Student Name : Kiew Ten Wei

Group : SCS2

Date : 22/03/2024

LAB 4: ANALZING NETWORK DATA LOG

You are provided with the data file, in .csv format, in the working directory. Write the program to extract the following informations.

EXERCISE 4A: TOP TALKERS AND LISTENERS

One of the most commonly used function in analyzing data log is finding out the IP address of the hosts that send out large amount of packet and hosts that receive large number of packets, usually know as TOP TALKERS and LISTENERS. Based on the IP address we can obtained the organization who owns the IP address.

List the TOP 5 TALKERS

Rank	IP address	# of packets	Organisation
1	193.62.192.8	3041	EUR-BIO-INST
2	155.69.160.32	2975	NTUNET1
3	130.14.250.11	2604	NLM-ETHER
4	14.139.196.58	2452	NKN-IIT-GUW
5	140.112.8.139	2056	T-NTU.EDU.TW-NET

TOP 5 LISTENERS

Rank	IP address	# of packets	Organisation
1	103.37.198.100	3841	A-STAR-AS-NP
2	137.132.228.15	3715	NUSNET
3	202.21.159.244	2446	RPNET
4	192.101.107.153	2368	PNNL
5	103.21.126.2	2056	IITB-IN

EXERCISE 4B: TRANSPORT PROTOCOL

Using the IP protocol type attribute, determine the percentage of TCP and UDP protocol

	Header value	Transport layer protocol	# of packets
1	6	TCP	56064
2	17	UDP	9462
3			

EXERCISE 4C: APPLICATIONS PROTOCOL

Using the Destination IP port number determine the most frequently used application protocol.

(For finding the service given the port number https://www.adminsub.net/tcp-udp-port-finder/)

Rank	Destination IP port number	# of packets	Service
1	443	13423	HTTPS
2	80	2647	HTTP
3	52866	2068	Dyanamic/Private Ports
4	45512	1356	Unassigned
5	56152	1341	Dynamic/Private Ports

EXERCISE 4D: TRAFFIC

The traffic intensity is an important parameter that a network engineer needs to monitor closely to determine if there is congestion. You would use the IP packet size to calculate the estimated total traffic over the monitored period of 15 seconds. (Assume the sampling rate is 1 in 2048)

Total Traffic(MB)	126519.18359375

EXERCISE 4E: ADDITIONAL ANALYSIS

Please append ONE page to provide additional analysis of the data and the insight it provides. Examples include:
Top 5 communication pairs;
Visualization of communications between different IP hosts;
etc.

