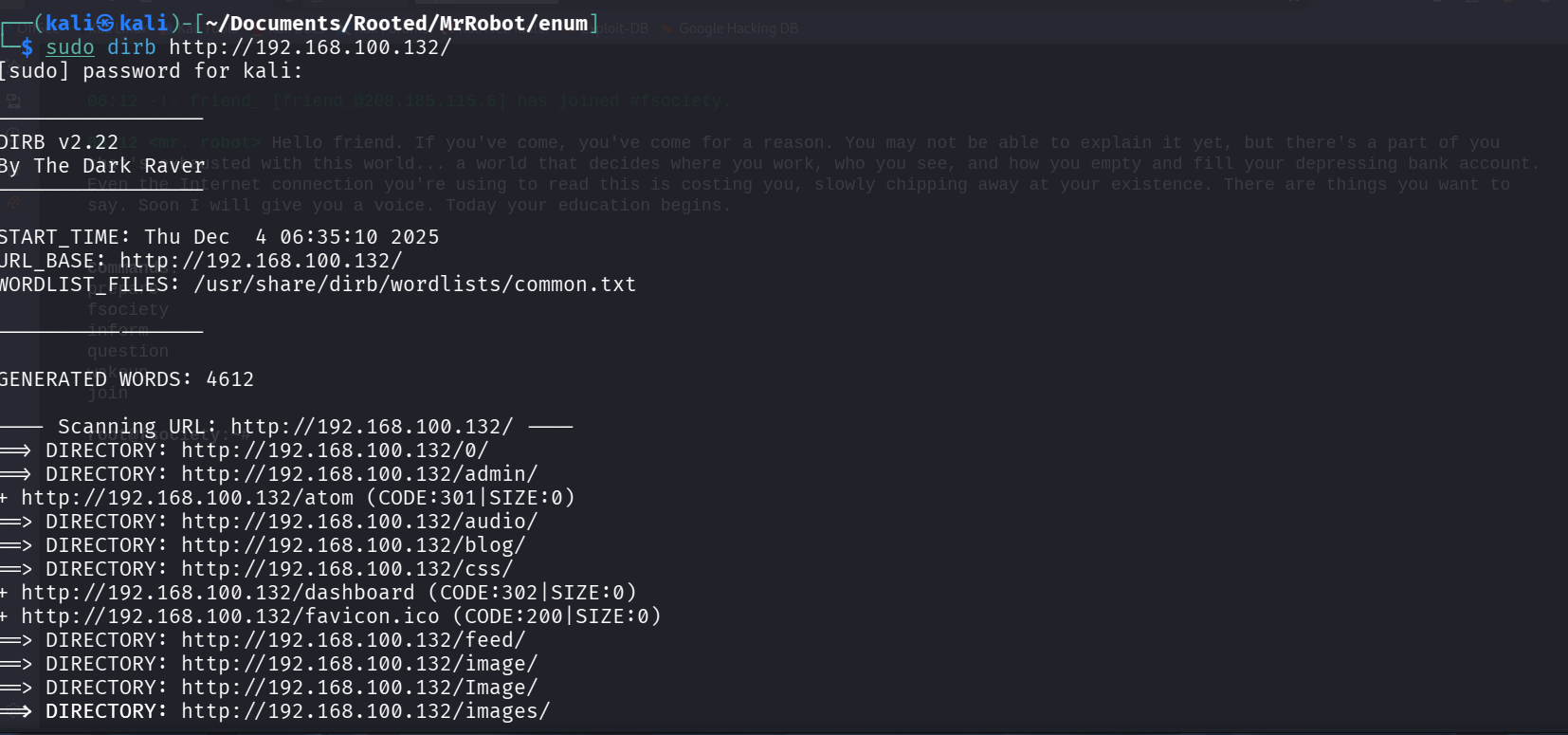


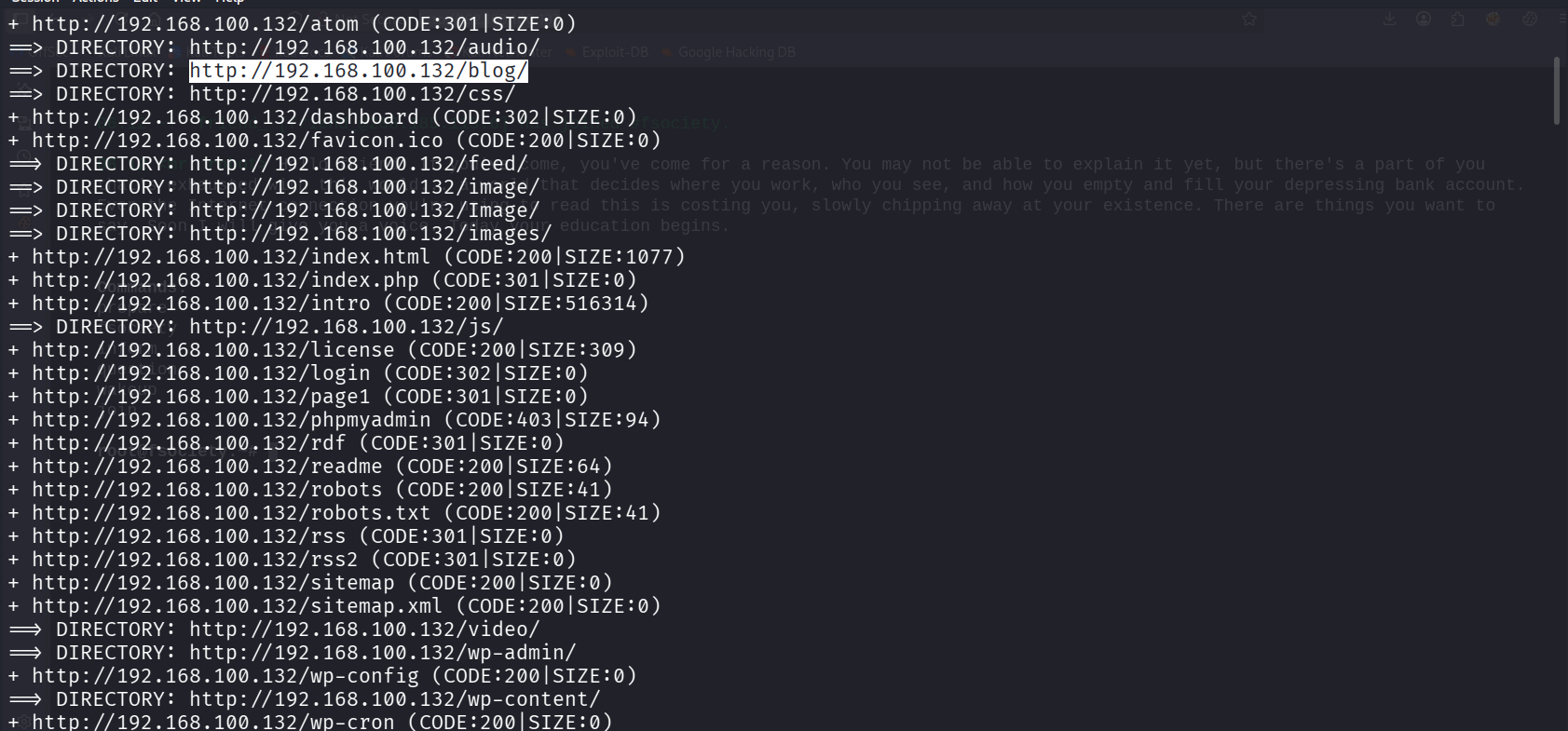
The Scan Resulted in three Open services and the scan output was saved for further processing.



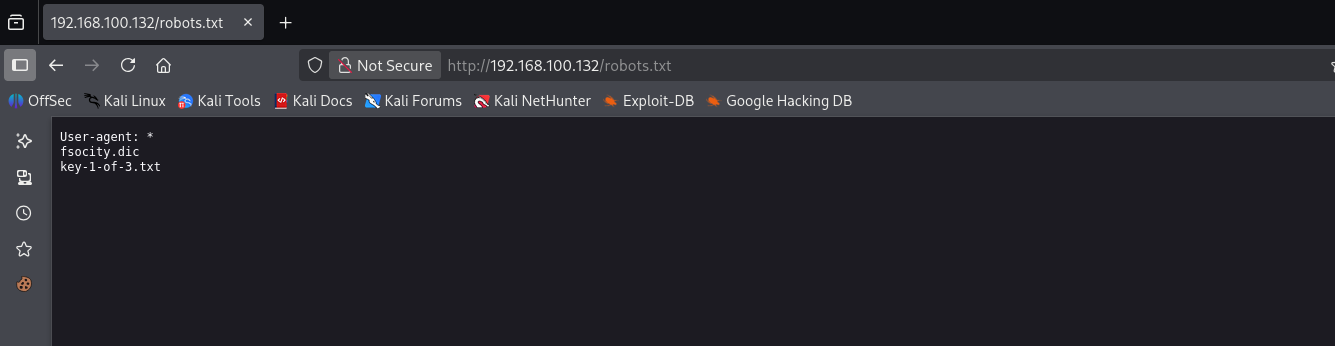
First Look of the website of Mr Robot.

