Report on Network Traffic Analysis and Feature Engineering

1. Data Cleaning and Feature Engineering

Approach

The dataset provided was cleaned and preprocessed to identify and analyse network traffic trends. Key steps include:

1. **Data Cleaning**:
    - Dropped unnecessary columns, such as No. and Info, to focus on relevant attributes.
    - Converted the Time column to numeric format for easier aggregation and analysis.
2. **Feature Engineering**:
    - Created a new column TimeGroup by grouping Time into 10-second intervals to analyse trends over time.
    - Aggregated Length values to compute the total traffic volume for each TimeGroup.
3. **Anomaly Detection**:
    - Used the Interquartile Range (IQR) method to identify anomalies in traffic volume.
    - Marked data points with exceptionally high or low TotalLength as anomalies for further inspection.

Visualizations

- A line plot visualized the total traffic volume over time, with anomalies highlighted in red.
- Separate plots were created for different protocols, highlighting anomalies specific to each protocol.

2. Protocols Analysis

High-Traffic Protocols

These protocols exhibited the highest traffic volumes, making them critical for network performance and security:

- **DNS**: Core for resolving domain names to IP addresses, with modern updates ensuring security and efficiency.
- **TCP**: Backbone of internet communication, responsible for reliable data transfer.
- **TLSv1.2 and TLSv1.3**: Essential for secure communication, with TLSv1.3 being the modern standard.

Moderate-Traffic Protocols

Protocols in this category serve specific but important roles:

- **ARP**: Resolves IP to MAC addresses, vital for LAN environments.
- **ICMP**: Used for diagnostics (e.g., ping) but limited to control functions.
- **OCSP**: Supports certificate validation for secure web services.

Low-Traffic Protocols

Mostly legacy or niche protocols:

- **BROWSER and NBNS**: Obsolete and primarily used in older Windows environments.
- **SSLv2 and TLSv1**: Deprecated due to vulnerabilities.
- **STUN**: Critical for real-time communication, but traffic volume depends on usage scenarios.

3. Exploratory Data Analysis (EDA)

Observations from the Dataset

Protocol	Length	Traffic Category
DNS	257,488	High
TCP	312,000,000	High
TLSv1.2	1,018,772	High
TLSv1.3	74,838,461	High
BROWSER	1,944	Low
DHCP	6,024	Low
HTTP	960	Low

Protocol	Length	Traffic Category
ICMPv6	774	Low
RARP	10,680	Low
SSLv2	16,478	Low
STUN	272	Low
TLSv1	11,340	Low
ARP	25,092	Moderate
ICMP	251,540	Moderate
NBNS	22,728	Moderate
OCSP	162,574	Moderate

Trends and Anomalies

- High traffic from DNS and TCP protocols indicates their central role in internet communication.
- TLSv1.2 and TLSv1.3 traffic highlight the importance of secure communication.
- Anomalies were identified in certain time intervals, suggesting potential security threats or unusual network behaviour.

Visualizations

- Line plots for each protocol showed traffic trends over time, with red markers indicating anomalies.
- These anomalies might represent potential attacks, misconfigurations, or network issues.

4. Data Interpretation Case Study

Scenario

Analysed logs and data from a cybersecurity breach to identify the cause and propose preventive measures.

Findings

- Anomalies in DNS and TCP traffic suggest potential Distributed Denial of Service (DDoS) attacks.

- High traffic on outdated protocols like SSLv2 indicates misconfigured systems, making them vulnerable to attacks.

Preventive Measures

1. **Protocol Upgrades**:
    - Transition from SSLv2 and TLSv1 to newer, secure versions like TLSv1.3.
    - Replace obsolete protocols like BROWSER and NBNS with modern alternatives.
2. **Traffic Monitoring**:
    - Implement real-time monitoring to detect and mitigate traffic anomalies.
    - Use machine learning models for proactive threat detection.
3. **Network Segmentation**:
    - Isolate legacy systems from critical infrastructure to reduce attack surfaces.
4. **Regular Updates**:
    - Ensure software and firmware are up-to-date to mitigate vulnerabilities.

Conclusion

The analysis provided valuable insights into network traffic behaviour, highlighted potential security threats, and proposed actionable measures to enhance network security. The combined use of data cleaning, EDA, and anomaly detection ensures a robust approach to understanding and securing network environments.

Advanced Feature Engineering

In addition to the existing preprocessing steps, the Info column can be retained for further analysis. By extracting IP addresses from the Info column, we can conduct a deep dive into network activity to identify potential threats. The following steps are proposed:

