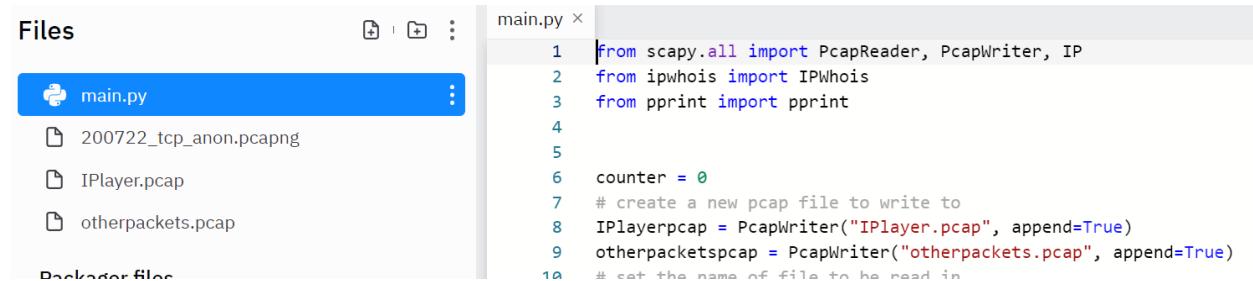


Submit detailed lab report with screenshots of results, code snippets, and description of what has been done and observed. Submit any time before Wednesday, Feb. 16, 11:59pm NY time.

Task 1 – Filter Pcap capture to select packets in Python/Scapy

I copied and pasted the existing line of code in “line 8” and modified the file name output into two separate files, titled “IPlayer.pcap” and “otherpackets.pcap”



The screenshot shows a code editor interface with a sidebar labeled "Files". Inside the sidebar, there is a blue-highlighted file named "main.py" and other files like "200722_tcp_anon.pcapng", "IPlayer.pcap", and "otherpackets.pcap". The main editor area contains the following Python code:

```
main.py x
1 from scapy.all import PcapReader, PcapWriter, IP
2 from ipwhois import IPWhois
3 from pprint import pprint
4
5
6 counter = 0
7 # create a new pcap file to write to
8 IPlayerpcap = PcapWriter("IPlayer.pcap", append=True)
9 otherpacketspcap = PcapWriter("otherpackets.pcap", append=True)
10 # set the name of file to be read in
```

Below this, another section of code is shown:

```
12 for packet in PcapReader(filename):
13     try:
14         # if an IP layer exists, keep that packet
15         if packet[IP]:
16             # write the packet to new pcap file
17             IPlayerpcap.write(packet)
18             #contains IP layer packets
19         else:
20             otherpacketspcap.write(packet)
```

List items a) and b) describe the logic of the code:

- a) Lines 12 through 17 puts all packets with IP layers in IPlayerpcap file
- b) Else command then writes other packets to otherpacketspcap file, we don't need to clarify contents since the first if statement will remove all packets with IP layers for us and we might not get any packers In the second file since it is possible that there are all the packets have IP layers

Task 2 – Analysis with Python and tools (IP addresses)

- a) Within the if statement for packet[IP] I was able to append each packet destination using an array function and the append .dst function. The list output is shown on the right hand side.
- b) Using pprint(len(IP_dst)) I was able to get the total IP destinations which was a total of 101.

```
#IP locations addresses
print("Destination")
pprint(IP_dst)

# Length
print("Length")
pprint(len(IP_dst))

# use pprint to see a formatted version of the result
print("\n\npretty printing the entire ipwhois result:")
#pprint(result)

# use IPWhois module
whoisinfo = IPWhois("8.8.8.8")
result = whoisinfo.lookup_rdap(depth=1)

# find out all the available keys in the dictionary returned as a result
print("keys in the ipwhois results:\n")
for key in result.keys():
    print(key)

    # prints contents within the asn key
```

	'10.106.81.220', '10.106.81.220', '10.106.81.220', '10.106.81.220', '23.216.84.240', '10.106.81.220', '68.67.161.208', '10.106.81.220', '68.67.161.208', '68.67.161.208', '68.67.161.208', '10.106.81.220', '10.106.81.220', '68.67.161.208', '68.67.161.208', '68.67.161.208', '68.67.161.208', '68.67.161.208', '68.67.161.208', '68.67.161.208', '68.67.161.208', '68.67.161.208', '10.106.81.220', '10.106.81.220', '10.106.81.220', '10.106.81.220']
Length	101

- c) Using uniqueIP = set (IP_dst), I was able to find the total number of unique IP addresses, the total number was 18, I made sure to use set to eliminate duplicate IP addresses from the list.

```
#unique IP 2
uniqueIP = set(IP_dst)
print("Unique IP")
print(len(uniqueIP))
```

- d) Using whoisinfo I was able to find out that 1 of the 18 unique IP addresses was from Great Britain and the rest are US based, the date range for the IP addresses spanned from 03-14-2007 to 09-24-2017 being the latest.

```

24     except:
25         pass
26     if counter > 100: # limit the number of packets saved
27         break
28
29     #unique IP 2
30     uniqueIP = set(IP_dst)
31     print("Unique IP")
32     print(len(uniqueIP))
33
34     # use IPWhois module for unique IPs task 2 part d
35     for ip in uniqueIP:
36         try:
37             whoisinfo = IPWhois(ip)
38             result = whoisinfo.lookup_rdap(depth=1)
39             print(result["asn_country_code"])
40             print(result["asn_date"])
41         except:
42             pass
43
44     #IP locations addresses
45     print("Destination")
46     pprint(IP_dst)
47
48
49     # Length
50     print("Length")
51     pprint(len(IP_dst))
52
53     # use pprint to see a formatted version of the result

```

Unique IP	Country	Date
18	US	2014-03-28
US	US	2014-10-23
US	US	2009-02-12
US	US	2011-05-16
US	US	2013-08-26
US	US	2007-03-14
US	US	2013-07-12
GB	GB	2001-03-21
US	US	2013-07-12
US	US	2017-09-29
US	US	2017-09-29
US	US	2016-09-12

Task 3 – Analysis with Python and tools (other packets)

Nothing exist inside other packets file.