1. **Subdomain enumeration**

We will further enumerate this website using Dirbuster 



It resulted in several different subdomains. Let’s check robots.txt



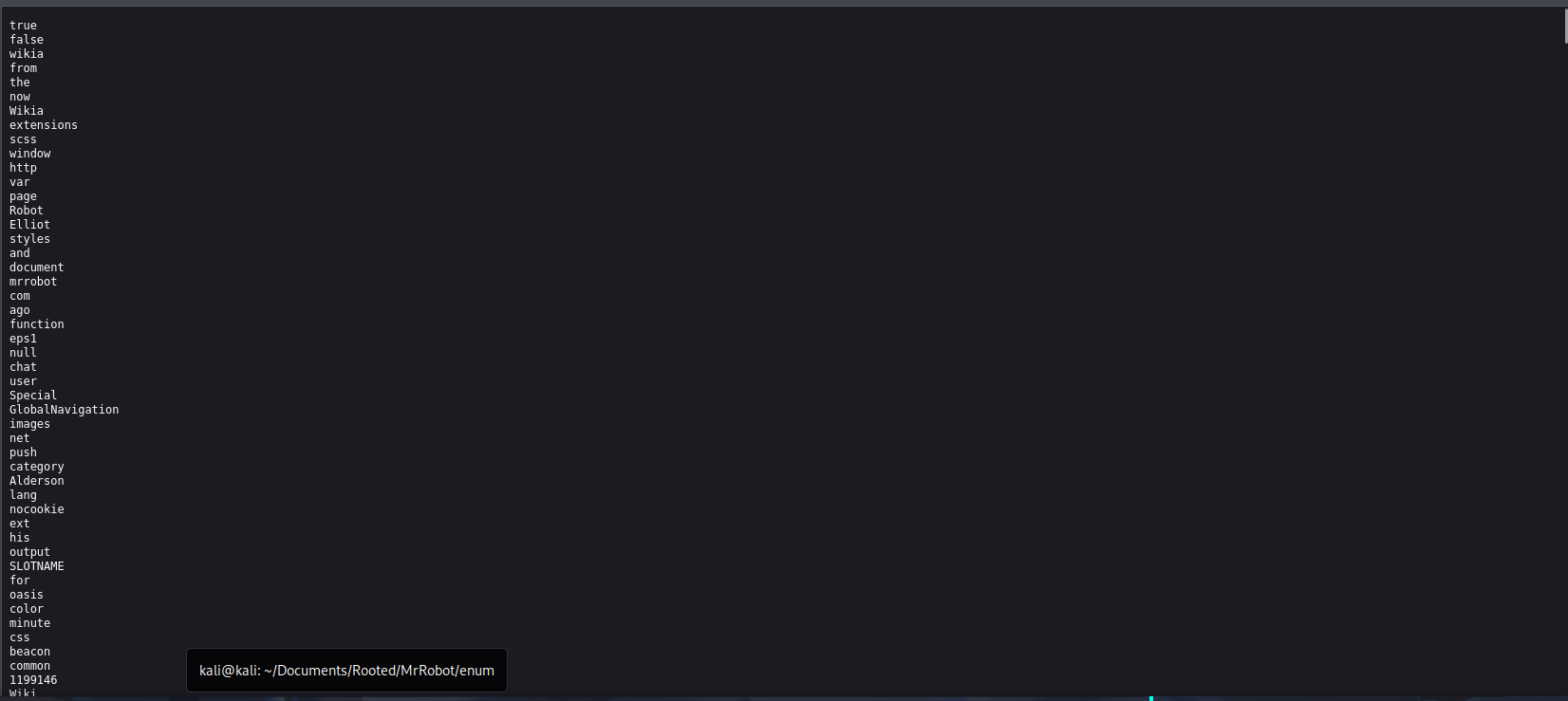
There are two files available in the robots.txt

1. Fsociety.dic
2. Key-1-of-3.txt

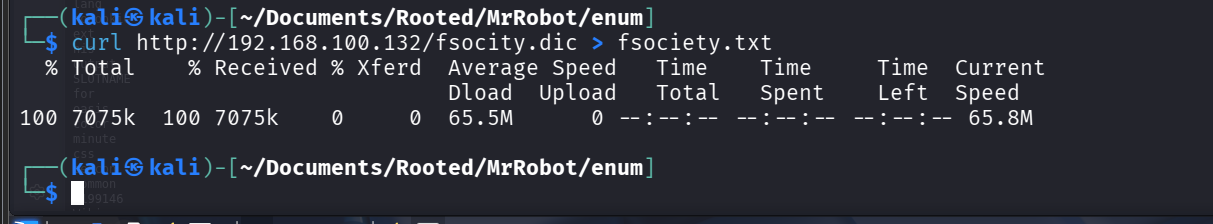
The second file must be one of the flags.



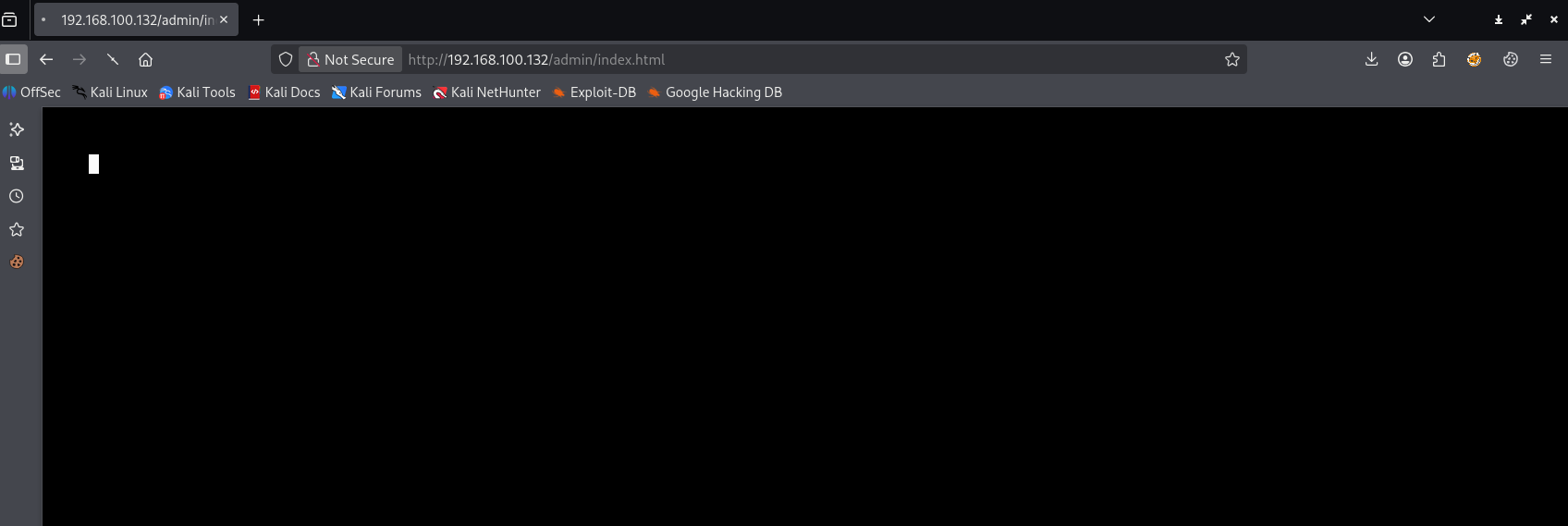
Yes, it’s one of the keys from the task.



The second file seems to be wordlist of some sort.

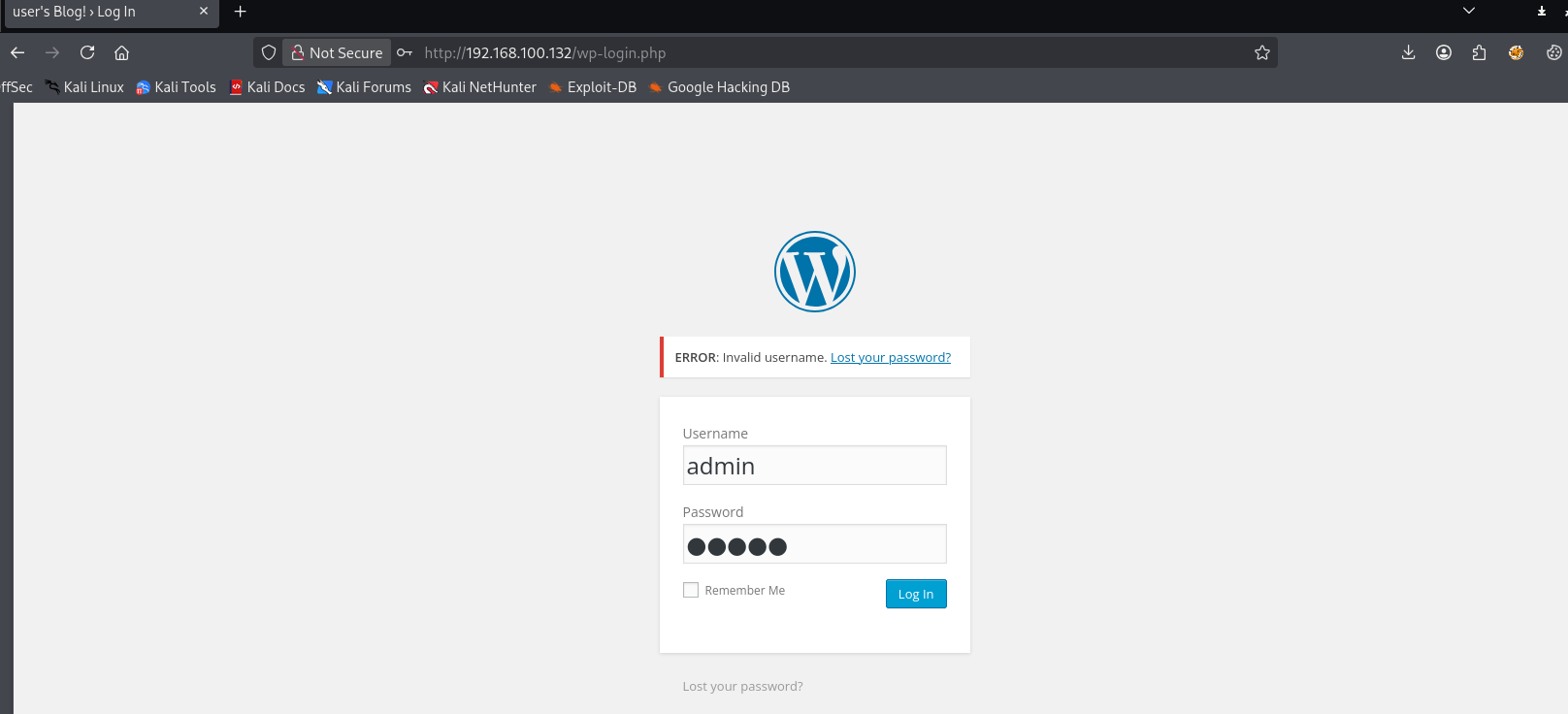


Downloading and keeping it for further testing. From the subdomain enumeration we also found out an admin page.



This admin page keeps redirecting. There is also WordPress admin page.