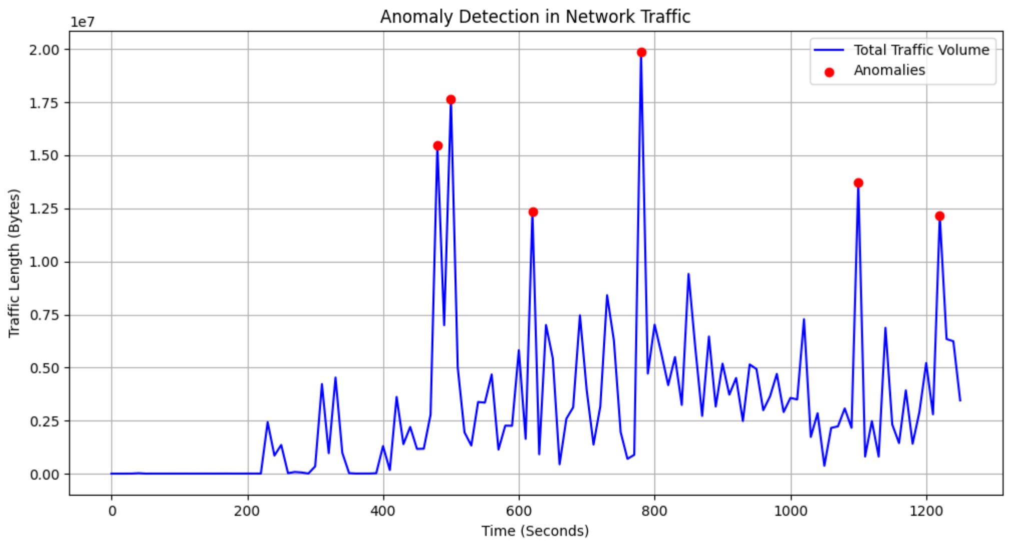
1. **Extract IP Addresses**: Parse the Info column to isolate source and destination IP addresses.
2. **Analyse IP Activity**: Identify:
    - **Most Frequent IPs**: Detect IP addresses with the highest frequency of occurrences, as these might indicate critical nodes or heavily used devices.

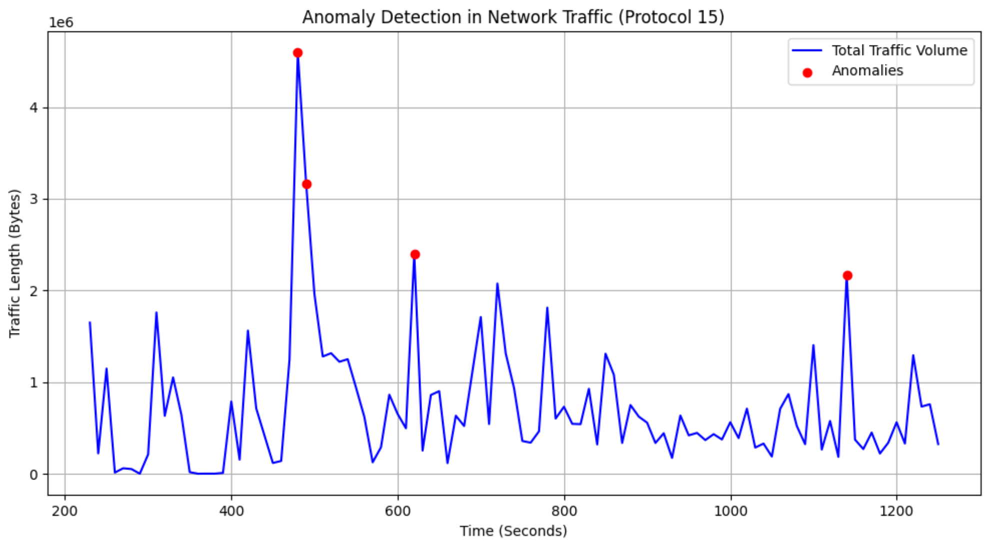
- **Suspicious Activity**: Highlight IPs sending multiple requests to different destinations or receiving high traffic from varied sources, which could signal potential security threats such as Distributed Denial-of-Service (DDoS) attacks.

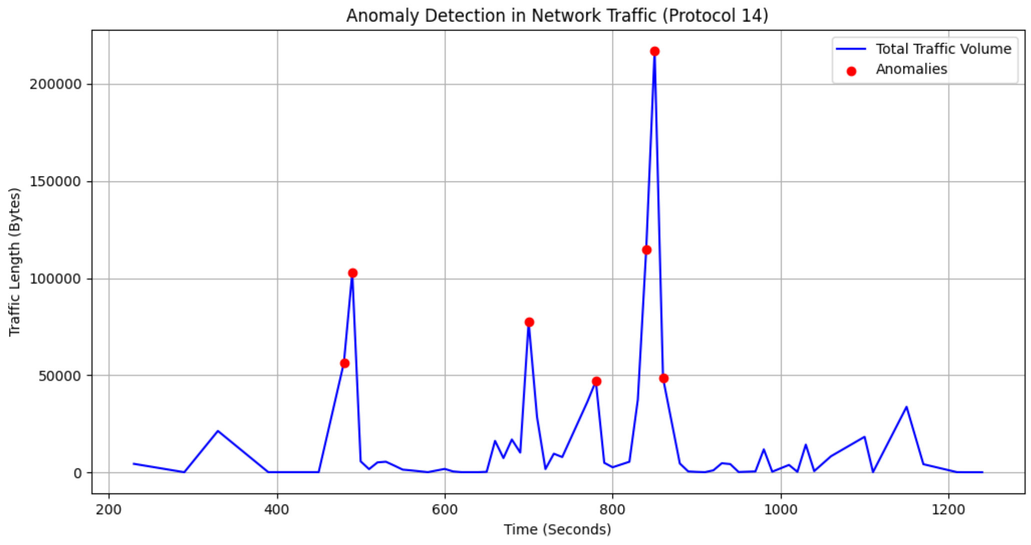
3. **Correlation with Anomalies**: Cross-reference these IPs with the anomalies detected earlier in traffic volume and protocol usage to strengthen threat detection.

