Network Systems and Security

Assignment 4: Network Traffic Analysis

Final Report

Course: SIL765 (Network Systems and Security)
Submission Date: 6 April 2025

Author: Rohit Patidar
Entry no. : 2024JCS2042
Institution: Indian Institute Of Science And Technology ,Delhi

Contents

1 Introduction **2**

2 Task 1: Basic Network Traffic Statistics (25 Marks) **2**
- 2.1 Approach . . . 2
- 2.2 Result : All outputs above correspond to Basic Network Traffic Statistics . . . 5

3 Task 2: Traffic Estimation Using Sublinear Space (30 Marks) **6**
- 3.1 Objective . . . 6
- 3.2 (a) Estimating Unique IPs(10 Marks) . . . 6
  - 3.2.1 Approach . . . 6
- 3.3 Result : All outputs correspond to Estimating the number of unique IP addresses 8
- 3.4 (b) Identifying Frequently Contacted Destination IPs(10 Marks) . . . 8
  - 3.4.1 Approach: . . . 8
- 3.5 Result : All outputs correspond to Identifying frequently contacted destination IPs . . . 11
- 3.6 (c) Efficient Membership Testing (Bloom Filter)(10 Marks) . . . 11
  - 3.6.1 Approach : . . . 11
- 3.7 Key Findings . . . 13

4 Task 3: Advanced Anomaly Detection (25 Marks) **13**
- 4.1 Objective . . . 13
- 4.2 (a) Statistical Traffic Analysis(10 Marks) . . . 13
  - 4.2.1 Approach : . . . 13
  - 4.2.2 Result : All outputs correspond to Statistical Traffic Analysis . . . 16
- 4.3 (b) Behavioral Analysis (10 Marks) . . . 17
  - 4.3.1 Result : All output corresponds to Behavioral Analysis . . . 20
- 4.4 (c) Suspicious Communication Patterns (5 Marks) . . . 21
  - 4.4.1 Approach : . . . 21

5 Task 4: Deep Security Threat Analysis (25 Marks) **24**
- 5.1 Objective . . . 24
- 5.2 (a) Detecting Complex Attack Patterns (10 Marks) . . . 24
  - 5.2.1 Approach : . . . 25
- 5.3 (b) Malicious Payload Identification (10 Marks) . . . 26
- 5.4 (c) Threat Attribution and Risk Analysis (5 Marks) . . . 28
  - 5.4.1 Approach : . . . 28
  - 5.4.2 Detailed Reasoning for Risk Categorization . . . 30
- 5.5 Overall Risk Assessment . . . 31

6 Conclusion and Summary of Findings **32**

1 Introduction

This report presents an end-to-end analysis of network traffic logs provided in CSV format. The tasks include:

- **Task 1: Basic Network Traffic Statistics**
- **Task 2: Traffic Estimation Using Sublinear Space**
- **Task 3: Advanced Anomaly Detection**
- **Task 4: Deep Security Threat Analysis**

Throughout these tasks, different techniques (both exact and approximate) have been implemented to identify patterns, anomalies, and potential security threats in the traffic data. We also leveraged sublinear approaches (like Bloom filters and approximate counting) to handle memory constraints. Finally, we conducted a risk analysis and threat attribution based on the detected anomalies.

2 Task 1: Basic Network Traffic Statistics (25 Marks)

2.1 Approach

The network traffic data is loaded from a CSV file using the `pandas` library. The data is structured with various columns such as source IP, destination IP, protocol name, source bytes, destination bytes, and timestamp.

- **Step 1: Total Number of Flows**
  The first statistic computed is the total number of flows in the dataset. This is simply the count of rows in the dataset, which corresponds to the total number of network connections or transactions.

$$\text{Total Number of Flows} = \text{len(df)}$$

- **Step 2: Top Protocols**
  The next statistic identifies the top 5 protocols in the dataset. This is achieved by counting the frequency of each protocol using the `value_counts()` method on the `protocolName` column.

$$\text{Top Protocols} = \texttt{df['protocolName'].value\_counts().head(5)}$$

- **Step 3: Top Source and Destination IPs**
  The top 10 most active source IPs and destination IPs are determined using the `value_counts()` method on the `source` and `destination` columns, respectively.

$$\text{Top Source IPs} = \texttt{df['source'].value\_counts().head(10)}$$

$$\text{Top Destination IPs} = \texttt{df['destination'].value\_counts().head(10)}$$

- **Step 4: Average Packet Size**
  The average packet size is calculated by summing the `totalSourceBytes` and `totalDestinationBytes` for each flow, and then computing the mean of this new column.

$$\text{Average Packet Size} = \text{mean}(df['packet\_size'])$$

- **Step 5: Most Common Source-Destination Pair**
  The most common source-destination pair is identified by grouping the data by both the `source` and `destination` columns, then counting the occurrences of each pair.

$$\text{Most Common Pair} = \texttt{df.groupby(['source', 'destination']).size().sort\_values(ascending=F}$$

- **Step 6: Consistent IPs Across Hours**
  The next statistic looks for IPs that communicate consistently across every hour. This is done by grouping the data by hour and checking for common IPs that appear in every hourly group.

$$\text{Consistent IPs} = \bigcap \text{IPs across all hourly groups}$$

- **Step 7: Traffic Spikes Detection**
  Traffic spikes are detected by resampling the data by hour and calculating the mean and standard deviation of the hourly traffic counts. A threshold is set at the mean plus two times the standard deviation, and any traffic count exceeding this threshold is flagged as a spike.

$$\text{Threshold} = \text{mean}(\text{traffic\_counts}) + 2 \times \text{std}(\text{traffic\_counts})$$

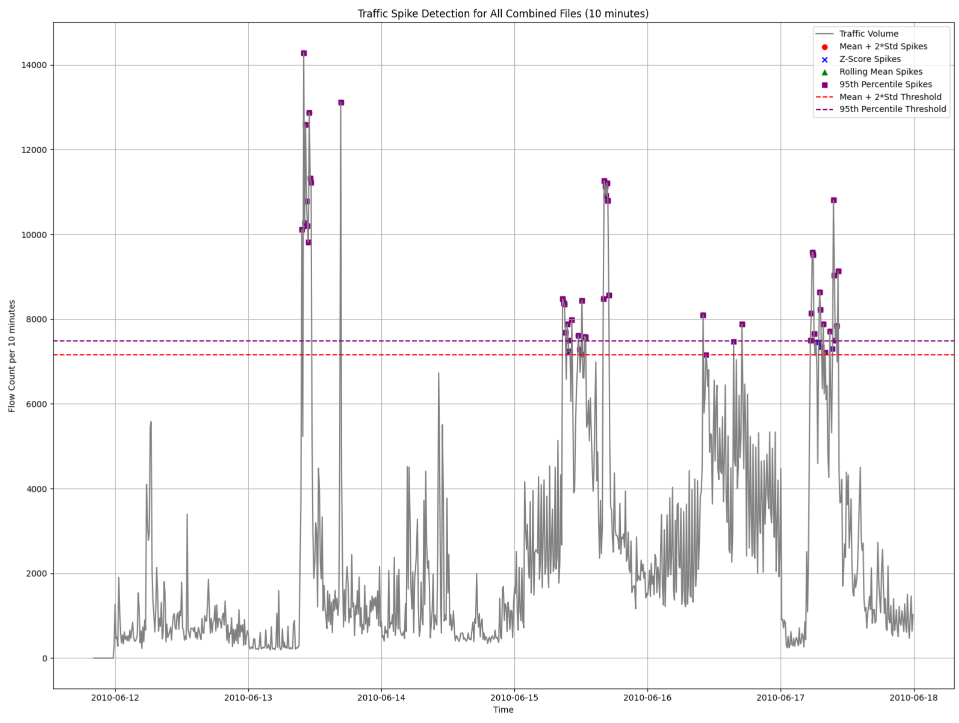


Figure 1: spike detection of all Combined Filesfor 10 minutes

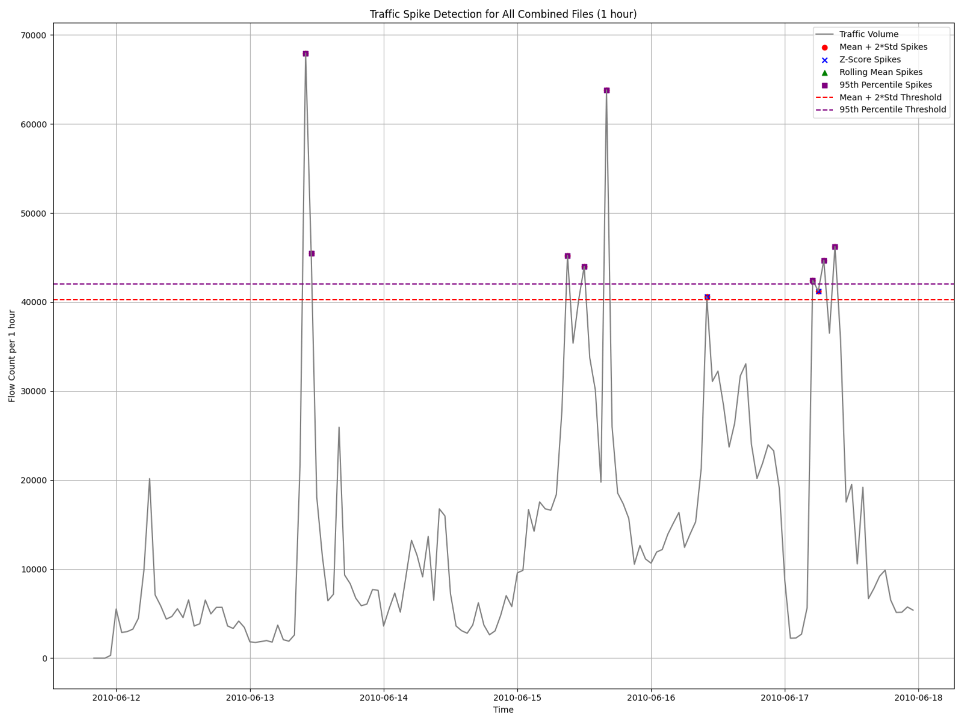
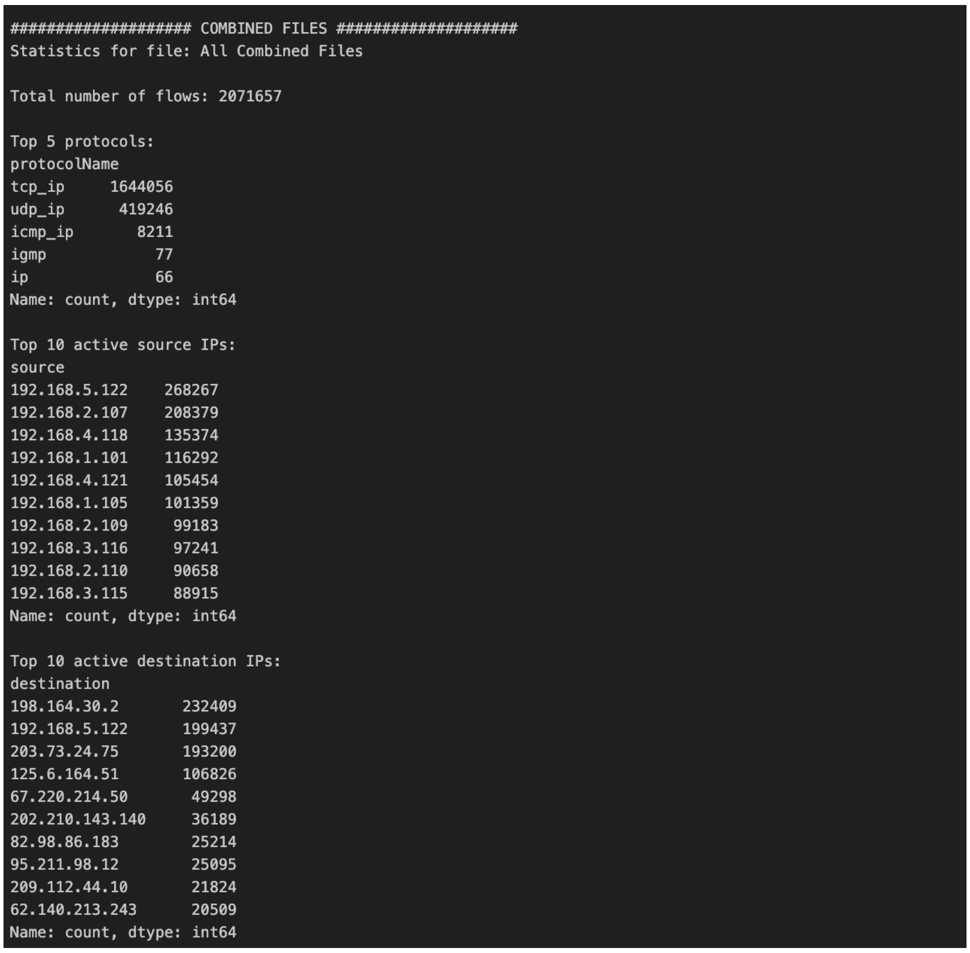