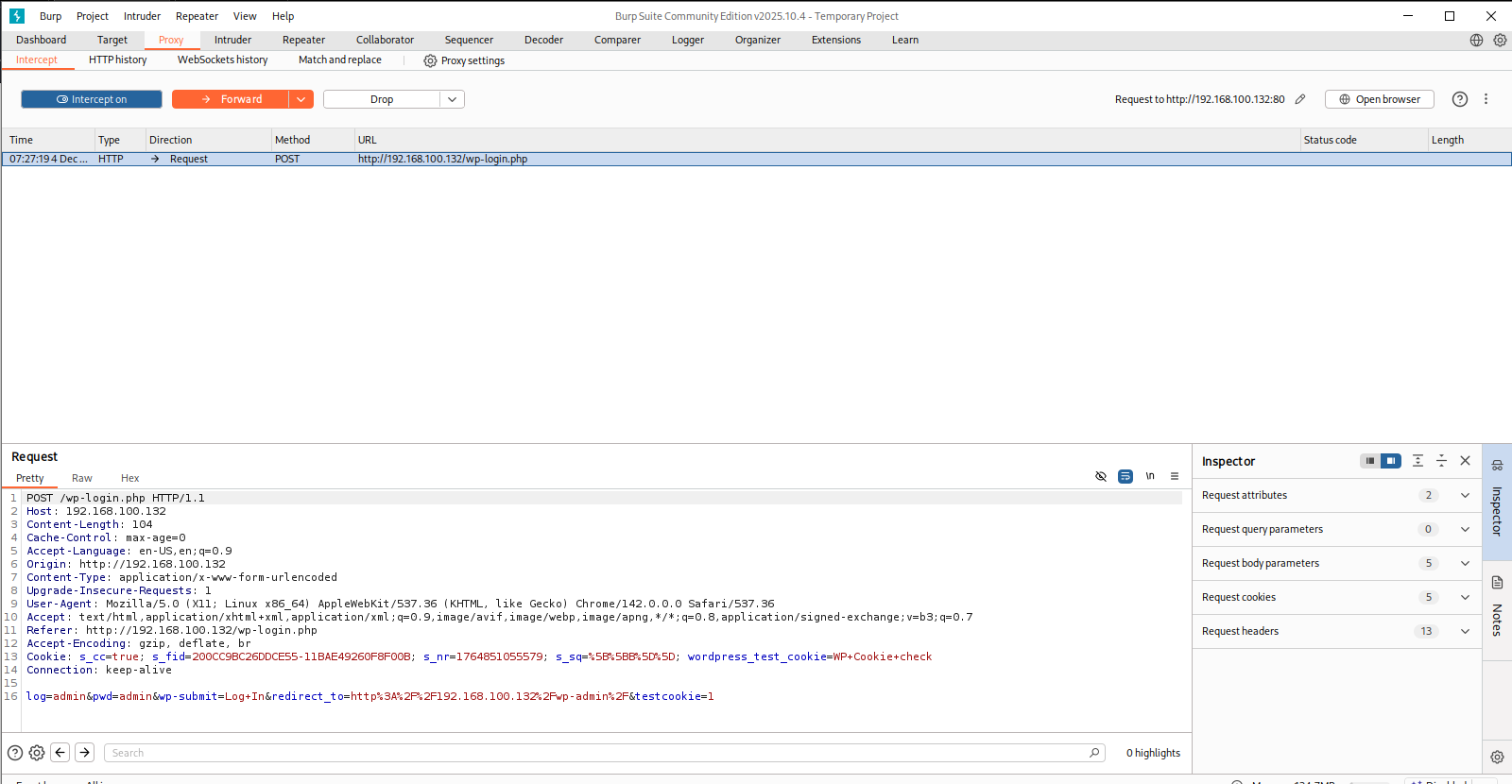




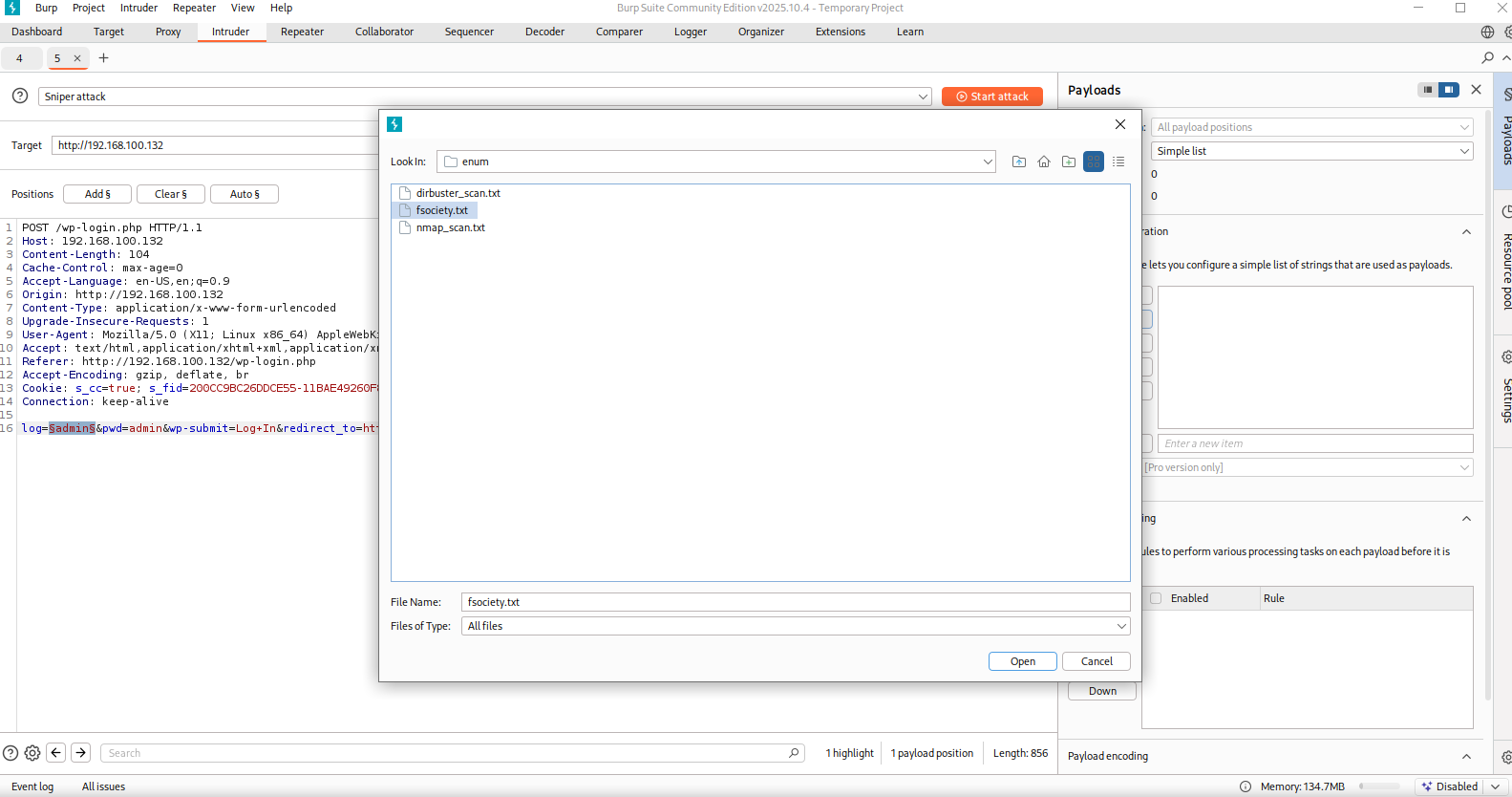
When we enter default credentials the site responds invalid username. By this we can determine that user enumeration might be possible.

1. **Using Burp suite for User Enumeration**

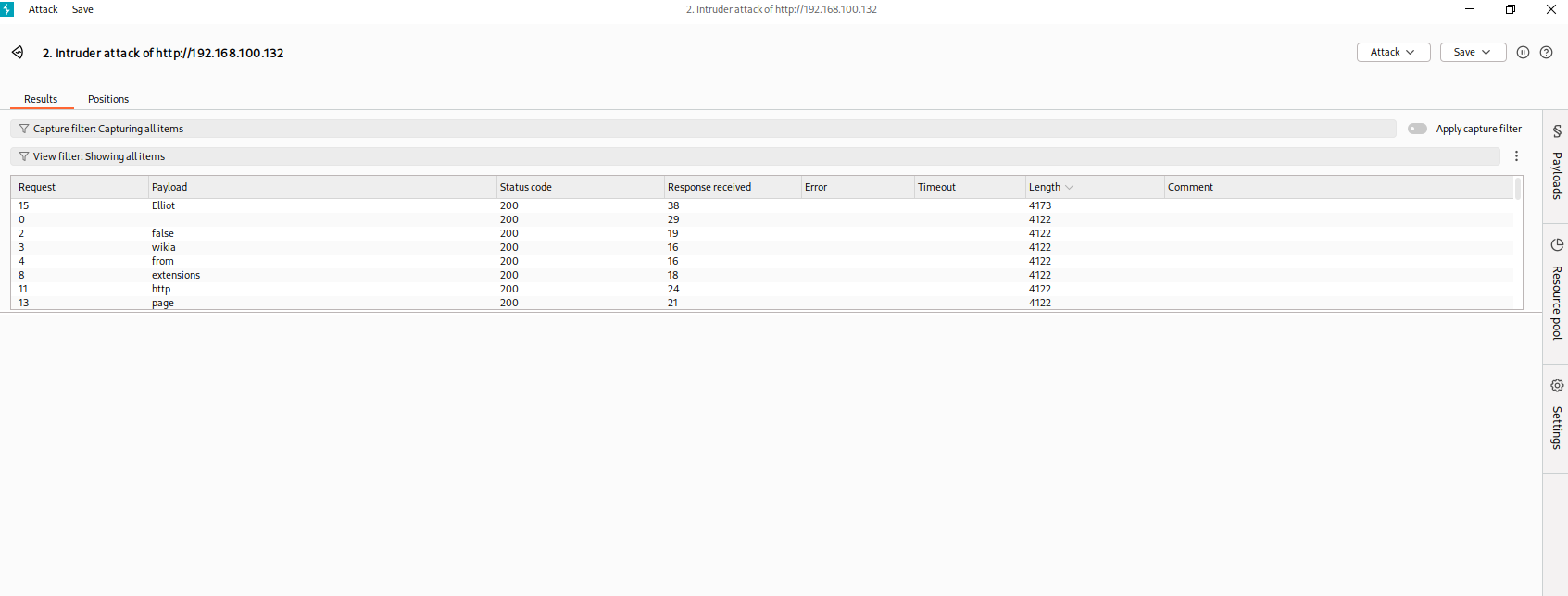
Burp suite Intruder can be used to perform brute force attacks on web applications.