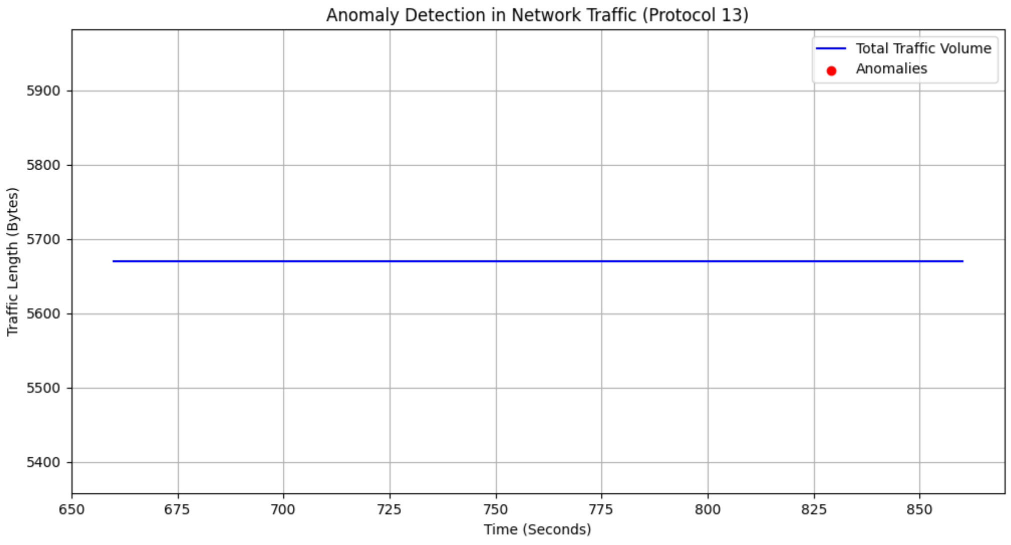
This step can provide additional insights into unusual network behaviour and assist in preemptive threat mitigation.

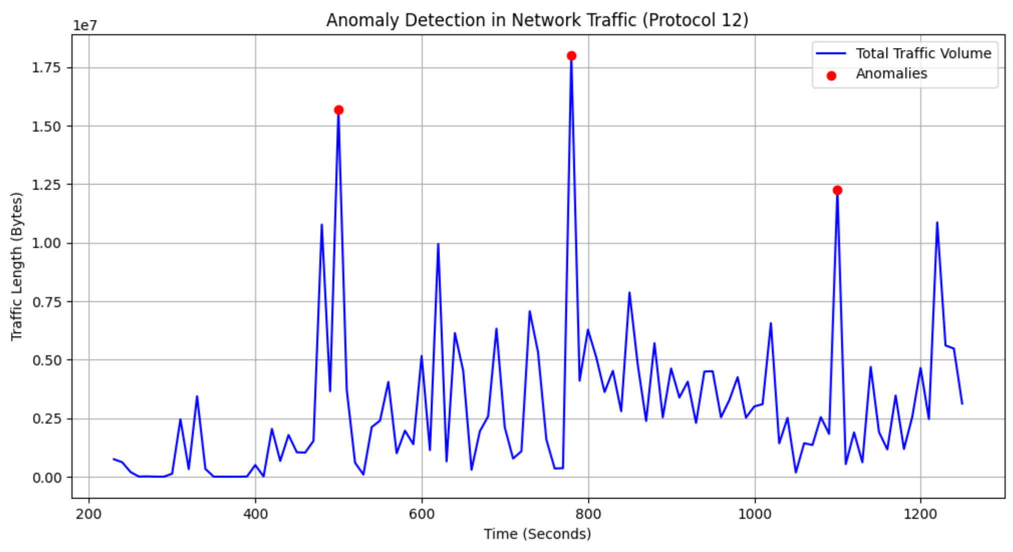
Graphs:

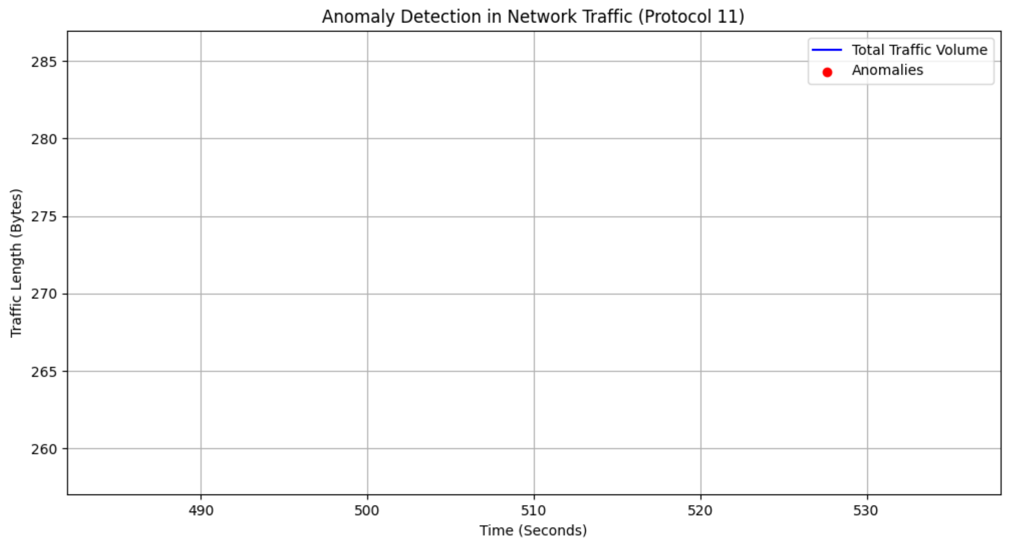


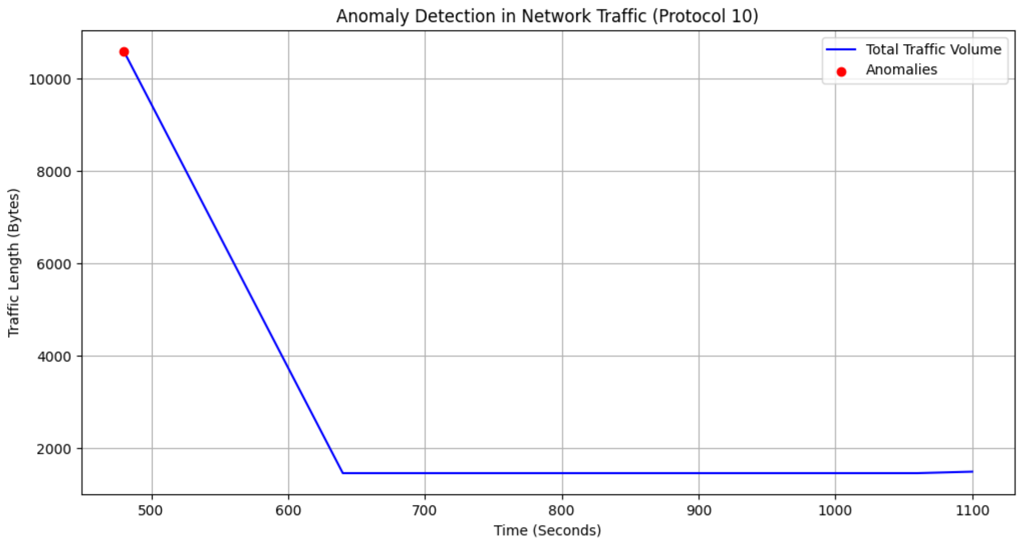

Anomaly Detection in Network Traffic (Protocol 15)
1e6
Traffic Length (Bytes)
Time (Seconds)
Total Traffic Volume
Anomalies

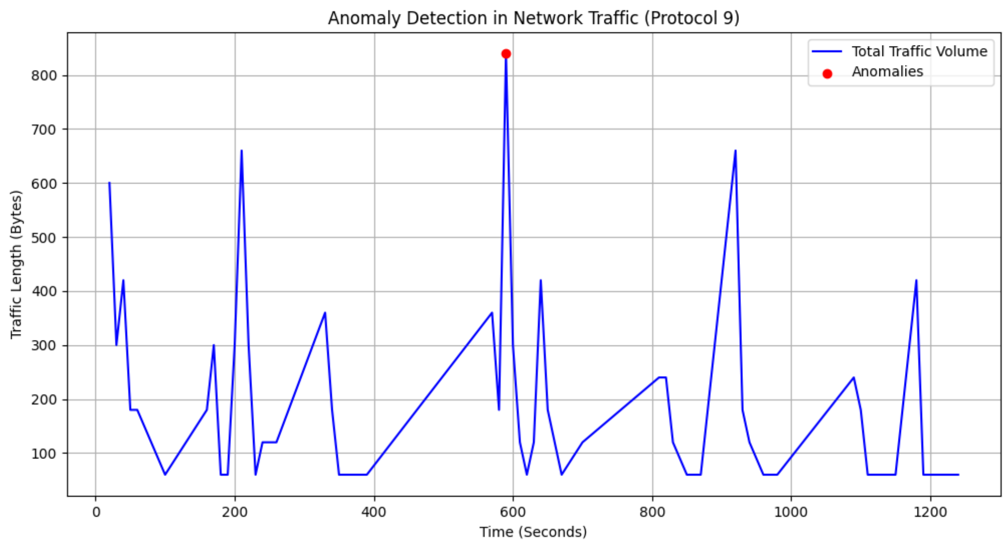

Anomaly Detection in Network Traffic (Protocol 14)
Traffic Length (Bytes)
Time (Seconds)
Total Traffic Volume
Anomalies


Anomaly Detection in Network Traffic (Protocol 13)
Traffic Length (Bytes)
Time (Seconds)
Total Traffic Volume
Anomalies

