Lab 6: Network Security

Jonathan Harijanto

February 22, 2017

CS 373: Defense Against the Dark Arts - Winter 2017

Abstract

The purpose of this blog is to describe my observation from the interactive labs provided by Intel team. This blog will cover the testing methodology and the knowledge that was gain from completing the lab. Also, this blog has a dedicated section for the answers to challenge questions.

I. Blog

A. What you looked at:

The topic for this week is Network Security. I studied about new terminologies and technologies (tools) used in securing the network. Also, there are two lab assignments that we have to do this week. The first one was about deriving intelligence from packet meta-data. The tools required for this lab was Sublime – a text editor, to modify the given Python script. Essentially, I had to rewrite the script to extract specific information from a packet data (.csv file). The second lab was about deriving intelligence from captured packet content. The tools I used were Wireshark – a network protocol analyzer and Truecrypt – a disk encryption & decryption software. In this lab, I must explore the packet capture in Wireshark to be able to find some sensitive information.

B. How you looked at it & What you found:

1) Notes:

In this blog, Lab 1 is the Python script exercise, and Lab 2 is the Wireshark.

2) Lab 1:

In this lab, I was provided with a Python script that can do for-loop to count how many IP address used in particular protocol number. For example, the script could print something like `IP Protocol 17: 59995` which tells 59995 IP addresses used protocol number 17 (UDP). The first question asked in Lab 1 was to determine whether these port numbers are TCP or UDP. Since I couldn't bring the script out of the VM, I decided to provide the problem-solving process instead. These are the steps that I did for question 1:

1) I created two empty lists with size 1024 for UDP and TCP.
2) For each packet contained in the .csv file, I checked whether their destination port number is TCP or UDP. The checking can be done by using a simple If-Else clause.
3) For each packet with UDP destination port number, I added it to the UDP list. I did the same thing for TCP port number.
4) When my script reached the end of the page (.csv file), I wrote an if-statement to check the existence of `-stat` flag.
5) If the answer is yes, there is a new for-loop that goes through each list (TCP and UDP) and print the values.

An example that my script was written look something like `TCP 139 - 9455`. The output means that 9455 IP addresses used TCP port number 139. *Note: Please visit the next section entitled "Answer for Lab 1 and Lab 2" to see more about the outputs.*

The next question was to create a list of distinct IP addresses with their usage counts. These are the steps that I did for question 2:

1) I created a new list to contain all these IP addresses.
2) For each packet contained in the .csv file, I grabbed only the destination IP address and stored it to the list.
3) I used a Python module called counter to count the distinct IP and print those addresses descending order using `most_common()`.
4) I wrote an if-statement to check the existence of `-countip` flag.
5) If the answer is yes, there is another new for-loop that goes through the list (TCP and UDP) and print the contents.

Based on the output generated by my script, a destination network number `10.5.63.230` dominated the traffic in R.csv file, and a destination network number `192.245.12.221` dominated the traffic in O.csv file. *Note: Please visit the next section entitled "Answer for Lab 1 and Lab 2" to see more about the outputs.*

In question three, I had to display the IP addresses that used GRE (Generic Routing Encapsulation), IPSEC (Internet Protocol Security), and OSPF (Open Shortest Path First). Below are the steps to answer this problem:

1) I did research to find the protocol number for the three of them. Finally, I was able to know that GRE, IPSec, and OSPF number are 47, 50, 89.
2) Starting off with GRE, I added an If-statement to check whether the protocol number is equal to 47 right before my script grabbed the destination IP.
3) If the answer is yes, then that destination IP can be appended to the list.
4) I recorded the result and repeated the process by replacing the number 47 in the If-statement with 50 (IPSec) and 89 (OSPF).