Figure 2: spike detection All Combined Files 1 for hour

- **Step 8: Variance of Average Source Packet Sizes**
  The final statistic computed is the variance of the average source packet sizes. This is calculated by first computing the average packet size for each flow and then computing the variance of these averages.**Note:**Low variance suggests consistent packet sizes across flows

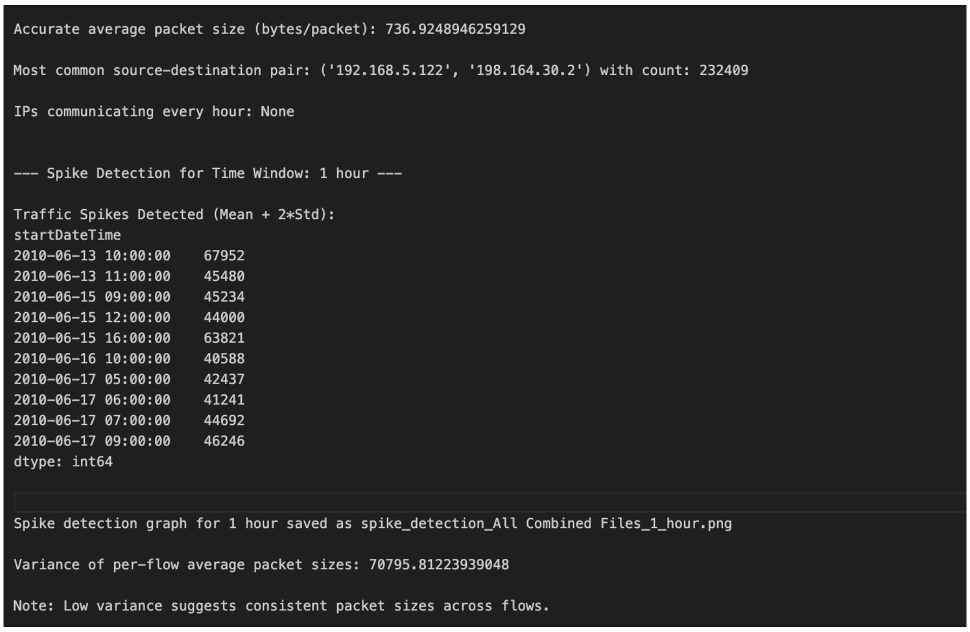
$$\text{Variance of Average Source Packet Sizes} = \text{var}(df['avg\_packet\_size'])$$

2.2 Result : All outputs above correspond to Basic Network Traffic Statistics



```
################### COMBINED FILES ###################
Statistics for file: All Combined Files

Total number of flows: 2071657

Top 5 protocols:
protocolName
tcp_ip     1644056
udp_ip      419246
icmp_ip       8211
igmp            77
ip              66
Name: count, dtype: int64

Top 10 active source IPs:
source
192.168.5.122    268267
192.168.2.107    208379
192.168.4.118    135374
192.168.1.101    116292
192.168.4.121    105454
192.168.1.105    101359
192.168.2.109     99183
192.168.3.116     97241
192.168.2.110     90658
192.168.3.115     88915
Name: count, dtype: int64

Top 10 active destination IPs:
destination
198.164.30.2       232409
192.168.5.122      199437
203.73.24.75       193200
125.6.164.51       106826
67.220.214.50       49298
202.210.143.140     36189
82.98.86.183        25214
95.211.98.12        25095
209.112.44.10       21824
62.140.213.243      20509
Name: count, dtype: int64
```

Figure 3:



```
Accurate average packet size (bytes/packet): 736.9248946259129

Most common source-destination pair: ('192.168.5.122', '198.164.30.2') with count: 232409

IPs communicating every hour: None


--- Spike Detection for Time Window: 1 hour ---

Traffic Spikes Detected (Mean + 2*Std):
startDateTime
2010-06-13 10:00:00    67952
2010-06-13 11:00:00    45480
2010-06-15 09:00:00    45234
2010-06-15 12:00:00    44000
2010-06-15 16:00:00    63821
2010-06-16 10:00:00    40588
2010-06-17 05:00:00    42437
2010-06-17 06:00:00    41241
2010-06-17 07:00:00    44692
2010-06-17 09:00:00    46246
dtype: int64

Spike detection graph for 1 hour saved as spike_detection_All Combined Files_1_hour.png

Variance of per-flow average packet sizes: 70795.81223939048

Note: Low variance suggests consistent packet sizes across flows.
```

Figure 4:

3 Task 2: Traffic Estimation Using Sublinear Space (30 Marks)

3.1 Objective

Due to memory constraints, exact counting of all IPs is impractical in large-scale networks. Hence, we:

1. Estimated the number of unique IPs using sublinear space.
2. Identified heavy-hitter destination IPs with an approximate approach.
3. Performed efficient membership tests using Bloom filters.

3.2 (a) Estimating Unique IPs(10 Marks)

3.2.1 Approach

The goal of this analysis is to estimate the number of unique IP addresses in network traffic data using two different methods: a sublinear HyperLogLog (HLL) bucket-based approach and an exact method using the set of unique IPs. The approach can be divided into the following steps:

- **Step 1: Loading the Network Traffic Data**

  The network traffic data is read from CSV files using the `pandas` library. The required columns, such as `source` and `destination`, are extracted and combined into a single list of IP addresses for analysis.

- **Step 2: HyperLogLog Estimation Method**

  HyperLogLog is a probabilistic data structure used to estimate the cardinality of a set (the number of unique elements). In this approach, the IP addresses are processed in the following manner:

  – Compute the MD5 hash of each IP address.
  – The hash is converted to a 128-bit binary string, and the first p bits are used to index a bucket.
  – The remaining bits are analyzed for the number of leading zeros, which is used to estimate the number of unique elements.
  – The HyperLogLog estimate is then calculated using the harmonic mean of the bucket values.

  The formula used for HyperLogLog is as follows:

$$E = \alpha_m \cdot m^2 / Z$$

  Where:

  – E is the estimated number of unique elements.
  – α_m is a correction constant, computed as $\alpha_m = \frac{0.7213}{1+\frac{1.079}{m}}$.
  – m is the number of buckets (2^p).
  – Z is the harmonic sum of the buckets.

- **Step 3: Exact Unique IP Count**

  The exact unique IP count is computed by simply finding the number of unique IP addresses in the dataset. This is achieved by using the `unique()` method in `pandas` on the combined list of source and destination IPs.

- **Step 4: Error Rate Calculation**

  The error rate between the approximate HyperLogLog estimate and the exact unique IP count is calculated as follows:

$$\text{Error Rate} = \frac{|E - \text{Exact Unique}|}{\text{Exact Unique}} \times 100$$

  Where E is the HyperLogLog estimate and Exact Unique is the exact count of unique IPs.

- **Step 5: Time and Space Complexity Analysis**

  For both methods (HyperLogLog and exact), the time taken to process the data is recorded. HyperLogLog operates in constant time per element, while the exact count method requires examining each element individually. Memory usage is also measured for both methods: HyperLogLog uses a fixed amount of space (determined by the number of buckets), whereas the exact method stores the entire set of unique IPs.

- **Step 6: Trade-off Analysis**

  The trade-offs between the HyperLogLog and exact methods are analyzed in terms of:

  - **Space**: HyperLogLog uses a fixed, sublinear amount of memory (2^p buckets), which is much more memory-efficient than storing all unique IP addresses exactly.
  - **Accuracy**: HyperLogLog achieves high accuracy with a small error margin (1-3%), but the error can increase if the parameter p is not properly tuned.
  - **Computation Time**: HyperLogLog operates in constant time per element, making it faster for large datasets compared to the linear approach.

- **Step 7: Visualization**

  Finally, the results for each file are visualized using bar charts. The following metrics are plotted:

  - Error rate for each file.
  - Comparison of exact and approximate unique IP counts.
  - Computation time for both methods.
  - Memory usage for both methods.

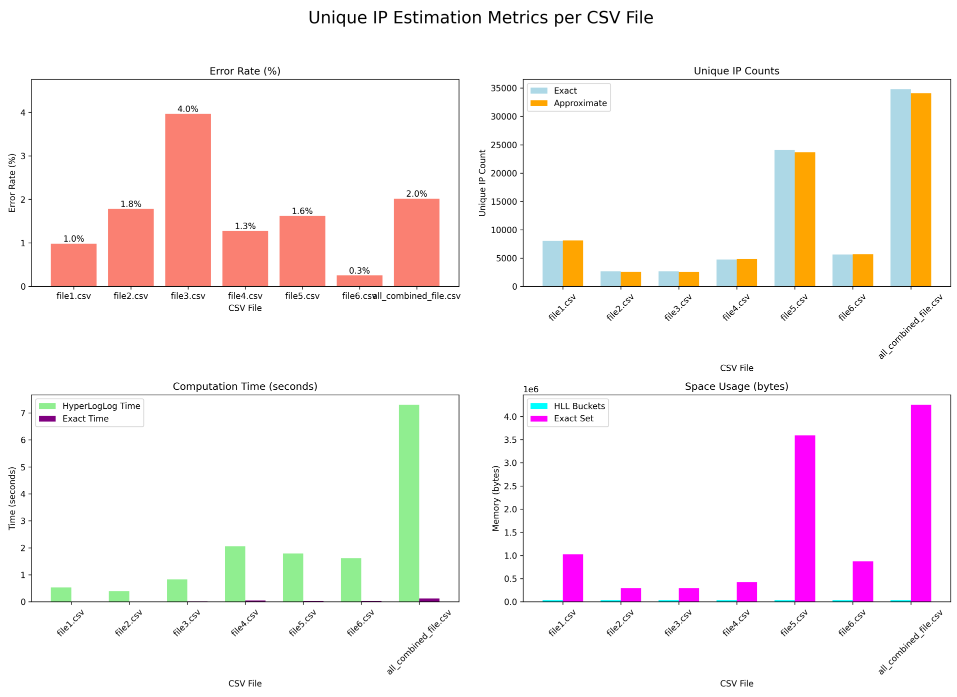
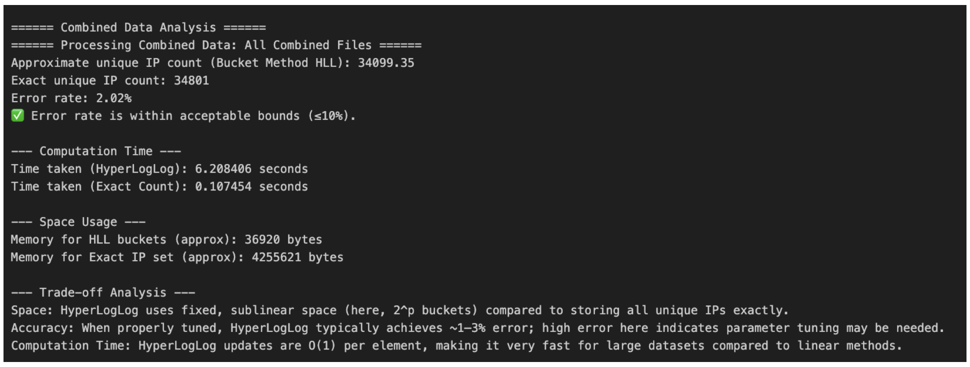


Figure 5:

3.3 Result : All outputs correspond to Estimating the number of unique IP addresses



```
====== Combined Data Analysis ======
====== Processing Combined Data: All Combined Files ======
Approximate unique IP count (Bucket Method HLL): 34099.35
Exact unique IP count: 34801
Error rate: 2.02%
✅ Error rate is within acceptable bounds (≤10%).

--- Computation Time ---
Time taken (HyperLogLog): 6.208406 seconds
Time taken (Exact Count): 0.107454 seconds

--- Space Usage ---
Memory for HLL buckets (approx): 36920 bytes
Memory for Exact IP set (approx): 4255621 bytes

--- Trade-off Analysis ---
Space: HyperLogLog uses fixed, sublinear space (here, 2^p buckets) compared to storing all unique IPs exactly.
Accuracy: When properly tuned, HyperLogLog typically achieves ~1-3% error; high error here indicates parameter tuning may be needed.
Computation Time: HyperLogLog updates are O(1) per element, making it very fast for large datasets compared to linear methods.
```

Figure 6:

3.4 (b) Identifying Frequently Contacted Destination IPs(10 Marks)

3.4.1 Approach:

The goal of this analysis is to identify heavy hitters in the network traffic data, specifically focusing on destination IP counts, using the Count-Min Sketch (CMS) algorithm. This approach approximates the frequency of the most common IP addresses while balancing space efficiency and computational time. The methodology consists of the following steps:

- **Step 1: Loading the Network Traffic Data**

  The network traffic data is read from CSV files using the `pandas` library. The required column, `destination`, which contains the destination IP addresses, is extracted for analysis.