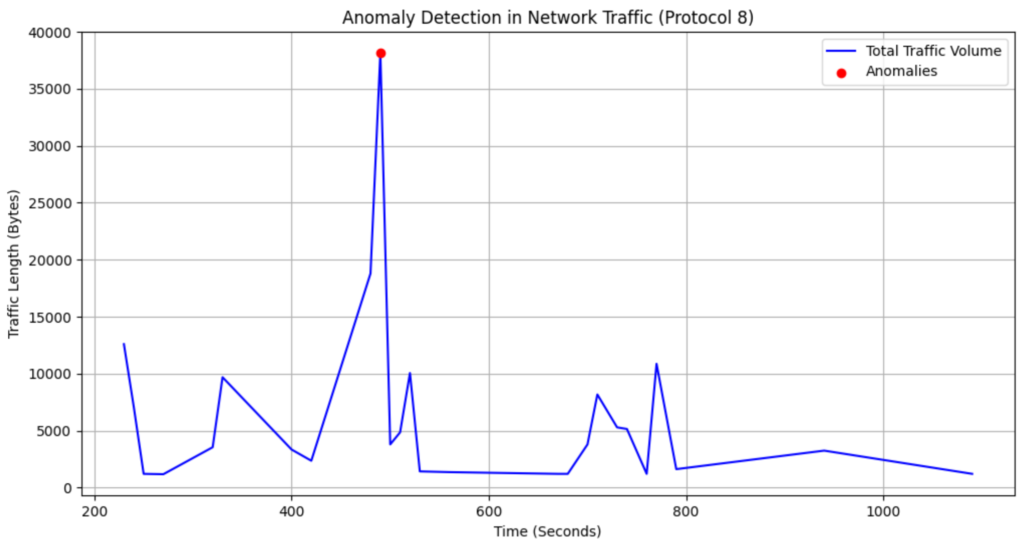

Anomaly Detection in Network Traffic (Protocol 12)
1e7
Traffic Length (Bytes)
Time (Seconds)
Total Traffic Volume
Anomalies


Anomaly Detection in Network Traffic (Protocol 11)
Traffic Length (Bytes)
Time (Seconds)
Total Traffic Volume
Anomalies

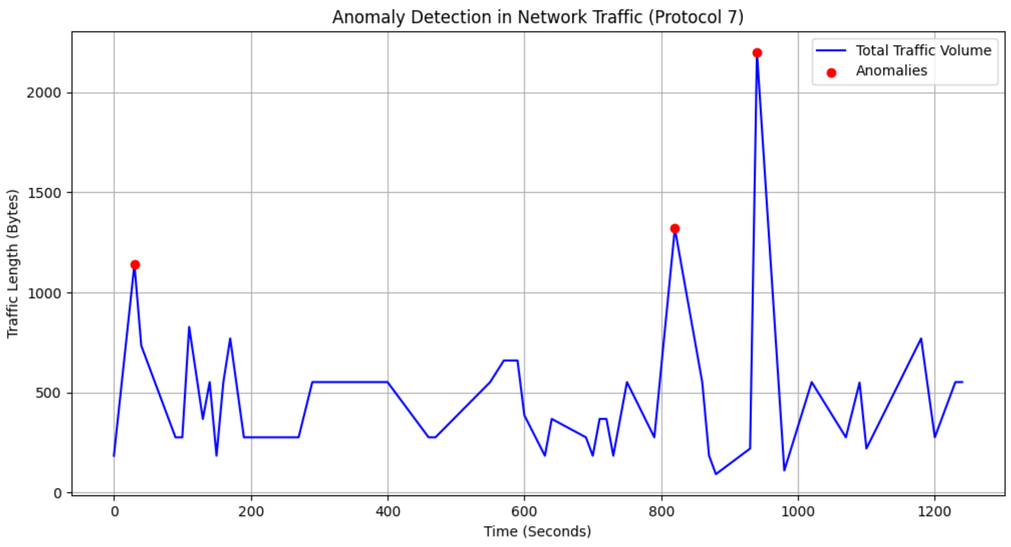

Anomaly Detection in Network Traffic (Protocol 10)
Traffic Length (Bytes)
Time (Seconds)
Total Traffic Volume
Anomalies



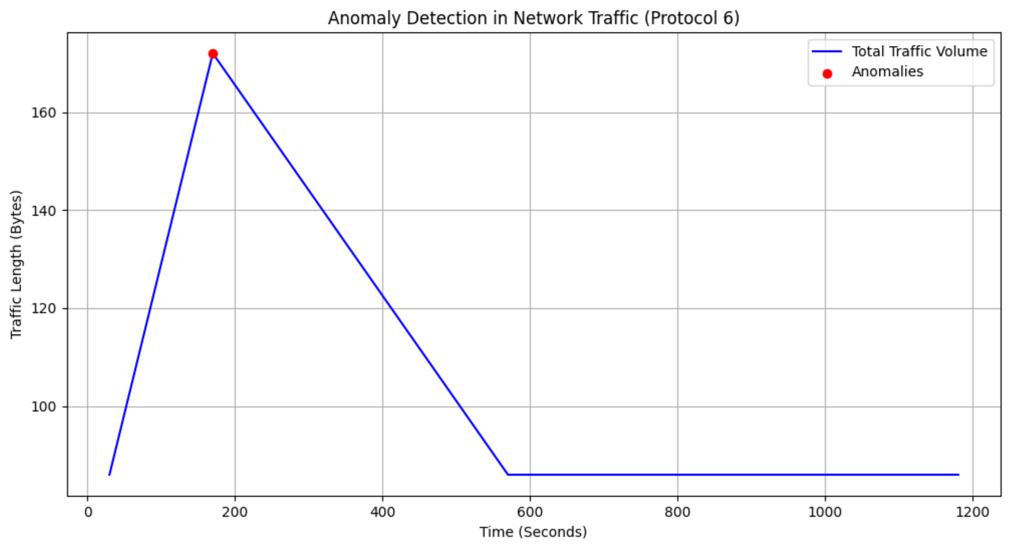
Anomaly Detection in Network Traffic (Protocol 9)
Total Traffic Volume
Anomalies
Traffic Length (Bytes)
Time (Seconds)
100
200
300
400
500
600
700
800
0
200
400
600
800
1000
1200



