**Intern Intelligence**

**Report**

**Penetration Testing of “SMOL CTF” in TryHackMe**

**15.03.2025**

**Cavanshir Sheyxov**

**Contents**

1.Reconnaissance

2.Enumeration

3.Exploiting

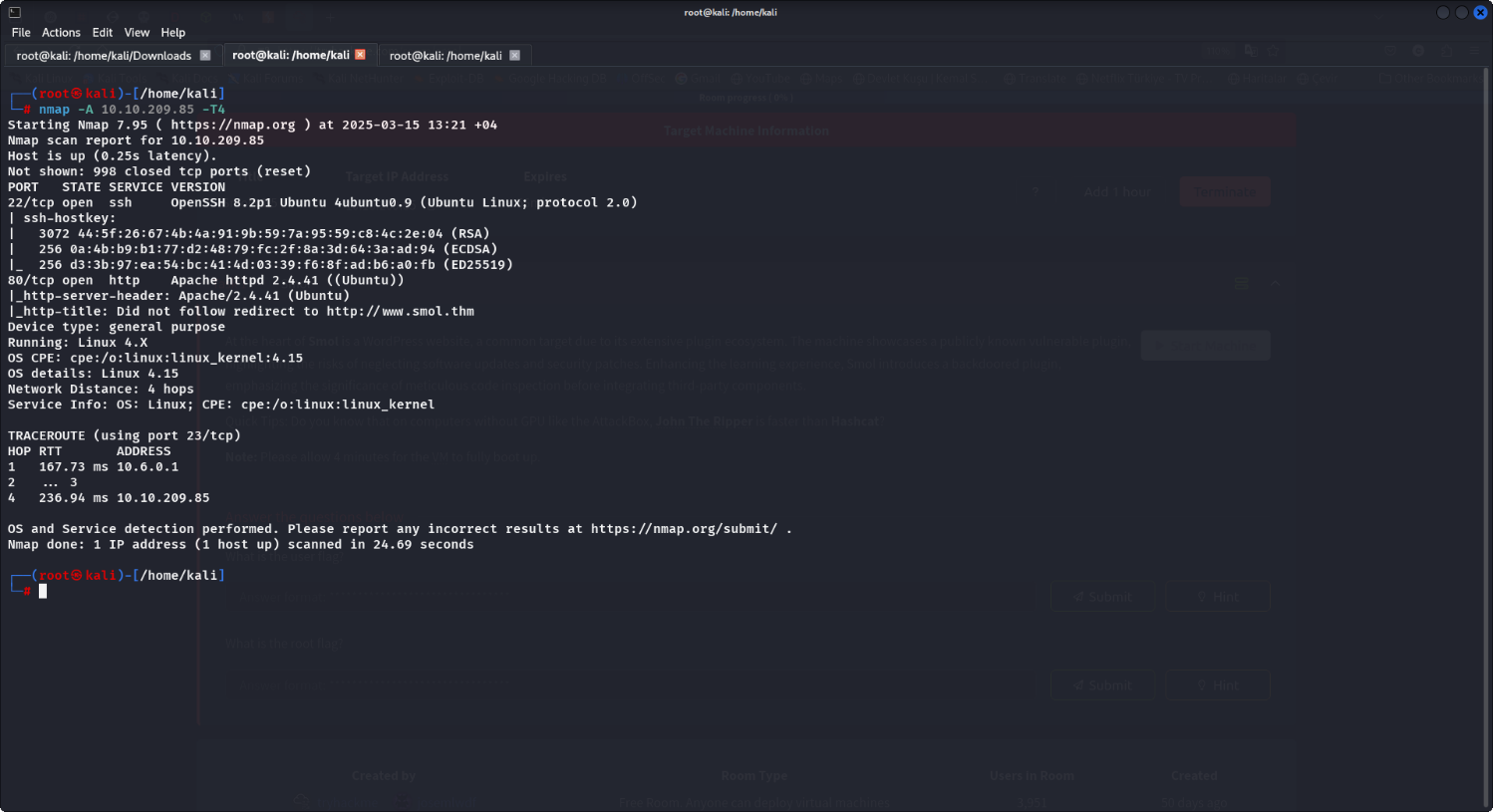
4.Privilege Escalation

**This penetration test evaluated the client's virtual environment using a black box methodology. The assessment revealed vulnerabilities in external, web application, and internal components, including outdated software and critical injection flaws. Misconfigurations in the internal network allowed lateral movement and privilege escalation, resulting in the capture of both User.txt and Root.txt flags. The findings underscore the urgent need for immediate remediation measures and regular security assessments before production.**

**1.Reconnaissance**

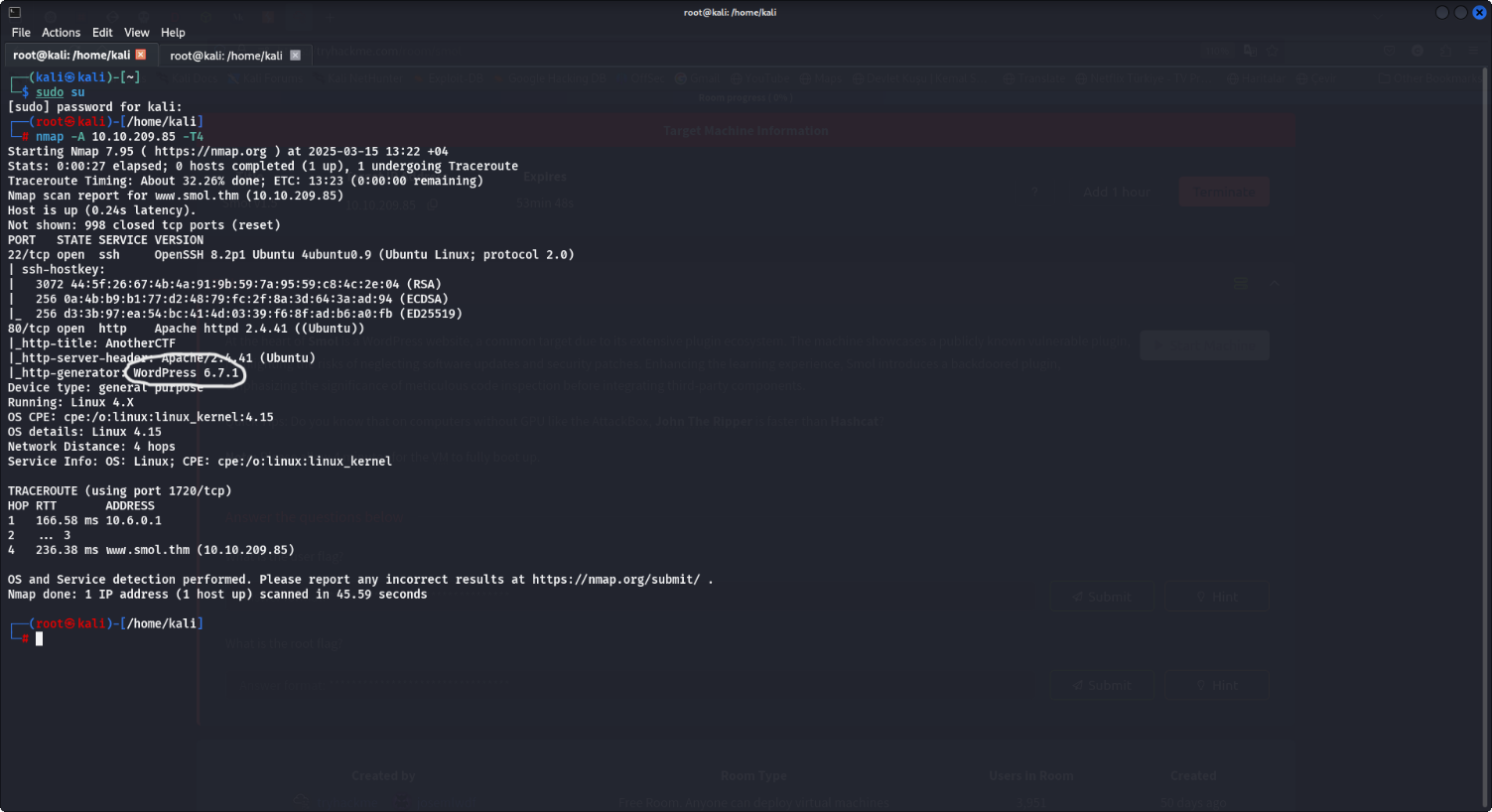
1. **The target is a WordPress website with a vulnerable plugin.**
2. **There is a backdoored plugin installed.**
3. **Hash cracking will be required (hinted by the note about John The Ripper).**

I began by adding www.smol.thm to /etc/hosts and scanning the target for open ports and services with **nmap**



