Aalto-yliopisto
Aalto-universitetet
Aalto University

ELEC-E7130 - Internet Traffic Measurements and Analysis

Date: 22.10.2023
Name: Mohammed Al-Humairi
Student ID: 101708984
Email: mohammed.al-humairi@aalto.fi

Assignment 1. Basic programming and processing data

Task 1: Programming Tools

1. **What is the function of the command awk? How does the awk command work? Could you give at least three examples highlighting its usefulness?**

- **Function of awk:** awk is a versatile text processing tool in Unix and Unix-like operating systems. Its primary function is to process and manipulate text or data files. It can search for patterns within text, and when patterns are found, it can perform actions on the matched lines. It's particularly useful for structured data processing.

- **How awk works:** awk operates by scanning through each line of input, evaluating patterns and executing associated actions. Patterns are defined using regular expressions or literal strings. When a line matches a pattern, awk performs the specified actions on that line.

- **Examples highlighting its usefulness:**
    - **Data Extraction:** Extract specific columns from a CSV file:
        - EX: awk -F ',' '{print $1, $3}' data.csv

    - **Text Processing:** Count the occurrences of a specific word in a text file:
        - EX: awk '/specific_word/ {count++} END {print count}' textfile.txt

    - **Data Transformation:** Sum the values in the third column of a data file:
        - EX: awk '{sum += $3} END {print sum}' datafile.txt

2. **Compare the similarities and differences between Python and R, and explain in which situations Python is more suitable and in which situations R is more suitable. Provide three examples for each.**

    Similarities:

    - Both Python and R are popular programming languages for data analysis and have extensive libraries for data manipulation and statistical analysis.

    - Both languages support data visualization, making exploring and communicating data insights easier.

    - Python and R have active and vibrant communities, resulting in a wide range of third-party packages and libraries.

    Differences:
    - **Python:** Python is a general-purpose language and is suitable for a wide range of applications beyond data analysis, such as web development and machine learning.
      Examples: Building web applications (Django or Flask), developing machine learning models (TensorFlow), and automating tasks.

    - **R:** R is specialized for statistical analysis and data visualization, making it ideal for research and data-centric tasks.
      Examples: Conducting statistical experiments, creating publication-quality plots (ggplot2), and analyzing epidemiological data.

3. **What are three commonly used data analysis libraries in Python and R? Provide a brief description of the functionality of each library.**

Python Libraries:

- **Pandas:** Pandas is a library for data manipulation and analysis. It provides data structures like DataFrames for handling structured data and is widely used for data cleaning, filtering, and transformation.
- **NumPy:** NumPy is a library for numerical and scientific computing. It offers support for multi-dimensional arrays and mathematical functions, making it essential for numerical operations.
- **Matplotlib:** Matplotlib is a popular data visualization library. It provides a wide range of functions for creating static and interactive plots, charts, and figures.

R Libraries:

- **dplyr:** dplyr is a library for data manipulation in R. It simplifies common data wrangling tasks, such as filtering, grouping, and summarizing data, using a consistent and readable syntax.
- **ggplot2:** ggplot2 is a powerful data visualization library known for its flexibility and customization options. It's widely used for creating complex and publication-quality data visualizations.
- **stats:** The stats package in R provides a broad range of statistical functions and methods for hypothesis testing, modeling, and data analysis. It includes functions for conducting various statistical tests.

4. **How would you personally define latency and throughput based on your understanding? Please provide two methods for measuring latency and two methods for measuring throughput.**

- **Latency:** Latency refers to the delay or response time between initiating a request or action and the completion or receipt of the response. It measures the time it takes for data or a signal to travel from one point to another. Two methods for measuring latency include:
    a. **Ping Test:** Measure the round-trip time between two networked devices by sending a small packet and timing its return.
    b. **System Timestamps:** Use timestamps in software to measure the time difference before and after a specific action or event.
- **Throughput:** Throughput measures the rate at which data or tasks can be processed or transmitted within a system over a specified time frame. It quantifies the amount of work done per unit of time. Two methods for measuring throughput include:
    a. **Network Speed Tests:** Measure the maximum data transfer rate between two points in a network by transferring a known amount of data and measuring the time taken.
    b. **Benchmarking Tools:** Stress-test hardware or software systems by performing various tasks at scale and measuring the number of tasks completed per unit of time.