- **Step 2: Initializing the Count-Min Sketch**

  A Count-Min Sketch (CMS) data structure is initialized with a predefined width and depth. This structure uses multiple hash functions to map items (IP addresses) into a table. The CMS is used to approximate the frequency of destination IPs.

  The parameters `width` and `depth` control the space usage and error bounds of the CMS, where:

  - `width` determines the number of columns (buckets) in the sketch table.
  - `depth` represents the number of hash functions used.

- **Step 3: Adding IP Addresses to the Count-Min Sketch**

  Each destination IP in the dataset is added to the Count-Min Sketch. For each IP, the hash functions are applied to update the corresponding bucket in the table. This step allows the CMS to store approximate frequency counts of the destination IPs without explicitly storing all unique IP addresses.

- **Step 4: Estimating Heavy Hitters**

  After processing all IPs, the CMS estimates the frequency of each unique IP address by querying the corresponding buckets in the table. The top 10 most frequent destination IPs are selected based on the estimated counts, which are then compared with the exact counts.

- **Step 5: Error Rate Calculation**

  The error rate between the approximate counts obtained from the Count-Min Sketch and the exact counts is calculated for each of the top 10 heavy hitters. The error rate is defined as:

  $$\text{Error Rate} = \frac{|E - \text{Exact}|}{\text{Exact}} \times 100$$

  Where E is the approximate count and Exact is the true count of the destination IP.

- **Step 6: Time and Space Complexity Analysis**

  The time taken to process the data and the memory usage of both the Count-Min Sketch method and the exact method are measured. The CMS operates in constant time per element, whereas the exact method requires checking all unique destination IPs.

  Space complexity is analyzed by comparing the memory usage of the CMS (which uses a fixed number of buckets) with the space required to store the exact set of unique IP addresses.

- **Step 7: Trade-off Analysis**

  The trade-offs between the Count-Min Sketch and exact methods are analyzed in terms of:

  - **Space**: The Count-Min Sketch is memory-efficient, requiring much less space than storing all unique IP addresses.
  - **Accuracy**: While the CMS achieves high accuracy with low error, there may be some variance in estimates, especially for the smallest frequency counts.
  - **Computation Time**: The CMS operates faster than the exact method for large datasets, as it avoids storing all IP addresses.

- **Step 8: Visualization**

  The results are visualized through bar charts that show the comparison between the exact and approximate counts for the top heavy hitter IPs. The following metrics are plotted:

  – Average error rate for each file.

  – Comparison of exact and approximate counts for the top heavy hitter.

  – Computation time for both methods.

  – Memory usage for both methods.

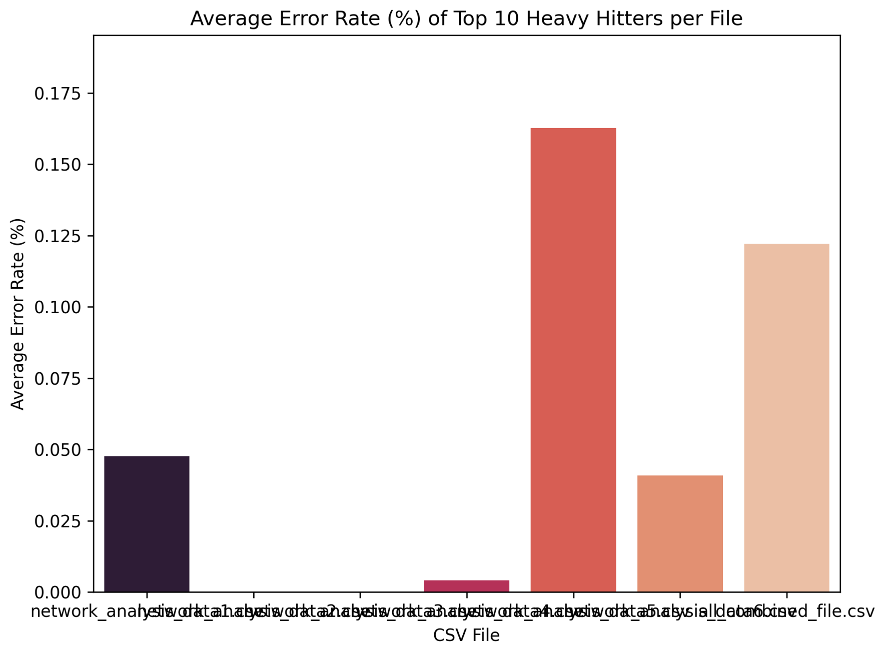


Figure 7:

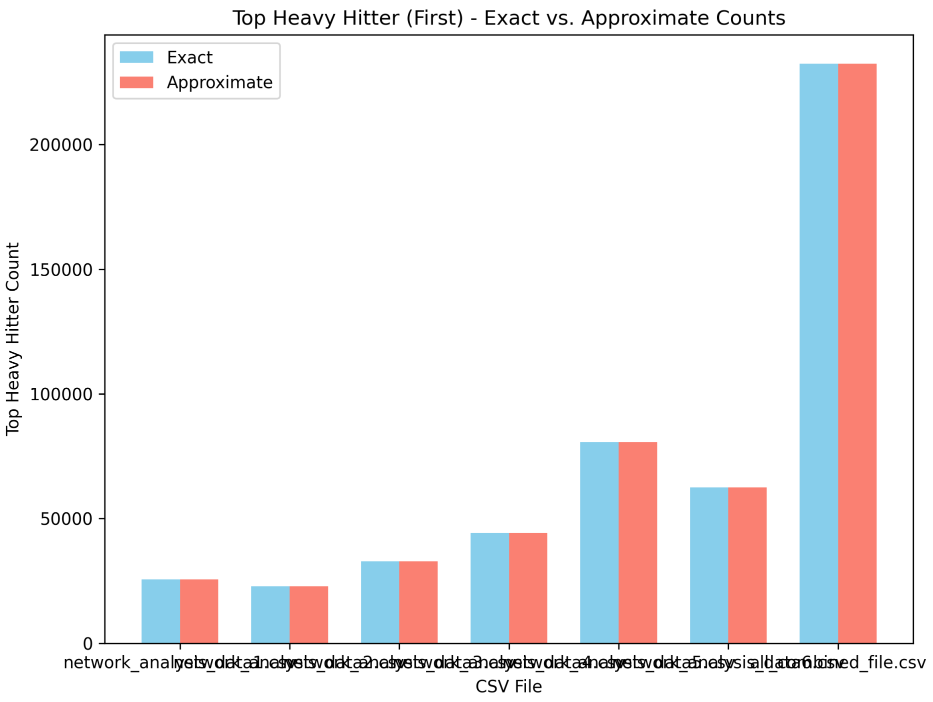
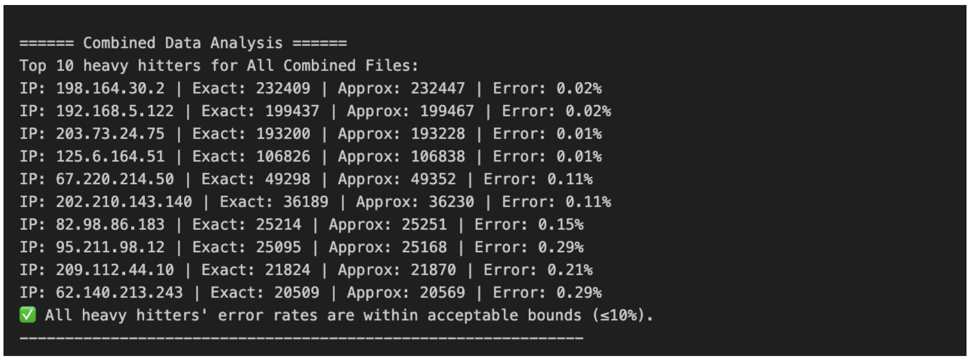


Figure 8:

3.5 Result : All outputs correspond to Identifying frequently contacted destination IPs



```
====== Combined Data Analysis ======
Top 10 heavy hitters for All Combined Files:
IP: 198.164.30.2 | Exact: 232409 | Approx: 232447 | Error: 0.02%
IP: 192.168.5.122 | Exact: 199437 | Approx: 199467 | Error: 0.02%
IP: 203.73.24.75 | Exact: 193200 | Approx: 193228 | Error: 0.01%
IP: 125.6.164.51 | Exact: 106826 | Approx: 106838 | Error: 0.01%
IP: 67.220.214.50 | Exact: 49298 | Approx: 49352 | Error: 0.11%
IP: 202.210.143.140 | Exact: 36189 | Approx: 36230 | Error: 0.11%
IP: 82.98.86.183 | Exact: 25214 | Approx: 25251 | Error: 0.15%
IP: 95.211.98.12 | Exact: 25095 | Approx: 25168 | Error: 0.29%
IP: 209.112.44.10 | Exact: 21824 | Approx: 21870 | Error: 0.21%
IP: 62.140.213.243 | Exact: 20509 | Approx: 20569 | Error: 0.29%
✅ All heavy hitters' error rates are within acceptable bounds (≤10%).
---------------------------------------------------------------
```

Figure 9:

3.6 (c) Efficient Membership Testing (Bloom Filter)(10 Marks)

3.6.1 Approach :

The goal of this analysis is to test membership in a blocklist of IP addresses using a Bloom filter, a probabilistic data structure. The Bloom filter allows us to check if an IP address is part of the blocklist with a configurable error rate, offering significant space and time efficiency for large datasets. The approach is as follows:

- **Step 1: Loading the Network Traffic Data**

  The network traffic data is read from CSV files using the `pandas` library. The required column, `source`, which contains the source IP addresses, is extracted and cleaned for use in the Bloom filter.

- **Step 2: Bloom Filter Initialization**

  A Bloom filter is initialized with a capacity equal to the number of unique IP addresses in the blocklist and a desired error rate (default: 1%). This filter is used to perform membership tests efficiently. The Bloom filter's space complexity is proportional to the number of elements and the desired error rate, which is much smaller than the size required for storing the exact set of IPs.

  The steps involved in initializing the Bloom filter include:

  - Building the blocklist by extracting unique source IPs from the dataset.
  - Adding each IP address from the blocklist to the Bloom filter.

- **Step 3: Membership Test for Known IP**

  A membership test is performed for a known IP from the blocklist. The Bloom filter is queried to check if the IP is part of the blocklist, and the result is compared with the expected outcome (True).

- **Step 4: Estimating the False Positive Rate**

  To estimate the false positive rate, a set of random IP addresses (that are not in the blocklist) is tested. For each random IP, if the Bloom filter incorrectly indicates membership, it counts as a false positive. The false positive rate is calculated based on 10,000 random IP tests, and is expressed as:

$$\text{False Positive Rate} = \frac{\text{False Positives}}{\text{Total Tests}} \times 100$$

- **Step 5: Final Analysis**

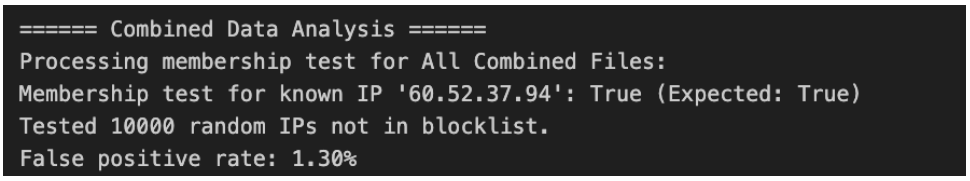
  The following analyses are performed:

  - The membership test for a known IP should always return True (i.e., the IP is part of the blocklist).
  - The false positive rate for random IPs is calculated and reported.
  - An exact membership test using a Python set (for comparison) shows 0% false positives, indicating the Bloom filter's efficiency relative to the exact method.

- **Step 6: Results and Metrics**

  The results for each file are stored and presented, including:

  - The Bloom filter's false positive rate.
  - The exact membership test result for the known IP.



```
====== Combined Data Analysis ======
Processing membership test for All Combined Files:
Membership test for known IP '60.52.37.94': True (Expected: True)
Tested 10000 random IPs not in blocklist.
False positive rate: 1.30%
```

Figure 10:

3.7 Key Findings

4 Task 3: Advanced Anomaly Detection (25 Marks)

4.1 Objective

To detect suspicious behaviors:

- **Statistical anomalies:** e.g., packet size outliers, flow count spikes, unusual protocol distributions.
- **Behavioral shifts:** IPs with sudden changes in traffic volume or pattern.
- **Suspicious communication patterns:** Long-duration connections, multi-protocol usage in short time, etc.