****

**2. WordPress Plugin Enumeration**

****

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

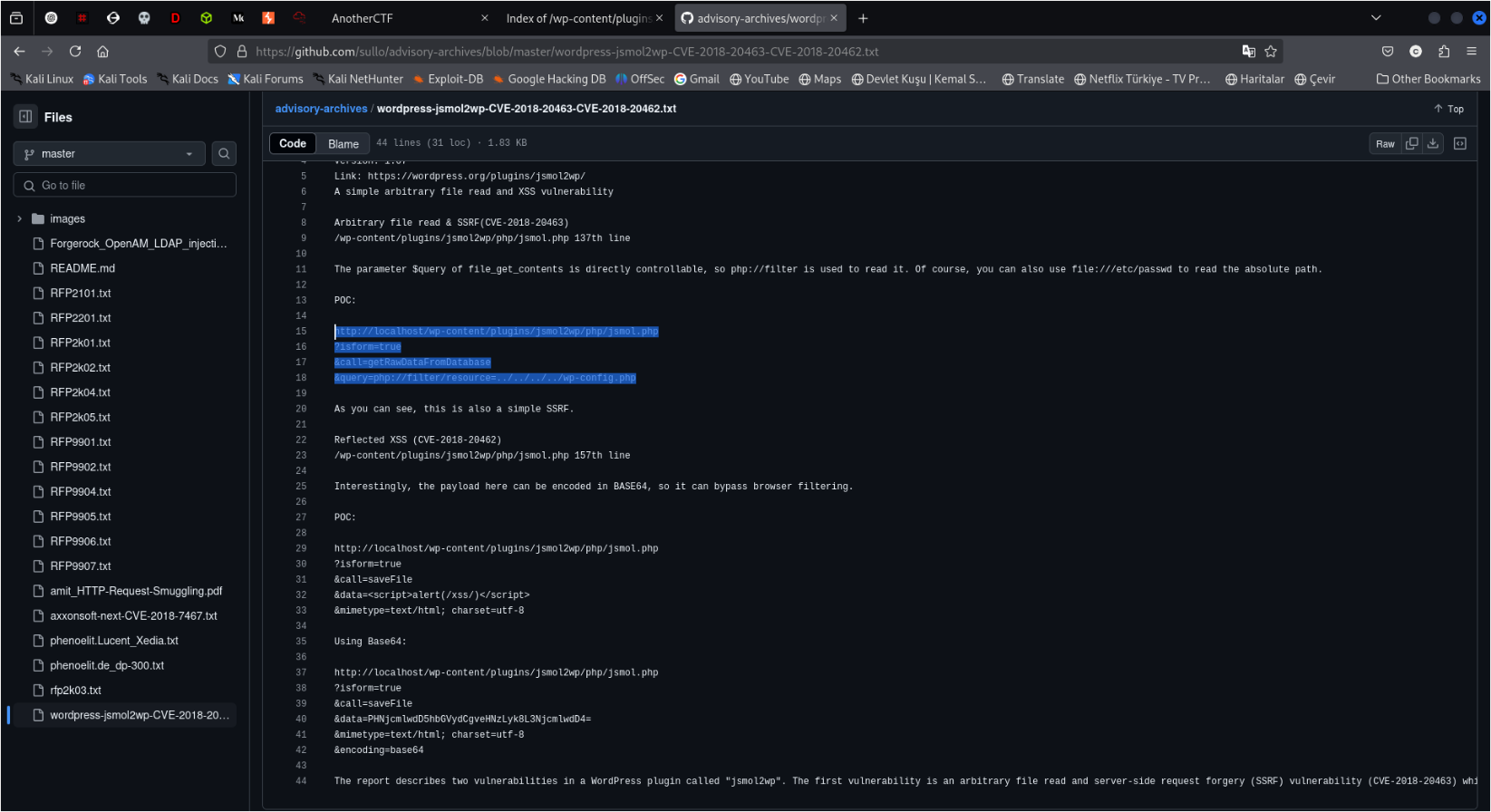
Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

Knowing the target is a WordPress site with a vulnerable plugin, I used wpscan to enumerate plugins:

metin, ekran görüntüsü, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

The scan identified the jsmol2wp plugin. A quick search led me to the following GitHub repository documenting its vulnerabilities:

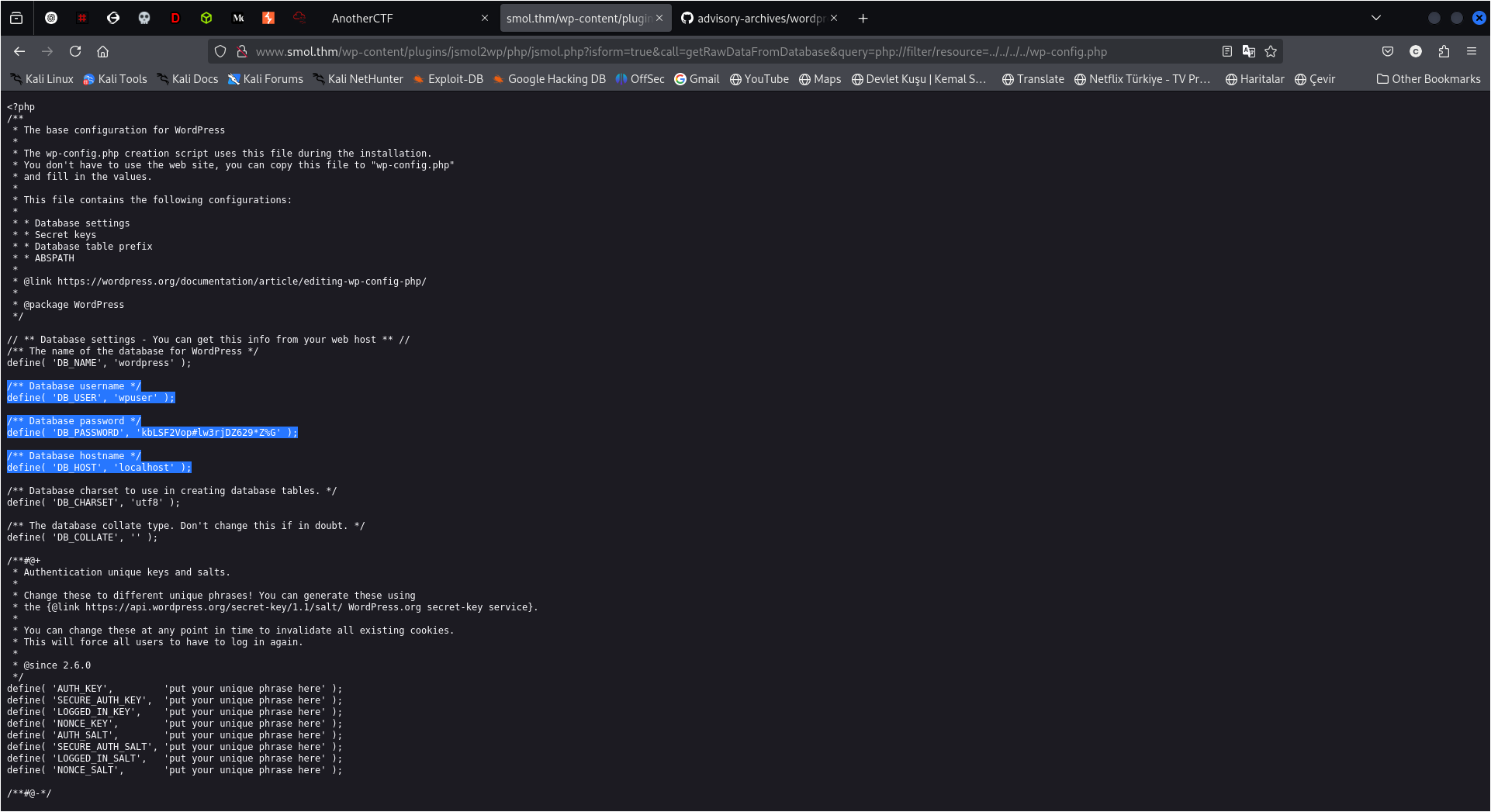


According to the repository, jsmol2wp is vulnerable to **LFI**, **XSS**, and **SSRF**. The **LFI (Local File Inclusion)** vulnerability appeared the most useful for our purposes.