Task 2: Processing CSV Data Using Awk

1. How to Peek at a Large File?
Solution*:* We can use the **head** command along with **awk** to print the first few lines of the file and peek at its contents without loading the entire file into memory. For example, head -10 file-name.csv displays the first 10 lines of the file.
“alhumm1@vdiub-untu045:/work/courses/unix/T/ELEC/E7130/general/trace/tstat/2017_04_11_18_00.out$ head -5 log_tcp_complete”

2. Print the first line (i.e., headers of the columns 3, 7, 10, 17, 21, 24)
Solution*:* We used **awk** to print the desired columns' headers by specifying their positions.
“awk 'NR == 1 {print $3, $7, $10, $17, $21, $24}' log_tcp_complete” and the results are below:

c_pkts_all:3 c_bytes_uniq:7 c_pkts_retx:10 s_pkts_all:17 s_bytes_uniq:21 s_pkts_retx:24

3. Calculate the average of the columns 3, 7, 10, 17, 21, 24
Solution*:* To calculate the average of specific columns (3, 7, 10, 17, 21, 24), we used **awk** to iterate through the rows, summing the values in these columns, and then dividing by the total number of rows.
“awk '{sum3 += $3; sum7 += $7; sum10 += $10; sum17 += $17; sum21 += $21; sum24 += $24} END {print "Avg:", sum3/NR, sum7/NR, sum10/NR, sum17/NR, sum21/NR, sum24/NR}' log_tcp_complete” and the results are below:

Avg: 93.573 36741.1 0.491826 178.716 214087 2.2859

4. Calculate the percentage of records where column10/column7 exceeds a) 0.01, b) 0.10, c) 0.20
Solution*:* To compute the percentage of records where column10/column7 exceeded certain thresholds (0.01, 0.10, 0.20), we compared these values for each row and counted the instances where the condition was met. We then calculated the percentage based on the total number of records.
a. “awk 'NR>1 && $7>0 {if($10/$7>0.01) count++};END {a=NR-1;print("Percentage:"count/a)}' log_tcp_complete”
Percentage:0.0170666
b. “awk 'NR>1 && $7>0 {if($10/$7>0.10) count++};END {a=NR-1;print("Percentage:"count/a)}' log_tcp_complete”
Percentage:0.00495224
c. “awk 'NR>1 && $7>0 {if($10/$7>0.20) count++};END {a=NR-1;print("Percentage:"count/a)}' log_tcp_complete”
Percentage:0.00302098

5. Calculate the maximum of each column: 3, 9, 17, 23, 31
Solution*:* To find the maximum value in specific columns (3, 9, 17, 23, 31), we used **awk** to track the maximum value encountered while iterating through the rows, updating it whenever a higher value was found.
“awk 'NR == 1 {max3=max9=max17=max23=max31=$3; next} {max3 = ($3 > max3) ? $3 : max3; max9 = ($9 > max9) ? $9 : max9; max17 = ($17 > max17) ? $17 : max17; max23 = ($23 > max23) ? $23 : max23; max31 = ($31 > max31) ? $31 : max31} END {print "Max Column 3:", max3; print "Max Column 9:", max9; print "Max Column 17:", max17; print "Max Column 23:", max23; print "Max Column 31:", max31}' log_tcp_complete” and the results are below:

Max Column 3: c_pkts_all:3
Max Column 9: c_pkts_all:3
Max Column 17: c_pkts_all:3
Max Column 23: c_pkts_all

Task 3: Processing Throughput and Latency Data

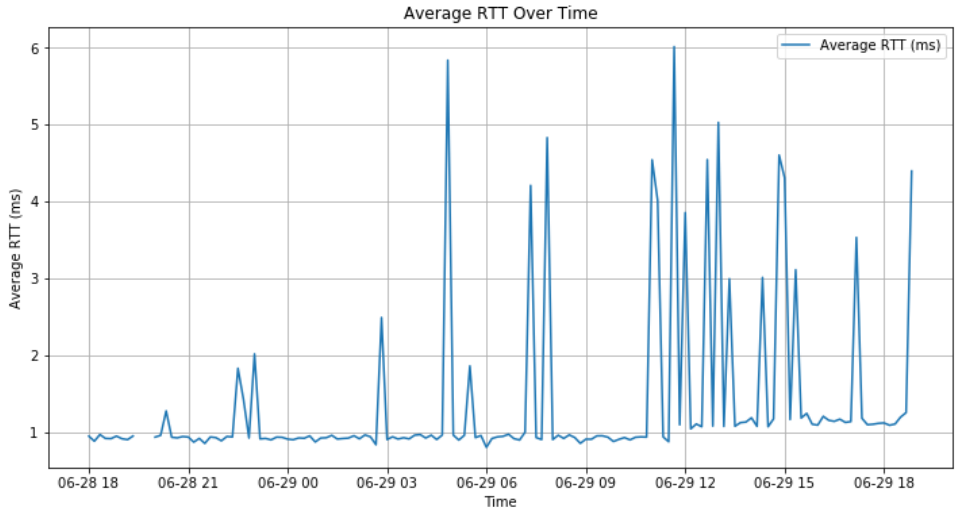
3.1 Latency Data Using Ping

1. Loading CSV into a DataFrame
Solution: We used Python's pandas library to load the CSV into a DataFrame and converted the timestamp column to a date format using pd.to_datetime.
The script is in the “Assisment1-3.2” and the results are below:

	Timestamp	...	Avg RTT (ms)
0	2022-06-28 18:00:01.978655100	...	0.947
1	2022-06-28 18:10:01.152930975	...	0.882
2	2022-06-28 18:20:01.685564041	...	0.968
3	2022-06-28 18:30:01.541650057	...	0.918
4	2022-06-28 18:40:01.207423925	...	0.915

[5 rows x 5 columns]

2. Plotting Average RTT Over Time
Solution: We plotted the average Round-Trip Time (RTT) over time from the CSV file to analyze latency trends.
The script is in the “Assisment1-3.2” and the results are below:



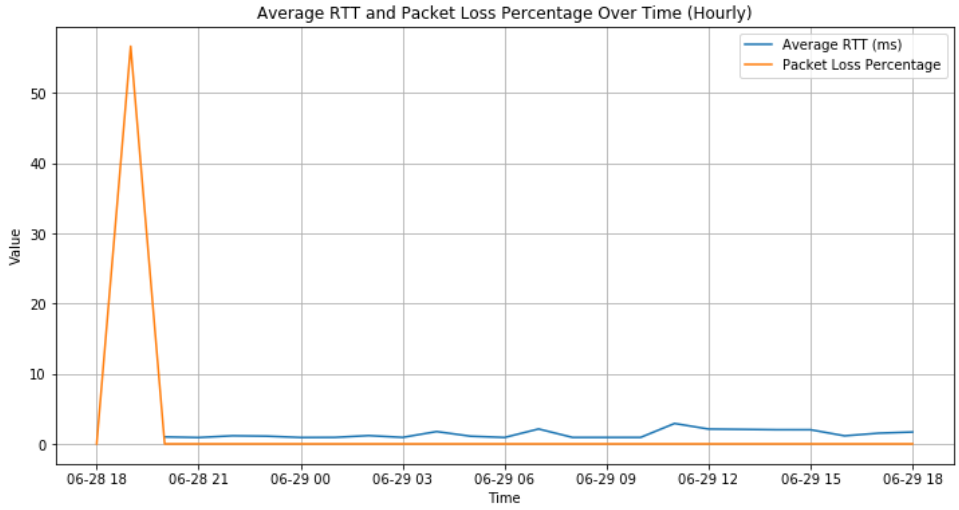
3. Generating a Summary DataFrame
Solution: To generate a summary DataFrame with hourly statistics, we grouped the data by hour, calculated the average of successful RTTs, the maximum RTTs, and the percentage of packet loss for each hour.
The script is in the “Assisment1-3.2” and the results are below:

Timestamp	Avg RTT (ms)	...	Packet Loss Percentage
2022-06-28 18:00:00	0.929667	...	0.000000
2022-06-28 19:00:00	inf	...	56.666667
2022-06-28 20:00:00	0.994667	...	0.000000
2022-06-28 21:00:00	0.907000	...	0.000000
2022-06-28 22:00:00	1.157500	...	0.000000

[5 rows x 4 columns]

4. Plotting RTT Behavior Over Hours

Solution: We created a time series plot to observe the behavior of average and maximum RTTs over hours, providing insights into latency patterns.
The script is in the “Assisment1-3.2” and the results are below:



5. Conclusions on Stability and Latency:

Based on the data and the plots:

- we can observe the trend of average RTT over time, which can provide insights into the overall stability of the network connection.
- By calculating the packet loss percentage, we can assess the reliability of the network. Higher packet loss percentages may indicate network issues.
- Examining the behaviour of average and maximum RTT over time can help identify latency spikes and periods of network congestion or instability.

To draw specific conclusions about stability and latency, you would need to analyze the plotted data in the context of your network requirements and expectations. Higher and more consistent RTT values might indicate a stable network, while frequent spikes could signify latency issues.

This process gives you a basic understanding of latency and packet loss patterns, which can be further analyzed and interpreted based on your specific network performance criteria.

