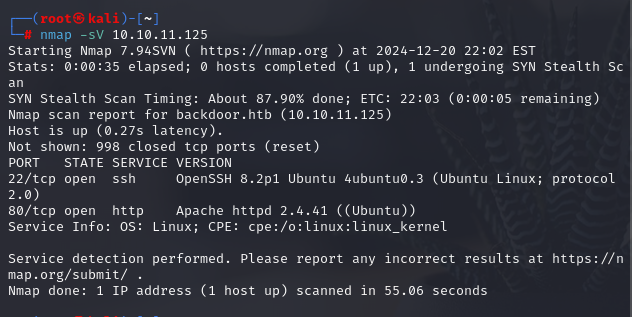
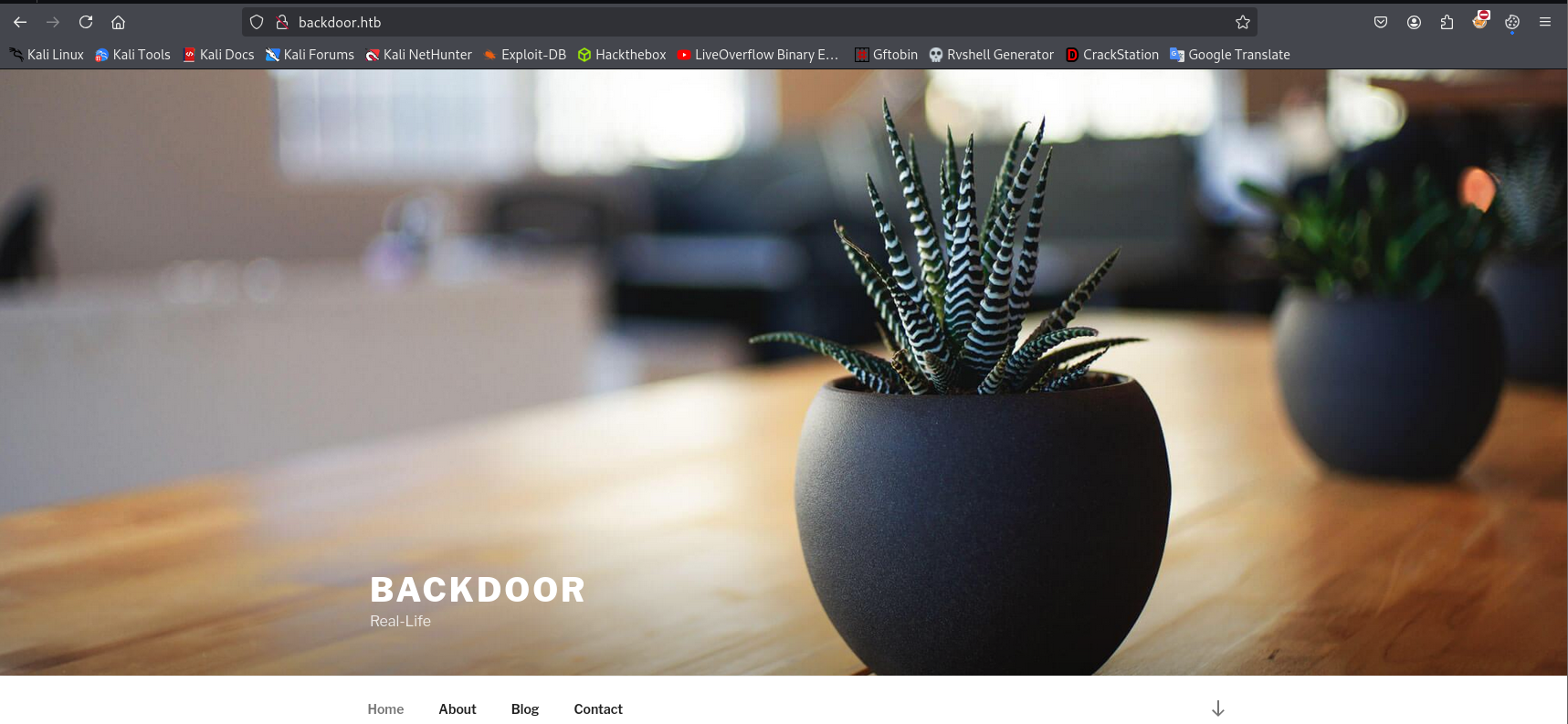
# BACKDOOR

# INFORMATION GATHERING

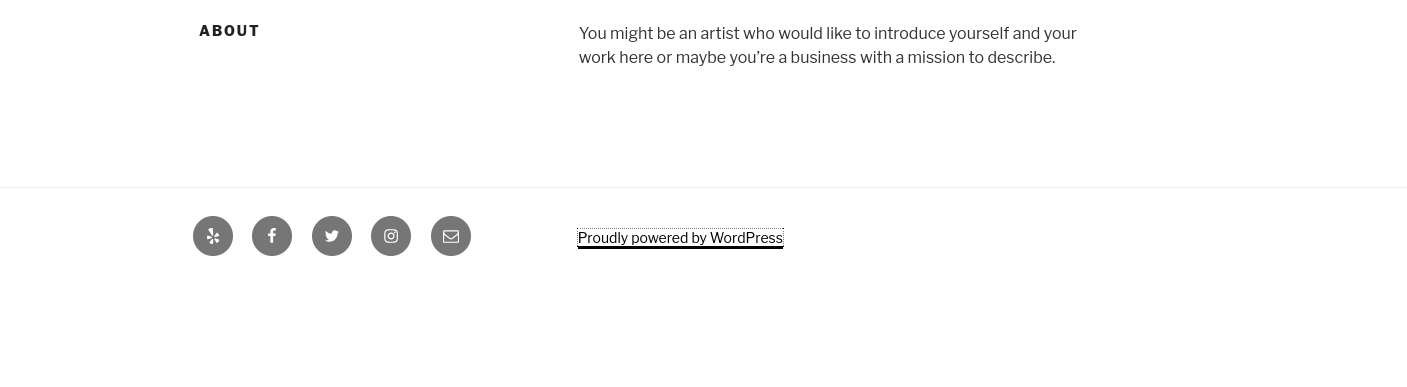
Looking at the nmap results, we can see that SSH is running on port 22 and Apache Web server is running on port 80. Also port 1337 is open, this port may be hidden and we can focus on this port