I find it interesting to know that the R.csv file didn't have any IP that used GRE, IPSec, or OSPF. However, in O.csv file, I could see that most of the IP are using IPSec protocol. *Note: Please visit the next section entitled "Answer for Lab 1 and Lab 2" to see more about the outputs.*

For the final question, I had to count the number of packets sent to each service on the network. In other words, I have to be able to count how many distinct source IP are connecting to one particular destination IP. In addition, I should also provide the protocol number being used by the IP of origin. This problem was tough because I had to use a dictionary that contains a list of tuples. I was scared that things are going to get messy real quick when I had used three data structures. However, I still did it anyway, and these are the steps:

1) I created a new dictionary and assigned a list inside of it.
2) For each packet in the .csv file, I checked whether the packet had a TCP or UDP protocol number.
3) I set each destination IP as the dictionary key and append a tuple containing the source IP and protocol number as the key value.
4) I used print manipulation technique to generate the dictionary output in descending order.

Based on the output yielded from my script, 4800 distinct IP sources connected to `10.5.63.22`. *Note: Please visit the next section entitled "Answer for Lab 1 and Lab 2" to see more about the outputs.*

3) Lab 2:

The second lab was quite tricky because I need to use Truecrypt to unlock the questions and the packet captures. Regarding the structure of the questions, this lab has three different topics which are called 'Hunting a Spy', 'Tracking a Spy', and 'Hacked'. In my opinion, this lab was fun because we had to perform a forensic investigation to obtain some sensitive data.

The lab started with the 'Hunting a Spy' case. A bit about the background story, there were two people, Greg and Betty, contacting each other to set up a meeting. Then, as a secret agent, I had to sniff the given packet capture and figure out when's their meeting time. The first thing that I did after my Wireshark opened the packet successfully was to find any string related to the word "Betty" or "Greg". I did this because I was hoping that their conversation will somehow appear in a packet. To find these strings, I used Wireshark's 'Find Package' feature, chose 'Search by String' in a 'Packet Byte', and typed 'Betty' as the filter word.

I was surprised to see that Wireshark managed to find 'Betty' string and pointed me to an individual packet data with IP source `46.165.193.136` and IP destination `172.29.1.55`. The next step I did was to use 'Follow TCP Stream' feature, and I saw a short conversation in the stream like the following:

```
:betty!PRIVMSG #S3cr3tSp0t : Hi Greg :)
PRIVMSG #S3cr3tSp0t: Hi Betty, what day do you want to meet up?
```

After these two sentences, the rest of the message was in hex format like `&#x41&#62&#52`. Thus, I opened my browser to search for HEX-ASCII online converter. After I successfully decoded all the hex, here's the continuation of their conversation:

```
:betty! PRIVMSG #S3cr3tSp0t: How does Wednesday sound?
PRIVMSG #S3cr3tSp0t: Great :) what time?
```

```
:betty! PRIVMSG #S3cr3tSp0t: ah 2pm
PRIVMSG #S3cr3tSp0t: Ok, I can't wait!
```

From this message, it is evident now that they were planning a meeting Wednesday.

In the next question, the story continues when Greg and Betty were planning another meeting in a classified location. As a secret agent, I had to sniff another packet capture to locate their meeting point (city). I handle the problem by applying the same technique which was used 'Find Package' feature and searched a relevant string. For the second time, I typed 'Betty' again as the filter word, and Wireshark still managed to point me to an individual packet data that contain the string "betty_swindoll". I followed that packet stream and discovered a huge chunk of text written both ASCII and hex like `I\%20wanted\%20to\%20let`. Honestly, that content was horrid to be read.

Anyhow, thanks to Wireshark search pointer, I managed to find a hidden email from `betty_swindoll@aol.com` in the stream. The summary from the email was Betty apologized to Greg for not showing up at the previous meeting. She did that because she wanted to make sure that Greg didn't bring any friend. However, she wanted to meet him again, and she gave Greg a password that will lead him to the next meeting point which was `S3cr3tVV34p0n`.

After I had known the secret password, I decided to locate the encrypted file. I used the 'Find Package' feature and searched for 'PRIVMSG' because it appeared quite often in the previous case. As I suspected, I found a chat history in the packet stream that looks like:

```
NOTICE D34thM3rch4nt :DCC Send r3nd3zv0us (172.29.1.50)
PRIVMSG D34thM3rch4nt :.DCC SEND r3nd3zv0us 2887582002 1024 819200
```

After I read this, I had a hunch that that `S3cr3tVV34p0n` was the password required to unlock this `r3nd3zv0us` file.

