Name: Yutong Du

Date: 2023-11-25

Email: Yutong.du@aalto.fi

Student ID: 101481573

Task 1: Capturing data

* **Acquiring packet capture data**

1. **What kind of trace file and tool/s you are using to perform the packet capture?**

**Wireshark**

1. **Date, time, duration, measurement setting (in terms of profile if you are using the Wireshark) or file name if you are using some public traces.**

**25/11/2023: From 17:31 to 19:31. Measurement setting: Default measurement setting of Wireshark. File name: final\_a.pcap.**

* **Provide a short sample (10 lines or so) of the data taken from your capture file.**

**Wireshark) or file name if you are using the some public traces.**

|  |
| --- |
|  |

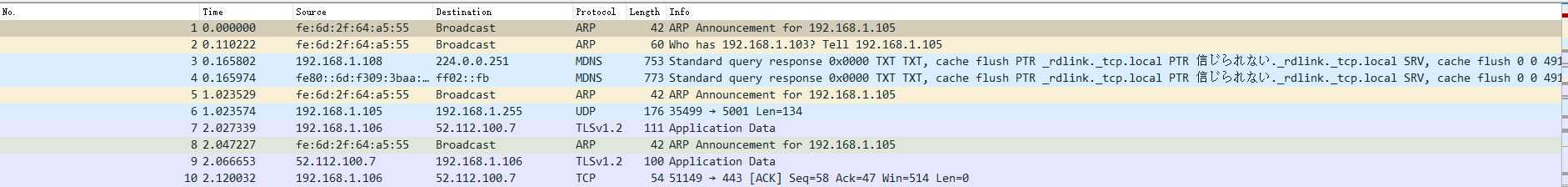
* **Data pre-processing**

**Commands:**

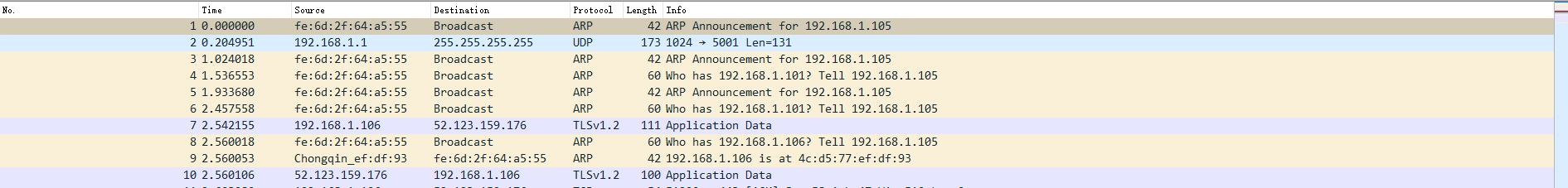
|  |
| --- |
| **tshark -r final\_a.pcap -Y "frame.time\_relative >= 0.0 and frame.time\_relative <= 2334" -w part1.pcap**  **tshark -r final\_a.pcap -Y "frame.time\_relative >= 2335 and frame.time\_relative <= 4667" -w part2.pcap**  **tshark -r final\_a.pcap -Y "frame.time\_relative >= 4668" -w part3.pcap** |

**Short samples:**

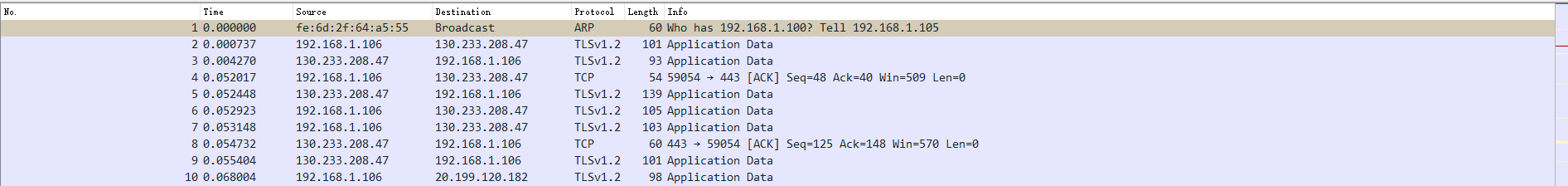
**part1.pcap:**

****

**part2.pcap**

****

**part3.pcap**

****

* **Packet data PS1**

**1.1: Visualise packet distribution by port numbers.**

**Code:**

|  |
| --- |
| import pyshark import matplotlib.pyplot as plt from collections import Counter  tshark\_path = 'D:\\0x00\_Softwares\\Wireshark\\tshark.exe' file\_path = 'files/part1.pcap'  cap = pyshark.FileCapture(file\_path, tshark\_path=tshark\_path, keep\_packets=True)  port\_counts = Counter() for packet in cap:  if 'TCP' in packet or 'UDP' in packet:  layer = packet.tcp if 'TCP' in packet else packet.udp  port\_counts[layer.dstport] += 1  cap.close()  plt.figure(figsize=(10, 10)) plt.pie(port\_counts.values(), labels=port\_counts.keys(), autopct='%1.1f%%', startangle=140) plt.title('Packet Distribution by Port Numbers') plt.show() |

