COMPETITION IN NETWORK TRAFFIC ANALYSIS

Antonio Capone, Matteo Cesana, Francesco Musumeci, Achille Pattavina,
Elisabetta Di Nitto, Armin Okic, Bin Xiang
Politecnico di Milano
Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB)

Contents

1 Introduction **3**

2 Videos **3**

3 Competition details and objectives. **4**

3.1 Submission list 5

4 Parsing .pcap file **5**

5 Dataset **6**

6 Coding hints **6**

6.1 Libraries 7

6.2 Useful materials 8

7 Contact **8**

1 Introduction

This document is created in order to help students for the network data traffic analysis competition. It is organized to provide details about the competition objectives and challanges, together with datasets details used for the competition and as well to give useful coding hints for students. Aside from this document, two videos are created which are explaining structure of the competition and giving tutorial on how to extract the network data traffic from the Linux network interface, analyze and parse it with Python code and extract simple statistics. In both cases, Python code samples are provided, which can be used as a starting point towards the final solution. This competition is putting the main emphasis on the network data traffic analysis, while placing in a second plane the coding skills. For that reason, we have included a "Coding hints" section into the documentation file and provided a set of example Python codes.
If you are planning to participate to the competition it is necessary to register on the following link: `https://forms.gle/BUze1ZR5vFPzXrTS9`

2 Videos

To enrich the description of the competition and as well to better help students to fulfill the goals of a competition, we have additionally provided two videos explaining in details specific things about network traffic extraction, processing, filtering and representing. Here is a short description of two videos:

- Video 1: `https://youtu.be/Nh5KQLsb8mc` - **Competition details and datasets** - This video is created in order to provide more details about the dataset used in this competion, mainly focusing on the purposes and goals of the competition.
- Video 2: `https://youtu.be/yzE_i6-1hjg` - **How to parse .pcap file with Python?** - This video is explaining how to extract the .pcap file from a network interface, how you can parse an extracted .pcap file using Python and how to get simple statistics.

3 Competition details and objectives.

The competition is focusing on the network data traffic analysis using Python. The main goal of the competition is for students to improve their knowledge about networking, network traffic analysis and as well to improve and enrich coding skills.
The competition is structured in two parts:

1. **Analysis of the provided network traffic stored in .pcap file.** This part is composed of two tasks:
   - **Mandatory** task for all groups is to create a requested subset of statistics and export them into specific format. This task is used as a threshold for deciding if the group can approach to the next steps of the competition.
   - **Creative** task is again including extraction of statistics from the provided dataset, but giving to students much more freedom of exploring the dataset. For this task, groups will need to create at least two and maximum up to five statistics, not done before.
2. **Presentation of the performed analysis.** The evaluation committee will select the best projects (in the range of 15-20 max) and will invite their authors to present their work. Each group will have 5 minutes SHARP for the presentation followed by questions. The presentation will be given in English, possibly by both team members, and will focus on the presenting the creative part and on discussing the results achieved with the analysis.

As mentioned previously the analysis is done in two steps. For the first mandatory phase, the following tasks are required:

1. Sort the **standardized** ports based on the total amount of traffic generated. Header format of the resulting .csv file: "port_number", "amount_of_traffic"
2. Find the top 10 IP addresses with highest amount of traffic generated and create a graph showing unique IPs (x-axis) and corresponding traffic (y-axis). Header format of the resulting .csv file: "ip_addr", "amount_of_traffic"
3. For top 10 IP addresses (generating the most traffic) find the most used protocol, most used source and destination port. For each Header format of the resulting .csv file: "ip_addr", "amount_of_total_traffic", "protocol", "amount_of_traffic_for_specific_protocol", "source_port", "amount_for_spec_source_port", "destination_port", "amount_for_spec_destination_port"
4. Calculate min, max, average and variance values of time to live (TTL) values of whole traffic. Header format of the resulting .csv file: "min", "max", "average", "variance"

All resulting .csv files need to follow the required structure and the name of the file should be in the following form *task_N.csv*, where N corresponds to the number of a task.

The second step for students is to be creative and extract interesting network traffic statistics. To give directions for the further analysis: you can create heatmaps with IP addresses as points, distribution analysis and histograms, data representations in time etc.
The evaluation of the second task will be done based on following criteria:

- Creativity,
- Importance of a created statistic, from the **network** point of view,
- Representation of results,
- Originality and quality of the attached code[1].

3.1 Submission list

Here is the final list of resulting files that need to be submitted:

- First phase (mandatory tasks):
  1. Task: task_1.csv + python code
  2. Task: task_2.csv + graph + python code
  3. Task: task_3.csv + python code
  4. Task: task_4.csv + python code
- Second phase (creative tasks):
  1. Task: creative_task_1.* + python code
  2. Task: creative_task_2.* + python code . . .