Open website on port 80

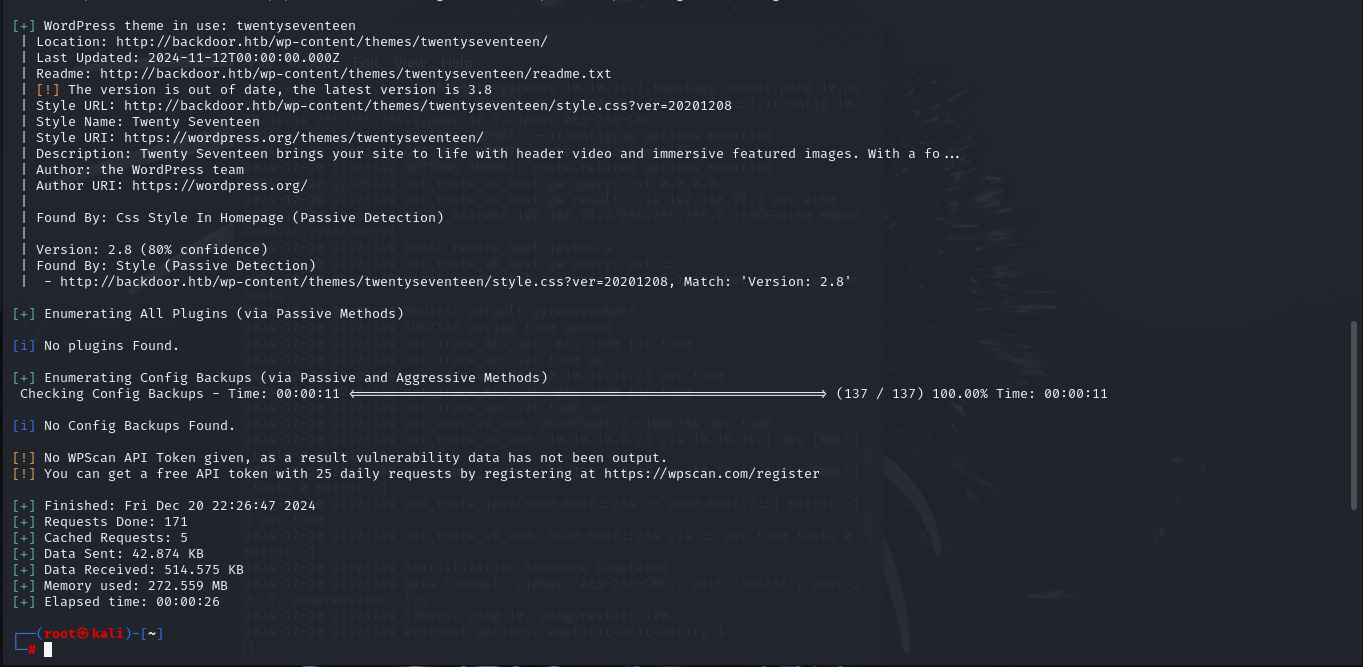


We see that the website is made with Wordpress.



Wp-scan is a WordPress security scanner that detects security vulnerabilities, outdated plugins and themes, and checks system configuration. It can scan WordPress versions, find vulnerabilities, check for weak users and passwords. We use wp scan to scan our target.

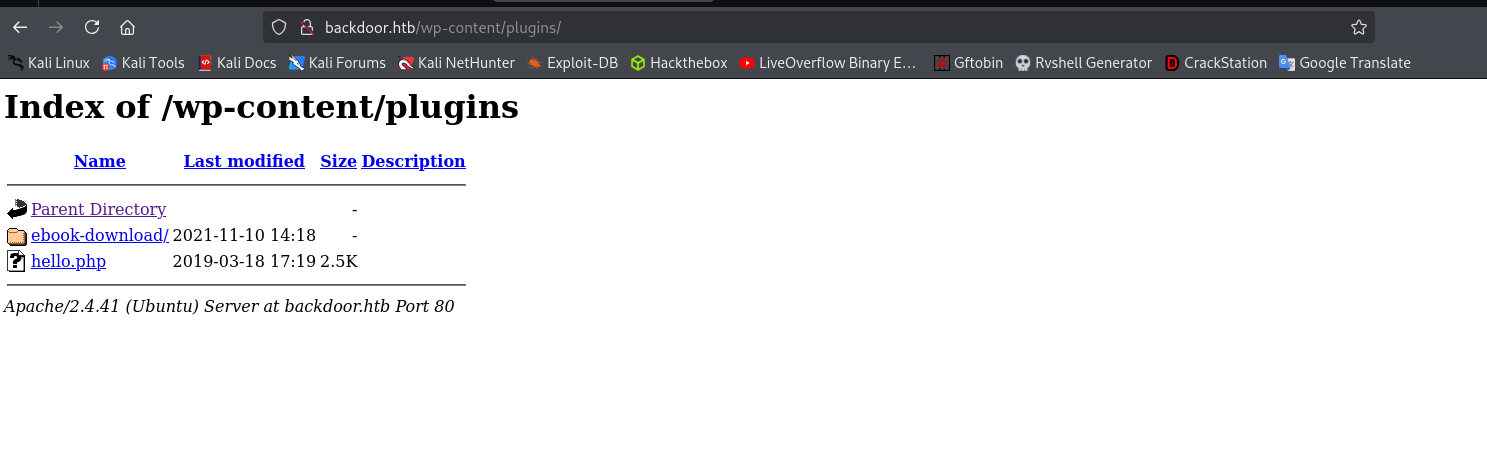




We dont see anything

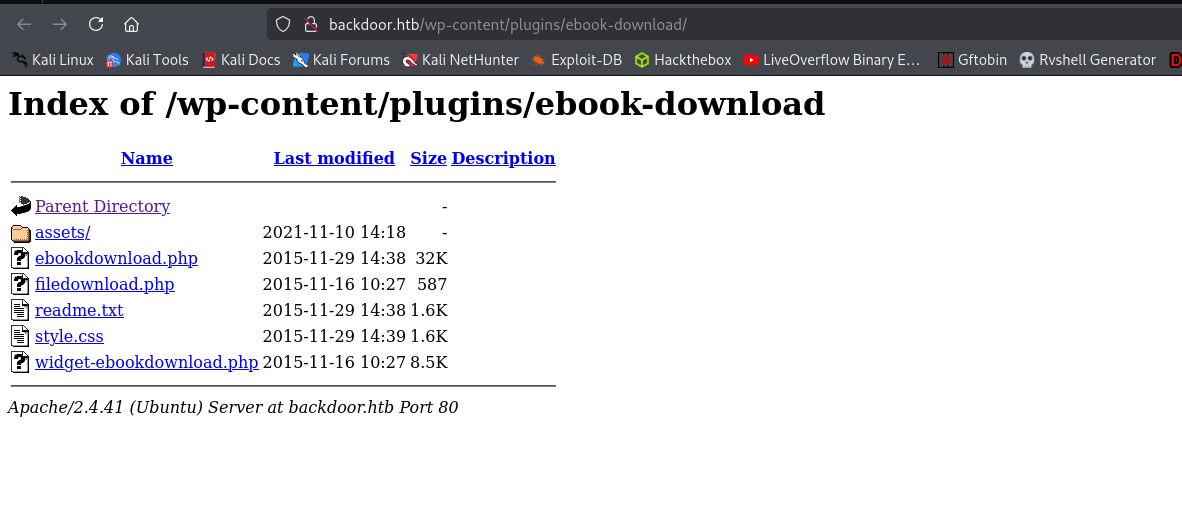
# EXPLOIT

Lets check our target website, because we know that website is made by Wordpress, so we can check some default URLs that we already know for Wordpress.