**Packet distribution:**

|  |
| --- |
|  |

**1.2: Plot traffic volume as a function of time with at least two sufficiently different time scales.**

**Exported the part1.pcap as part1.csv.**

**Code:**

|  |
| --- |
| import pandas as pd import matplotlib.pyplot as plt  csv\_file\_path = 'files/part1.csv'  df = pd.read\_csv(csv\_file\_path)  df['Time'] = pd.to\_datetime(df['Time'], unit='s')  traffic\_volume = df.groupby('Time')['Length'].sum().reset\_index()  traffic\_per\_second = traffic\_volume.set\_index('Time').resample('S').sum().fillna(0) traffic\_per\_minute = traffic\_volume.set\_index('Time').resample('T').sum().fillna(0)  plt.figure(figsize=(14, 7))  plt.subplot(1, 2, 1) plt.plot(traffic\_per\_second.index, traffic\_per\_second['Length'], marker='o', linestyle='-') plt.title('Traffic Volume per Second') plt.xlabel('Time (second)') plt.ylabel('Traffic Volume (bytes)') plt.xticks(rotation=45)  plt.subplot(1, 2, 2) plt.plot(traffic\_per\_minute.index, traffic\_per\_minute['Length'], marker='o', linestyle='-', color='orange') plt.title('Traffic Volume per Minute') plt.xlabel('Time (minute)') plt.ylabel('Traffic Volume (bytes)') plt.xticks(rotation=45)  plt.tight\_layout() plt.show() |

**Result:**

|  |
| --- |
|  |

**1.3: Plot packet length distribution (use bins of width 1 byte), its empirical cumulative distribution function and key summary statistics.**

**Code:**

|  |
| --- |
| import pandas as pd import matplotlib.pyplot as plt import numpy as np  csv\_file\_path = 'files/part1.csv'  df = pd.read\_csv(csv\_file\_path)  plt.figure(figsize=(14, 7))  plt.figure(figsize=(14, 7))  plt.subplot(1, 2, 1) bin\_width = 100 bins = range(min(df['Length']), max(df['Length']) + bin\_width, bin\_width) plt.hist(df['Length'], bins=bins, color='blue', alpha=0.7, log=True) plt.title('Packet Length Distribution (Log Scale)') plt.xlabel('Packet Length (bytes)') plt.ylabel('Frequency (Log Scale)')  plt.subplot(1, 2, 2) sorted\_length = np.sort(df['Length']) yvals = np.arange(1, len(sorted\_length) + 1) / len(sorted\_length) plt.plot(sorted\_length, yvals, marker='.', linestyle='none') plt.title('Empirical Cumulative Distribution Function (ECDF)') plt.xlabel('Packet Length (bytes)') plt.ylabel('ECDF')  plt.tight\_layout() plt.show()  print("Summary statistics for packet lengths:") print(df['Length'].describe()) |

**Result:**

|  |
| --- |
|  |

**Key summary statistics:**

|  |
| --- |
| **Summary statistics for packet lengths:**  **count 178393.000000**  **mean 1058.292758**  **std 1753.207170**  **min 42.000000**  **25% 218.000000**  **50% 1292.000000**  **75% 1292.000000**  **max 65006.000000** |