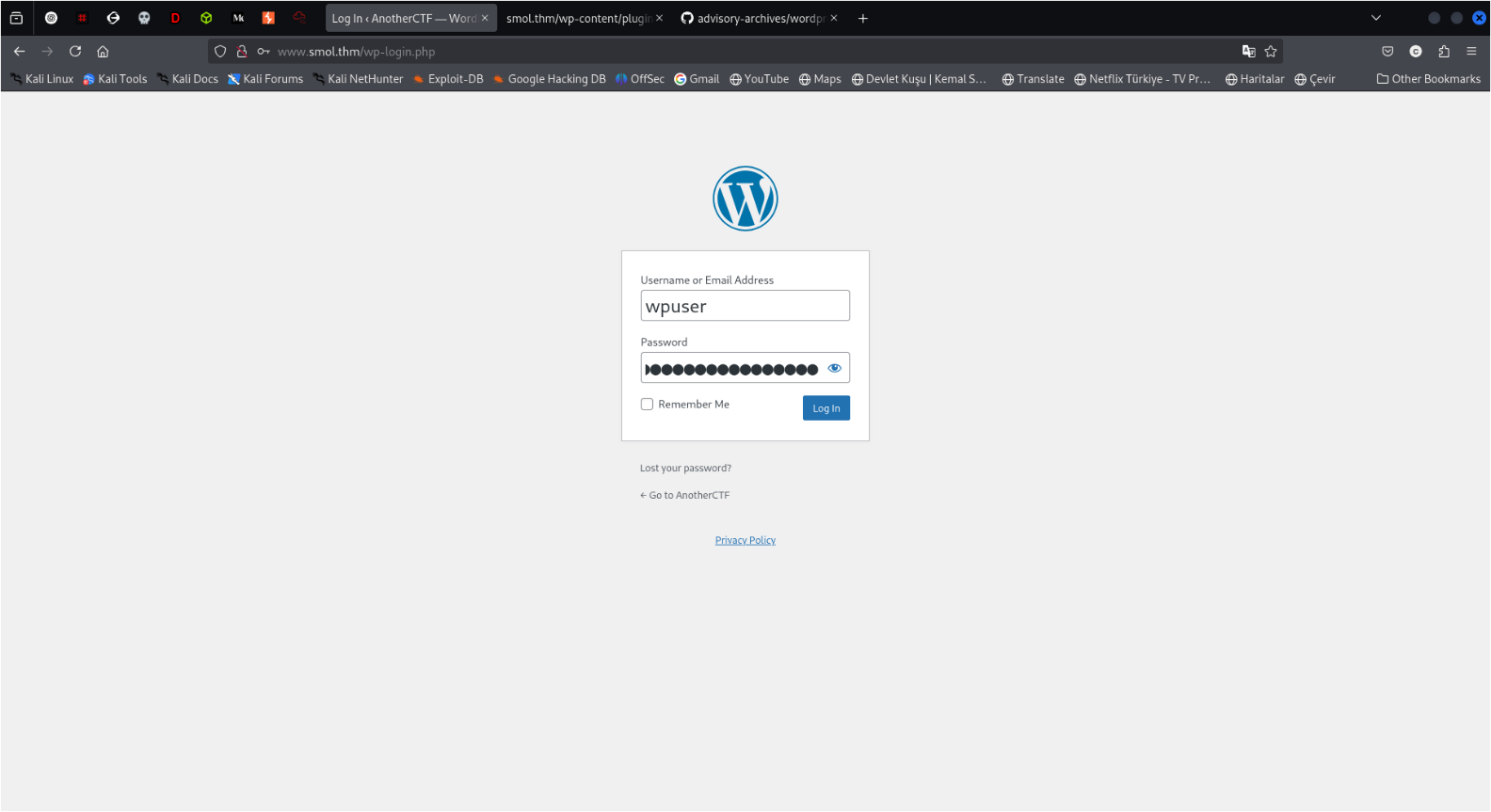
**3. Exploiting LFI**

**Using the LFI vulnerability, I accessed the wp-config.php file with the following URL:**

**http://www.smol.thm/wp-content/plugins/jsmol2wp/php/jsmol.php?isform=true&call=getRawDataFromDatabase&query=php://filter/resource=../../../../wp-config.php**

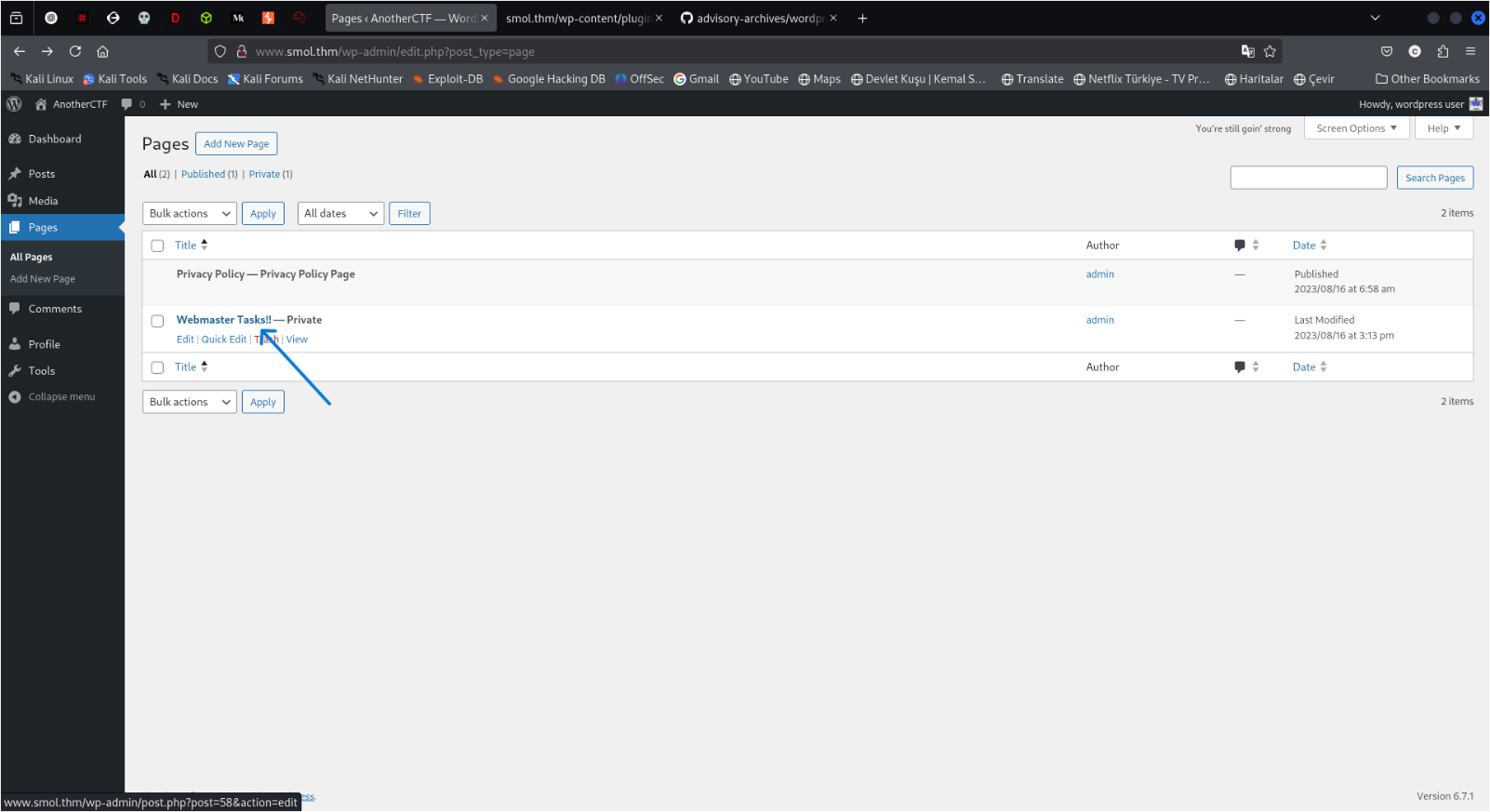


The credentials in the wp-config.php file allowed me to log in to the WordPress admin dashboard at:



**4. Discovering Vulnerable Plugins**

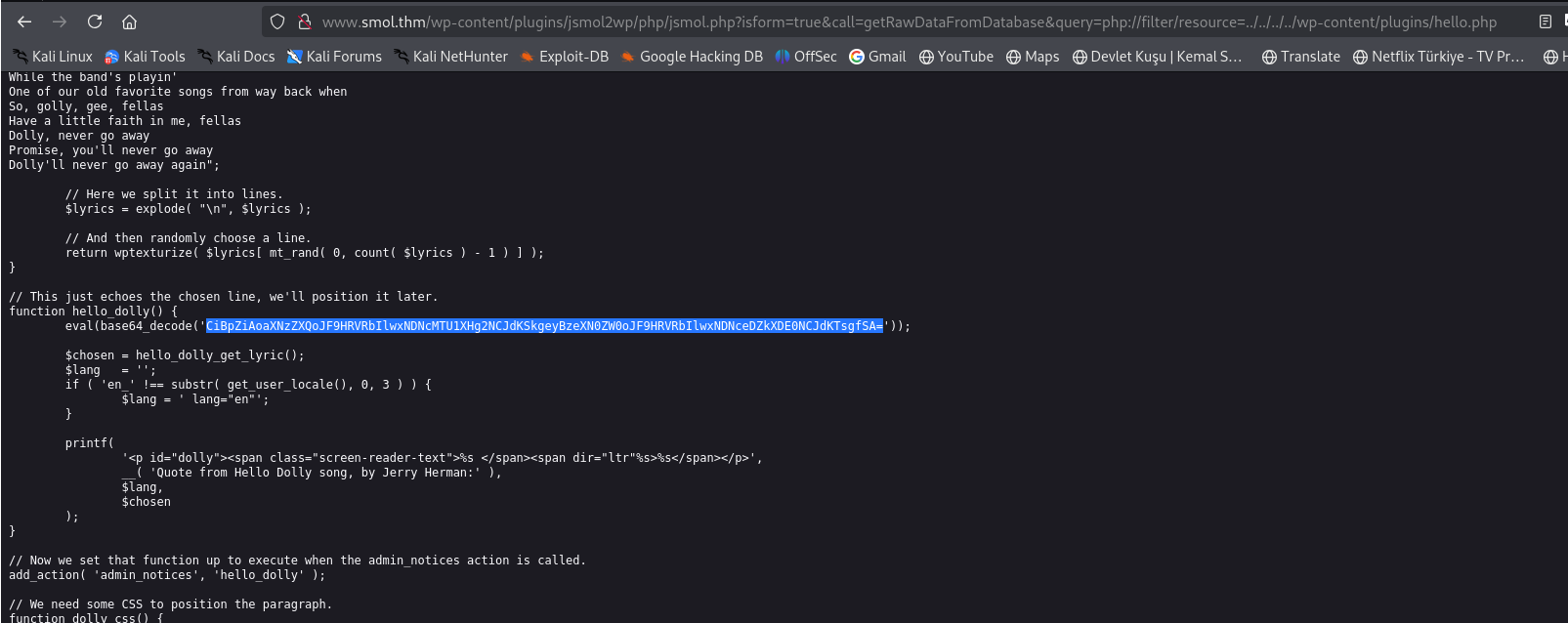
In the WordPress dashboard, I found a private page titled “Webmaster Tasks!!”.



The page revealed the presence of the **Hello Dolly** plugin, which is vulnerable. However, since the user didn’t have direct access to plugins through the dashboard, I leveraged the jsmol2wp LFI vulnerability again to read the plugin file:

**metin, ekran görüntüsü, yazılım, bilgisayar simgesi içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.URL |** **http://www.smol.thm/wp-content/plugins/jsmol2wp/php/jsmol.php?isform=true&call=getRawDataFromDatabase&query=php://filter/resource=../../../../wp-content/plugins/hello.php**

****

The plugin code contained a Base64-encoded string. I decoded it using **CyberChef** and unescaped the characters to reveal a backdoor that allowed **RCE (Remote Code Execution)** via the cmd URL parameter.

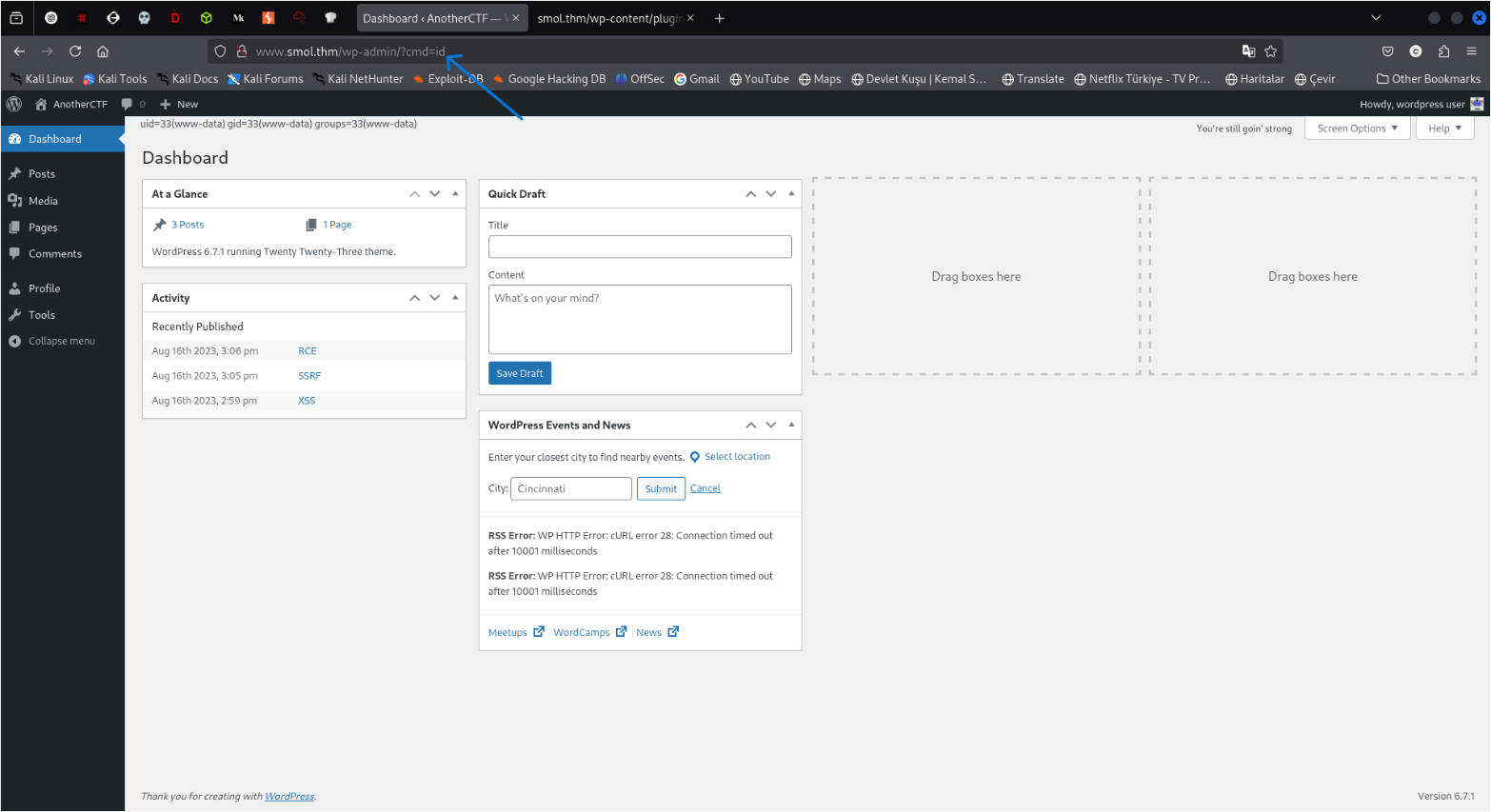
metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