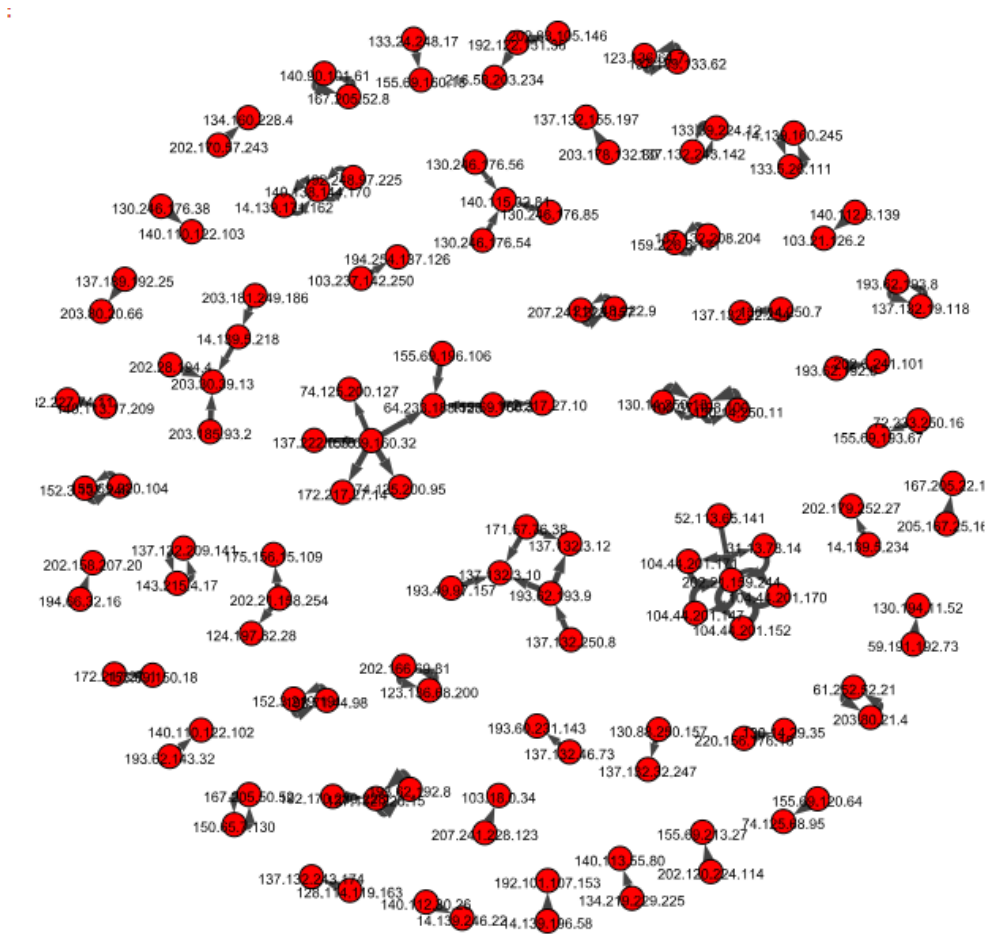
Please limit your results within one page (and any additional results that fall beyond one page limit will not be assessed).

Top 5 Communication Pair

Index	Source_IP	Destination_IP	Number of Packets
1	193.62.192.8	137.132.228.15	3041
2	130.14.250.11	103.37.198.100	2599
3	14.139.196.58	192.101.107.153	2368
4	140.112.8.139	103.21.126.2	2056
5	137.132.228.15	193.62.192.8	1910

Visualization of communications between different IP hosts

We can now analyze and comprehend network connectivity patterns and traffic intensity, using notebook, it shows a directed graph representing connections between source and destination IP addresses, and visualizes it. Based on the data, it assigns vertices and edges attributes like size, label, and width. The picture that is produced helps locate important nodes in the network and spot any possible congestion or unusual activity.



EXERCISE 4F: SOFTWARE CODE

Please also submit your code to the NTULearn lab site.

Lab 4

Kiew Ten Wei
U2221627A

```
import numpy as np
import pandas as pd
import math
import whois
from ipwhois import IPWhois
from ipwhois.exceptions import IPDefinedError
import networkx as nx
import matplotlib.pyplot as plt
from igraph import Graph, plot
import cairo
import networkx as nx
import matplotlib.pyplot as plt
%matplotlib inline
import mpld3
mpld3.enable_notebook()
from scapy.all import *
from mpl_interactions import ioff, panhandler, zoom_factory
```

```
SFlow_att_cat = ['Type', 'flow_agent_address', 'inputPort', 'outputPort', 'scr_MAC', 'dst_MAC', 'ethernet_type', 'in_vlan', 'out_
data = pd.read_csv('Data_3.csv', names=SFlow_att_cat)
data = data.drop(data.columns[-1], axis=1)
data.head(10)
```