In /wp-content/plugins/ we see some files and folder

Let’s see Ebook-download folder



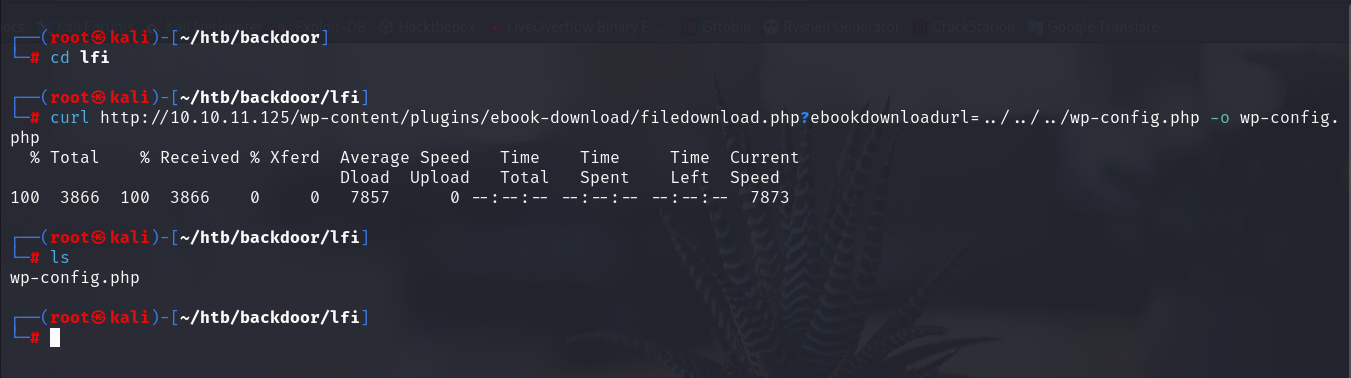
Here web see the ebookdownload.php. So lets check it out on Exploitdb with the keyword “ebook download”

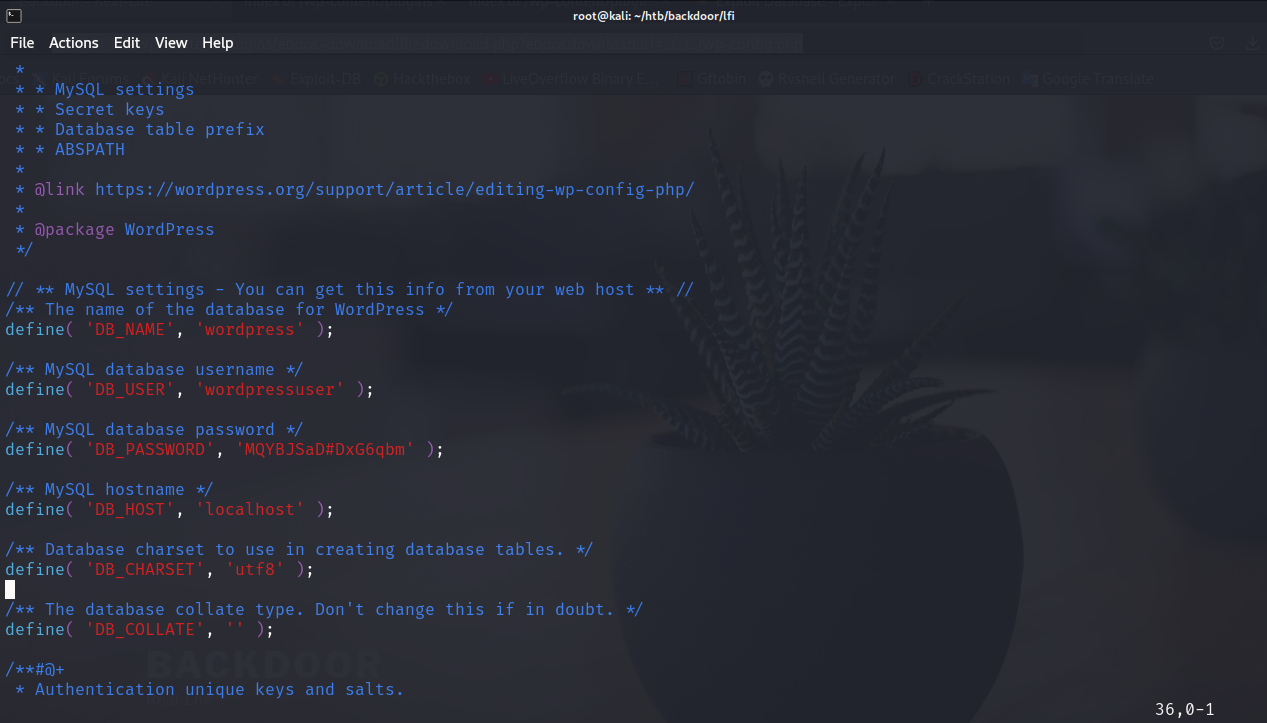


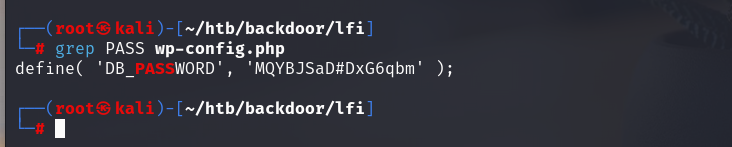
And we found a Directory Traversal Exploit for our target <https://www.exploit-db.com/exploits/39575>