* **Flow data PS2**

**1.4: Visualise flow distribution by port numbers.**

**Use the command below to convert part2.cap to flow data:**

|  |
| --- |
| **tshark -r part2.pcap -q -z conv,tcp > part2.txt** |

**Code:**

|  |
| --- |
| import pandas as pd import numpy as np import matplotlib.pyplot as plt from matplotlib.lines import Line2D  def convert\_to\_byte(row):  units = {'bytes': 1, 'kb': 1024, 'mb': 1024\*\*2}   try:  ld\_bytes\_unit = str(row['ld\_bytes\_unit']).lower()  factor = units[ld\_bytes\_unit]  ld\_kb = row['ld\_bytes'] \* factor   rd\_bytes\_unit = str(row['rd\_bytes\_unit']).lower()  factor = units[rd\_bytes\_unit]  rd\_kb = row['rd\_bytes'] \* factor   total\_bytes\_unit = str(row['total\_bytes\_unit']).lower()  factor = units[total\_bytes\_unit]  total\_kb = row['total\_bytes'] \* factor   return pd.Series({'ld\_bytes': ld\_kb, 'rd\_bytes': rd\_kb, 'total\_bytes': total\_kb, 'server\_ip': row['second\_ip\_interface']})  except KeyError as e:  print(f"Error processing row {row}: {e}")  raise ValueError("Invalid unit. Supported units are 'bytes', 'kb', 'mb.")  df = pd.read\_csv('files/part2.txt', sep='\s+', skiprows=5, header=None, skipfooter=1, engine='python')  new\_column\_names = ["first\_ip\_interface", "arrow", "second\_ip\_interface", "ld\_frames", "ld\_bytes", "ld\_bytes\_unit",  "rd\_frames", "rd\_bytes", "rd\_bytes\_unit", "total\_frames", "total\_bytes", "total\_bytes\_unit",  "start", "duration"]  df.columns = new\_column\_names  pd.set\_option('display.max\_columns', None)  df = df.assign(\*\*df.apply(convert\_to\_byte, axis=1))  df['port'] = df['second\_ip\_interface'].str.split(':').str[1].astype(str)  port\_flow\_count = df.groupby('port').size()  plt.figure(figsize=(10, 6)) bar\_plot = plt.bar(port\_flow\_count.index, port\_flow\_count.values)  for bar in bar\_plot:  yval = bar.get\_height()  plt.text(bar.get\_x() + bar.get\_width() / 2, yval, int(yval), va='bottom', ha='center')  plt.xlabel('Port Number') plt.ylabel('Number of Flows') plt.title('Flow Count Distribution by Port Numbers') plt.xticks(rotation=45) plt.show() |

**Flow distribution:**

|  |
| --- |
|  |

**1.5: Plot traffic volume as a function of time with at least two sufficiently different time scales.**

**Exported the part2.pcap as part2.csv.**

|  |
| --- |
|  |

**1.6: Visualise flow distribution by country.**

**Code:**

|  |
| --- |
| import pandas as pd import matplotlib.pyplot as plt from geoip2.database import Reader  def convert\_to\_byte(row):  units = {'bytes': 1, 'kb': 1024, 'mb': 1024\*\*2}   try:  ld\_bytes\_unit = str(row['ld\_bytes\_unit']).lower()  factor = units[ld\_bytes\_unit]  ld\_kb = row['ld\_bytes'] \* factor   rd\_bytes\_unit = str(row['rd\_bytes\_unit']).lower()  factor = units[rd\_bytes\_unit]  rd\_kb = row['rd\_bytes'] \* factor   total\_bytes\_unit = str(row['total\_bytes\_unit']).lower()  factor = units[total\_bytes\_unit]  total\_kb = row['total\_bytes'] \* factor   return pd.Series({'ld\_bytes': ld\_kb, 'rd\_bytes': rd\_kb, 'total\_bytes': total\_kb, 'server\_ip': row['second\_ip\_interface']})  except KeyError as e:  print(f"Error processing row {row}: {e}")  raise ValueError("Invalid unit. Supported units are 'bytes', 'kb', 'mb.")  df = pd.read\_csv('files/part2.txt', sep='\s+', skiprows=5, header=None, skipfooter=1, engine='python')  new\_column\_names = ["first\_ip\_interface", "arrow", "second\_ip\_interface", "ld\_frames", "ld\_bytes", "ld\_bytes\_unit",  "rd\_frames", "rd\_bytes", "rd\_bytes\_unit", "total\_frames", "total\_bytes", "total\_bytes\_unit",  "start", "duration"]  df.columns = new\_column\_names  pd.set\_option('display.max\_columns', None)  df = df.assign(\*\*df.apply(convert\_to\_byte, axis=1))  geoip\_reader = Reader('others/GeoLite2-Country.mmdb')  def get\_country(ip):  try:  response = geoip\_reader.country(ip)  return response.country.name  except:  return "Unknown"  df['country'] = df['second\_ip\_interface'].str.split(':').str[0].apply(get\_country)  country\_traffic = df.groupby('country').size()  plt.figure(figsize=(12, 8)) country\_traffic.plot(kind='bar') plt.xlabel('Country') plt.ylabel('Total Traffic (Bytes)') plt.title('Flow Distribution by Country') plt.xticks(rotation=45) plt.show() |

**Result:**

|  |
| --- |
|  |