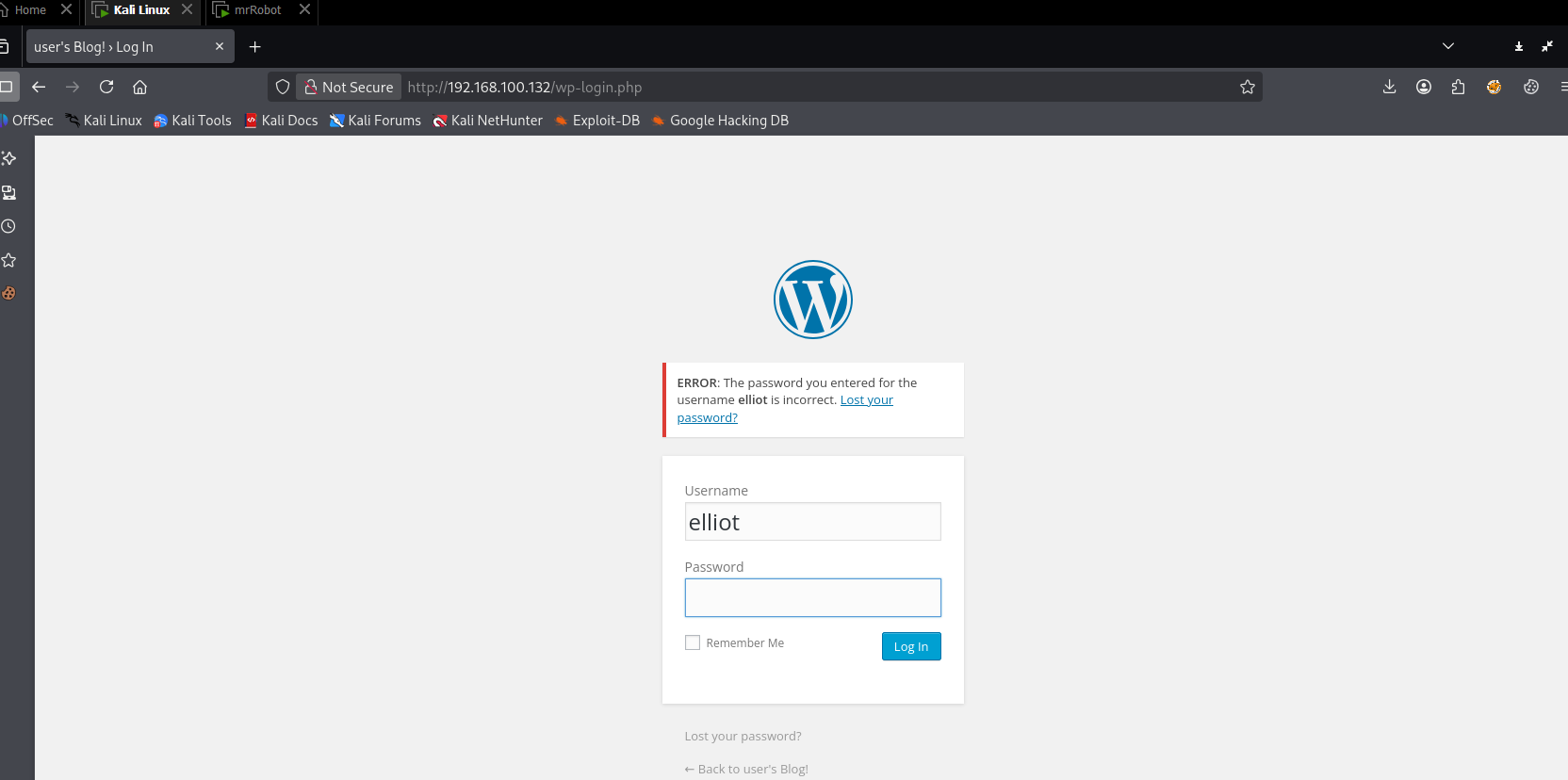
We are going to capture the login request in the burp suite browser and send it to intruder.



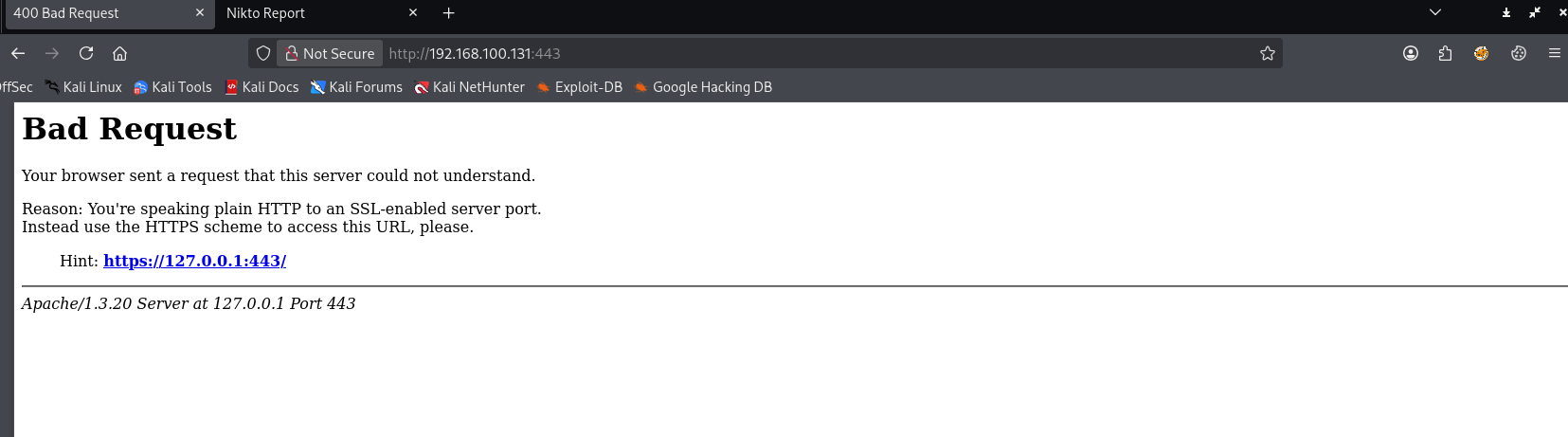
We are configuring to performing user enumeration using the wordlist we got from robots.txt.



If we filter the response by length, we can find out that one results in different length. We can check that username.

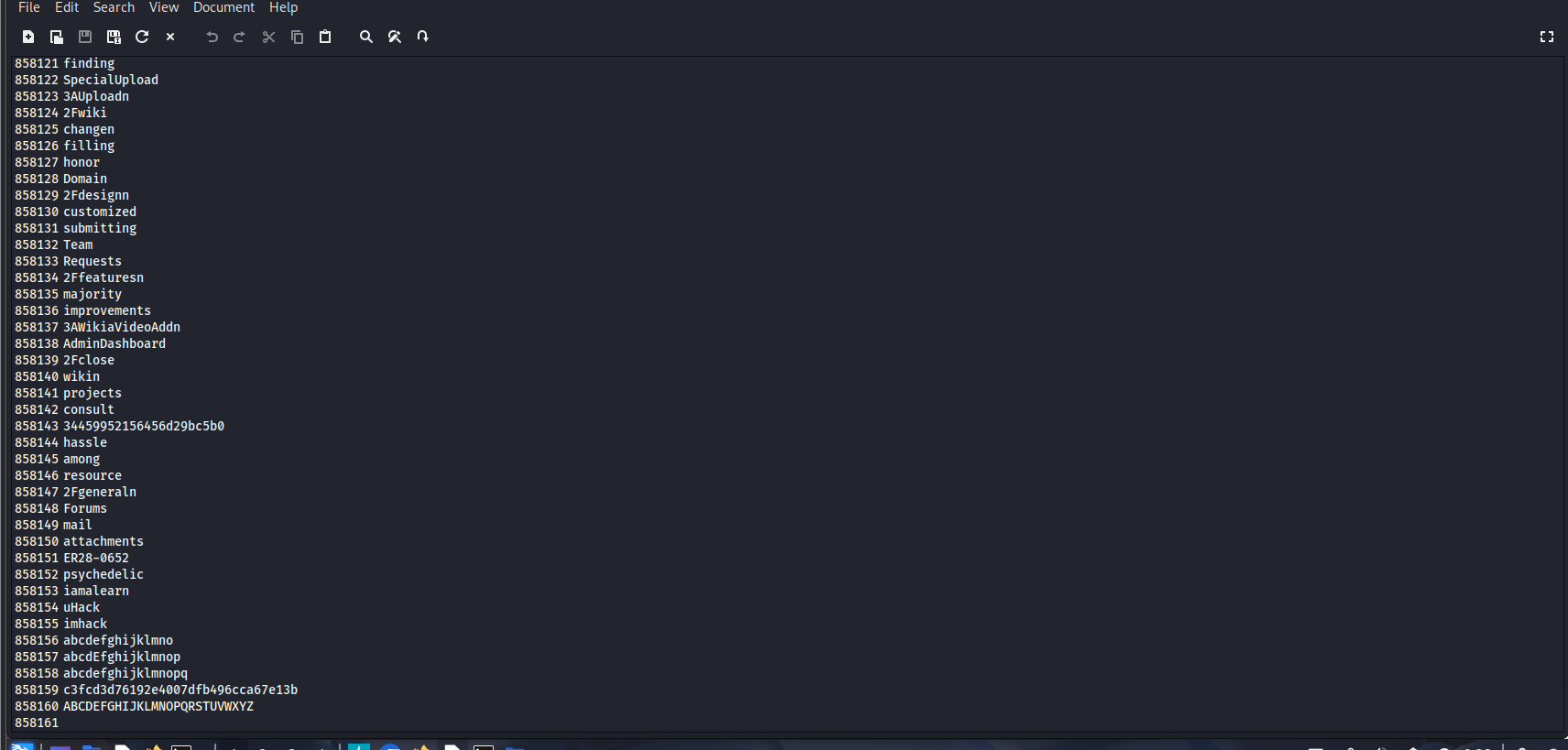


We were able to determine the user Elliot account is present.

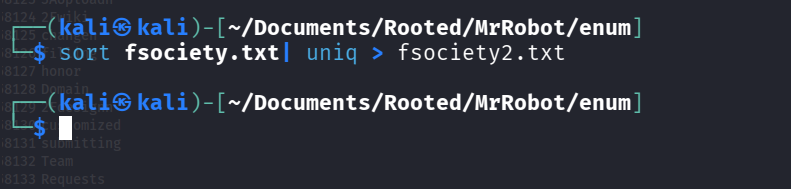


1. **Performing Dictionary attack using Hydra**

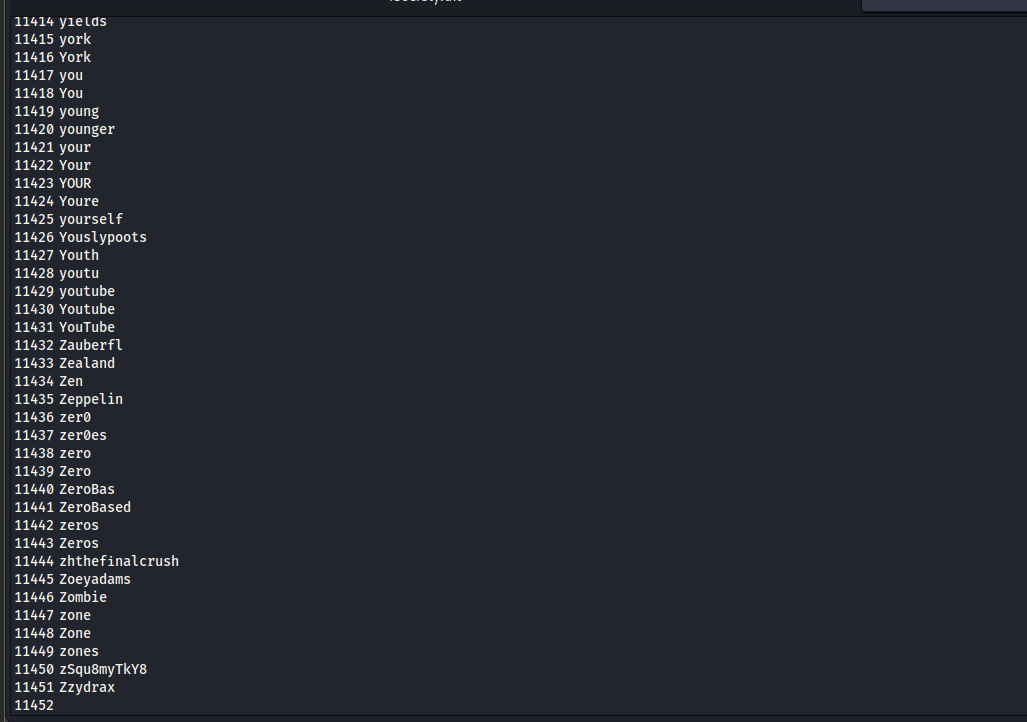
Hydra is a tool which helps us to find username and password by automating the dictionary attack. It provides several dictionary attacks on different services. Before Performing the attack, we need to configure the wordlist.