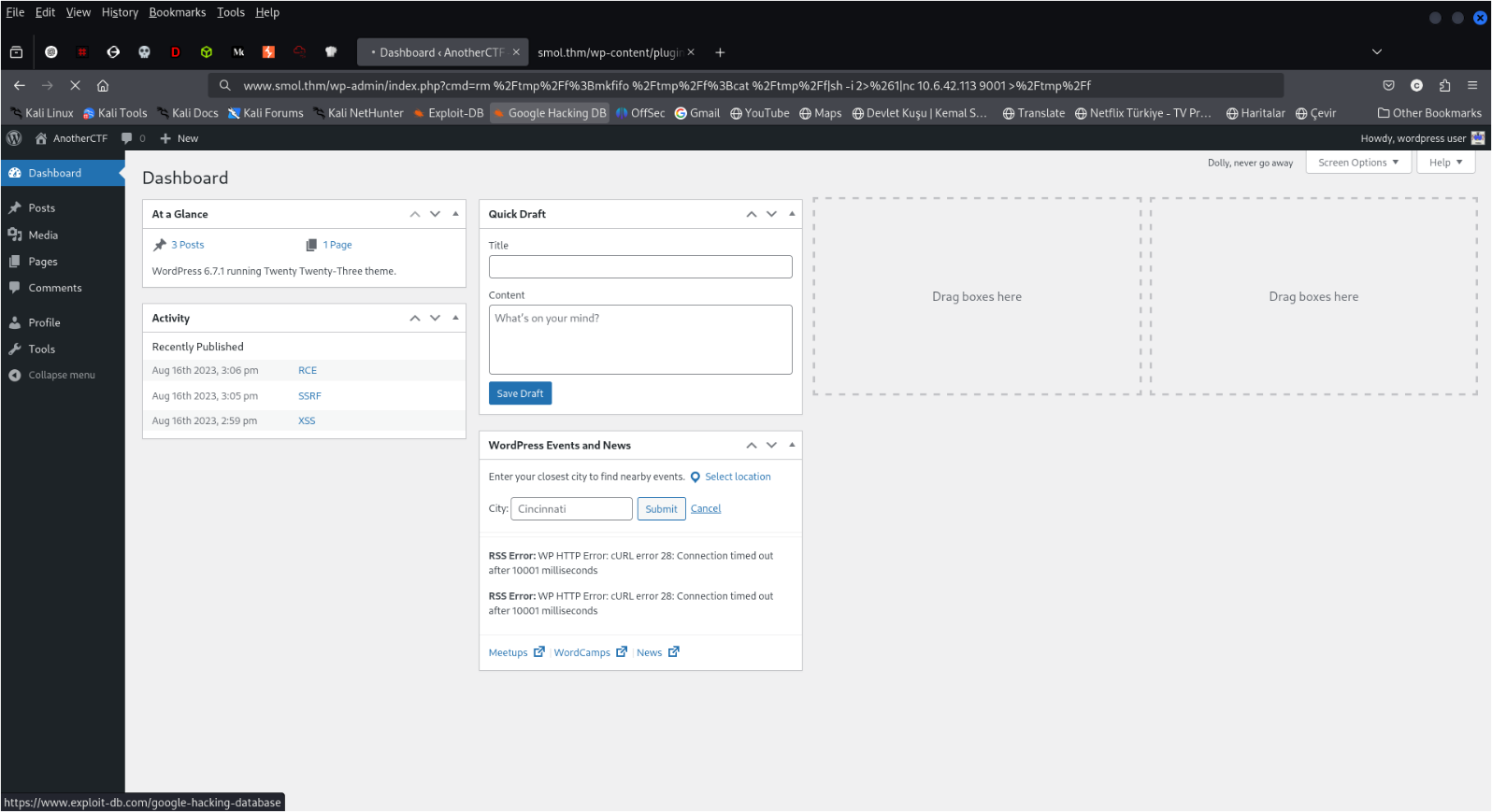
**5. Reverse Shell**

I verified RCE by testing a payload :

Next, I used the following reverse shell payload

ekran görüntüsü, yazılım, multimedya yazılımı, metin içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

Before executing, I set up a Netcat listener:

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

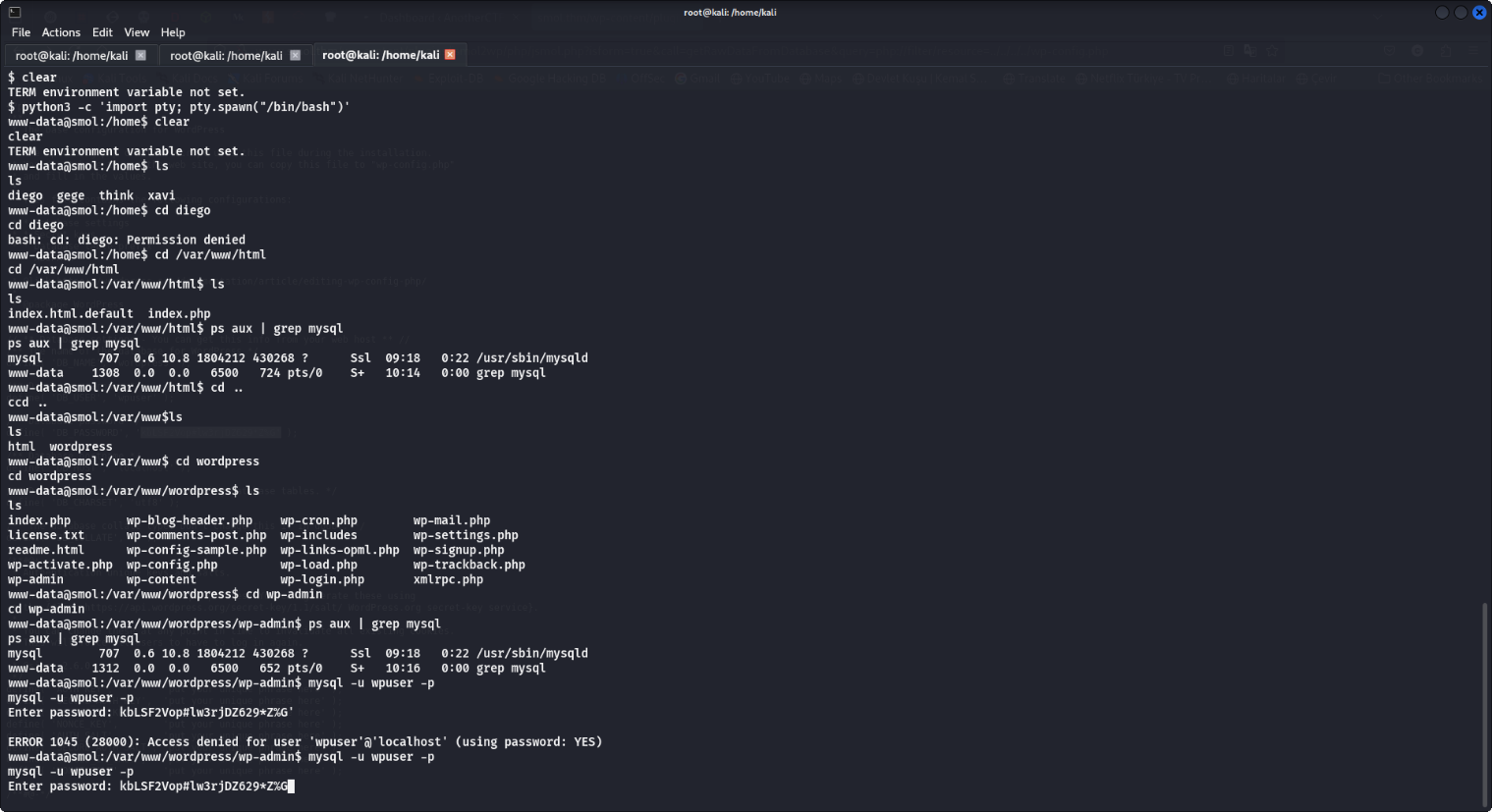
Once the shell was active, I upgraded it to a full TTY Shell:

**Command :** **python3 -c 'import pty; pty.spawn("/bin/bash")'**

**6. Exploring the System**

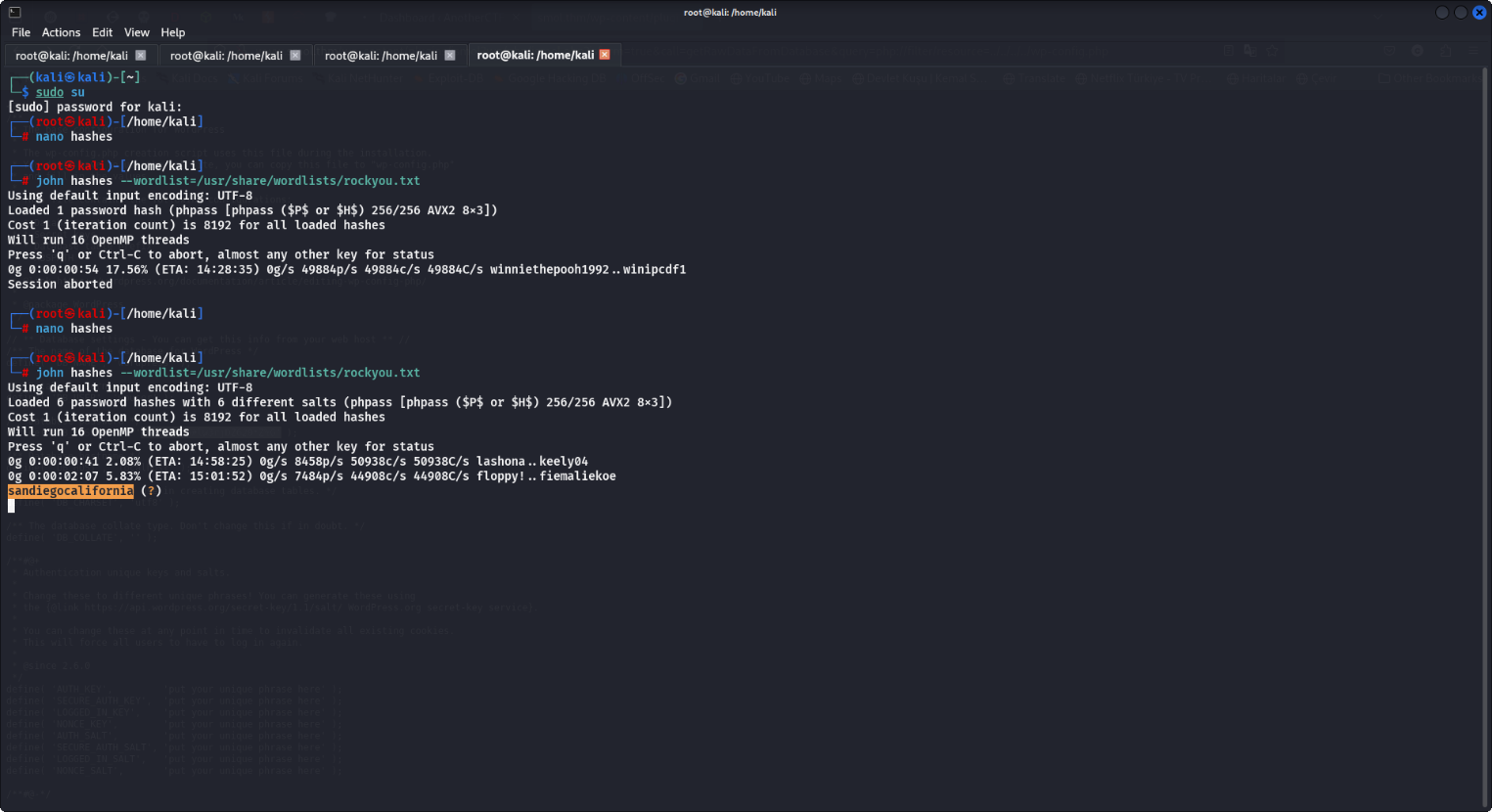
I checked the MySQL database using credentials from the wp-config.php file