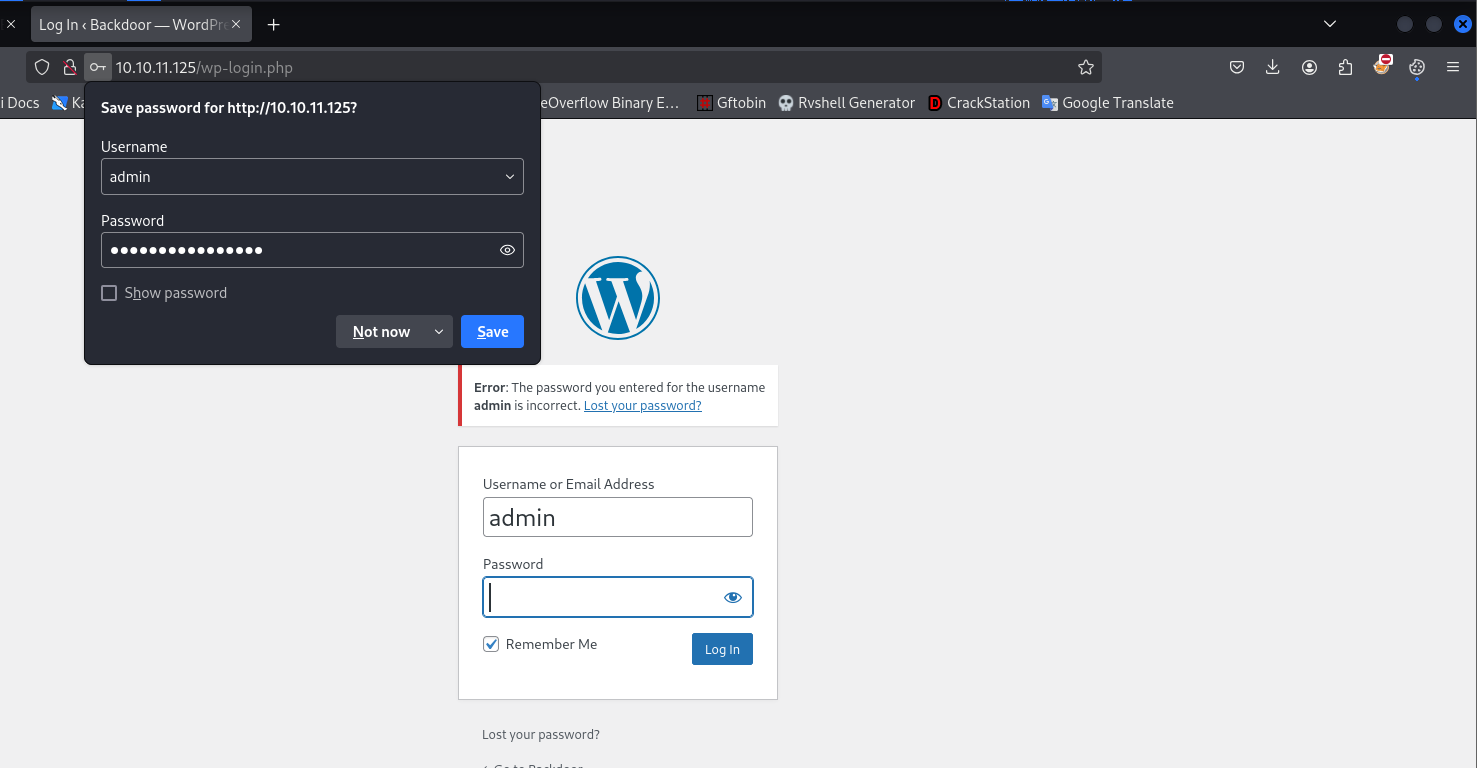
First, download it on our Kali with the name wp-config.php and use vim to read file







And we found a password, so lets try to login to wordpress login page.