  Where asterix ("*"), refers to which ever file format you decide to submit (e.g. .png, .csv, .txt etc.)

All of the listed files need to be stored into single .zip file. The resulting .zip file name needs to be in a following form: "NameOfAGroupLeader_LastNameOfAGroupLeader.zip" This file needs to be uploaded in the submission folder in a Beep competition web page.

4 Parsing .pcap file

In this part we are giving you an example of how to extract the network traffic from your network interface and parse it with Python. We have added the code and description of the code in the additional file, which is also explained in the second video. The link to access this file can be found here: `https://tinyurl.com/yylyosr3`.

[1] We will assess your project for code originality checking whether you have copied your code from external sources or from colleagues. We will assess it for code quality verifying that the code is well-structured and commented.

5 Dataset

This part is containing explainations of the datasets provided for the competition. Two datasets are attached for the competition:

- network_traffic.pcap - is the main dataset containing details about the network traffic that needs to be analyzed.
- ip_locations.csv - is the file containing details about locations of IP addresses from the .pcap file, which can be used to represent data on the map.

The network traffic dataset is extracted from the opensource .pcap trace and it is representing only a subset of the trace. The full dataset with coresponding details can be found here:

https://mcfp.weebly.com/the-ctu-13-dataset-a-labeled-dataset-with-botnet-normal-and-background-traffic.html

The dataset was captured from the main router in CVUT University (Prague, Czech Republic) by the Stratosphere Lab for research purposes. The payload of the original .pcap file is truncated to ensure not to break any privacy regulations. The .pcap file is truncated to have only following starting bytes of each packet: TCP-54 bytes, UDP-42 bytes, ICMP-66 bytes; this means that payload data is missing, but still all the relevant information for the competition purposes is maintained. The dataset is containing around 30 minutes of the traffic.
The IP locations dataset is assigning x and y coordinates to each IP address.

6 Coding hints

This part is organized in the way to help students to solve some of the coding problems and to give the ideas of statistics creation.

Timestamp

The timestamp of the packet is in the Unix time (seconds), which is representing the number of seconds from the fixed data January 1st 1970. It can be extracted from the packet in the following way:

```
from scapy.all import * #importing scapy library
import time #importing time lib

pcap_data = rdpcap('small_test_data.pcap') #reading the file
sessions = pcap_data.sessions #storing sessions
for session in sessions: #looping through sessions
    packet = sessions[session][0] #getting packet from session
#printing the formatted time
print(time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(packet.time)))
```

CSV read and write

Read:

```
import csv

```

```
with open("file.csv", mode = "r") as csv_file:
    csv_file_reader = csv.reader(csv_file, delimiter = ",")

    for line in csv_file_reader:
        print(line)
```

Write:

```
import csv
with open("file.csv", mode = "w") as csv_file:
    csv_file_writer = csv.writer(csv_file, delimiter = ",")
```

More details can be found here: `https://realpython.com/python-csv/`

6.1 Libraries

The coding envorionment is Python 3.6, with the set of libraries listed below.

Scapy Library

The parsing of .pcap file with Python can be done by the usage of several libraries, i.e. Scapy, dpkt, pyshark, libpcap etc. We decided to use Scapy library.

The installation of a Scapy library is very simple and it can be done by running the following command in terminal:

```
pip install scapy
```

Pandas library

Pandas is a opensource Python library used for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

The installation of a Pandas library is very simple and it can be done by running the following command in terminal:

```
pip install pandas
```

Short guide to how to use Pandas can be found here: `https://pandas.pydata.org/pandas-docs/stable/getting_started/10min.html`

Matplotlib

For the plotting of results we propose to use matplotlib Python library.

The installation of a Matplotlib library is very simple and it can be done by running the following command in terminal:[2]

```
pip install matplotlib
```

Official tutorials on how to use Matplotlib can be found here: `https://matplotlib.org/tutorials/index.html`

[2] If some problems are existing with the pip command, it is suggested to try with pip3 command instead.

6.2 Useful materials

How to use Scapy to change packet values of a .pcap file - https://www.youtube.com/watch?v=ADDYo6CgeQY
Scapy usage - https://itgeekchronicles.co.uk/2014/12/02/scapy-sessions-or-streams/
Using Scapy to extract the payload from a pcap file - https://medium.com/@vworri/extracting-the-payload-from-a-pcap-file-using-python-d938d7622d71
Scapy readpcap() usage - https://www.programcreek.com/python/example/103591/scapy.all.rdpcap
Pandas tutorials - https://www.tutorialspoint.com/python_pandas/index.htm

7 Contact

If you are having problem to understand some parts of the competition structure or understanding the provided codes, you are free to contact us via email:

- Armin Okic: armin.okic@polimi.it
- Bin Xiang: bin.xiang@polimi.it