**Commands :**

**mysql -u username -p  
SHOW DATABASES;  
USE wordpress;  
SHOW TABLES;  
SELECT \* FROM wp\_users;**

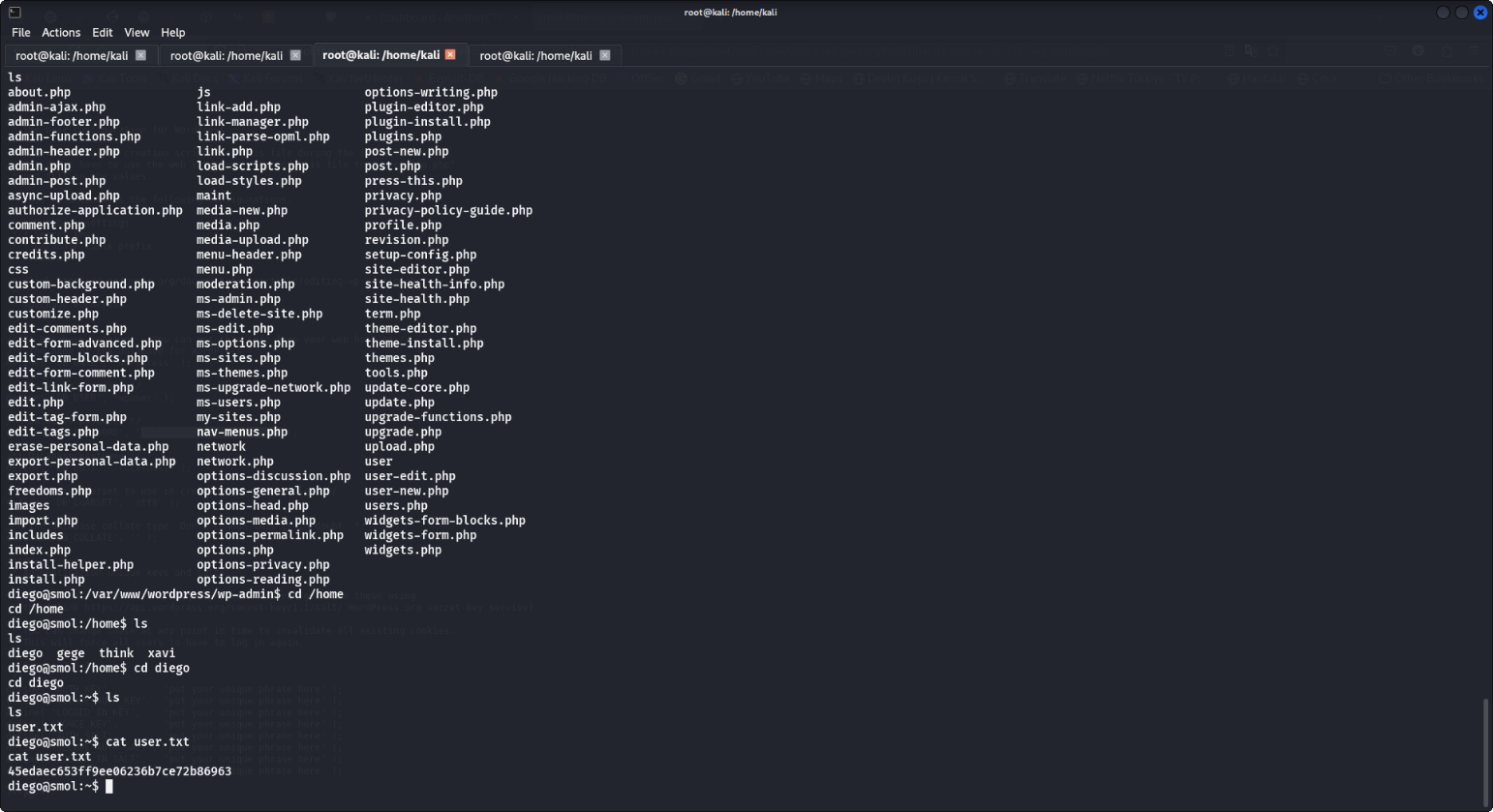
**ekran görüntüsü, metin, ekran, görüntüleme, yazılım içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.**

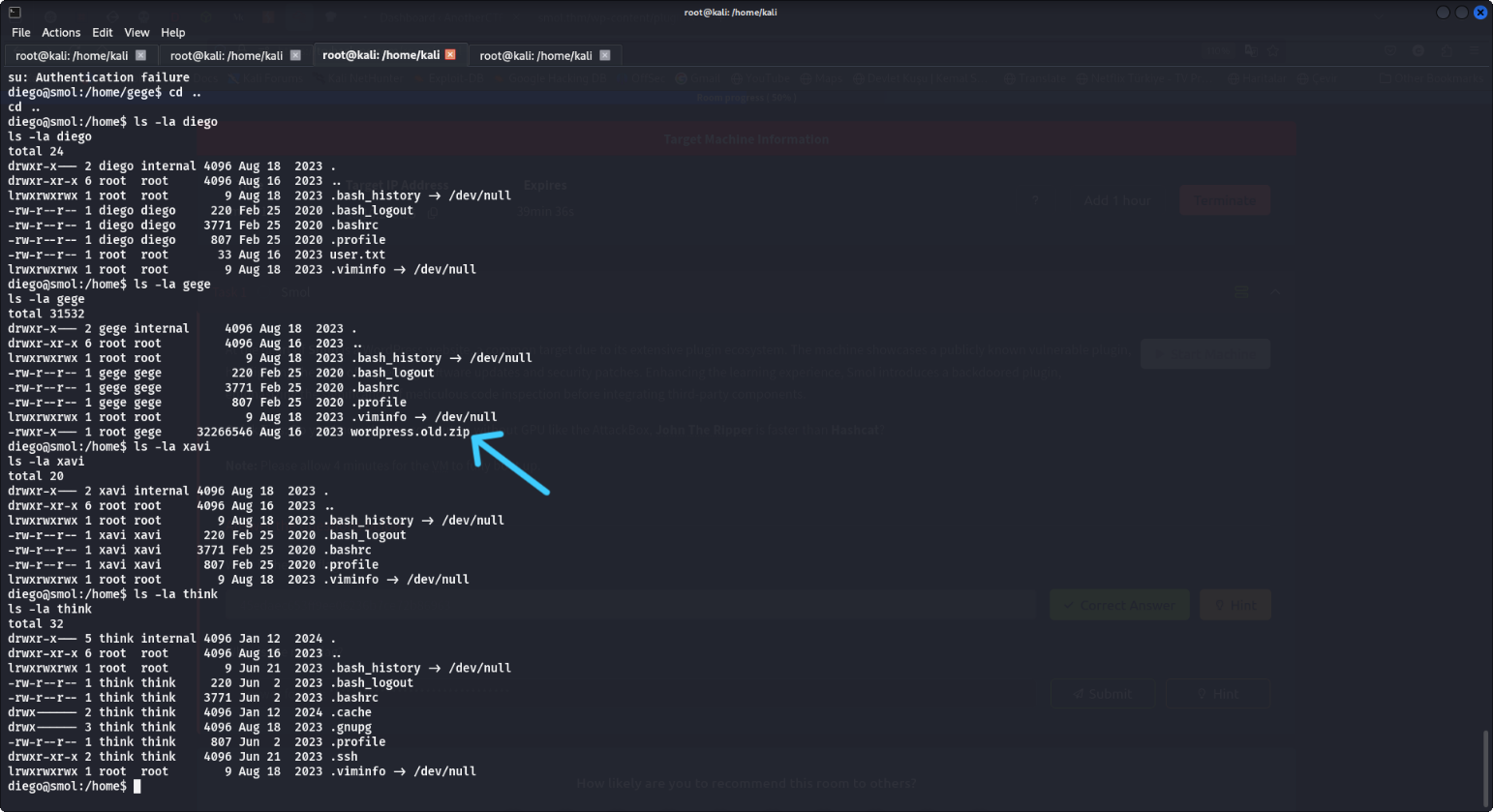
Following this, I cracked the WordPress password hash for user ‘diego’