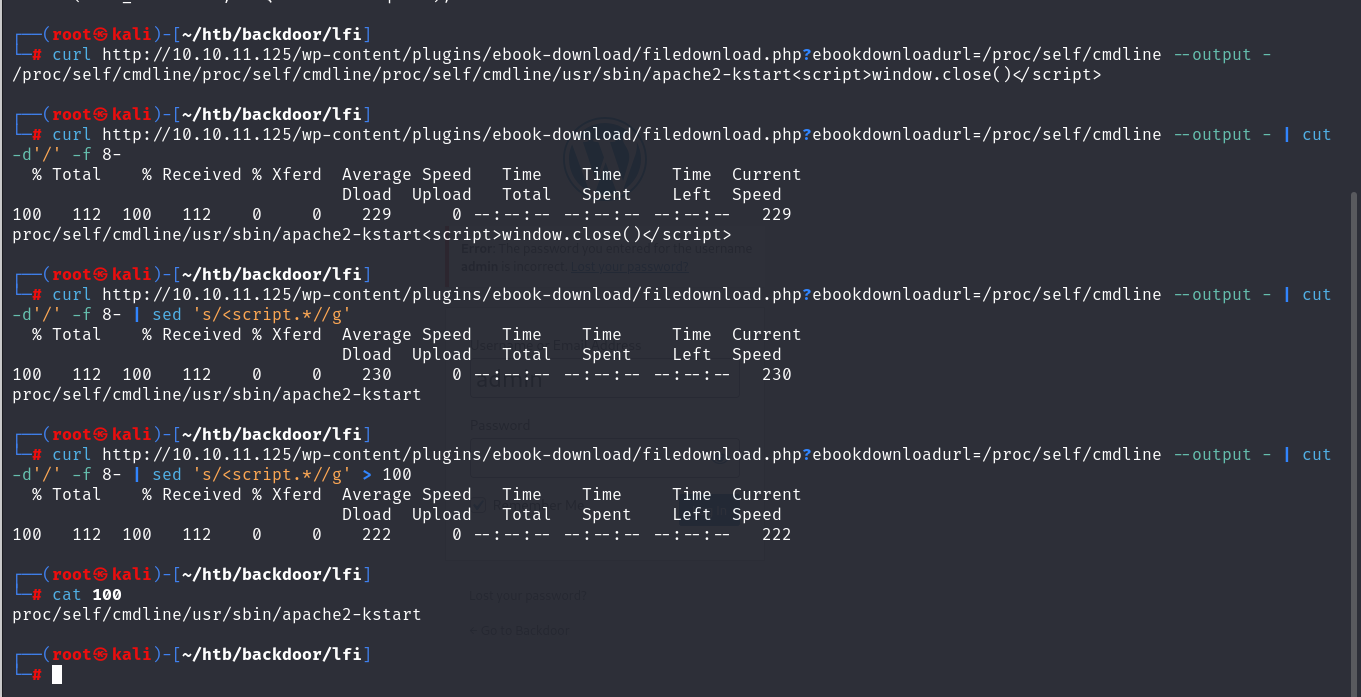


Login fail, so lets try another way

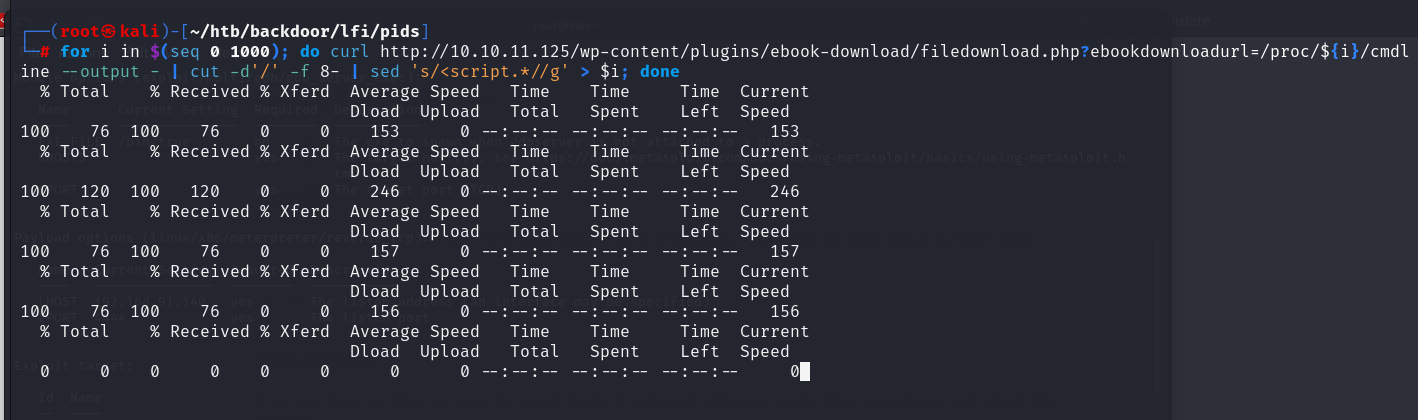
*curl http://10.10.11.125/wp-content/plugins/ebook-download/filedownload.php?ebookdownloadurl=/proc/self/cmdline --output - | cut -d'/' -f 8- | sed 's/<script.\*//g' > 100*

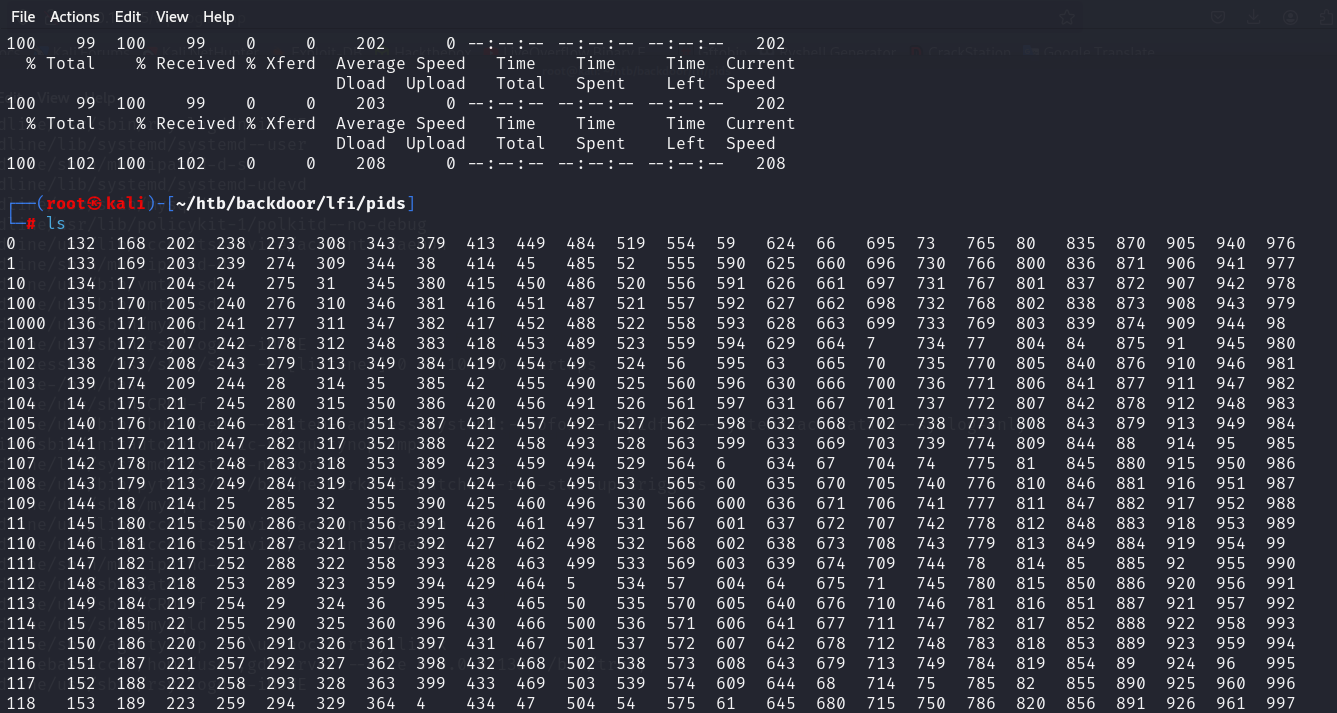
The ebookdownloadurl=/proc/self/cmdline parameter is an exploit, where /proc/self/cmdline can retrieve information about the command or process running on the server.

* **cut -d'/' -f 8-:** The cut command splits data by the / delimiter. This command takes from the 8th field onwards in the URL path string.
* **sed 's/<script.\*//g':** The sed command is used to remove the tail script code.