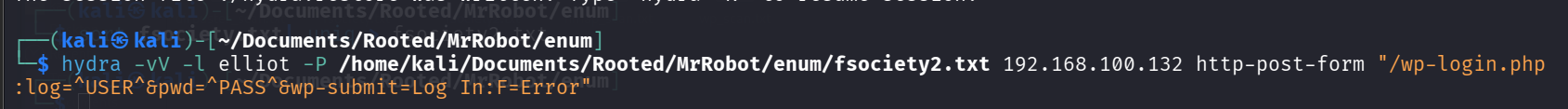
As we can see that this word list is way to big and it takes time to perform the dictionary attack. Let’s shorten this by removing all the duplicates present.



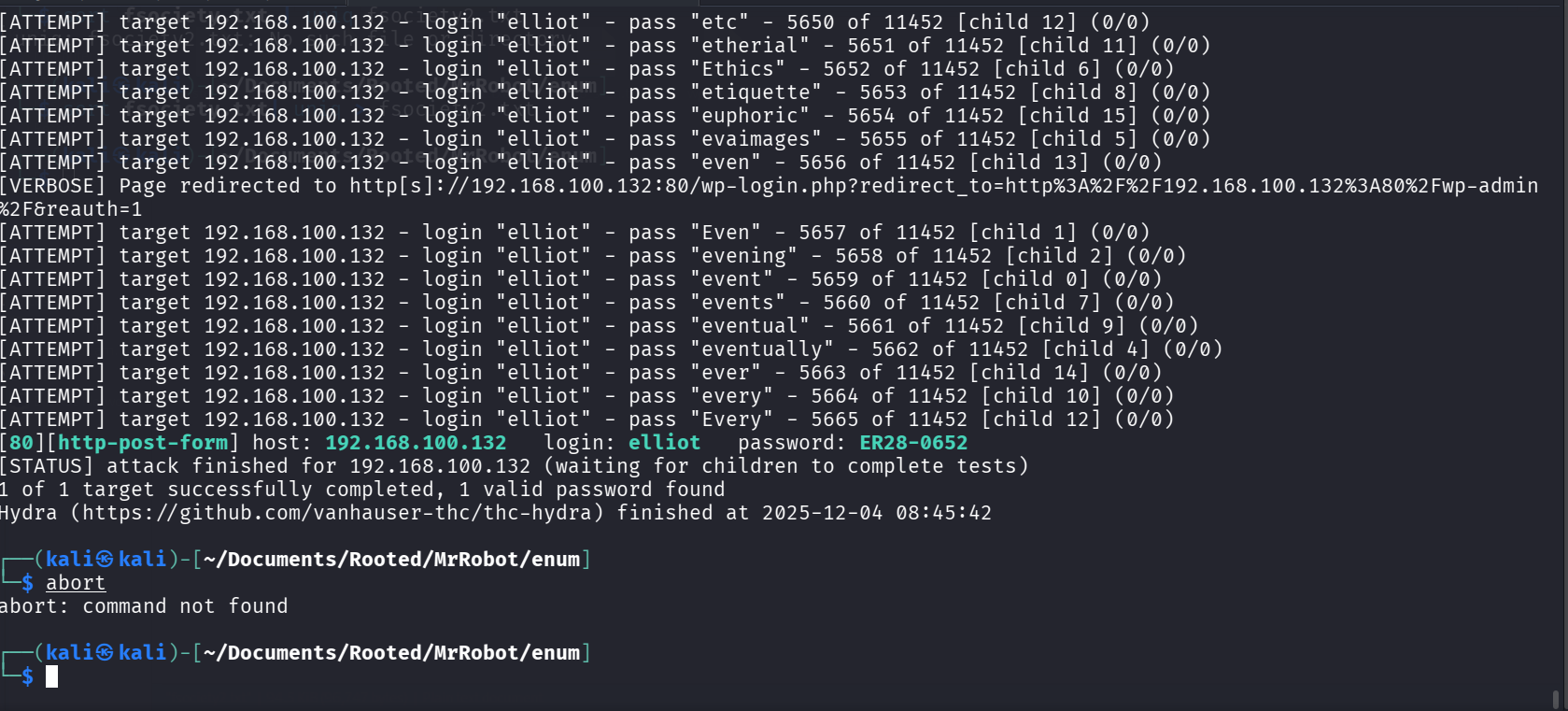
The above commands remove the duplicate words from the wordlist and creates a new wordlist that has no duplicates named fsociety2.txt



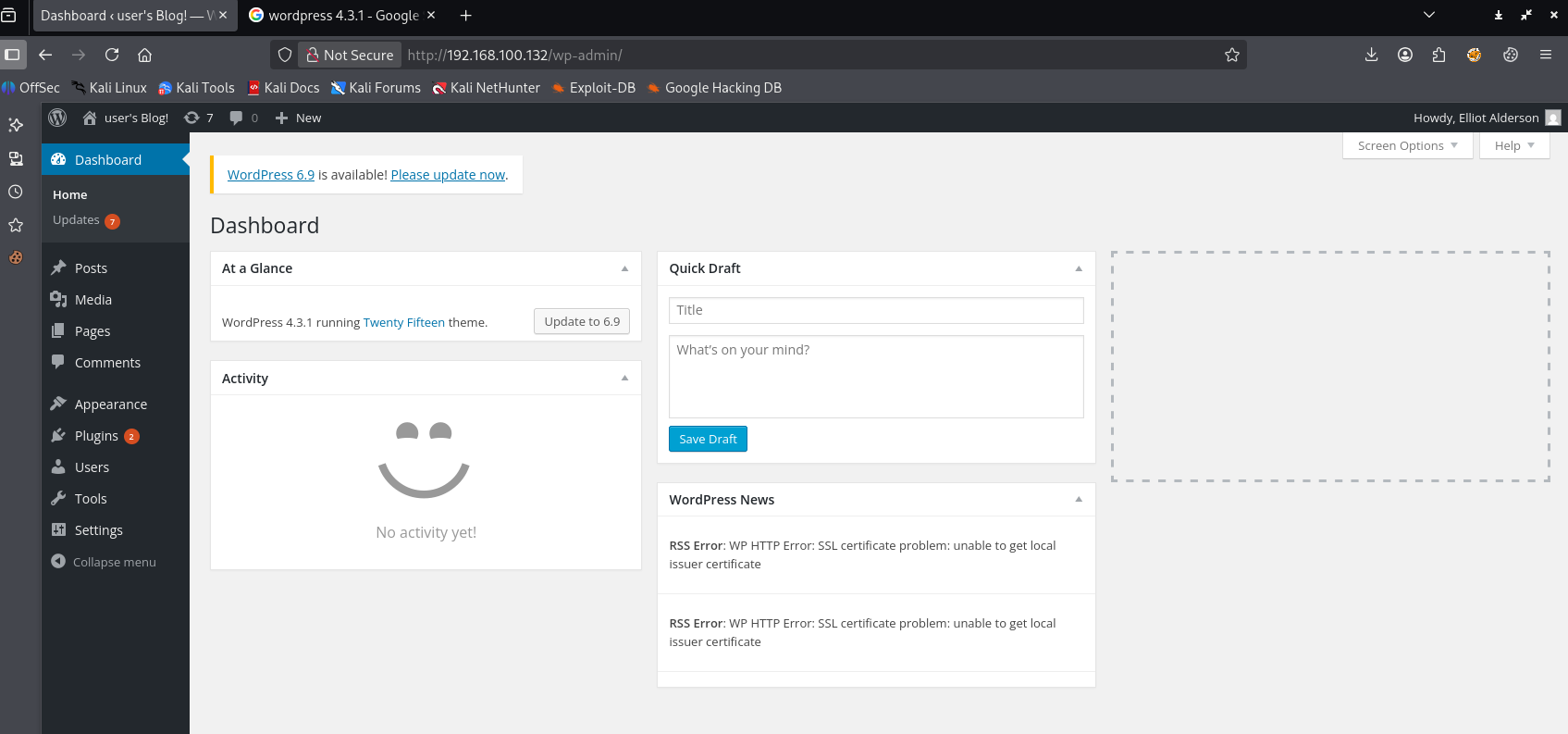
Now the new wordlist is significantly shorter we can perform Dictionary attack significantly faster.



Now we use the above command to perform Dictionary attack on the /wp-login.php page.