4.2 (a) Statistical Traffic Analysis(10 Marks)

4.2.1 Approach :

The goal of this analysis is to explore network traffic data, detect anomalies using statistical methods and machine learning techniques, and summarize key metrics for further investigation. The process can be divided into several steps:

- **Step 1: Data Cleaning**

  The network traffic data is loaded from CSV files using the `pandas` library. Initially, rows with missing values in key columns such as `startDateTime`, `source`, `destination`, and `totalSourceBytes` are dropped. The `startDateTime` column is then converted to the `datetime` format for easier time series analysis.

- **Step 2: Descriptive Statistics for Numeric Columns**

  Descriptive statistics are computed for numeric columns such as `totalSourceBytes`, `totalDestinationBytes`, `totalSourcePackets`, and `totalDestinationPackets`. The following metrics are calculated:

  - Mean, Median, Standard Deviation
  - IQR (Interquartile Range)
  - Anomalous packet identification based on mean $\pm$ 3 thresholds and IQR-based thresholds

  Anomalous packets are flagged based on these thresholds and the number of such packets is reported.

- **Step 3: Flow Counts and Thresholds Over Time**

  Time series analysis is performed on the network traffic by resampling the data to hourly and daily intervals. The following statistics are computed:

  - Hourly and Daily Flow Counts (mean, standard deviation)
  - Detection of anomalous hourly and daily flow counts based on thresholds computed as mean + 3

  Anomalous hours and days are identified by checking which time intervals exceed the computed thresholds.

- **Step 4: Outlier Detection by Traffic Volume per IP**

  Traffic volume per source IP is analyzed to identify outlier IPs. The top 10 IPs by flow count are reported, and outliers are flagged based on a threshold of mean + 3 of IP flow counts.

- **Step 5: Machine Learning-Based Anomaly Detection**

  An advanced anomaly detection technique, the Isolation Forest, is applied using features `totalSourceBytes` and `totalSourcePackets`. The following steps are taken:

  – Apply Isolation Forest to detect anomalies based on the traffic features

  – Use PCA (Principal Component Analysis) to reduce the dimensionality of the dataset

  – Visualize anomalies in a 2D PCA scatter plot with red points indicating anomalies and blue points indicating normal observations

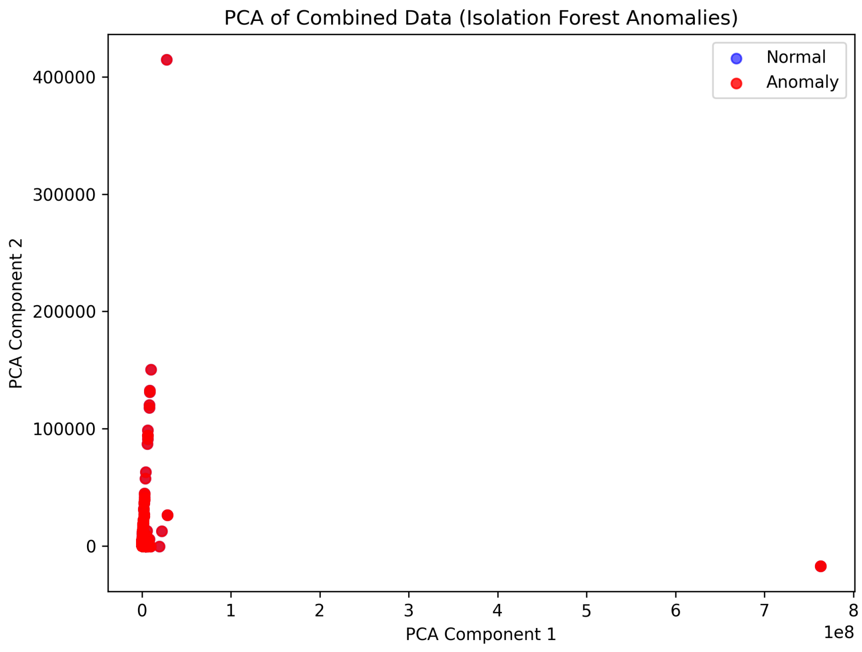


Figure 11:

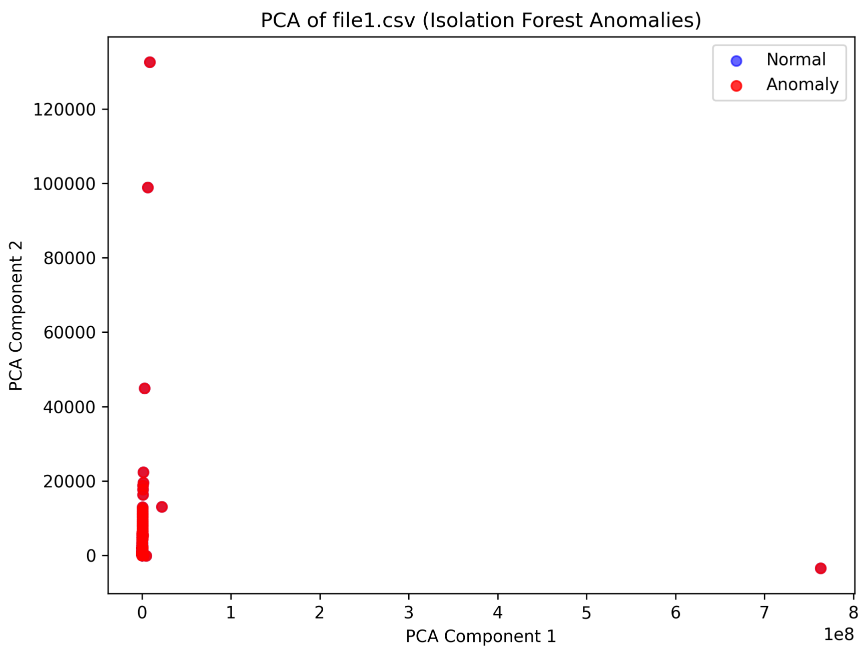


Figure 12:

A PCA scatter plot is generated and saved for each dataset to visualize the distribution of anomalies.

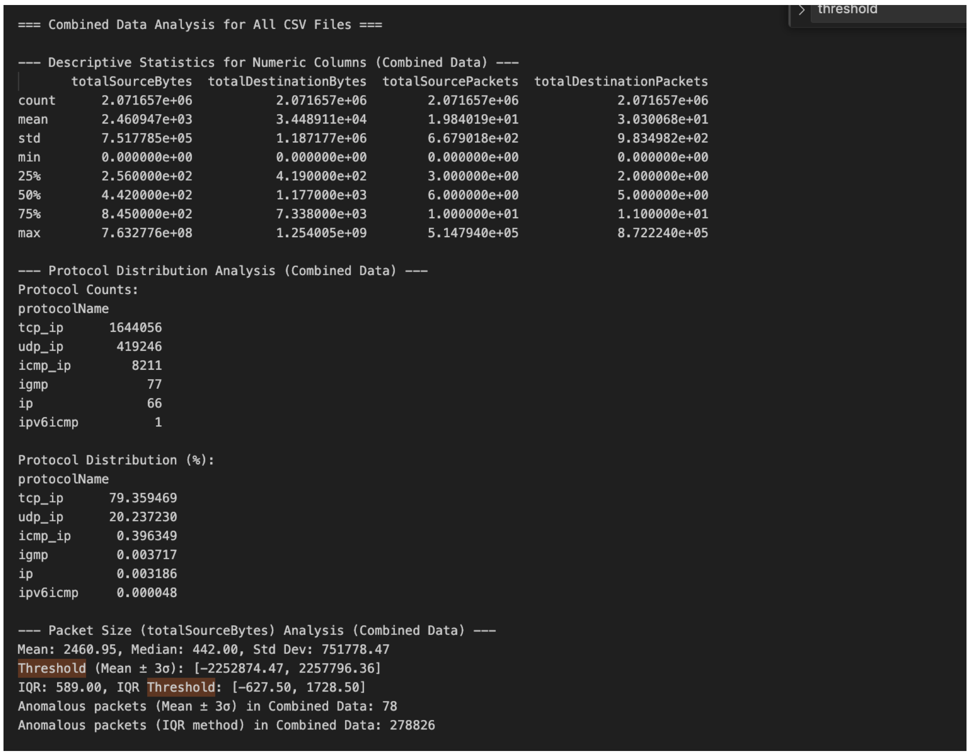
- **Step 6: Visualization**

  The results of the analysis, including anomaly detection and PCA visualizations, are plotted using `matplotlib` and `seaborn`. The following visualizations are created:

  – Flow count anomalies over time (hourly and daily)

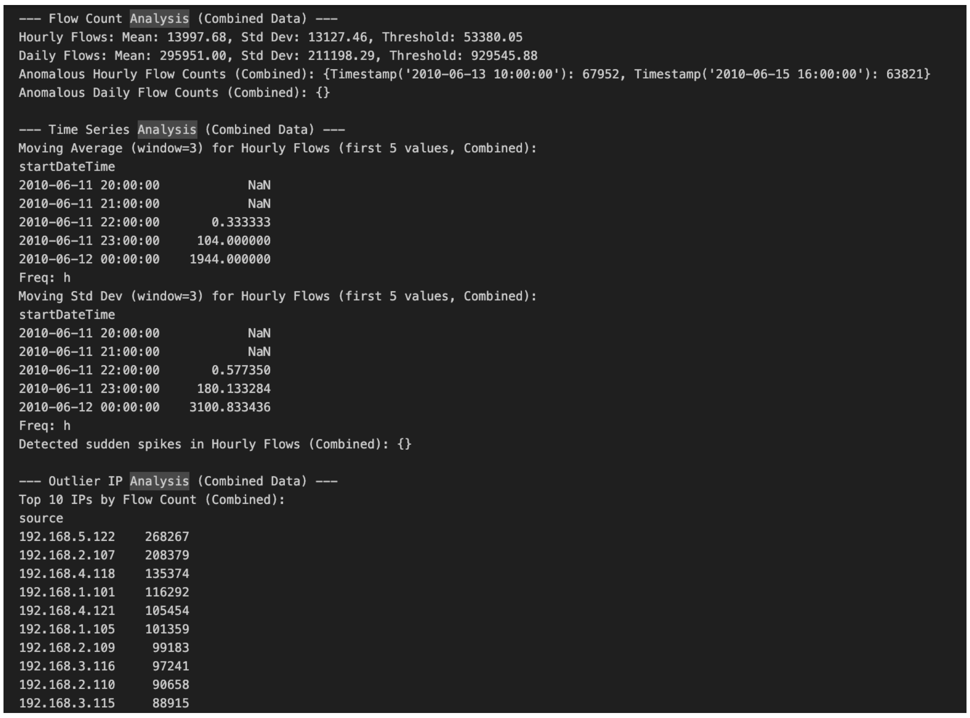
  – PCA scatter plots of network traffic data, showing anomalies detected by Isolation Forest

4.2.2 Result : All outputs correspond to Statistical Traffic Analysis



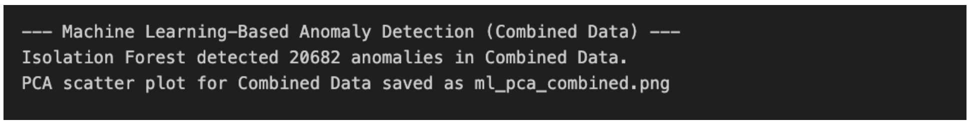
```
=== Combined Data Analysis for All CSV Files ===

--- Descriptive Statistics for Numeric Columns (Combined Data) ---
       totalSourceBytes  totalDestinationBytes  totalSourcePackets  totalDestinationPackets
count      2.071657e+06           2.071657e+06        2.071657e+06             2.071657e+06
mean       2.460947e+03           3.448911e+04        1.984019e+01             3.030068e+01
std        7.517785e+05           1.187177e+06        6.679018e+02             9.834982e+02
min        0.000000e+00           0.000000e+00        0.000000e+00             0.000000e+00
25%        2.560000e+02           4.190000e+02        3.000000e+00             2.000000e+00
50%        4.420000e+02           1.177000e+03        6.000000e+00             5.000000e+00
75%        8.450000e+02           7.338000e+03        1.000000e+01             1.100000e+01
max        7.632776e+08           1.254005e+09        5.147940e+05             8.722240e+05

--- Protocol Distribution Analysis (Combined Data) ---
Protocol Counts:
protocolName
tcp_ip      1644056
udp_ip       419246
icmp_ip        8211
igmp             77
ip               66
ipv6icmp          1

Protocol Distribution (%):
protocolName
tcp_ip      79.359469
udp_ip      20.237230
icmp_ip      0.396349
igmp         0.003717
ip           0.003186
ipv6icmp     0.000048

--- Packet Size (totalSourceBytes) Analysis (Combined Data) ---
Mean: 2460.95, Median: 442.00, Std Dev: 751778.47
Threshold (Mean ± 3σ): [-2252874.47, 2257796.36]
IQR: 589.00, IQR Threshold: [-627.50, 1728.50]
Anomalous packets (Mean ± 3σ) in Combined Data: 78
Anomalous packets (IQR method) in Combined Data: 278826
```