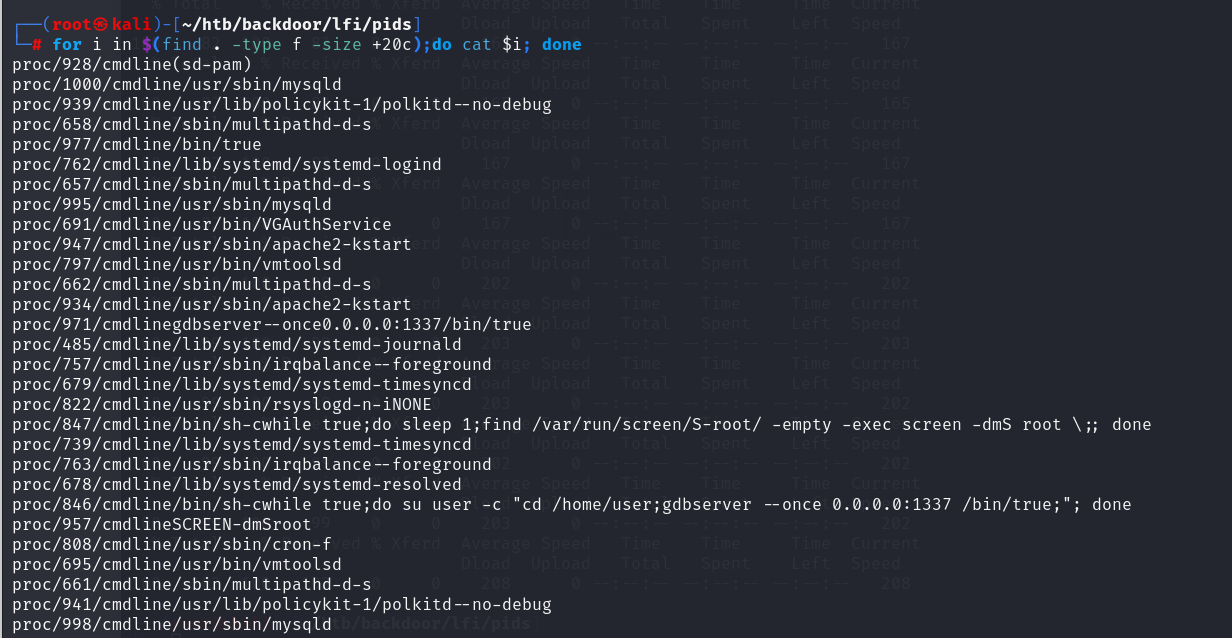
for i in $(seq 0 1000); do curl http://10.10.11.125/wp-content/plugins/ebook-download/filedownload.php?ebookdownloadurl=/proc/${i}/cmdline --output - | cut -d'/' -f 8- | sed 's/<script.\*//g' > $i; done



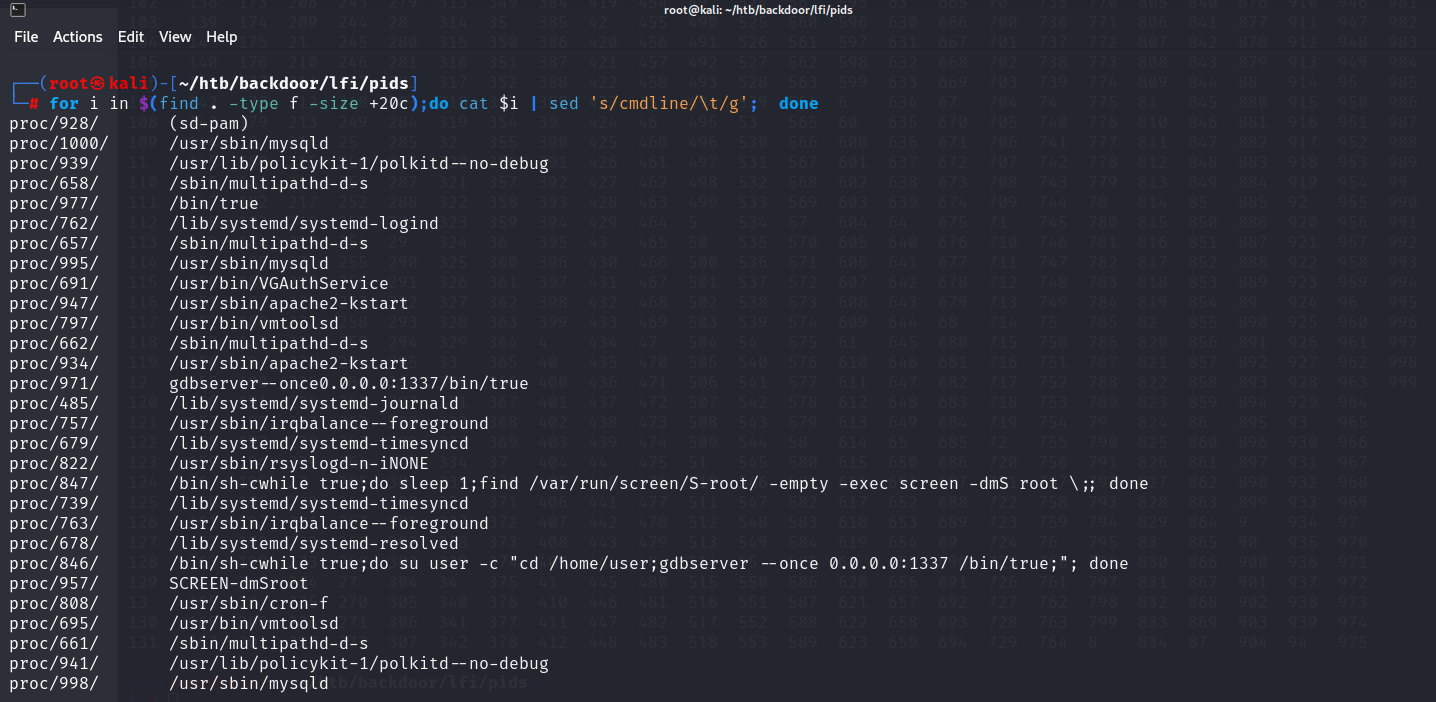


Find all files larger than 20 bytes in the current directory and then in the contents of each file using cat, using the following command:

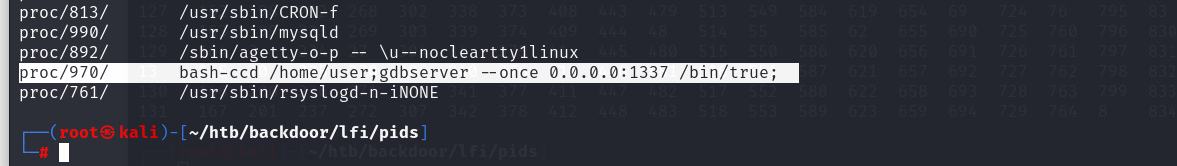
*for i in $(find . -type f -size +20c);do cat $i; done*