We were able to determine the password for Elliot i.e. **ER28-0652**



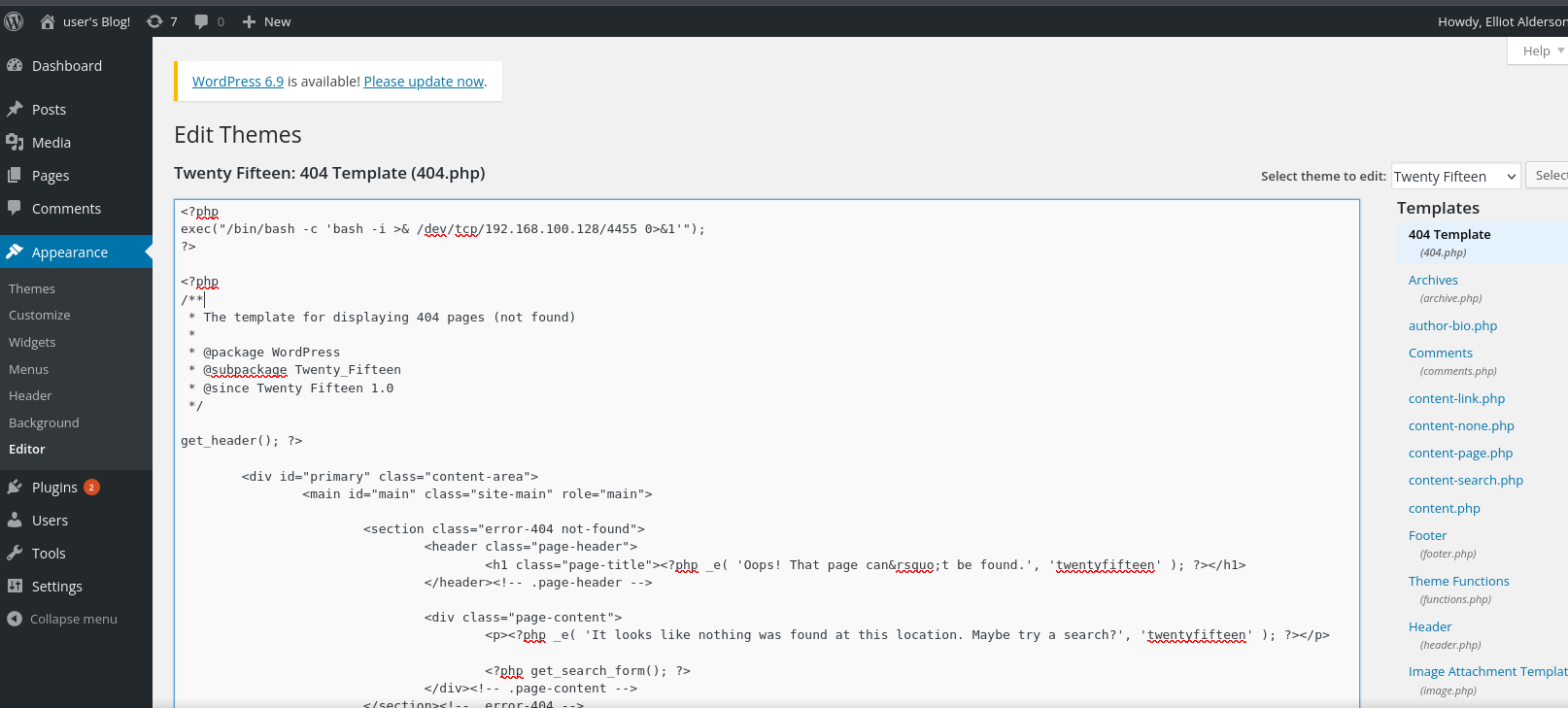
We were successfully able to login in to WordPress login page.

1. **Creating a Reverse shell using php code**

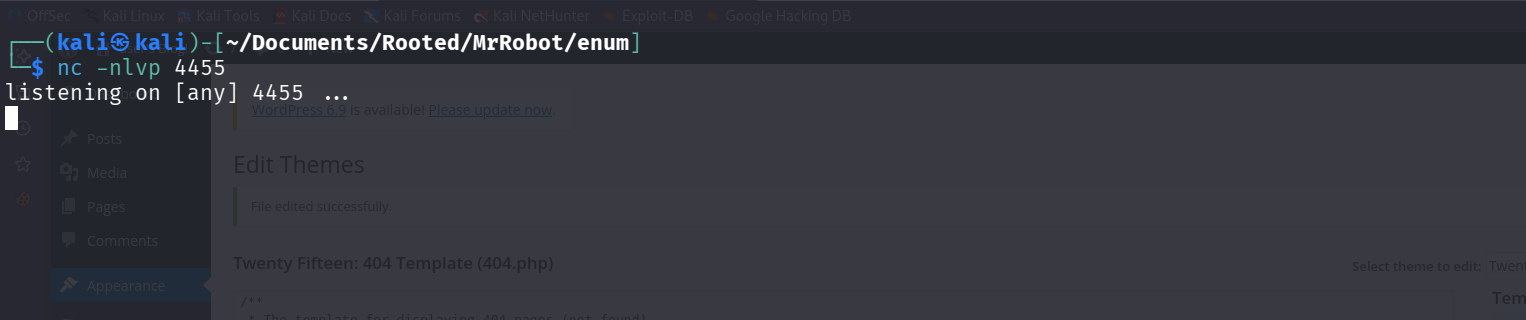
We are going to insert a php reverse shell payload in the 404-template using the WordPress editor tool.



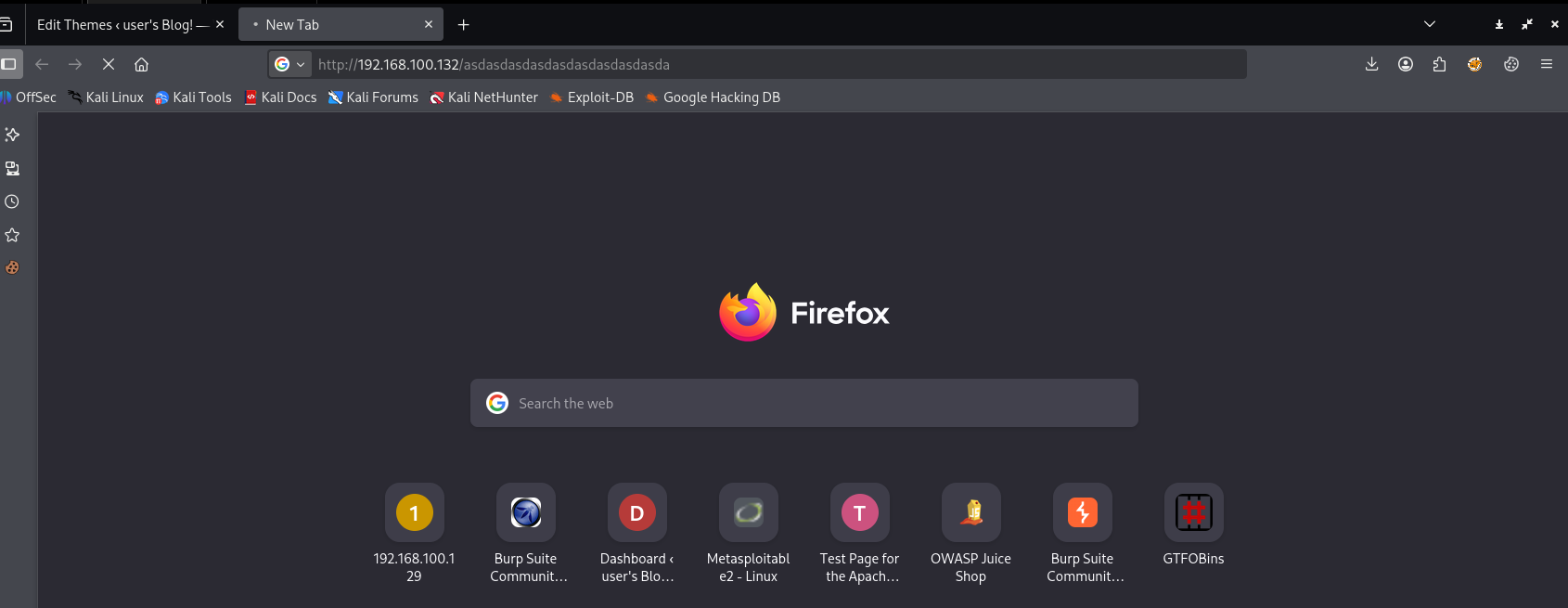
We are going to add the code at the begging of this webpage so it gets executed.

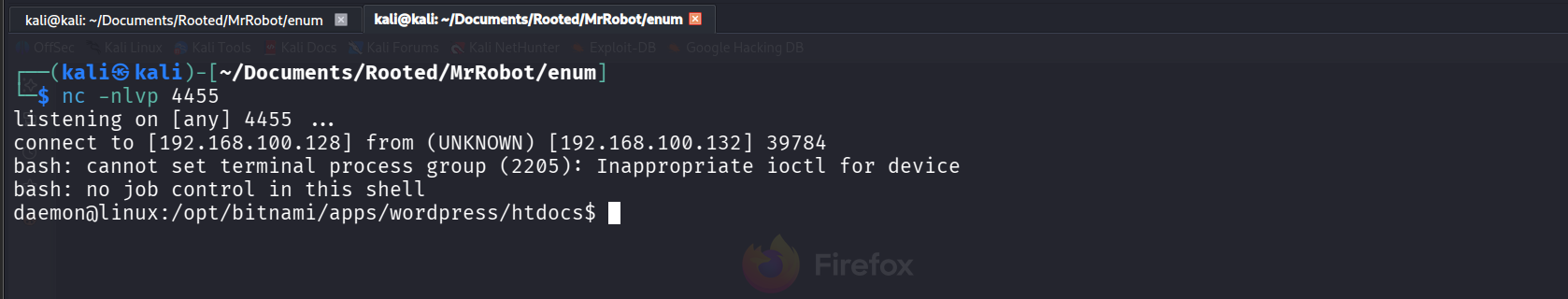


We will start the listener in our attacker machine

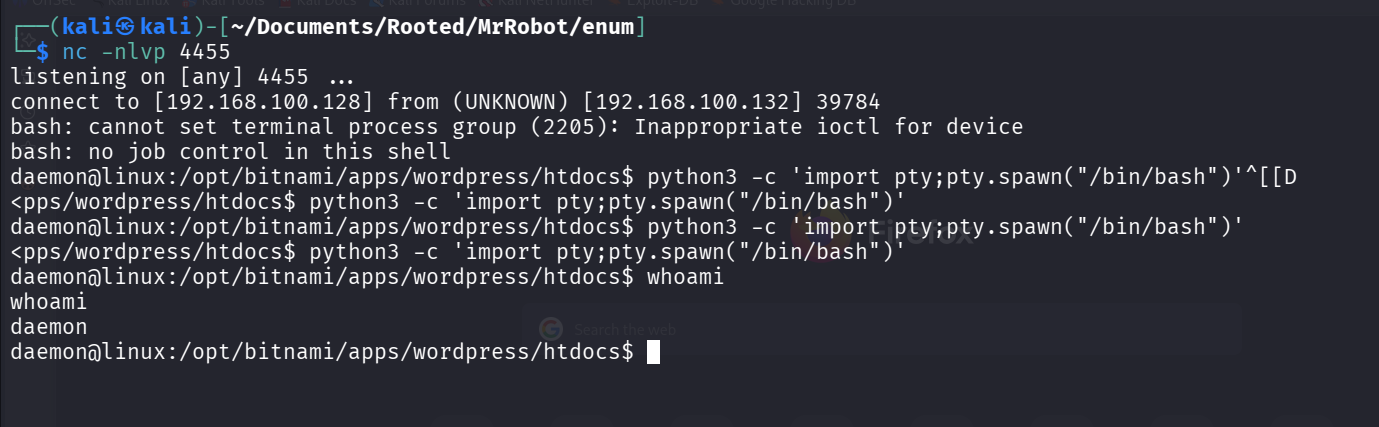


Then we will trigger this 404 page by going to any random link on the website.