Next, I examined the packet that contained the following content `PRIVMSG D34thM3rch4nt:.DCC SEND r3nd3zv0us 2887582002 1024 819200` by analyzing this packet's neighbors. In the end, I was able to discover `r3nd3zv0us` file by extracting a packet that's located literally below the examined packet. The reason I was so confident that I found `r3nd3zv0us` because the packet's conversation size was 820128 bytes, which is unusual for an ordinary conversation. After I had finished extracting the file, I used Truecrypt to decrypt the file and answered the passphrase with the word `S3cr3tVV34p0n`. Inside this file, I saw a photo of "Welcome to Fabulous Las Vegas Nevada" sign. Hence, it is evident now that they were going to meet in Las Vegas.

The background story for the last question was our friend successfully hacked Betty's email account and accidentally opened an embedded malware. As a good friend, we had to assist him to find the size of the malware. The first thing that I did when I heard the word email was to look for HTTP protocol in the packet capture. I noticed four packet data used HTTP protocol and only two of them used GET request. The first request was to a file called `paimia&amp;am` and the other one was for `favicon.ico`. I never heard the word 'paimia', I determined to google it and saw a match in a malware detector website. Since I already knew that 'paimia' is malware, I decided to use 'HTTP Export Object' feature to see the size of every HTTP file available in the packet capture. I selected 'paimia' file and the number (file size) shown by the downloader was 3113 bytes.

II. Answer for Lab 1 and Lab 2

A. Lab 1: Python Script

1) Characterize the main functions on each well-known destination port numbers for TCP and UDP. What kind of a network is it?:

R.csv Data				
Port Number	Number of Output	Service Name	Service Usage	Type of Network
TCP 139	9455	NETBIOS Session Service	File and printer sharing	Work
TCP 80	1361	Hyper Text Transfer Protocol	Web traffic purposes	Work, Home
TCP 110	990	Post Office Protocol Version 3	Retrieve email from a remote service	Work
TCP 22	448	Secure Shell	Log in to a remote machine and execute commands	Data Center
UDP 53	428	Domain Name Service	Domain name resolution	Data Center

O.csv Data				
Port Number	Number of Output	Service Name	Service Usage	Type of Network
TCP 25	211205	Simple Mail Transfer Protocol	Sending and receiving email	ISP
TCP 80	156397	Hyper Text Transfer Protocol	Web traffic purposes	Work, Home
TCP 22	26383	Secure Shell	Log in to a remote machine and execute commands	Work
UDP 53	21563	Domain Name Service	Domain name resolution	Data Center
TCP 445	10867	Microsoft-DS Active Directory	Direct TCP/IP MS Networking access without the need for a NetBIOS layer	Work

2) Attempt to determine the network number (network prefix) that seems to dominate the traffic.:

In R.csv, the network number that dominates the traffic is `10.5.63.230`. That particular network number was being used 43338 times. On the other hand, in O.csv, the network number that dominates the traffic is `192.245.12.221`. I observed that network number was being used 169634 times.

3) In R.csv, generate sorted output from -countip for the IP protocols to identify all the IP addresses that use GRE, IPSEC, and OSPF:

IP address that use GRE	
Network Number	Number of Output
198.182.113.9	1397
209.104.16.215	1170
66.134.158.90	30
209.104.16.58	29

IP address that use IPSEC	
Network Number	Number of Output
198.182.113.1	690
207.182.35.50	350
128.196.69.2	285
209.104.16.119	34
151.193.130.121	34
192.70.160.132	26
207.182.45.254	23
207.182.45.153	15
207.182.45.178	12
12.9.142.163	7
204.17.35.131	6
207.182.36.166	1
216.133.8.30	1

IP address that use OSPF	
Network Number	Number of Output
207.182.45.58	8
207.182.45.49	6
207.182.45.50	4
207.182.45.60	2
207.182.45.47	2
207.182.45.55	2