Cut cmdline and replace with space tab for easy analysis



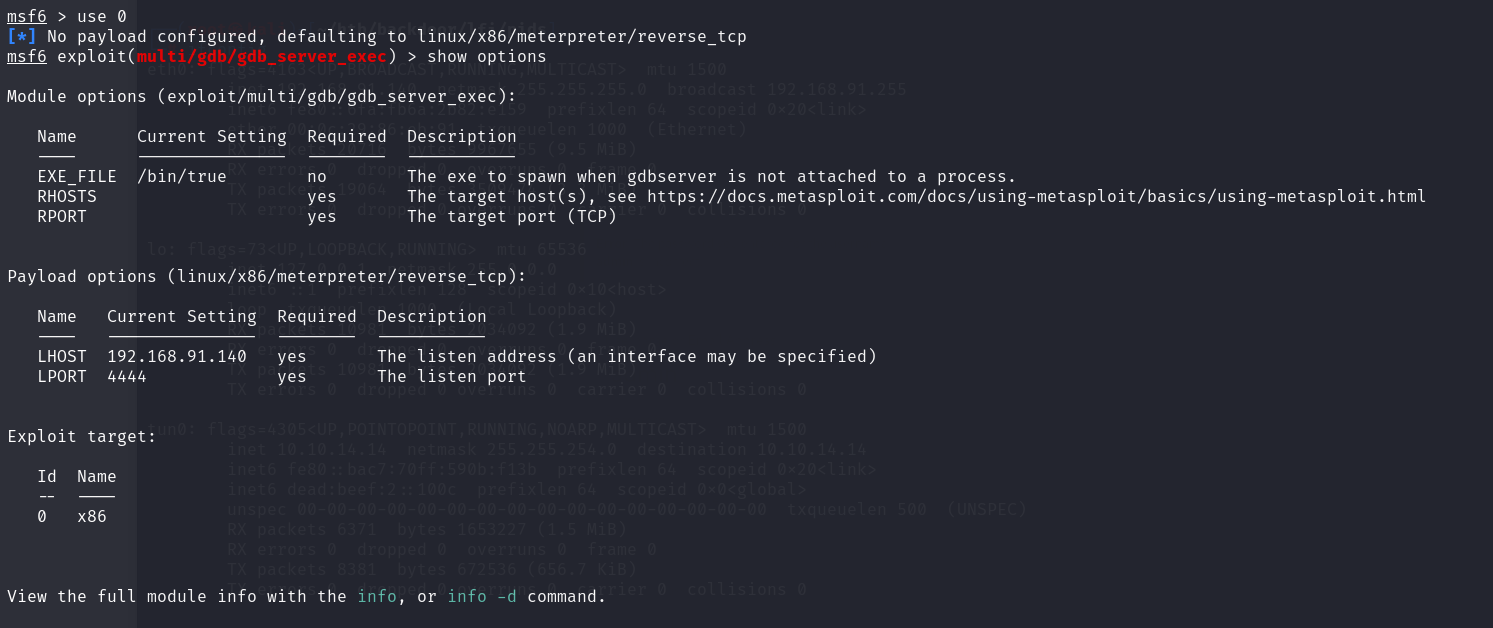
This process executes a bash command (a shell) to execute the command ccd /home/user and starts gdbserver to listen on address 0.0.0.0:1337. This may be relevant for debugging or inspecting remote processes. Now we know gdb is listening on port 1337.



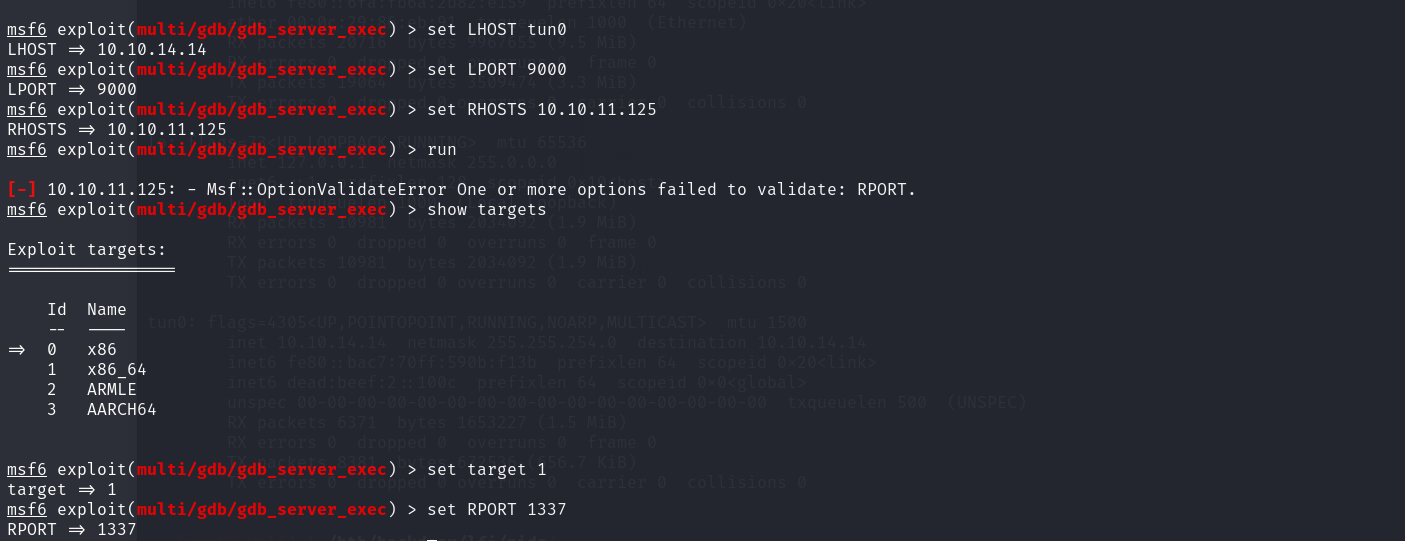
We can use Metasploit to find an exploit module for gdb. And we se the module name exploit/multi/gdb/gdb\_server\_exec, lets try it.