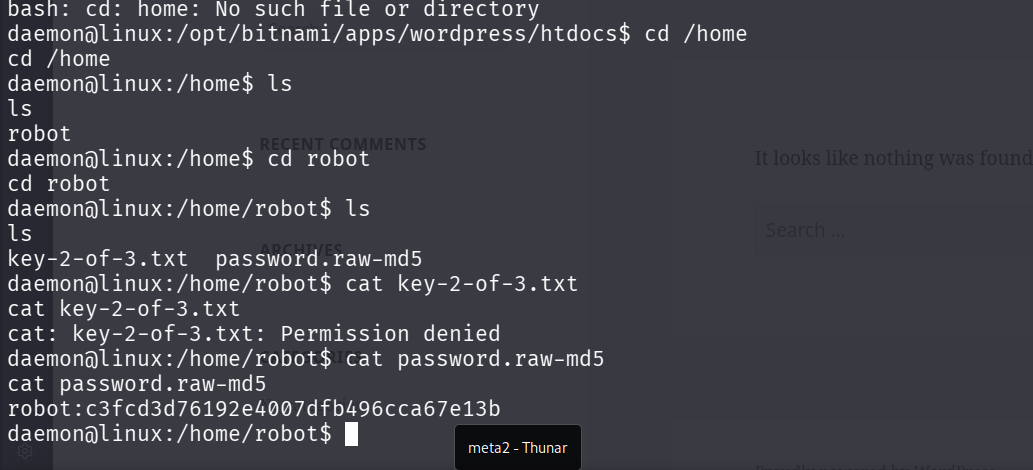




We were successful in getting the reverse shell.



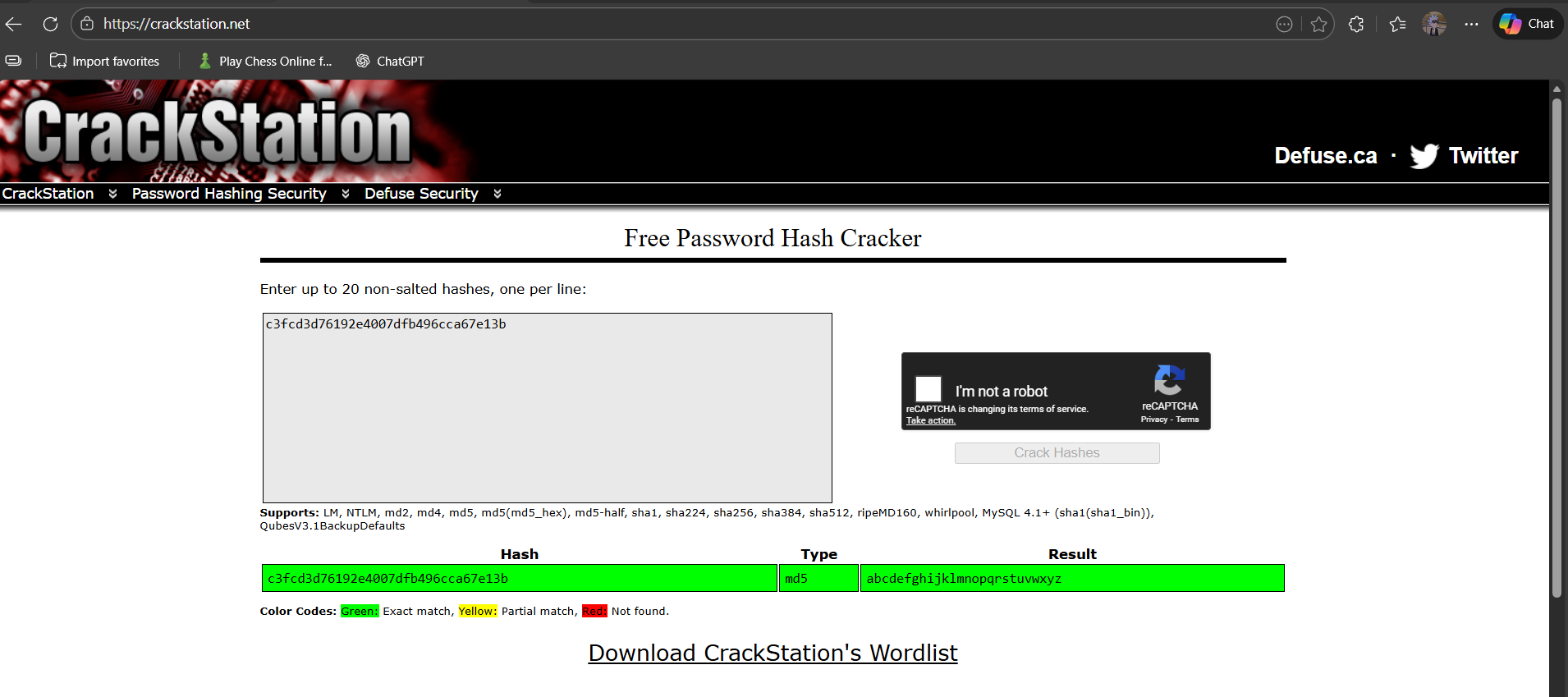
Upgrading the shell for better functional fully interactive shell.



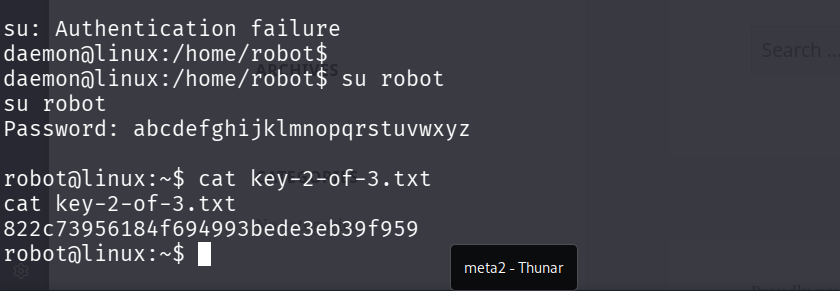
We were able to find the user flag i.e. key-2-of-3.txt but we don’t have appropriate permissions. So, we need to switch to robot user so that we can read this file.

1. **Using crack station for cracking hash**

From Previous snapshot we were able to find the hash of the password so we are going to attempt to crack it using the crack station website.

