Lab 3

Advanced Persistent Threats, Fuzzing and Exploitation

Due Date - October 26th

Part 1: APT Campaign Questions

APTs use a variety of tools and techniques to accomplish their goals. In this set of exercises, you will use openly available reports and information to learn more about some of their specific techniques. The goal is to familiarize you not only with some of the APT techniques but also some of the resources someone interested in enterprise network security could use to secure their network. All of the information you will need to answer these questions is given in the links provided.

The answers to each set of problems should fit on 1.5 typewritten pages or less. Do not write more than 2 pages per set of exercises.

Exercise Set 1

Read the following sections in Fireeye Mandiant's M-Trends 2019 report: "Executive Summary", "By the Numbers", "APT", and "Defensive Trends". For convenience, this report has been uploaded to canvas, but can also be found here: https://content.fireeye.com/m-trends.

Answers to all questions below can be found in these sections of the report and/or by using information on https://attack.mitre.org (Note: only question 3 requires the use of MITRE's ATT&CK as a source of information).

The goal of this set of problems is to build a general understanding of recent APT tactics, techniques, and procedures.

1-1. What is the definition of dwell time? What has affected the change in median dwell time in the last year? Optional bonus: Why is the median a good/bad metric for dwell time?

1-2. What are the general goals of each of the newly named APT groups?

1-3. For APT37, what are the known methods of initial compromise? Using MITRE's ATT&CK as additional information, provide a short definition of each method of initial compromise and how a defender can either detect or prevent this sort of behavior.

1-4. Based on pages 71-72, briefly summarize why "Lack of Investigation" and "Poorly Timed Remediation" are problematic. What should defenders do to avoid these problems?

Exercise Set 2

Use the information here https://attack.mitre.org/techniques/T1102/ to do this set of problems. The goal of this set of problems is to learn about MITRE's ATT&CK resource and to learn more about a specific type of APT behavior.

1-5. What part of the APT lifecycle is described? From the defender's viewpoint, why might this type of behavior be hard to detect?

1-6. Using the examples on the page, create a simple table listing the web service used and the number of times it is listed as an example. Format the table as follows:

Web Service	Examples
Service 1	10
Service 2	5
Service 3	1

For web services that are very similar, use your best judgment as to whether or not to group them into a single row. For example, one could list all cloud storage in separate rows or group them all together. The goal of creating the table is to collect enough information that would be useful as a defender to get a general understanding of what APT actors might do.

Sort the rows in descending order by the number of examples and use the sorted table as your homework answer.

1-7. What web services are most prevalent in the table created? How might one detect the three most prevalent web services being used for malicious purposes?

Exercise Set 3

Now that you are more familiar with the resources listed, the following questions will require using different parts of each resource to answer. The goal is to build additional knowledge of recent APT behavior and to gain further familiarity with useful, online resources.

1-8. Using the report on APT41, briefly summarize why an APT actor would be interested in targeting the video game industry. Note: For convenience, this report has been uploaded to canvas.

1-9. Using the report on APT29, describe the five stages of Hammertoss. Why are these difficult to detect? Note: For convenience, this report has been uploaded to canvas.

1-10. In the attack matrix, what is Sudo Caching? How can this be detected/stopped?

1-11. What does the malicious software MimiPenguin do? Your answer should incorporate the stages of the APT lifecycle.

Lab Setup

These steps are for preliminary setup for the next two parts.

1. Download the Windows victim VM from here and import it into Virtualbox

```
https://drive.google.com/file/d/1_sflYSISM8wwFwjDz9f57f0hJVhwPFJs/view
```

2. On our class Ubuntu VM, we will be setting up Metasploit. To do this, use these commands:

```
$ curl https://raw.githubusercontent.com/rapid7/metasploit-omnibus/master/
config/templates/metasploit-framework-wrappers/msfupdate.erb > msfinstall
$ chmod 755 msfinstall
$ ./msfinstall
```

3. Use Virtualbox to set up inter-VM connectivity between the Ubuntu class VM and the Windows VM. This can be done by setting up a NAT network within Virtualbox using these steps (steps are for Virtualbox 6.0):

- On the VirtualBox Manager window, go to File > Preferences
- Select Network on the left and add a new network. Name it whatever you wish.
- Open Settings for Ubuntu class VM. Under Network, switch to NAT network.
- Repeat for the Windows VM
- Note down the new local IP addresses of both the Ubuntu VM and Windows VM

Part 2: Fuzzing

Setup

1. Start the Windows VM and login as the class user. The password is the same as the Ubuntu VM: classpassword. Start Vulnserver using the

desktop shortcut

- This starts vulnserver listening on port 9999
- If asked about choosing a network type, you can choose “Public Network”

2. Verify you can connect to the server from your Ubuntu VM using netcat:

- `nc <WindowsVM-IP> 9999`

Exercise

In this exercise, you will be writing a fuzzer in python for vulnserver, a very simple server application. Familiarize yourself with vulnserver. Vulnserver accepts tcp connections; we recommend using netcat to initially communicate with and observe how the application behaves. Note the commands and arguments that vulnserver accepts and responds to.

With your knowledge of vulnserver’s communication protocols, write a simple python program to fuzz vulnserver. In addition to generating random data inputs for its various commands, consider providing static test input for particular fuzzing edge cases discussed in class. Record any abnormal behavior you recognize and the inputs that caused them.

Submit the python fuzzer code and answer the following questions in the report:

2-1. Did you observe any instances of notable behaviors? Summarize them briefly.

2-2. For one of those instances, provide your analysis on the cause of the behavior. Why might a particular input have such an effect? What type of exploit would be suitable to try given the results of your fuzzing?