4) In R.csv, run your -connto option, return the top 20 servers in:

Top 20 servers from connto output (R.csv Data)
ipdst 10.5.63.22 has 4895 distinct ipsrc on ports: 139, 23
ipdst 10.5.63.27 has 3400 distinct ipsrc on ports: 113, 137, 139
ipdst 10.5.63.6 has 1511 distinct ipsrc on ports: 110, 22, 25, 53
ipdst 10.5.63.7 has 681 distinct ipsrc on ports: 135, 137, 138, 139, 721, 80
ipdst 10.5.63.11 has 417 distinct ipsrc on ports: 137, 139
ipdst 10.5.63.28 has 301 distinct ipsrc on ports: 712
ipdst 32.97.225.112 has 293 distinct ipsrc on ports: 80
ipdst 209.67.181.11 has 268 distinct ipsrc on ports: 80
ipdst 10.5.63.18 has 239 distinct ipsrc on ports: 891
ipdst 209.67.181.20 has 133 distinct ipsrc on ports: 80
ipdst 10.5.63.255 has 126 distinct ipsrc on ports: 137, 138
ipdst 10.5.63.8 has 125 distinct ipsrc on ports: 515
ipdst 10.5.63.17 has 120 distinct ipsrc on ports: 137, 139
ipdst 208.10.192.175 has 117 distinct ipsrc on ports: 80
ipdst 10.5.63.200 has 111 distinct ipsrc on ports: 139, 80
ipdst 10.5.63.1 has 103 distinct ipsrc on ports: 113, 53
ipdst 10.5.63.230 has 91 distinct ipsrc on ports: 137, 138, 139
ipdst 193.164.170.30 has 91 distinct ipsrc on ports: 110
ipdst 208.10.192.202 has 83 distinct ipsrc on ports: 80
ipdst 216.101.171.2 has 80 distinct ipsrc on ports: 110

5) In R.csv, run your -connto option and identify the web servers, the printers, the mail servers, the DNS servers:

According to the internet, port TCP/139 and TCP/137 are dedicated for NetBIOS service. NetBIOS is a protocol used for file and print sharing. Thus, an IP address that is/ are connected to this port is a **printer**. Based on the R data, the following network numbers used either TCP/139 or TCP/137: `10.5.63.11`, `10.5.63.22`, `10.5.63.27`, `10.5.63.7`, `10.5.63.255`, `10.5.63.17`, and `10.5.63.230`.

Next, port TCP/25 is dedicated for Simple Mail Transfer Protocol service. It is a protocol used for sending and receiving email. Thus, an IP address that is/ are connected to this port is a **mail server**. Based on the R data, the following network numbers used TCP/25: `10.5.63.6`.

Port UDP/53 is dedicated for Domain Name System service. It is a protocol used for domain name resolution. Thus, an IP address that is/ are connected to this port is a **DNS server**. Based on the R data, the following network numbers used UDP/53: `10.5.63.6` and `10.5.63.1`.

Lastly, port TCP/80 is dedicated for Hypertext Transfer Protocol service. It is a protocol used for web service or web traffic purposes. Thus, an IP address that is/ are connected to this port is a **web server**. Based on the R data, the following network numbers used TCP/80: `10.5.63.7`, `32.97.225.112`, `209.67.181.11`, `209.67.181.20`, `208.10.192.175`, `10.5.63.200`, and `208.10.192.202`.

B. Lab 2: Wireshark

1) Hunting a Spy - What day of the week is the meeting scheduled for? :

Wednesday

2) Tracking a Spy - What city are they meeting?:

Las Vegas, NV

3) Hacked - How many bytes of data is the malicious payload? :

3113 bytes

III. Conclusion

I learned a lot of new things related to network security this week. First, I acquired some new tricks to get more information from a packet capture in Wireshark. Furthermore, I also had a chance to sharpen my Python skill by manipulating the given script. Last but not least, I learned new terminologies and concept about securing a network. Honestly, this week course has successfully drawn my interest to learn more about network security.