Figure 13:



```
--- Flow Count Analysis (Combined Data) ---
Hourly Flows: Mean: 13997.68, Std Dev: 13127.46, Threshold: 53380.05
Daily Flows: Mean: 295951.00, Std Dev: 211198.29, Threshold: 929545.88
Anomalous Hourly Flow Counts (Combined): {Timestamp('2010-06-13 10:00:00'): 67952, Timestamp('2010-06-15 16:00:00'): 63821}
Anomalous Daily Flow Counts (Combined): {}

--- Time Series Analysis (Combined Data) ---
Moving Average (window=3) for Hourly Flows (first 5 values, Combined):
startDateTime
2010-06-11 20:00:00            NaN
2010-06-11 21:00:00            NaN
2010-06-11 22:00:00       0.333333
2010-06-11 23:00:00     104.000000
2010-06-12 00:00:00    1944.000000
Freq: h
Moving Std Dev (window=3) for Hourly Flows (first 5 values, Combined):
startDateTime
2010-06-11 20:00:00            NaN
2010-06-11 21:00:00            NaN
2010-06-11 22:00:00       0.577350
2010-06-11 23:00:00     180.133284
2010-06-12 00:00:00    3100.833436
Freq: h
Detected sudden spikes in Hourly Flows (Combined): {}

--- Outlier IP Analysis (Combined Data) ---
Top 10 IPs by Flow Count (Combined):
source
192.168.5.122    268267
192.168.2.107    208379
192.168.4.118    135374
192.168.1.101    116292
192.168.4.121    105454
192.168.1.105    101359
192.168.2.109     99183
192.168.3.116     97241
192.168.2.110     90658
192.168.3.115     88915
```

Figure 14:



```
--- Machine Learning-Based Anomaly Detection (Combined Data) ---
Isolation Forest detected 20682 anomalies in Combined Data.
PCA scatter plot for Combined Data saved as ml_pca_combined.png
```

Figure 15:

4.3 (b) Behavioral Analysis (10 Marks)

The goal of this analysis is to detect suspicious communication patterns in network traffic data. We focus on two main aspects: long-duration connections and multiple protocols used in short time intervals. The steps involved are as follows:

1. **Data Loading and Preprocessing:**

   - CSV files matching a specified file pattern are loaded using Dask's `read_csv` function. Key columns are selected (by default, these include `startDateTime`, `stopDateTime`, `source`, `destination`, `totalSourceBytes`, and `totalDestinationBytes`).
   - A new column, `totalTraffic`, is computed for each flow as:

     $$\text{totalTraffic} = \text{totalSourceBytes} + \text{totalDestinationBytes}$$

   - The Dask dataframe is then computed into a Pandas DataFrame and sorted by `source` and `startDateTime` for subsequent analysis.

2. **Detection of Behavioral Anomalies:** The analysis implements three primary detection mechanisms:

   (a) **Sudden Traffic Spikes (R1):**

   - For each source IP, the time gap between consecutive flows is computed.
   - A flow is flagged as a sudden traffic spike if:

     $$\text{time\_gap} > \text{inactivity threshold (e.g., 3600 seconds)}$$

     and the `totalTraffic` exceeds a dynamic threshold, defined for each source as:

     $$\text{Threshold} = \text{mean(totalTraffic)} + 3 \times \text{std(totalTraffic)}$$

   (b) **Common Target Detection (R2):**

   - The data is grouped into 10-minute buckets by flooring the `startDateTime` to the nearest 10 minutes.
   - For each destination IP and time bucket, the number of unique source IPs is counted.
   - Destinations with a unique source count greater than a specified threshold (e.g., 5) are flagged as common targets.

   (c) **High Fan-in Destinations (R3):**

   - Similar to common target detection, the data is grouped into 10-minute intervals.

    - For each destination IP, if the unique source count exceeds a higher threshold (e.g., 50), the destination is flagged as having high fan-in, indicating potential malicious aggregation of traffic.

3. **Visualization and Reporting:**

   - Three types of graphs are generated:

     (a) A scatter plot showing sudden traffic spikes with `startDateTime` versus `totalTraffic`.

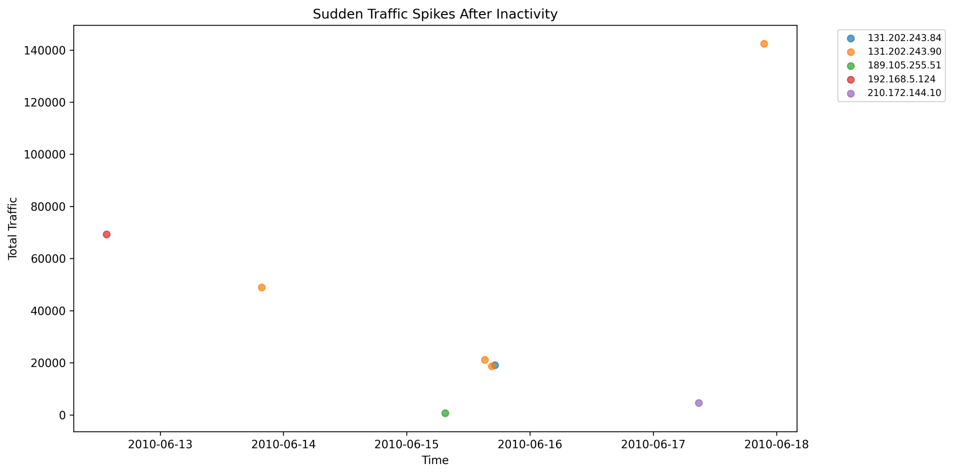


Figure 16:

     (b) A bar chart of common targets, displaying the top 10 destinations by total unique source counts within 10-minute windows.

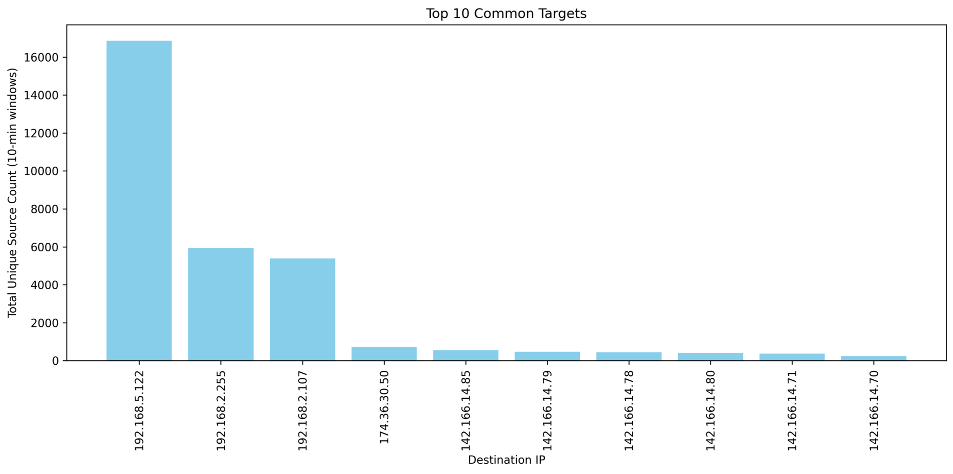


Figure 17:

     (c) A bar chart for high fan-in destinations, highlighting the top 10 destinations with the highest unique source counts.

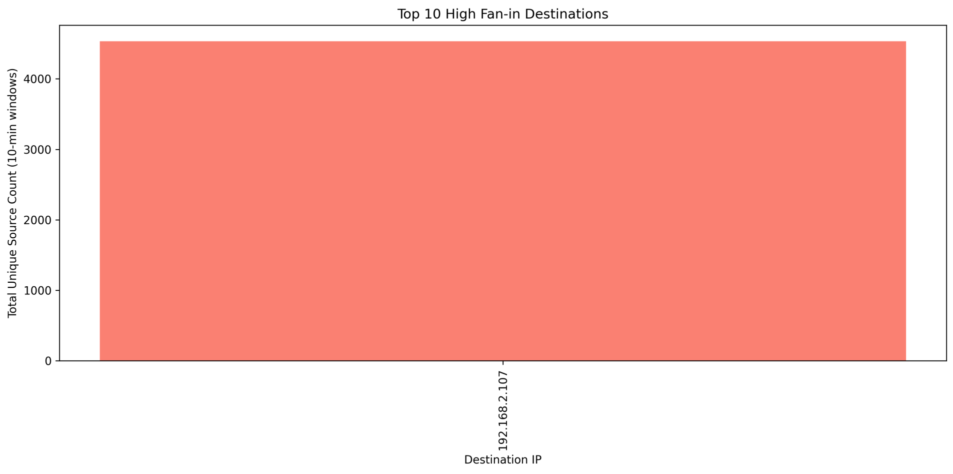
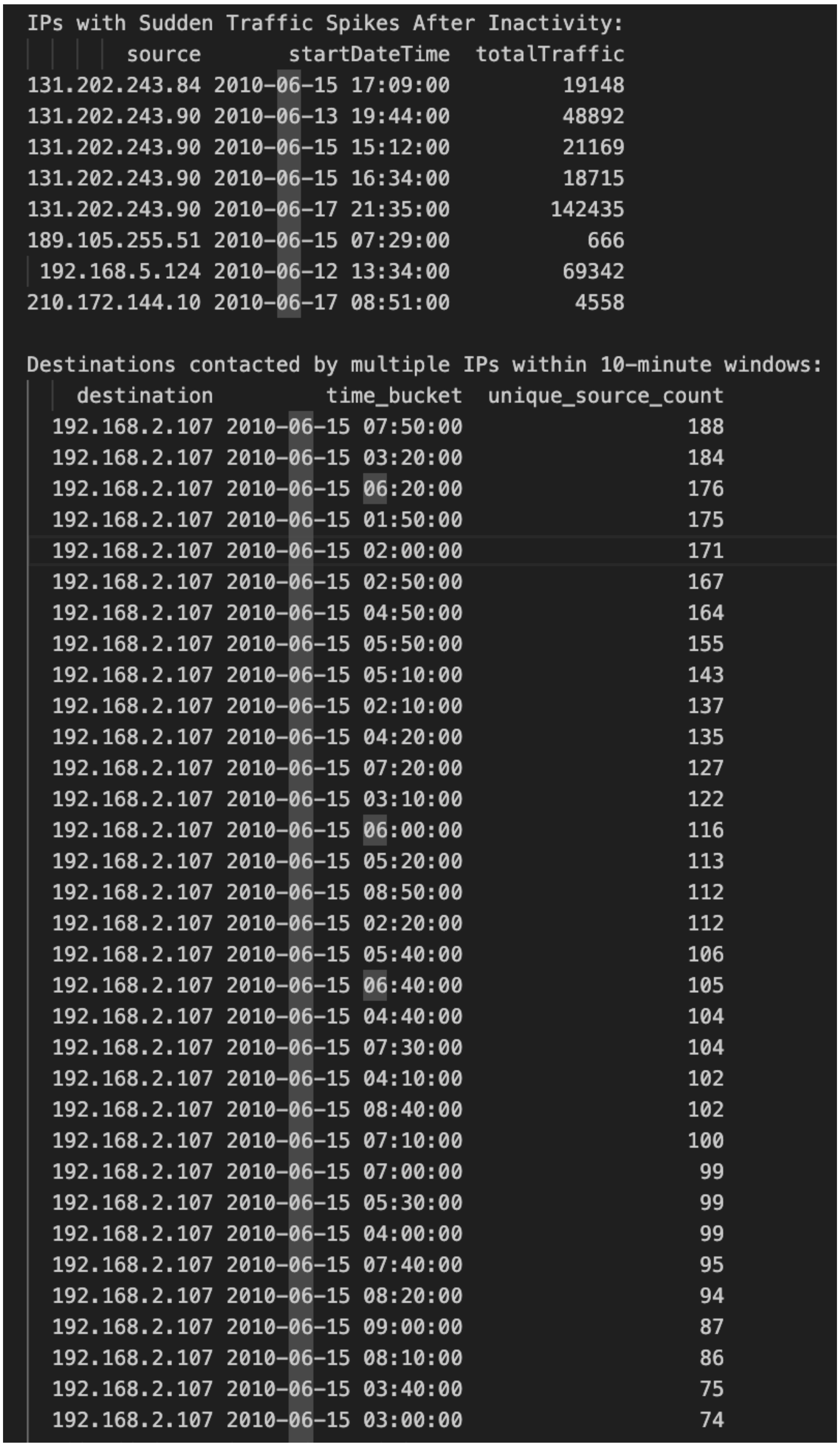


Figure 18:

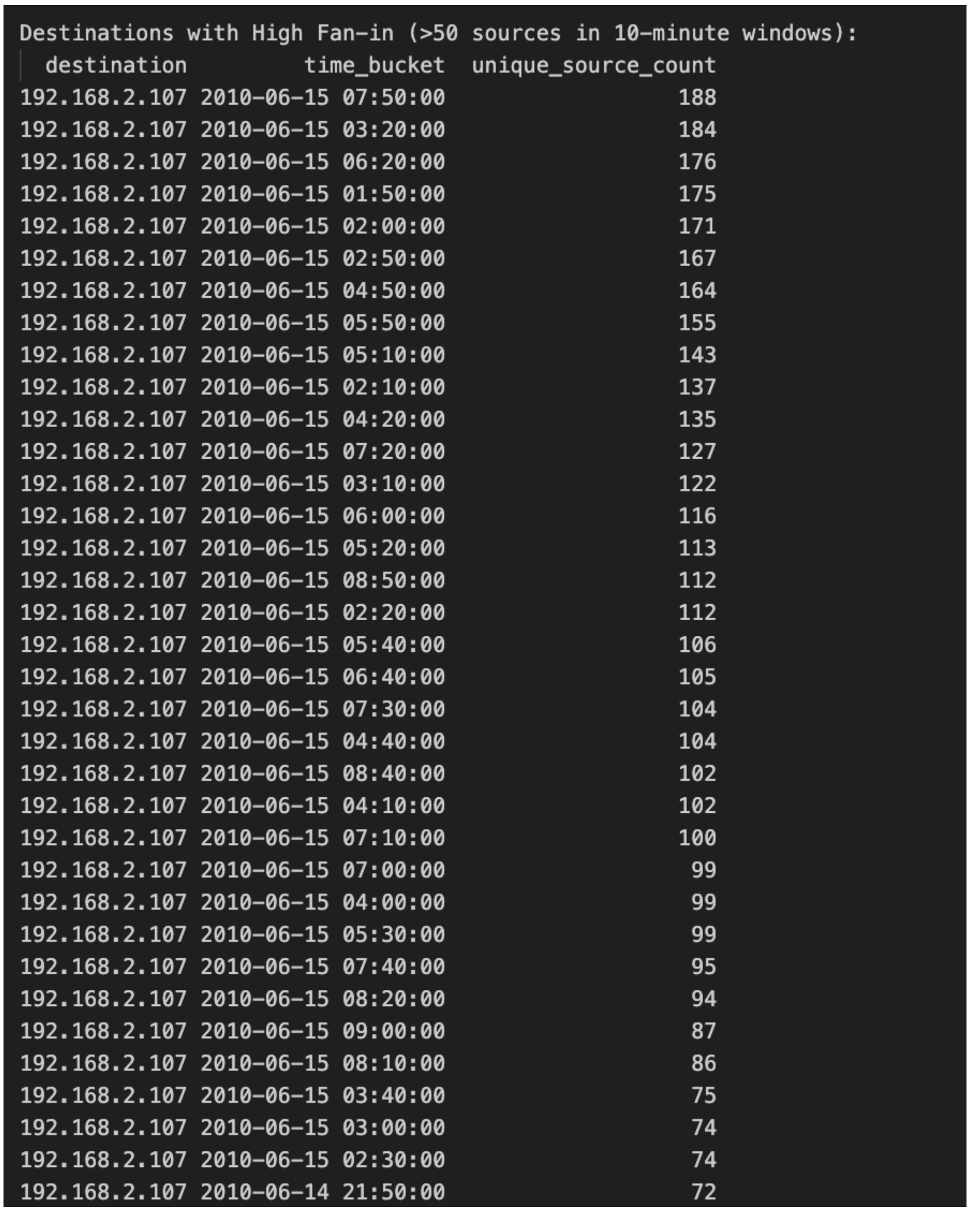
- The results from each detection method are consolidated and written to an output text file. Detailed reports include the list of flagged IPs and destinations, as well as statistical summaries.

4.3.1 Result : All output corresponds to Behavioral Analysis



```
IPs with Sudden Traffic Spikes After Inactivity:
            source       startDateTime  totalTraffic
131.202.243.84 2010-06-15 17:09:00         19148
131.202.243.90 2010-06-13 19:44:00         48892
131.202.243.90 2010-06-15 15:12:00         21169
131.202.243.90 2010-06-15 16:34:00         18715
131.202.243.90 2010-06-17 21:35:00        142435
189.105.255.51 2010-06-15 07:29:00           666
 192.168.5.124 2010-06-12 13:34:00         69342
210.172.144.10 2010-06-17 08:51:00          4558

Destinations contacted by multiple IPs within 10-minute windows:
      destination         time_bucket  unique_source_count
  192.168.2.107 2010-06-15 07:50:00                    188
  192.168.2.107 2010-06-15 03:20:00                    184
  192.168.2.107 2010-06-15 06:20:00                    176
  192.168.2.107 2010-06-15 01:50:00                    175
  192.168.2.107 2010-06-15 02:00:00                    171
  192.168.2.107 2010-06-15 02:50:00                    167
  192.168.2.107 2010-06-15 04:50:00                    164
  192.168.2.107 2010-06-15 05:50:00                    155
  192.168.2.107 2010-06-15 05:10:00                    143
  192.168.2.107 2010-06-15 02:10:00                    137
  192.168.2.107 2010-06-15 04:20:00                    135
  192.168.2.107 2010-06-15 07:20:00                    127
  192.168.2.107 2010-06-15 03:10:00                    122
  192.168.2.107 2010-06-15 06:00:00                    116
  192.168.2.107 2010-06-15 05:20:00                    113
  192.168.2.107 2010-06-15 08:50:00                    112
  192.168.2.107 2010-06-15 02:20:00                    112
  192.168.2.107 2010-06-15 05:40:00                    106
  192.168.2.107 2010-06-15 06:40:00                    105
  192.168.2.107 2010-06-15 04:40:00                    104
  192.168.2.107 2010-06-15 07:30:00                    104
  192.168.2.107 2010-06-15 04:10:00                    102
  192.168.2.107 2010-06-15 08:40:00                    102
  192.168.2.107 2010-06-15 07:10:00                    100
  192.168.2.107 2010-06-15 07:00:00                     99
  192.168.2.107 2010-06-15 05:30:00                     99
  192.168.2.107 2010-06-15 04:00:00                     99
  192.168.2.107 2010-06-15 07:40:00                     95
  192.168.2.107 2010-06-15 08:20:00                     94
  192.168.2.107 2010-06-15 09:00:00                     87
  192.168.2.107 2010-06-15 08:10:00                     86
  192.168.2.107 2010-06-15 03:40:00                     75
  192.168.2.107 2010-06-15 03:00:00                     74
```

Figure 19:



```
Destinations with High Fan-in (>50 sources in 10-minute windows):
  destination         time_bucket  unique_source_count
192.168.2.107 2010-06-15 07:50:00                  188
192.168.2.107 2010-06-15 03:20:00                  184
192.168.2.107 2010-06-15 06:20:00                  176
192.168.2.107 2010-06-15 01:50:00                  175
192.168.2.107 2010-06-15 02:00:00                  171
192.168.2.107 2010-06-15 02:50:00                  167
192.168.2.107 2010-06-15 04:50:00                  164
192.168.2.107 2010-06-15 05:50:00                  155
192.168.2.107 2010-06-15 05:10:00                  143
192.168.2.107 2010-06-15 02:10:00                  137
192.168.2.107 2010-06-15 04:20:00                  135
192.168.2.107 2010-06-15 07:20:00                  127
192.168.2.107 2010-06-15 03:10:00                  122
192.168.2.107 2010-06-15 06:00:00                  116
192.168.2.107 2010-06-15 05:20:00                  113
192.168.2.107 2010-06-15 08:50:00                  112
192.168.2.107 2010-06-15 02:20:00                  112
192.168.2.107 2010-06-15 05:40:00                  106
192.168.2.107 2010-06-15 06:40:00                  105
192.168.2.107 2010-06-15 07:30:00                  104
192.168.2.107 2010-06-15 04:40:00                  104
192.168.2.107 2010-06-15 08:40:00                  102
192.168.2.107 2010-06-15 04:10:00                  102
192.168.2.107 2010-06-15 07:10:00                  100
192.168.2.107 2010-06-15 07:00:00                   99
192.168.2.107 2010-06-15 04:00:00                   99
192.168.2.107 2010-06-15 05:30:00                   99
192.168.2.107 2010-06-15 07:40:00                   95
192.168.2.107 2010-06-15 08:20:00                   94
192.168.2.107 2010-06-15 09:00:00                   87
192.168.2.107 2010-06-15 08:10:00                   86
192.168.2.107 2010-06-15 03:40:00                   75
192.168.2.107 2010-06-15 03:00:00                   74
192.168.2.107 2010-06-15 02:30:00                   74
192.168.2.107 2010-06-14 21:50:00                   72
```

Figure 20:

4.4 (c) Suspicious Communication Patterns (5 Marks)

4.4.1 Approach :

1. **Data Loading and Preprocessing:**

   - CSV files matching a specified file pattern (e.g., `../network_analysis_data*.csv`) are read using `pandas`.
   - Only selected columns are loaded (by default, `startDateTime`, `stopDateTime`, `source`, `destination`, `totalSourceBytes`, `totalDestinationBytes`, and `protocolName`), with the date columns parsed into datetime objects.
   - A new field, `duration`, is computed as the time difference (in seconds) between `stopDateTime` and `startDateTime`.

2. **Detection of Long-Duration Connections:**

- The mean and standard deviation of the connection durations are calculated.
- A threshold is determined using:

$$\text{Threshold Duration} = \text{Mean Duration} + 3 \times \text{Standard Deviation}$$

- Flows exceeding this threshold are flagged as long-duration connections. A histogram of durations is also generated and saved as a graph.

3. **Detection of Multiple Protocol Usage in Short Time Windows:**

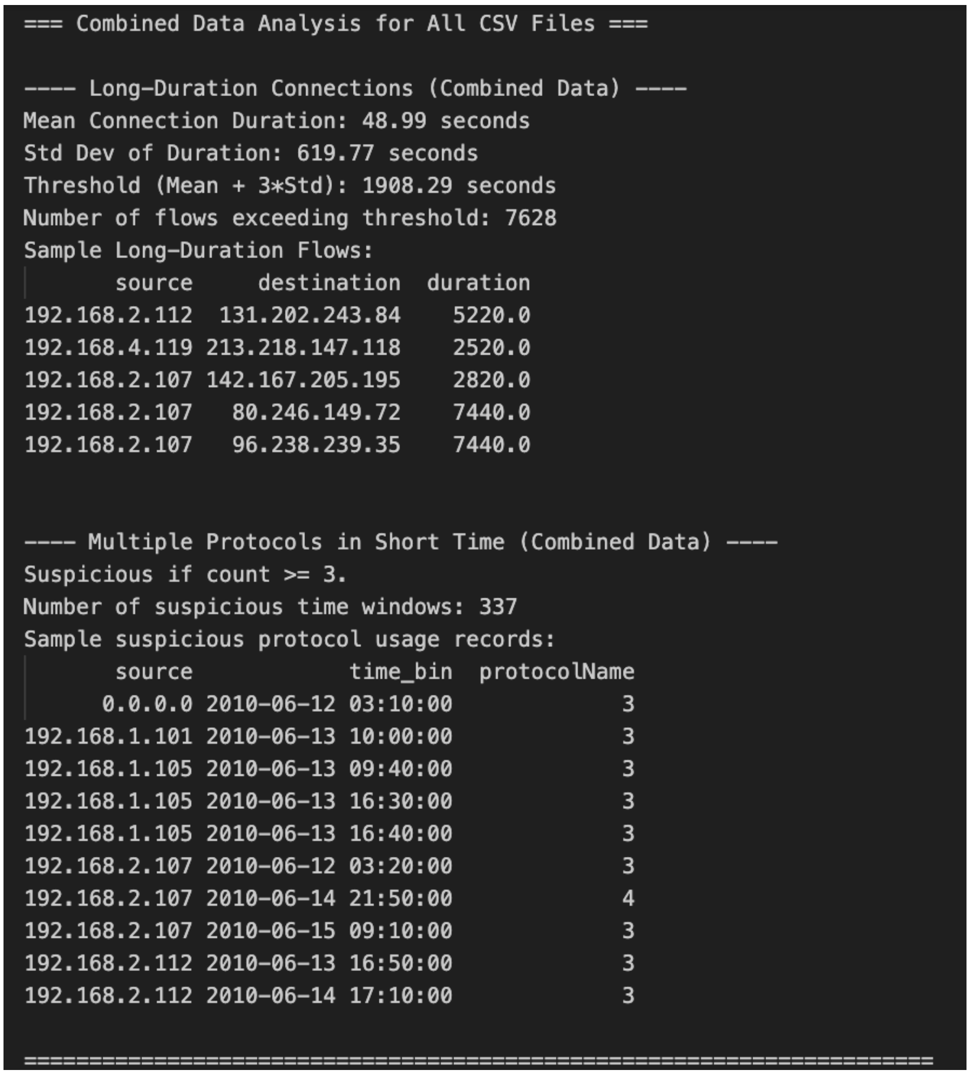
   - The data is grouped into 10-minute time bins by flooring the `startDateTime`.
   - For each source IP and time bin, the number of unique protocols used is computed.
   - Time windows where a source IP uses three or more distinct protocols are flagged as suspicious. A histogram of the protocol diversity is plotted to visualize the distribution.