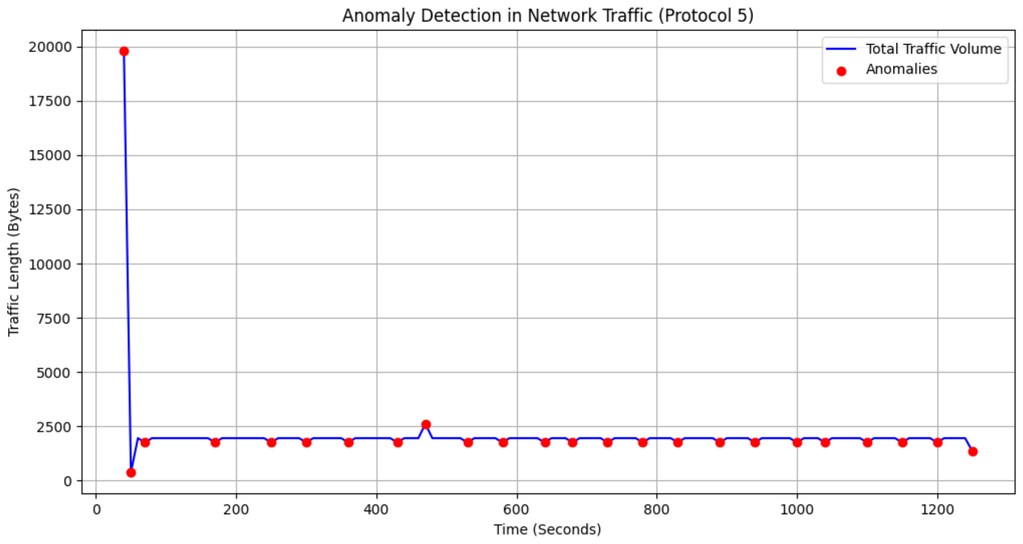
Anomaly Detection in Network Traffic (Protocol 8)
Total Traffic Volume
Anomalies
Traffic Length (Bytes)
Time (Seconds)
0
5000
10000
15000
20000
25000
30000
35000
40000
200
400
600
800
1000



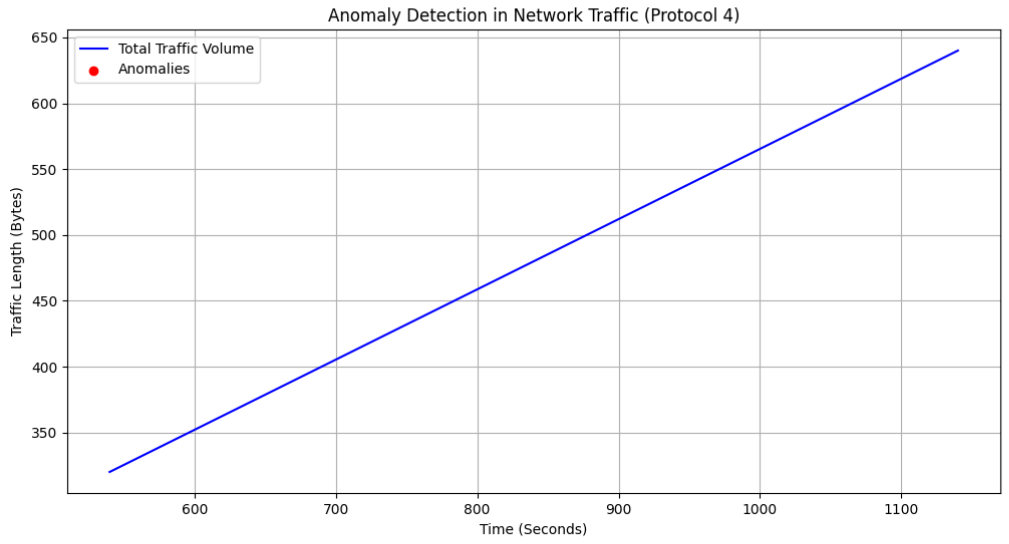
Anomaly Detection in Network Traffic (Protocol 7)
Total Traffic Volume
Anomalies
Traffic Length (Bytes)
Time (Seconds)
0
500
1000
1500
2000
0
200
400
600
800
1000
1200



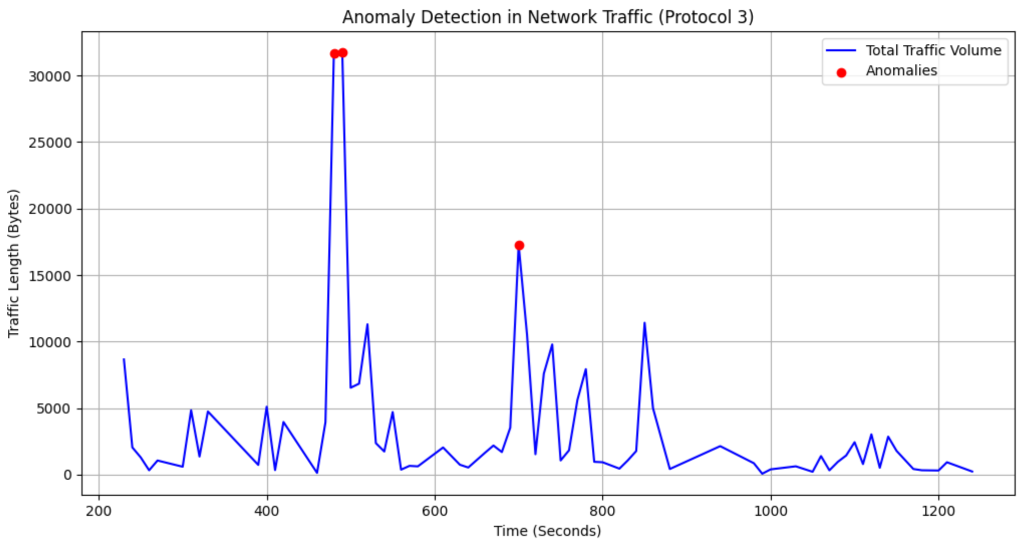
Anomaly Detection in Network Traffic (Protocol 6)
Total Traffic Volume
Anomalies
Traffic Length (Bytes)
Time (Seconds)