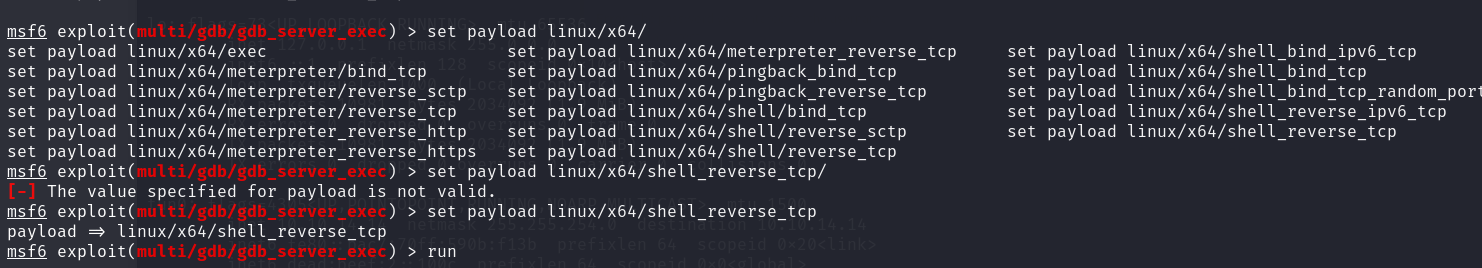


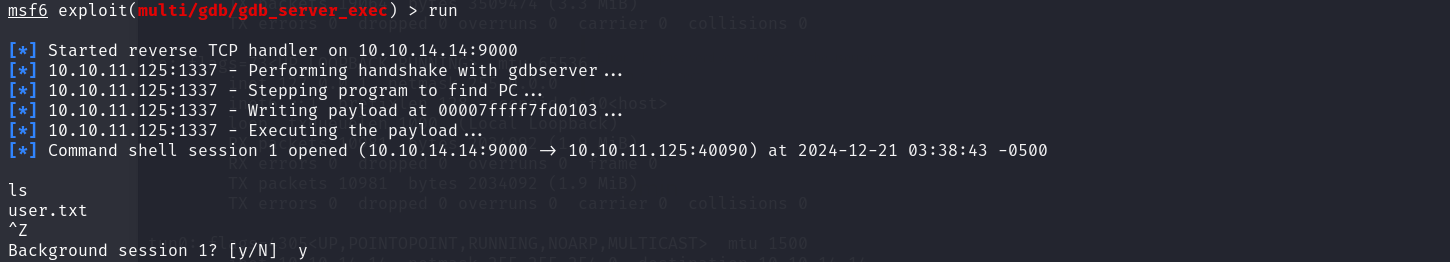
We use *show options* to display the options we need to run this exploit.



Set LHOST to tun0, which is the IP address of the attacker machine. Set LPORT to 9000 (the port on the attacker machine to listen on). Set RHOSTS to the target IP. We also need to set the target to 1 to exploit the x86\_64 system version and choose the appropriate payload for the target.

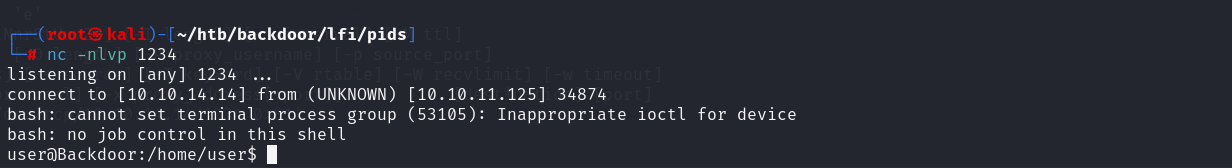




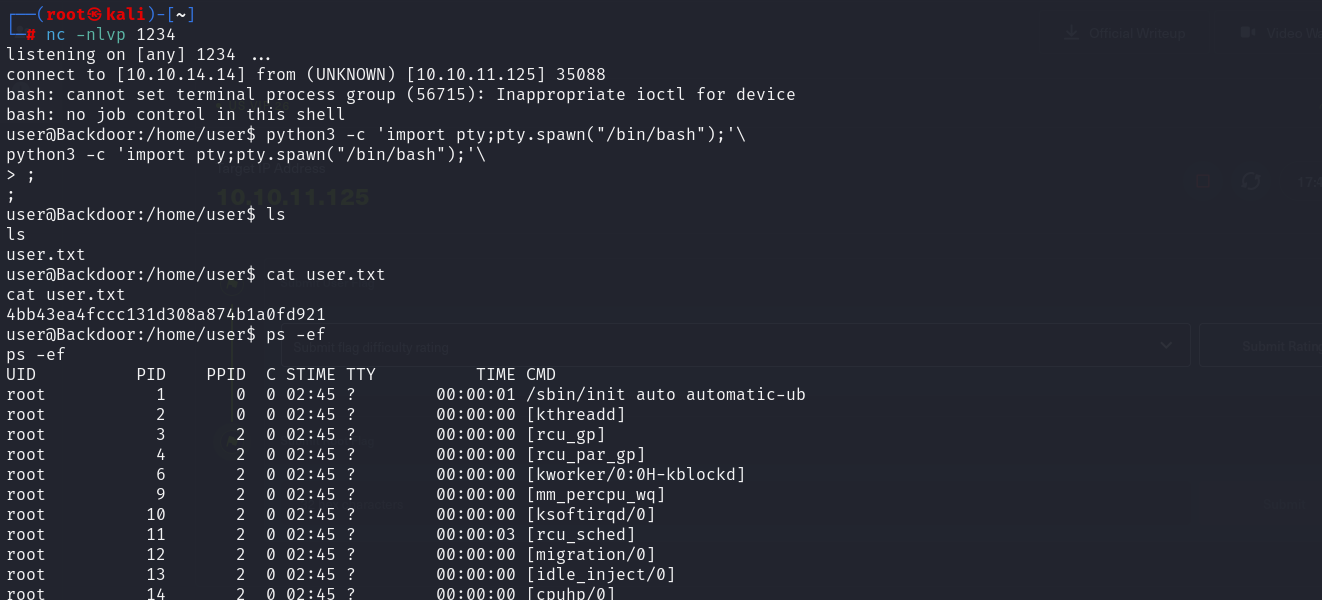


And we got a Meterpreter shell. The next step is to bring a shell to our Kali command line by using a reverse shell and using nc to listen on our Kali machine

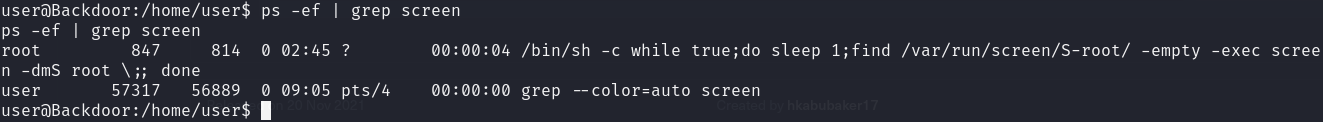




We have a shell on our Kali machine. Let's take the first flag.



Use the command *ps –ef* to list all the processes running on the system with detailed information.



We see a process, which is creating a screen session, inside a do-while loop. This process is being run as root, which means that this sceen session is also created by the root user and would have root privileges. The command inside the loop is as follows. *find /var/run/screen/S-root -empty -exec screen -dmS root ;*

The default screen syntax for attaching to a screen-session created for a different user is user/session\_name . Furthermore, to be able to attach to a screen session, the screen -x TERM environment variable needs to be set, as it defines the terminal type. In other words, it sets the terminal type for which output is to be prepared. We will set it to the value xterm :



Finally let's try to attach to the root screen session.



We have a root shell. The root.txt flag can be found at /root/root.txt