Part 3: Exploitation

For this part of the assignment, we will be running through an attack scenario. We will use the Metasploit framework on our class Ubuntu VM to attack a vulnerable application on a Windows machine and place an implant for Meterpreter. Meterpreter is an advanced shell, with helpful built-in commands useful for information gathering, pivoting, and privilege escalation. The write-up below will guide you through using some features of Metasploit, but please be aware you can always get a list of available commands using **help** on the msfconsole. Feel free to explore beyond this guide.

The app we will be exploiting, *Free Download Manager*, comes with an included remote control software that lets the user schedule downloads on their machine remotely. However, it is not very well written, fortunately for us.

Setup

1. Assuming you already have the Ubuntu and Windows VM setup for interconnectivity from above, start both VMs.
2. On the Windows VM, login as class and start *Free Download Manager* using the desktop shortcut. Also start *FDM remote control* using the desktop shortcut. (Note: The application starts hidden in the system tray)

- Verify the application is running and accessible by browsing to `http://<WindowsVM-IP>` on your Ubuntu VM. You should see a webpage that lets you view active downloads and create a new download.

3. Start Metasploit on your Ubuntu VM with the command **msfconsole**

- If Metasploit asks you to create a database for first time use, say no

4. Start Wireshark on your Ubuntu VM and start a capture on your local interface

- You might need Wireshark started as a root user to be able to capture packets on the local interface

Exercise

On the msfconsole, type **search fdm** to find all exploits for *Free Download Manager*. We are interested in the remote control buffer overflow, so now ask metasploit to use this module: `use exploit/windows/http/fdm_auth_header`

Look at the information and options of this exploit using commands **info** and **options**. Minimally, you will need to set the target IP in the RHOSTS variable using the **set** command. For example: `set RHOSTS <WindowsVM-IP>`

You can also see the available payloads that are possible to send using this exploit with the command **show payloads** and changing the payload with **set payload <payload-name>**. However, for now we will stick with the default, which is a Meterpreter reverse shell.

Once all options are correct, run the exploit using **exploit** on the console. If all goes well, you should have a *meterpreter>* command prompt on the remote system. Congratulations on getting a shell!

Before we go further, answer these questions about our exploit and what we just accomplished:

Take a look at your Wireshark capture.

3-1. What application protocol is the exploit using?

3-2. Which header field contains the payload?

3-3. Read about the usage of this header field. How is the payload encoded? Why is it encoded? Describe what you see once you decode the payload.

Take a look at the metasploit code (which is a short Ruby script) for this exploit. It should be available on your Ubuntu VM at:

`/opt/metasploit-framework/embedded/framework/modules/exploits/windows/http/fdm_auth_header.rb`

Use this link (https://netsec.ws/?p=262) as a resource on the structure of the module, and answer these questions:

3-4. How much overflow is required to the buffer before we overwrite EIP? What value are we overwriting it with?

3-5. Where does the above value return to?

3-6. What is the value of the Space variable? How does it affect our exploit?

3-7. How does this exploit exit? Why is this necessary to specify?

Search for this exploit on the Internet to answer these questions:

3-8. What is the CVE number for this exploit?

3-9. From the exploit page on the National Vulnerability Database (NVD), what is the CVSS base score for this exploit? Why?

Let switch back to our meterpreter shell. Use *help* to see available meterpreter commands. Using these commands, answer these questions:

3-10. What user are you?

3-11. Which directory are you in initially?

3-12. What is the OS kernel and build version?

3-13. What is the name and process ID (PID) of the process you are currently executing in?

3-14. How many network interfaces does the victim have? What is the MAC address of the one you are connected to?

3-15. What error message do you get when you attempt to access admin's user directory? (C:\Users\admin)

Next, we will attempt to break into the admin directory. To do this, we will need to escalate our privileges on this machine. Meterpreter comes with a handy command called **getsystem** to attempt to become the SYSTEM user on Windows (equivalent to root on Linux).

3-16. What happens when we run this command? Why did this happen?

Since we need another method to escalate privileges, we will need to try some other exploits. On the meterpreter console, use the command **background** to temporarily background the session. This should bring you back to msfconsole. Note the number of the session.

Metasploit comes with a handy module that can do some reconnaissance for us and suggest possible exploits that could work for privilege escalation. To use it, type: `use post/multi/recon/local_exploit_suggester` on the msfconsole.

Check the **info** for this module and note its options. Minimally, we need to give it our open meterpreter session, so type: `set SESSION <SESSION-NUMBER>`, where `<SESSION-NUMBER>` is the number you noted from earlier.

Run the module with **exploit**. This should give you a suggested list of exploits to try. These exploits will differ based on the architecture you are running your VM on. Try several of these until you find one that works by using the commands you have learned so far (**use**, **info**, **options**, **set**, **exploit**, etc). A successful exploit should open another meterpreter session.

Use your new elevated session to answer these questions:

3-17. What user are you?

3-18. Which privilege escalation method did you use?

3-19. Which process are you now in? (name and PID). Does it make sense to stay in this process? Why or why not?

Meterpreter has another convenient command called **hashdump** to obtain account names and password hashes that are stored in the Windows registry (similar to `/etc/passwd` and `/etc/shadow` on Linux). Use this command to obtain the hash for the admin user and crack it. The format of the hashdump output is:

```
username:UID:LM Hash:NTLM Hash:::
```

Use this information to answer these questions:

3-20. How did you crack the password?

3-21. What is admin's password?

3-22. What is admin's secret? (found in admin's documents)