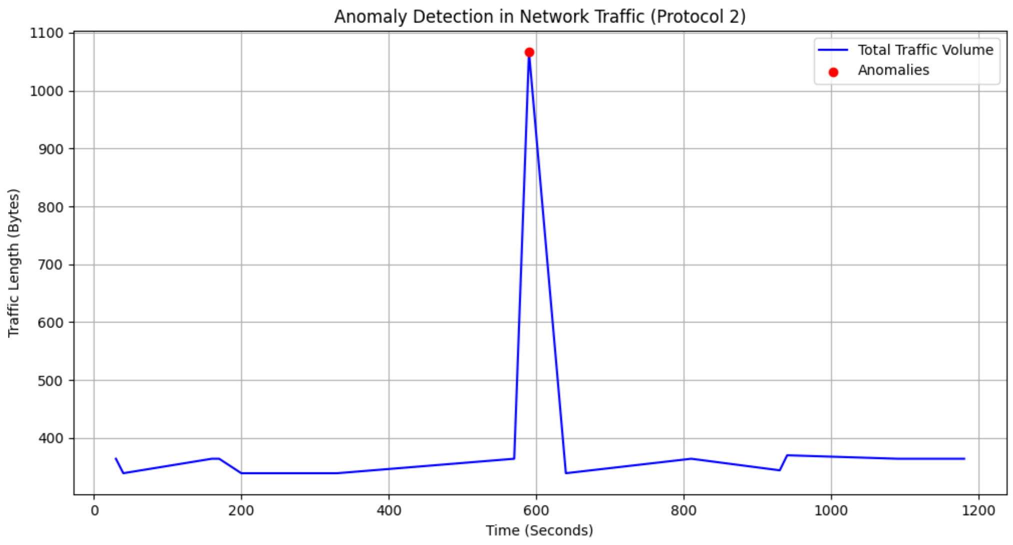


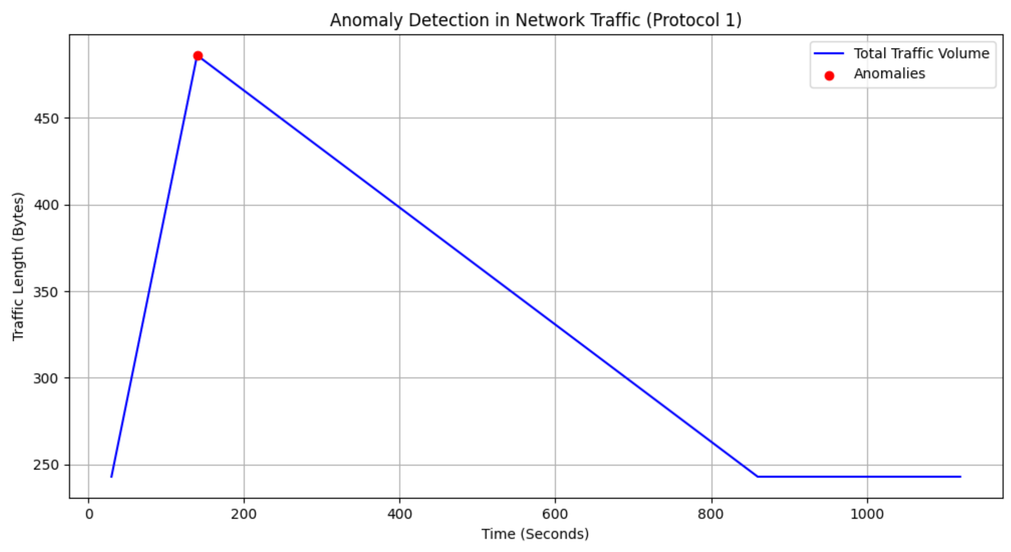
Anomaly Detection in Network Traffic (Protocol 5)
Total Traffic Volume
Anomalies
Traffic Length (Bytes)
Time (Seconds)