	Type	flow_agent_address	inputPort	outputPort	scr_MAC	dst_MAC	ethernet_type	in_vlan	out_vlan	src_IP	dst_IP	IP_protocol
0	FLOW	203.30.38.251	137	200	d404ff55fd4d	80711fc76001	0x0800	919	280	130.246.176.22	140.115.32.81	6
1	FLOW	203.30.38.251	129	193	609c9f851b00	0031466b23cf	0x0800	11	919	155.69.160.32	64.233.188.128	6
2	FLOW	203.30.38.251	137	200	d404ff55fd4d	80711fc76001	0x0800	919	280	130.246.176.53	140.115.32.83	6
3	FLOW	203.30.38.251	129	135	609c9f851b00	002688cd5fc7	0x0800	11	919	155.69.160.32	54.169.174.79	17
4	FLOW	203.30.38.251	130	199	00239cd087c1	544b8cf9a7df	0x0800	919	600	137.132.228.15	193.62.192.8	6
5	FLOW	203.30.38.251	129	135	609c9f851b00	002688cd5fc7	0x0800	11	919	155.69.160.32	54.255.221.151	17
6	FLOW	203.30.38.251	130	199	00239cd087c1	544b8cf9a7df	0x0800	919	600	137.132.250.8	193.62.193.9	6
7	FLOW	203.30.38.251	137	200	d404ff55fd4d	80711fc76001	0x0800	919	280	193.61.196.206	140.110.147.170	6
8	FLOW	203.30.38.251	200	3	80711fc76001	00235ed9b680	0x0800	280	32	137.189.133.62	123.136.64.7	6
9	FLOW	203.30.38.251	199	130	544b8cf9a7df	00239cd087c1	0x0800	600	919	193.62.192.8	137.132.228.15	6

```
def get_organization(ip_add):
    ip = IPWhois(ip_add)
    result = ip.lookup_rdap()
    return result.get('network', {}).get('name')
```

Top 5 Talker

```
top_five_talk = {}
for talker in data['src_IP']:
    if talker not in top_five_talk:
        top_five_talk[talker] = 1
    else :
        top_five_talk[talker] = top_five_talk[talker] + 1

talkerData = pd.DataFrame(top_five_talk.items(), columns=['src_IP', 'count'])
talkerData = talkerData.sort_values(by='count', ascending=False)

print("\tSource IP\t\tCount\t\tWhois:")
for num in range(1,6):
    organization = get_organization(talkerData.iloc[num-1]['src_IP'])
    print(str(num) + '\t' + talkerData.iloc[num-1]['src_IP'] + "\t\t" + str(talkerData.iloc[num-1]['count']) + "\t\t" + organizat
```

```
        Source IP               Count           Whois:
1       193.62.192.8            3041            EUR-BIO-INST
2       155.69.160.32           2975            NTUNET1
3       130.14.250.11           2604            NLM-ETHER
4       14.139.196.58           2452            NKN-IIT-GUW
5       140.112.8.139           2056            T-NTU.EDU.TW-NET
```

Top 5 Listeners

```
top_five_list = {}

for listener in data['dst_IP']:
    if listener not in top_five_list:
        top_five_list[listener] = 1
    else :
        top_five_list[listener] = top_five_list[listener] + 1

listData = pd.DataFrame(top_five_list.items(), columns=['dst_IP', 'count'])
listData = listData.sort_values(by='count', ascending=False)

print("\tDestination IP\t\tCount\t\tWho Is")
for num in range(1,6):
    organization = get_organization(listData.iloc[num-1]['dst_IP'])
    print(str(num) + '\t' + listData.iloc[num-1]['dst_IP'] + "\t\t" + str(listData.iloc[num-1]['count'])+ "\t\t" + organization)
```

	Destination IP	Count	Who Is
1	103.37.198.100	3841	A-STAR-AS-AP
2	137.132.228.15	3715	NUSNET
3	202.21.159.244	2446	RPNET
4	192.101.107.153	2368	PNNL
5	103.21.126.2	2056	IITB-IN