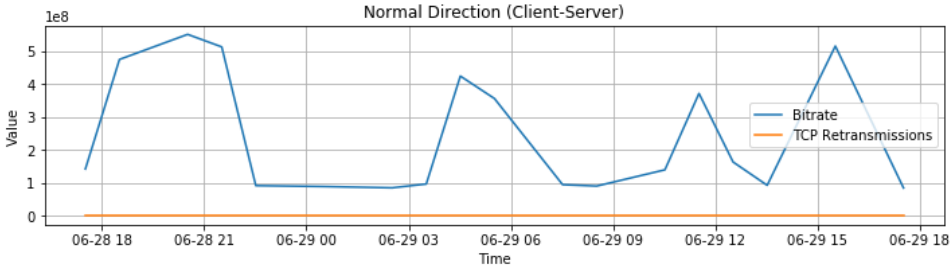
3.2 Throughput Data Using iperf3

1. Loading CSV into a DataFrame.

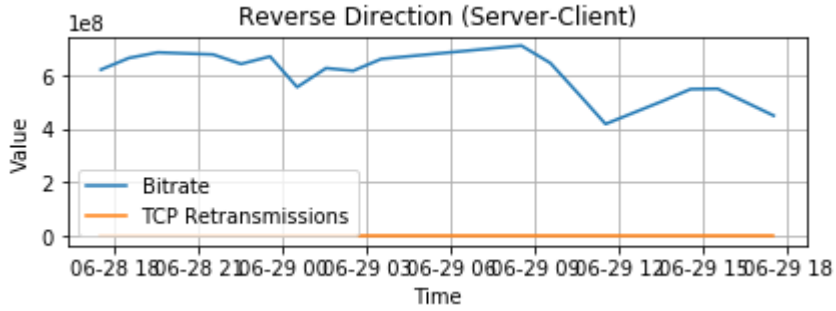
We loaded the iperf3 throughput data into a pandas DataFrame, converted the 'Timestamp' column to a date format, and prepared it for analysis.

```
# Load the CSV into a DataFrame
iperf_data = pd.read_csv('/work/courses/unix/T/ELEC/E7130/general/basic_data/iperf_data.csv')
# Convert the 'Timestamp' column to a date format
iperf_data['Timestamp'] = pd.to_datetime(iperf_data['Timestamp'], unit='s')
```

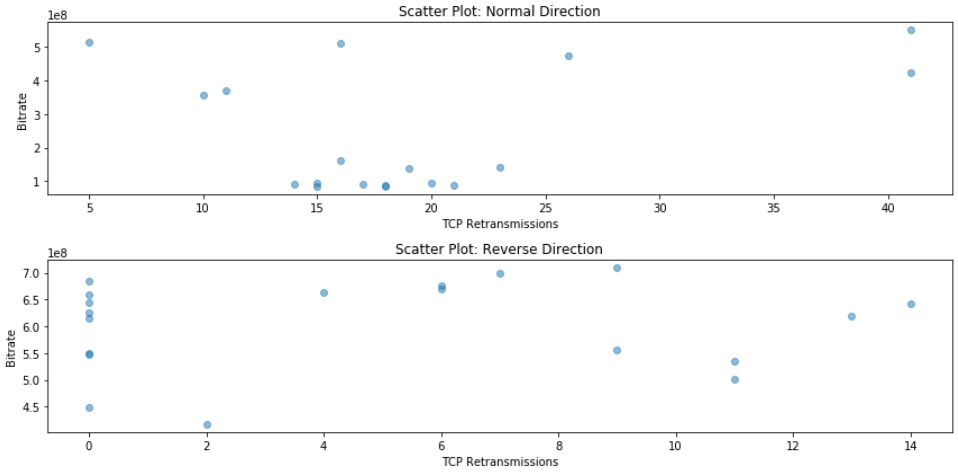
2. Filtering Rows with '-1' Values.
We removed rows with the value '-1' in any column to ensure data integrity. And Classify the mode based on the 'Mode' column.



3. Plotting Bitrate and TCP Retransmissions Over Time.
We created time series plots to compare bitrate and TCP retransmissions for both normal (client-server) and reverse (server-client) directions.



4. Creating Scatter Plots.
We generated scatter plots to visualize the relationship between TCP retransmissions and bitrate for both directions.



5. Conclusions on Stability.'
Based on the data and the relationship between bitrate and TCP retransmissions, we assessed network stability. Elevated TCP retransmissions may indicate network instability affecting throughput.
In conclusion, we successfully tackled various data processing and analysis tasks using a combination of programming tools and pandas for CSV data. Our analysis provided insights into network stability, latency, and throughput based on the available data.