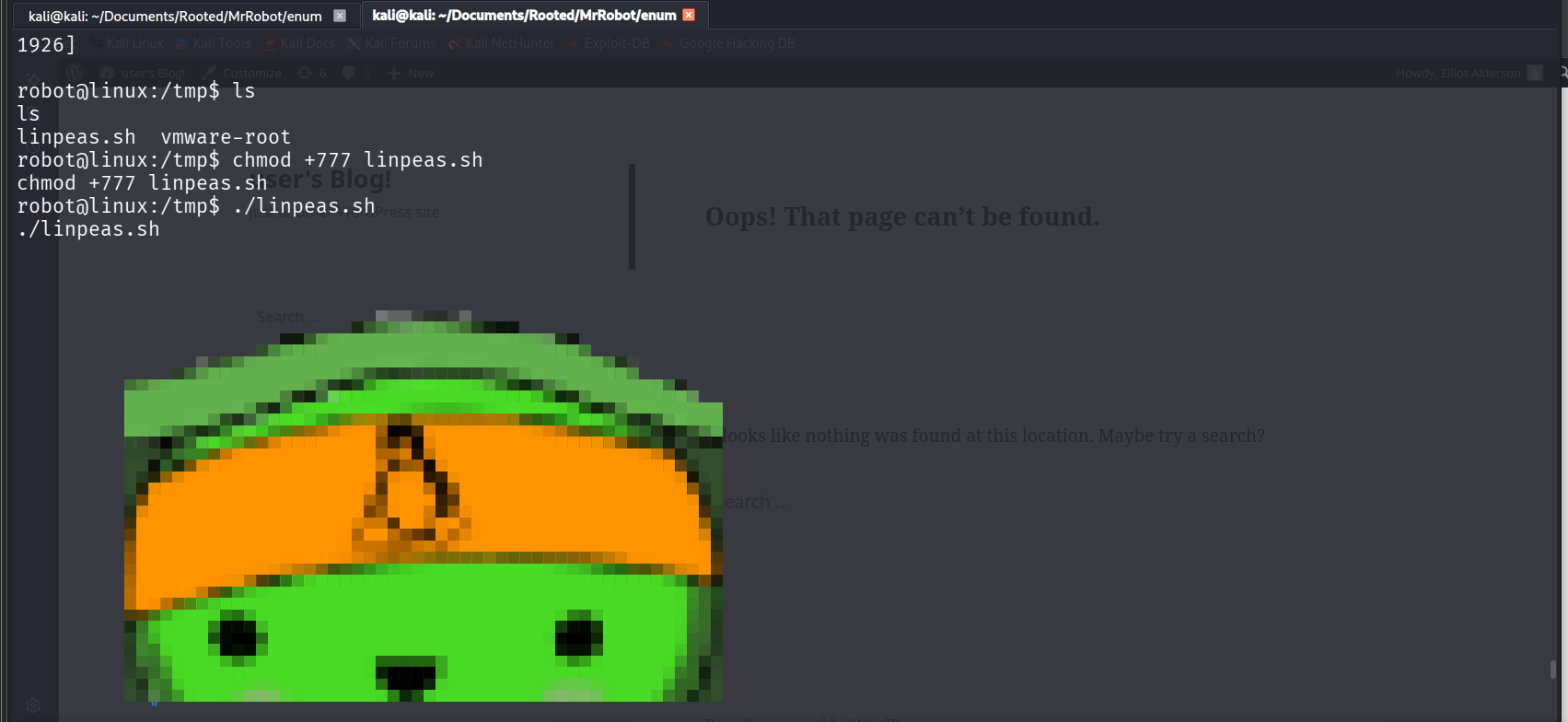


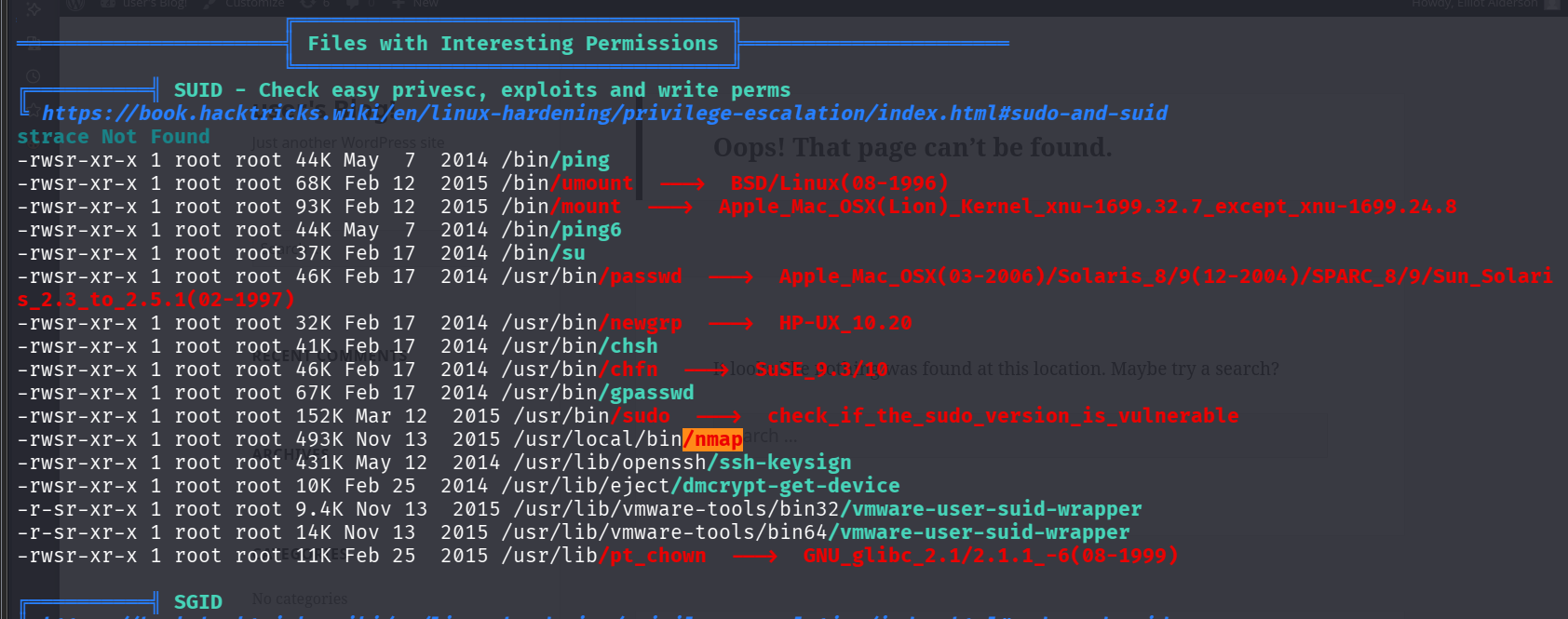
We were able to find the password ‘**abcdefghijklmnopqrstuvwxyz’**



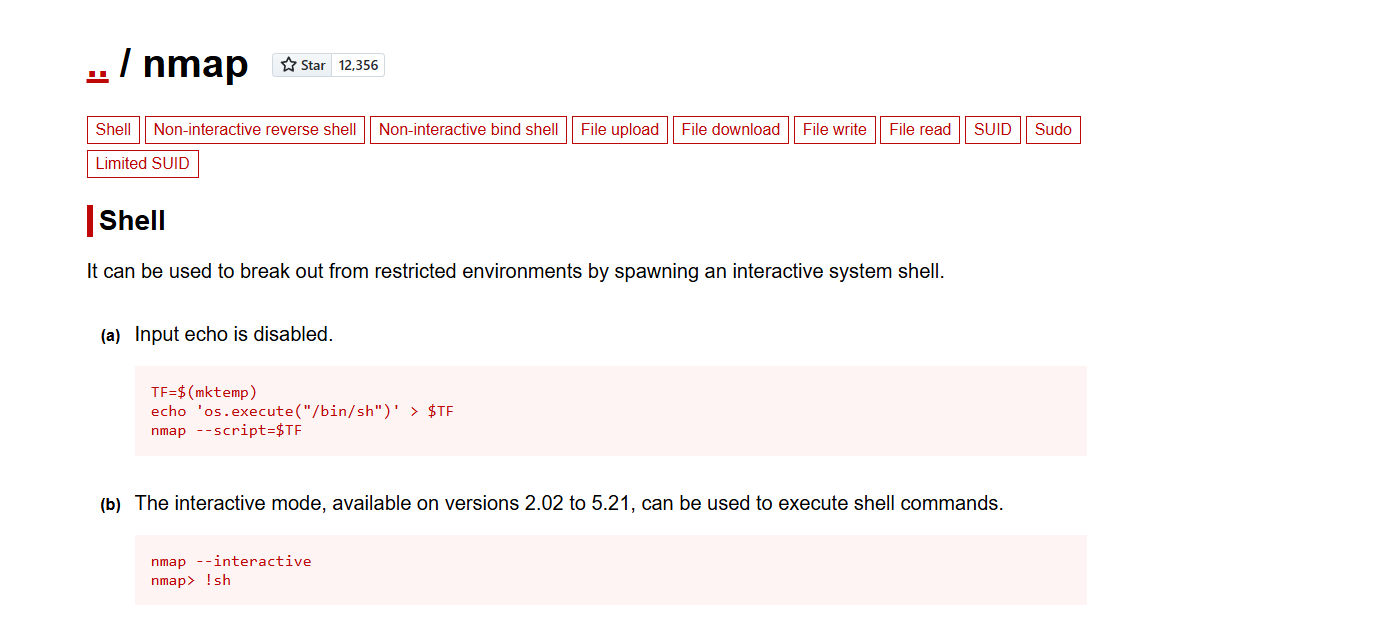
We got the flag 2.

1. **Escalating Privileges using Linpeas script.**

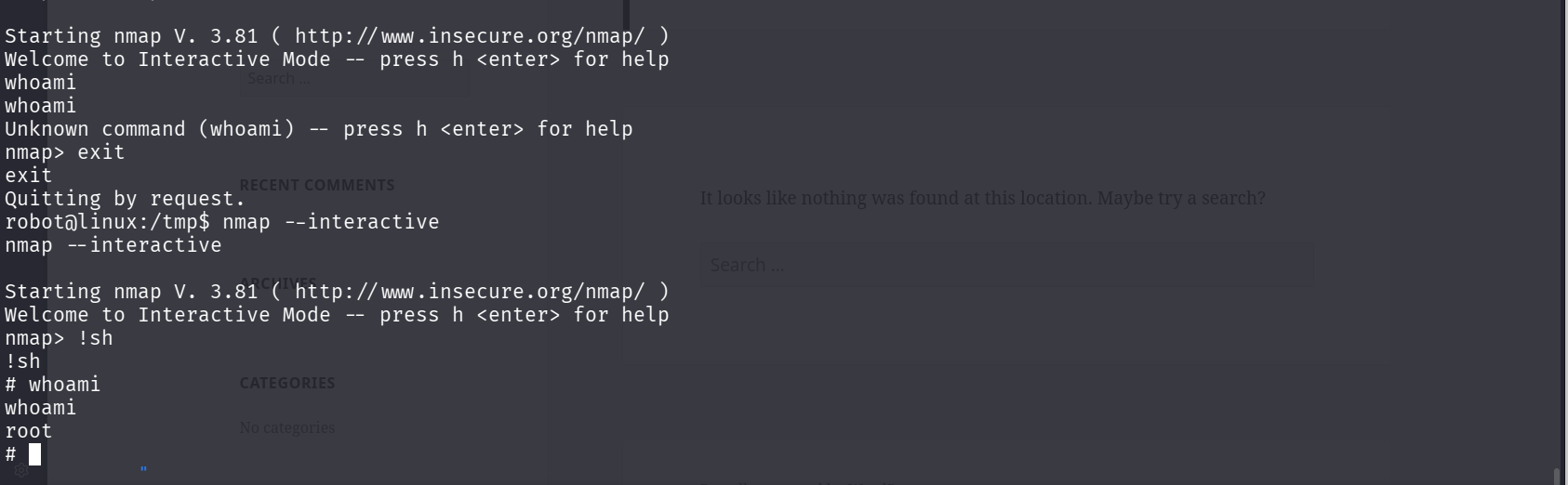




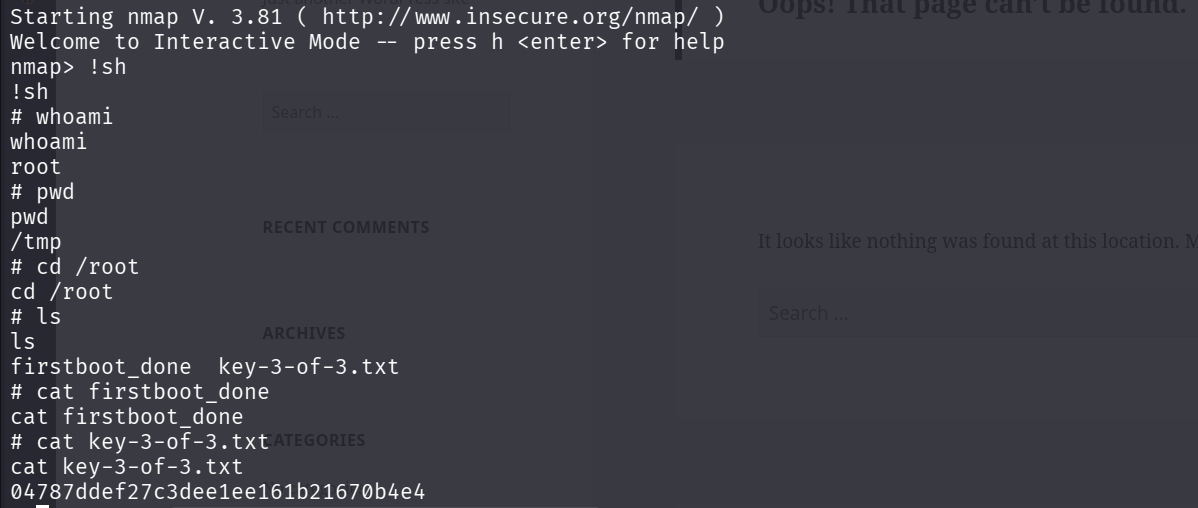
Nmap is being run with escalated privileges as root.



We can use GTFO bins to get snippets for privilege escalation.



We were successfully able to escalate our privileges to root.



We got the final Flag i.e. the root flag.

**Exploit chain Log:**

The Following table shows the chronological order of the exploitation process followed through out pentesting.

**Exploit Logs**

****

**Custom PoC summary:**

I found Publicly available PHP reverse shell proof of concept to suit the target Virtual machine i.e. Mr. Robot. I changed the attacker Ip address and port cleaned up for correct execution and inserted it in the WordPress 404 template. This custom template allows remote code execution for Mr. Robot.

**Findings Summary**

* WordPress authentication allows brute-force attacks.
* WordPress theme editor enables PHP code execution
* Nmap is being run with root privileges allowing root access.
* Unauthorized files accessible without proper permissions.

**Remediation Recommendations**

* Promote strong password policy and enable rate-limiting.
* Disable file editing in WordPress editor.
* Remove SUID bit from unnecessary binaries (nmap)
* Download and update WordPress Plugins Regularly.
* Perform system hardening by restring Sudo usage.
* Usage of Web application firewall is recommended to block malicious requests.