4. **Combined Analysis Across Multiple Files:**

   - The analyzer also performs a combined analysis by concatenating data from all CSV files.
   - The same analyses (long-duration connection detection and multiple protocols in short time windows) are applied to the combined dataset.
   - Summary statistics and sample records are generated, and corresponding graphs are saved.

5. **Reporting and Graph Generation:**

   - For each individual file and for the combined data, detailed results (such as the mean, standard deviation, thresholds, and counts of flagged events) are appended to an output report.



```
=== Combined Data Analysis for All CSV Files ===

---- Long-Duration Connections (Combined Data) ----
Mean Connection Duration: 48.99 seconds
Std Dev of Duration: 619.77 seconds
Threshold (Mean + 3*Std): 1908.29 seconds
Number of flows exceeding threshold: 7628
Sample Long-Duration Flows:
       source     destination  duration
192.168.2.112  131.202.243.84    5220.0
192.168.4.119 213.218.147.118    2520.0
192.168.2.107 142.167.205.195    2820.0
192.168.2.107   80.246.149.72    7440.0
192.168.2.107   96.238.239.35    7440.0


---- Multiple Protocols in Short Time (Combined Data) ----
Suspicious if count >= 3.
Number of suspicious time windows: 337
Sample suspicious protocol usage records:
       source            time_bin  protocolName
      0.0.0.0 2010-06-12 03:10:00             3
192.168.1.101 2010-06-13 10:00:00             3
192.168.1.105 2010-06-13 09:40:00             3
192.168.1.105 2010-06-13 16:30:00             3
192.168.1.105 2010-06-13 16:40:00             3
192.168.2.107 2010-06-12 03:20:00             3
192.168.2.107 2010-06-14 21:50:00             4
192.168.2.107 2010-06-15 09:10:00             3
192.168.2.112 2010-06-13 16:50:00             3
192.168.2.112 2010-06-14 17:10:00             3

======================================================================
```

Figure 21:

- Sample records from the flagged long-duration flows and suspicious protocol usage are included.
- Graphs (e.g., histograms of connection durations and protocol diversity) are generated and stored in a designated directory.

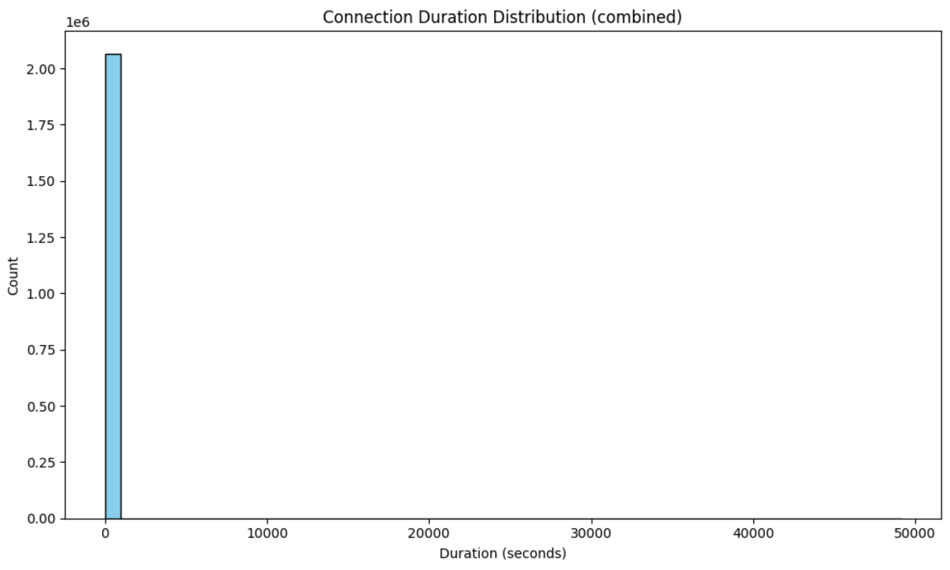


Figure 22:

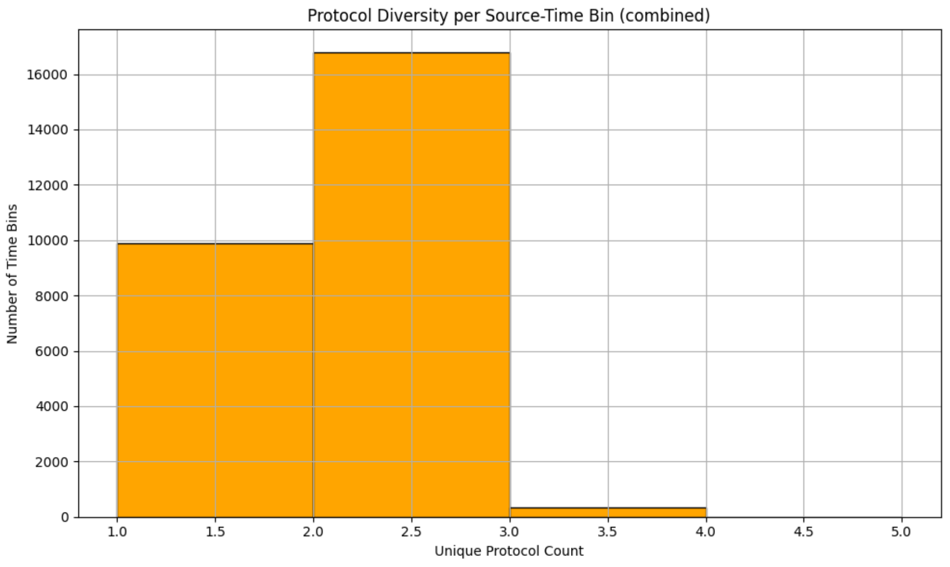


Figure 23:

5 Task 4: Deep Security Threat Analysis (25 Marks)

5.1 Objective

Beyond simple anomaly flags, investigate complex attacks, malicious payloads, and produce a risk attribution report.

5.2 (a) Detecting Complex Attack Patterns (10 Marks)

The goal of this analysis is to detect advanced network threats such as stealthy port scans, slow DDoS attacks, and IP hopping behavior. The approach can be broken down into the following steps:

5.2.1 Approach :

- **Step 1: Stealthy Port Scan Detection**

  The detection of stealthy port scans is performed by analyzing the number of distinct destination ports contacted by each source IP over the entire period and within 24-hour windows. A threshold is calculated using the mean and standard deviation of the distinct destination ports per source IP. Source IPs that exceed this threshold are flagged as potential stealthy port scanners.

  $$\text{Threshold} = \text{Mean Distinct Ports} + 3 \times \text{Std Distinct Ports}$$

  A 24-hour window analysis is performed to capture potential stealthy port scans over time, using a moving average and standard deviation to compute a dynamic threshold for each source IP.

- **Step 2: Slow DDoS Attack Detection**

  Slow DDoS attacks are detected by monitoring the flow counts per destination IP over time. The flow counts are resampled by hour, and a threshold is set using the mean and standard deviation of hourly flow counts. Destinations with flow counts exceeding this threshold are flagged as potential slow DDoS attack targets.

  $$\text{Threshold for Flow Count} = \text{Mean Flow Count} + 3 \times \text{Std Flow Count}$$

  This helps to identify destinations that experience a gradual increase in traffic, which is characteristic of slow DDoS attacks.

- **Step 3: IP Hopping Behavior Detection**

  IP hopping behavior is identified by tracking the number of distinct source IPs contacting each destination within short time windows (e.g., 10-minute intervals). If the number of distinct source IPs exceeds a predefined threshold (e.g., 5 distinct IPs), the destination is flagged as a potential IP hopping target.

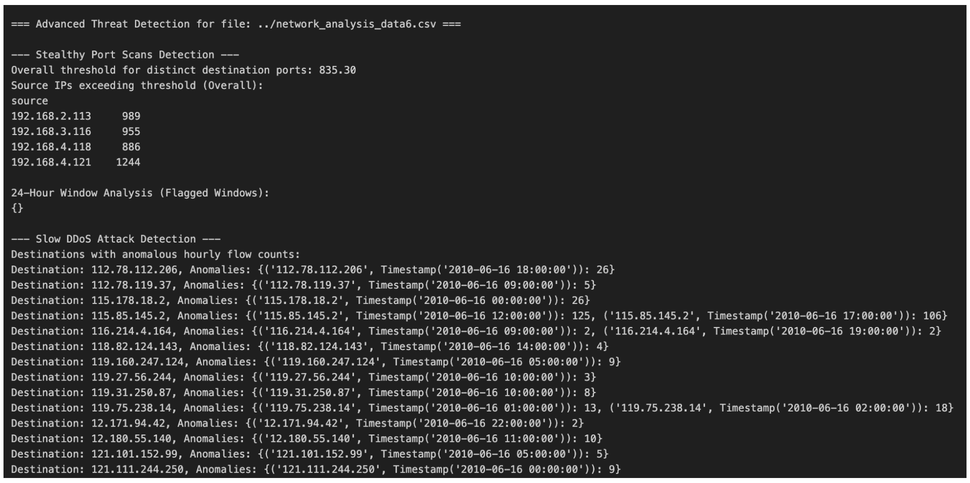
  $$\text{Rule: More than 5 distinct source IPs within 10 minutes}$$

  This approach helps identify suspicious behaviors where an attacker might use multiple IP addresses to contact a single destination.

- **Step 4: Reporting and Analysis**

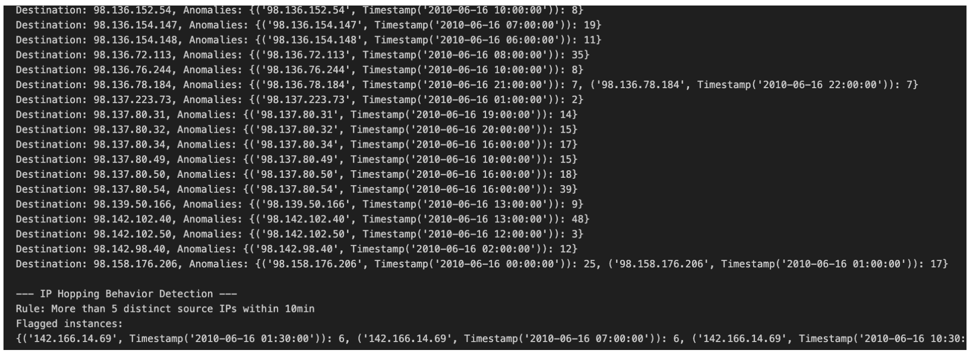
  For each of the three detection methods, detailed reports are generated that include the detected anomalies, flagged IPs, and thresholds used for detection. These reports help network administrators to identify and investigate potential network threats, such as port scans, slow DDoS attacks, and IP hopping patterns.