Top 5 Application

```
app_count = {}

for app in data['udp_dst_port/tcp_dst_port/icmp_code']:
    if app not in app_count:
        app_count[app] = 1
    else :
        app_count[app] = app_count[app] + 1

app_data = pd.DataFrame(app_count.items(), columns=['udp_dst_port/tcp_dst_port/icmp_code', 'count'])
app_data = app_data.sort_values(by='count', ascending=False)

print("\tapp\t\tCount")
for num in range(1,6):
    print(str(num) + '\t' + str(app_data.iloc[num-1]['udp_dst_port/tcp_dst_port/icmp_code']) + "\t\t" + str(app_data.iloc[num-1]
```

	app	Count
1	443	13423
2	80	2647
3	52866	2068
4	45512	1356
5	56152	1341

Total Traffic

```
total_traffic = 0
for packetSize in data['IP_size']:
    total_traffic += packetSize
total_traffic=total_traffic*2048/ (2**20)
print("Total traffic (MB): " + str(total_traffic))
```

```
Total traffic (MB): 126519.18359375
```

Proportion of TCP and UDP packet

```
no_of_proc = data['IP_protocol'].value_counts()
tcp_count = no_of_proc[6]
udp_count = no_of_proc[17]

tcp_percentage = tcp_count / len(data.index) * 100
udp_percentage = udp_count / len(data.index) * 100

print(f"TCP packet percentage: {tcp_count}, ({tcp_percentage}%)")
print(f"UDP packet percentage: {udp_count},({udp_percentage}%)")
```

```
TCP packet percentage: 56064, (80.81879775118928%)
UDP packet percentage: 9462,(13.639901974917112%)
```

Top 5 communication pair

```
# Top 5 unique communication pairs
pairs = data.groupby(['src_IP', 'dst_IP']).size().sort_values(ascending = False).to_frame()
pairs.columns = ['No. of Packets']
pairs = pairs.reset_index()
print(pairs['src_IP'])
top_comm_df = pairs[:5]
top_comm_df
```

```
0                                193.62.192.8
1                                130.14.250.11
2                                14.139.196.58
3                                140.112.8.139
4                                137.132.228.15
                          ...
6584                             155.69.193.14
6585                            155.69.193.139
6586                            155.69.193.115
6587                            155.69.193.112
6588    fe80:0000:0000:0000:b2a8:6e03:ca76:6716
Name: src_IP, Length: 6589, dtype: object
```

	src_IP	dst_IP	No. of Packets
0	193.62.192.8	137.132.228.15	3041
1	130.14.250.11	103.37.198.100	2599
2	14.139.196.58	192.101.107.153	2368
3	140.112.8.139	103.21.126.2	2056
4	137.132.228.15	193.62.192.8	1910

```
print("\tSrc IP\t\tSrc Org\t\tDest IP\t\tDest Org")

for i in range(1,6):
    organisation = get_organization(top_comm_df.iloc[i - 1]['src_IP'])
    organisation2 = get_organization(top_comm_df.iloc[i - 1]['dst_IP'])
    print(str(i) + '\t' + top_comm_df.iloc[i - 1]['src_IP'] + "\t\t"+ organisation +"\t\t" + top_comm_df.iloc[i - 1]['dst_IP'] +
```

```
        Src IP          Src Org         Dest IP         Dest Org
1       193.62.192.8            EUR-BIO-INST            137.132.228.15          NUSNET
2       130.14.250.11           NLM-ETHER               103.37.198.100          A-STAR-AS-AP
3       14.139.196.58           NKN-IIT-GUW             192.101.107.153         PNNL
4       140.112.8.139           T-NTU.EDU.TW-NET                103.21.126.2            IITB-IN
5       137.132.228.15          NUSNET          193.62.192.8            EUR-BIO-INST
```

Visualising the communication between different IP hosts

```
graph_series = data.groupby(["src_IP", "dst_IP"])\
             .size()\
             .nlargest(100)

graph_df = graph_series.to_frame().reset_index()
tuple_list = [tuple(edge) for edge in graph_df.head(n=100).to_numpy()]

g = Graph.TupleList(tuple_list, directed=True, weights=True)

g.vs["size"] = 15
g.vs["label_size"] = 9
g.vs["label"] = g.vs["name"]

g.es["width"] =  [max(math.log(weight)/1.5, 0) for weight in g.es['weight']]
g.es["arrow_size"] = [width / 5 for width in g.es["width"]]

plot(g, "network_graph.svg", bbox=(600,600))
```