**7. Privilege Escalation**

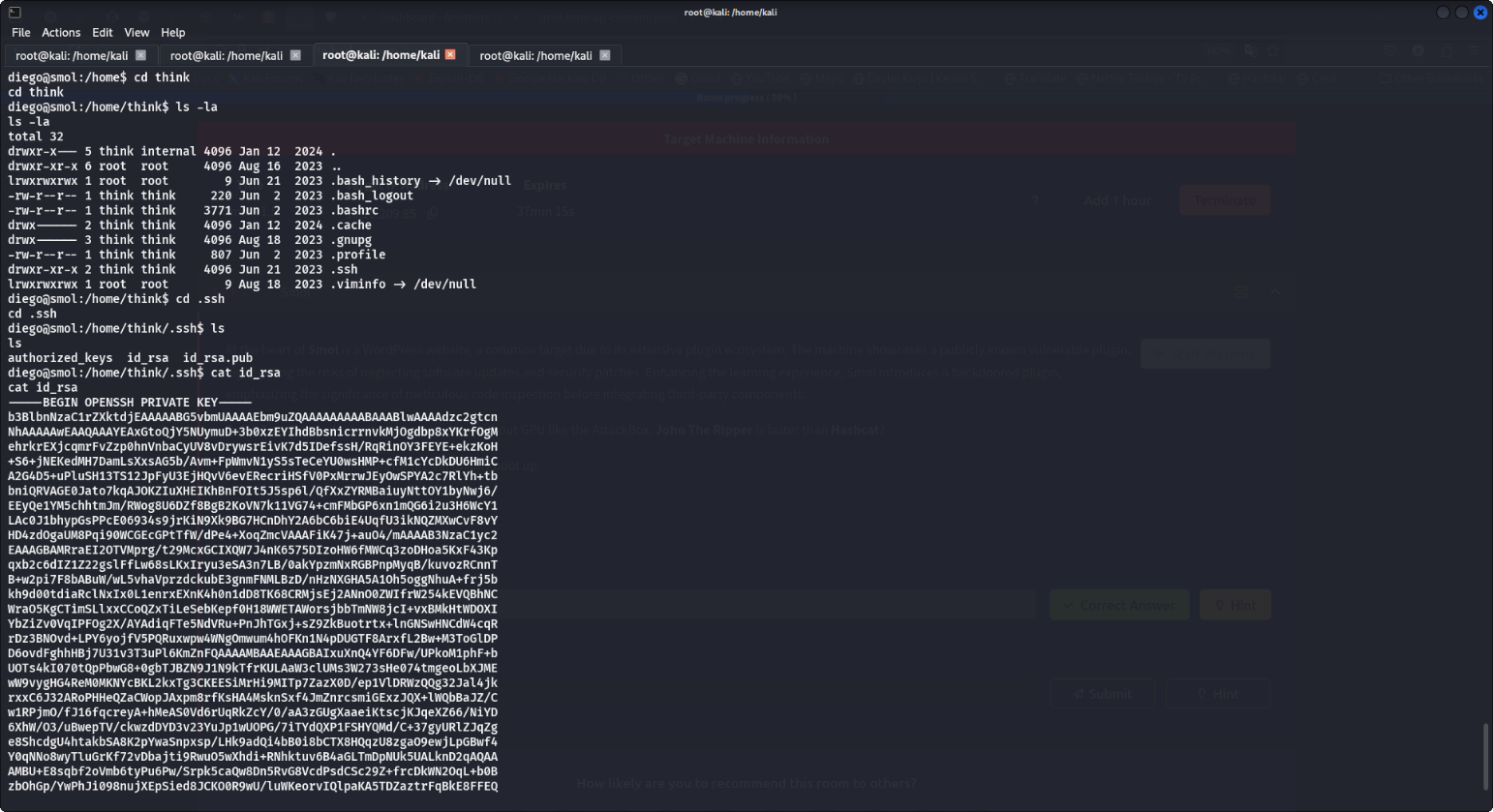
**User Flags:** After switching to the user **diego**, I located the first user flag in their home directory

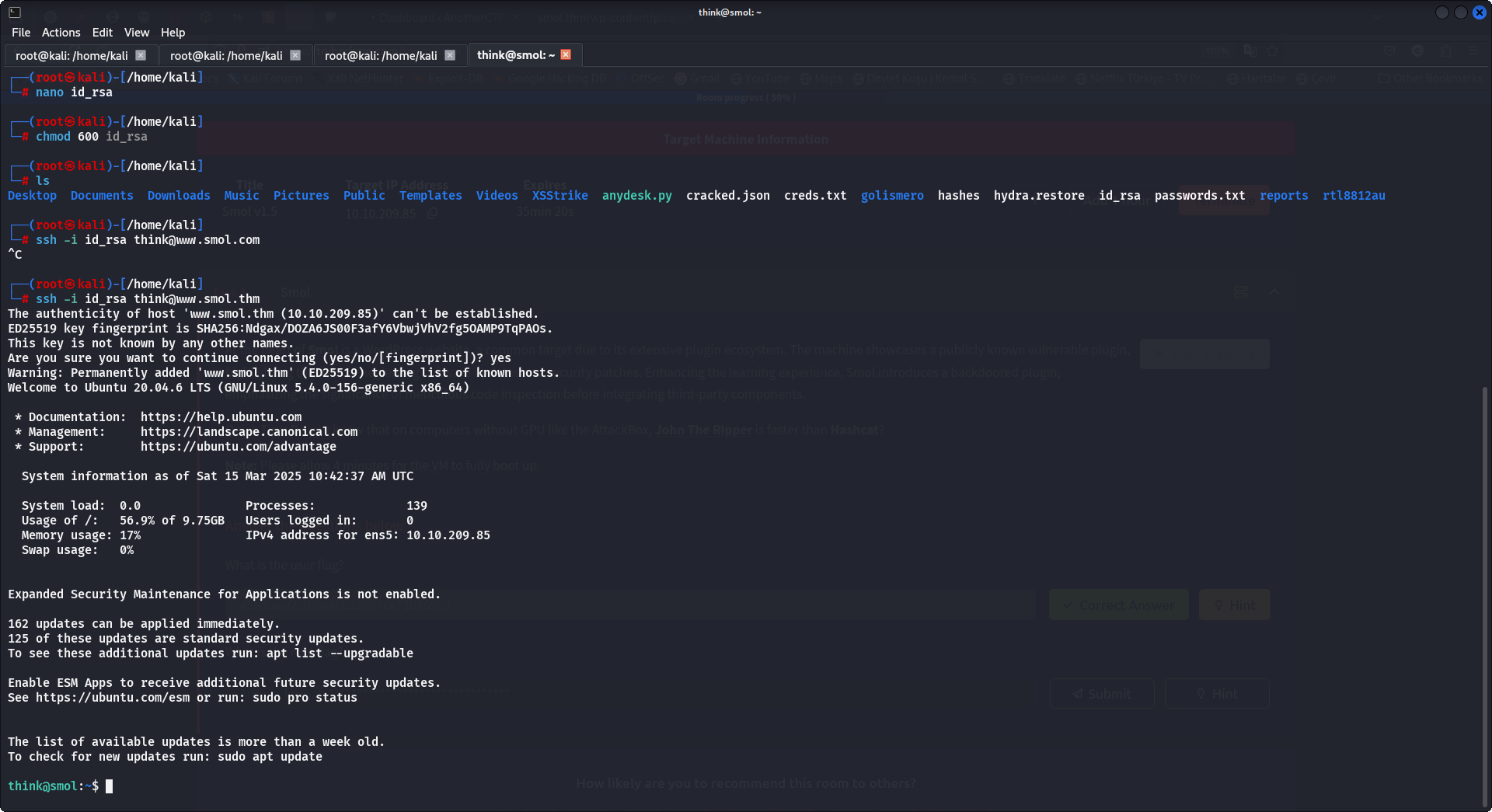


**Gege’s Archive:** In **gege**’s home directory, I found a wordpress.old.zip file, but couldn't unzip it due to permission issues