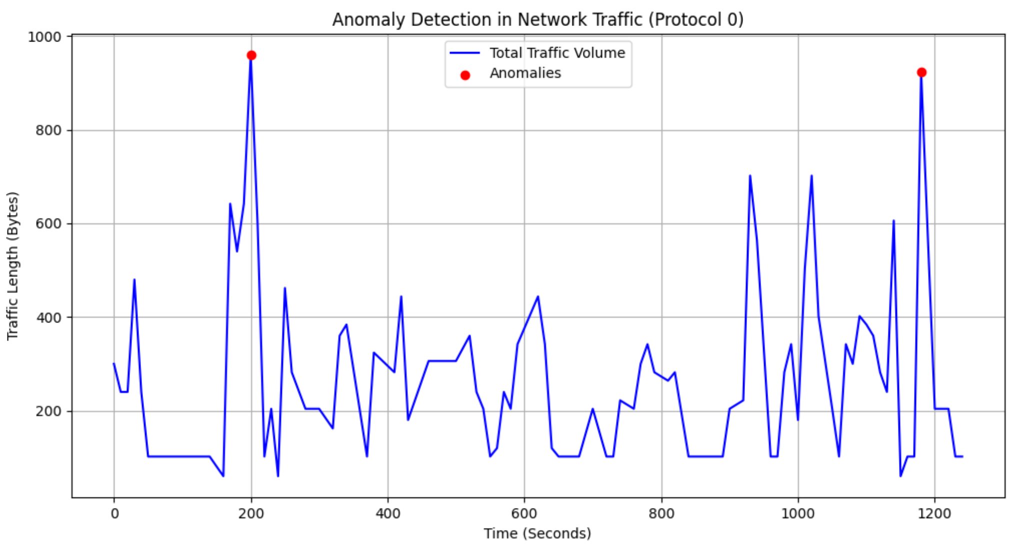


Anomaly Detection in Network Traffic (Protocol 4)
Total Traffic Volume
Anomalies
Traffic Length (Bytes)
Time (Seconds)


Anomaly Detection in Network Traffic (Protocol 3)
Total Traffic Volume
Anomalies
Traffic Length (Bytes)
Time (Seconds)
0
5000
10000
15000
20000
25000
30000
200
400
600
800
1000
1200


Anomaly Detection in Network Traffic (Protocol 2)
Total Traffic Volume
Anomalies
Traffic Length (Bytes)
Time (Seconds)
400
500
600
700
800
900
1000
1100
0
200
400
600
800
1000
1200





Code:

```
import pandas as pd

import numpy as np

data = pd.read_csv('Midterm_53_group.csv')


data = data.drop(['No.','Info'],axis = 1)


data.Time = pd.to_numeric(data.Time)
```

```
data['TimeGroup'] = (data['Time'] // 10) * 10

data1 = data.groupby(['TimeGroup']).agg(TotalLength = ('Length','sum')).reset_index()

data1

import matplotlib.pyplot as plt
plt.figure(figsize=(12, 6))
plt.plot(data1['TimeGroup'], data1['TotalLength'], label='Total Traffic Volume',
color='blue')

Q1 = data1.TotalLength.quantile(0.25)
Q3 = data1.TotalLength.quantile(0.75)
IQR = Q3-Q1
data1['Anomaly'] = (
    (data1.TotalLength < Q1 - 1.5 * IQR) |
    (data1.TotalLength > Q3 + 1.5 * IQR)
)

traffic_agg = data1
data1[data1['Anomaly']==True]



import matplotlib.pyplot as plt
plt.figure(figsize=(12, 6))
plt.plot(data1['TimeGroup'], data1['TotalLength'], label='Total Traffic Volume',
color='blue')
plt.scatter(
    traffic_agg['TimeGroup'][traffic_agg['Anomaly']],
    traffic_agg['TotalLength'][traffic_agg['Anomaly']],
    color='red',
    label='Anomalies',
    zorder=5
)
plt.xlabel("Time (Seconds)")
plt.ylabel("Traffic Length (Bytes)")
plt.title("Anomaly Detection in Network Traffic")
plt.legend()
plt.grid()
plt.show()



from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data.Protocol = le.fit_transform(data.Protocol)
data.Source = le.fit_transform(data.Source)
data.Destination = le.fit_transform(data.Destination)


data2 = data.groupby(['TimeGroup','Protocol']).agg(TotalLength = ('Length','sum')).reset_index()


data2


import matplotlib.pyplot as plt


unique_protocols = data2['Protocol'].unique()
unique_protocols = np.sort(unique_protocols)

for protocol in unique_protocols:

    data_protocol = data2[data2['Protocol'] == protocol].copy()


    Q1 = data_protocol['TotalLength'].quantile(0.25)

    Q3 = data_protocol['TotalLength'].quantile(0.75)

    IQR = Q3 - Q1


    data_protocol['Anomaly'] = (

      (data_protocol['TotalLength'] < Q1 - 2 * IQR) |

      (data_protocol['TotalLength'] > Q3 + 2 * IQR)

    )


    plt.figure(figsize=(12, 6))

    plt.plot(data_protocol['TimeGroup'], data_protocol['TotalLength'], label='Total Traffic
Volume', color='blue')

    plt.scatter(

      data_protocol['TimeGroup'][data_protocol['Anomaly']],

      data_protocol['TotalLength'][data_protocol['Anomaly']],

      color='red',

      label='Anomalies',

      zorder=5

    )

    plt.xlabel("Time (Seconds)")

    plt.ylabel("Traffic Length (Bytes)")

    plt.title(f"Anomaly Detection in Network Traffic (Protocol {protocol})")

    plt.legend()

    plt.grid()
    plt.show()


    print(f"Anomalies for Protocol {protocol}:")

    print(data_protocol[data_protocol['Anomaly'] == True])
```