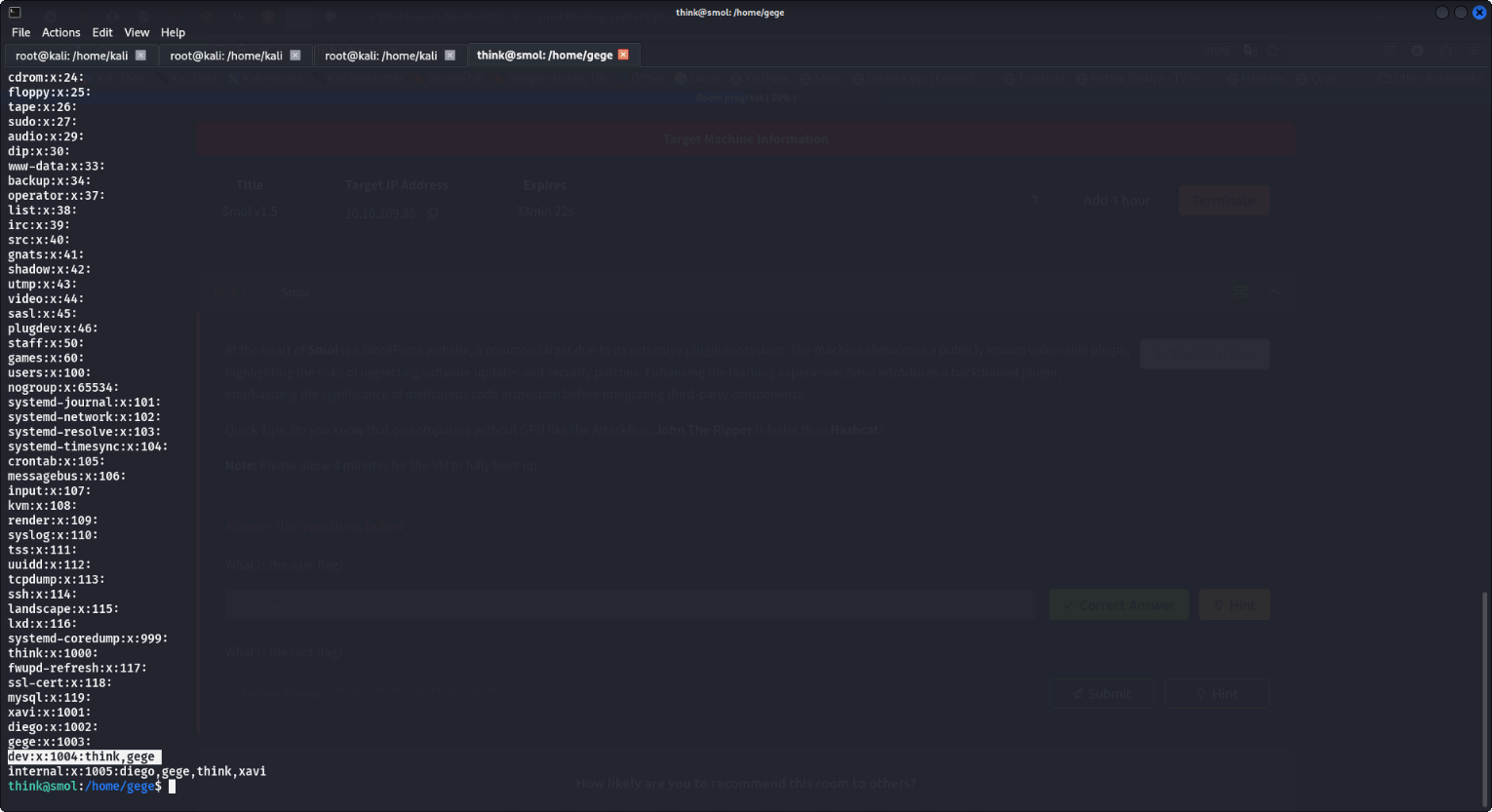




**Think’s SSH Key:** In **think**’s .ssh directory, I retrieved the private key and used it to SSH into the server as think

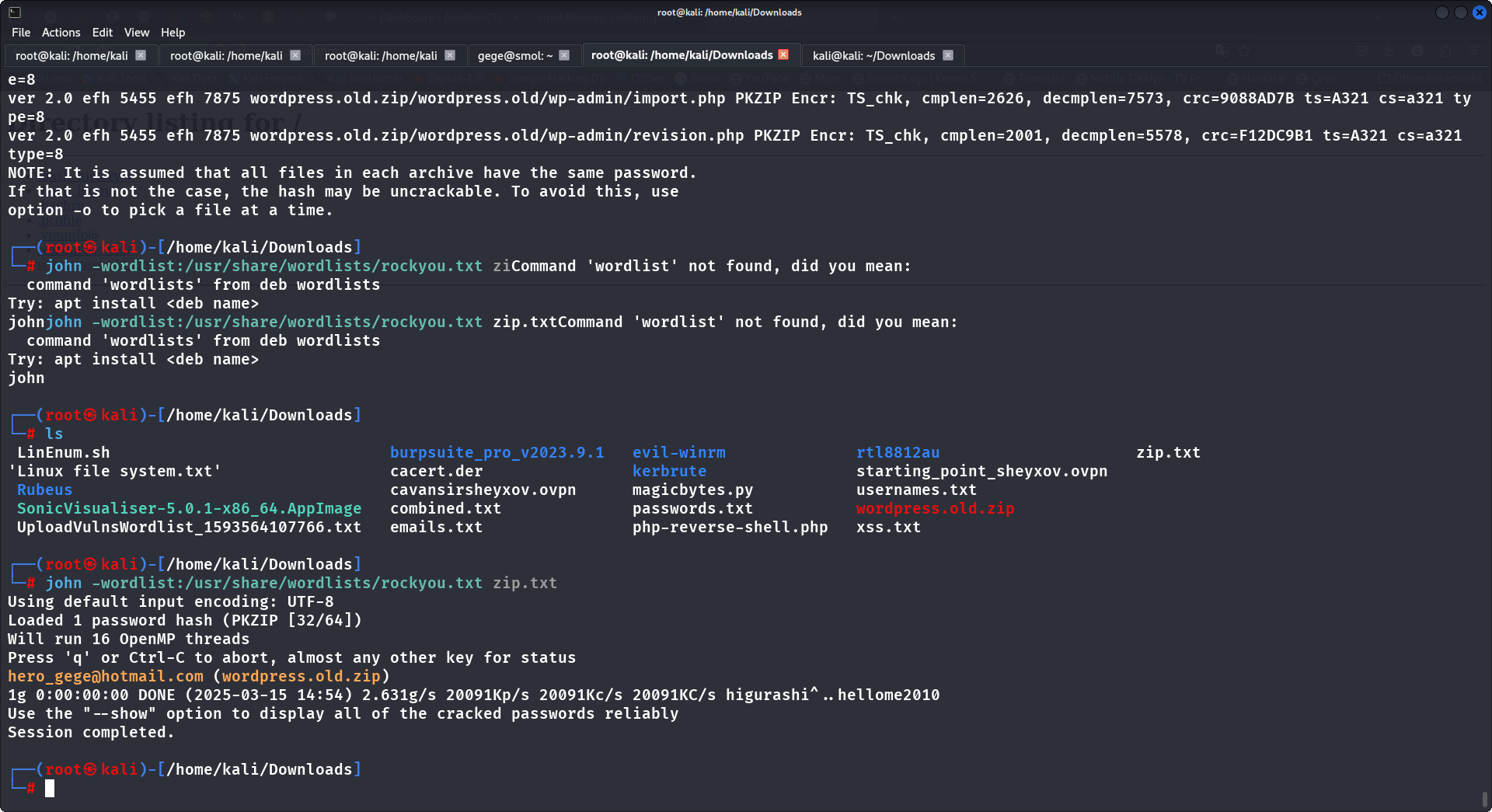


I discovered that gege and think are in the same group(dev) and I could switch to gege without a password:



**8. Cracking the Zip Archive**

I served the directory via Python to download the zip file and used **John The Ripper** to crack its password:



metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.Inside the archive, I found credentials for the user ‘xavi’ in the wp-config file.

**9. Root Access**

Finally, I switched to the **root** user and retrieved the **root.txt** flag from **/root**.