  The results are aggregated and outputted in a comprehensive text file, providing insights into the detected threats and their characteristics.



```
=== Advanced Threat Detection for file: ../network_analysis_data6.csv ===

--- Stealthy Port Scans Detection ---
Overall threshold for distinct destination ports: 835.30
Source IPs exceeding threshold (Overall):
source
192.168.2.113     989
192.168.3.116     955
192.168.4.118     886
192.168.4.121    1244

24-Hour Window Analysis (Flagged Windows):
{}

--- Slow DDoS Attack Detection ---
Destinations with anomalous hourly flow counts:
Destination: 112.78.112.206, Anomalies: {('112.78.112.206', Timestamp('2010-06-16 18:00:00')): 26}
Destination: 112.78.119.37, Anomalies: {('112.78.119.37', Timestamp('2010-06-16 09:00:00')): 5}
Destination: 115.178.18.2, Anomalies: {('115.178.18.2', Timestamp('2010-06-16 00:00:00')): 26}
Destination: 115.85.145.2, Anomalies: {('115.85.145.2', Timestamp('2010-06-16 12:00:00')): 125, ('115.85.145.2', Timestamp('2010-06-16 17:00:00')): 106}
Destination: 116.214.4.164, Anomalies: {('116.214.4.164', Timestamp('2010-06-16 09:00:00')): 2, ('116.214.4.164', Timestamp('2010-06-16 19:00:00')): 2}
Destination: 118.82.124.143, Anomalies: {('118.82.124.143', Timestamp('2010-06-16 14:00:00')): 4}
Destination: 119.160.247.124, Anomalies: {('119.160.247.124', Timestamp('2010-06-16 05:00:00')): 9}
Destination: 119.27.56.244, Anomalies: {('119.27.56.244', Timestamp('2010-06-16 10:00:00')): 3}
Destination: 119.31.250.87, Anomalies: {('119.31.250.87', Timestamp('2010-06-16 10:00:00')): 8}
Destination: 119.75.238.14, Anomalies: {('119.75.238.14', Timestamp('2010-06-16 01:00:00')): 13, ('119.75.238.14', Timestamp('2010-06-16 02:00:00')): 18}
Destination: 12.171.94.42, Anomalies: {('12.171.94.42', Timestamp('2010-06-16 22:00:00')): 2}
Destination: 12.180.55.140, Anomalies: {('12.180.55.140', Timestamp('2010-06-16 11:00:00')): 10}
Destination: 121.101.152.99, Anomalies: {('121.101.152.99', Timestamp('2010-06-16 05:00:00')): 5}
Destination: 121.111.244.250, Anomalies: {('121.111.244.250', Timestamp('2010-06-16 00:00:00')): 9}
```

Figure 24:



```
Destination: 98.136.152.54, Anomalies: {('98.136.152.54', Timestamp('2010-06-16 10:00:00')): 8}
Destination: 98.136.154.147, Anomalies: {('98.136.154.147', Timestamp('2010-06-16 07:00:00')): 19}
Destination: 98.136.154.148, Anomalies: {('98.136.154.148', Timestamp('2010-06-16 06:00:00')): 11}
Destination: 98.136.72.113, Anomalies: {('98.136.72.113', Timestamp('2010-06-16 08:00:00')): 35}
Destination: 98.136.76.244, Anomalies: {('98.136.76.244', Timestamp('2010-06-16 10:00:00')): 8}
Destination: 98.136.78.184, Anomalies: {('98.136.78.184', Timestamp('2010-06-16 21:00:00')): 7, ('98.136.78.184', Timestamp('2010-06-16 22:00:00')): 7}
Destination: 98.137.223.73, Anomalies: {('98.137.223.73', Timestamp('2010-06-16 01:00:00')): 2}
Destination: 98.137.80.31, Anomalies: {('98.137.80.31', Timestamp('2010-06-16 19:00:00')): 14}
Destination: 98.137.80.32, Anomalies: {('98.137.80.32', Timestamp('2010-06-16 20:00:00')): 15}
Destination: 98.137.80.34, Anomalies: {('98.137.80.34', Timestamp('2010-06-16 16:00:00')): 17}
Destination: 98.137.80.49, Anomalies: {('98.137.80.49', Timestamp('2010-06-16 10:00:00')): 15}
Destination: 98.137.80.50, Anomalies: {('98.137.80.50', Timestamp('2010-06-16 16:00:00')): 18}
Destination: 98.137.80.54, Anomalies: {('98.137.80.54', Timestamp('2010-06-16 16:00:00')): 39}
Destination: 98.139.50.166, Anomalies: {('98.139.50.166', Timestamp('2010-06-16 13:00:00')): 9}
Destination: 98.142.102.40, Anomalies: {('98.142.102.40', Timestamp('2010-06-16 13:00:00')): 48}
Destination: 98.142.102.50, Anomalies: {('98.142.102.50', Timestamp('2010-06-16 12:00:00')): 3}
Destination: 98.142.98.40, Anomalies: {('98.142.98.40', Timestamp('2010-06-16 02:00:00')): 12}
Destination: 98.158.176.206, Anomalies: {('98.158.176.206', Timestamp('2010-06-16 00:00:00')): 25, ('98.158.176.206', Timestamp('2010-06-16 01:00:00')): 17}

--- IP Hopping Behavior Detection ---
Rule: More than 5 distinct source IPs within 10min
Flagged instances:
{('142.166.14.69', Timestamp('2010-06-16 01:30:00')): 6, ('142.166.14.69', Timestamp('2010-06-16 07:00:00')): 6, ('142.166.14.69', Timestamp('2010-06-16 10:30:
```

Figure 25:

5.3 (b) Malicious Payload Identification (10 Marks)

The objective of this analysis is to extract features from network payloads and apply various methods for anomaly detection, including statistical analysis, machine learning, and signature-based methods. The approach is divided into the following steps:

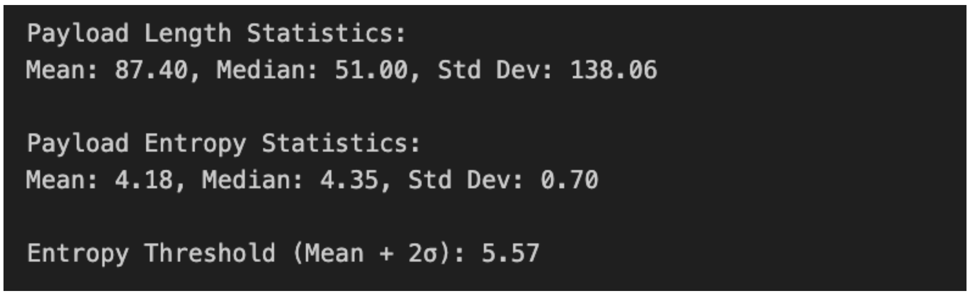
1. **Data Loading and Preprocessing:**

   - Network traffic data is loaded from CSV files (using a specified file pattern) with selected columns such as `sourcePayloadAsBase64`, `destinationPayloadAsBase64`, `protocolName`, and others.
   - The payloads are decoded from Base64 to UTF-8 and cleaned by removing non-printable characters. This ensures that the payload data is in a usable text format.
   - If not already present, an `encrypted` flag is computed for each flow using a custom function that examines the `protocolName`, the presence/absence of UTF and Base64 payloads, and common encrypted ports (e.g., 443).

2. **Feature Extraction:**

   - **Payload Length:** For each decoded payload, the length is computed.
   - **Shannon Entropy:** The randomness (or unpredictability) of the payload is quantified by calculating its Shannon entropy.

- Descriptive statistics (mean, median, standard deviation) are calculated for both payload length and entropy.



```
Payload Length Statistics:
Mean: 87.40, Median: 51.00, Std Dev: 138.06

Payload Entropy Statistics:
Mean: 4.18, Median: 4.35, Std Dev: 0.70

Entropy Threshold (Mean + 2σ): 5.57
```

Figure 26:

- A threshold is set for high entropy payloads using the formula:

$$\text{Entropy Threshold} = \text{Mean Entropy} + 2 \times \text{Std Dev of Entropy}$$

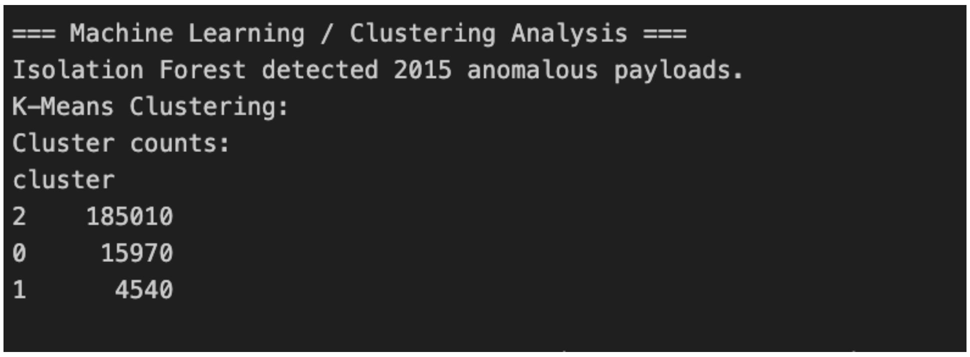
  Payloads exceeding this threshold are flagged as anomalous.

3. **Rule-Based Keyword and Signature Matching:**

   - A rule-based search identifies suspicious payloads by checking for specific keywords (e.g., `cmd`, `control`, `update`, `ping`, `malware`, `attack`).
   - Additionally, regular expressions are employed to match known malicious patterns, flagging payloads that resemble command-and-control (C&C) activity.

4. **Machine Learning-Based Anomaly Detection:**

   - **Isolation Forest:** An Isolation Forest is applied to the feature vectors (constructed from payload length and entropy) to detect outlier payloads.
   - **K-Means Clustering:** The payloads are also grouped into clusters using K-Means clustering. Small clusters (i.e., clusters with a size below 5% of the total payloads) are flagged as potential anomalies.

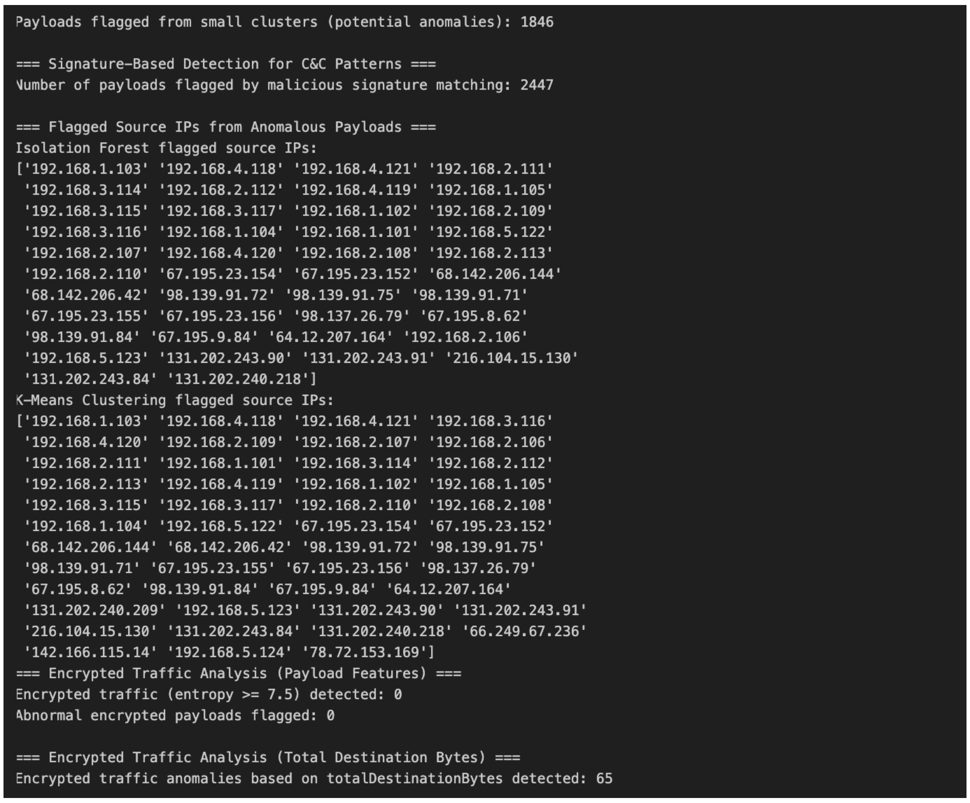


```
=== Machine Learning / Clustering Analysis ===
Isolation Forest detected 2015 anomalous payloads.
K-Means Clustering:
Cluster counts:
cluster
2    185010
0     15970
1      4540
```

Figure 27:

5. **Encrypted Traffic Analysis:**

- In addition to using the entropy-based threshold for anomaly detection, a specialized function further examines payloads with high entropy using an Isolation Forest to detect abnormal encrypted traffic.



```
Payloads flagged from small clusters (potential anomalies): 1846

=== Signature-Based Detection for C&C Patterns ===
Number of payloads flagged by malicious signature matching: 2447

=== Flagged Source IPs from Anomalous Payloads ===
Isolation Forest flagged source IPs:
['192.168.1.103' '192.168.4.118' '192.168.4.121' '192.168.2.111'
 '192.168.3.114' '192.168.2.112' '192.168.4.119' '192.168.1.105'
 '192.168.3.115' '192.168.3.117' '192.168.1.102' '192.168.2.109'
 '192.168.3.116' '192.168.1.104' '192.168.1.101' '192.168.5.122'
 '192.168.2.107' '192.168.4.120' '192.168.2.108' '192.168.2.113'
 '192.168.2.110' '67.195.23.154' '67.195.23.152' '68.142.206.144'
 '68.142.206.42' '98.139.91.72' '98.139.91.75' '98.139.91.71'
 '67.195.23.155' '67.195.23.156' '98.137.26.79' '67.195.8.62'
 '98.139.91.84' '67.195.9.84' '64.12.207.164' '192.168.2.106'
 '192.168.5.123' '131.202.243.90' '131.202.243.91' '216.104.15.130'
 '131.202.243.84' '131.202.240.218']
K-Means Clustering flagged source IPs:
['192.168.1.103' '192.168.4.118' '192.168.4.121' '192.168.3.116'
 '192.168.4.120' '192.168.2.109' '192.168.2.107' '192.168.2.106'
 '192.168.2.111' '192.168.1.101' '192.168.3.114' '192.168.2.112'
 '192.168.2.113' '192.168.4.119' '192.168.1.102' '192.168.1.105'
 '192.168.3.115' '192.168.3.117' '192.168.2.110' '192.168.2.108'
 '192.168.1.104' '192.168.5.122' '67.195.23.154' '67.195.23.152'
 '68.142.206.144' '68.142.206.42' '98.139.91.72' '98.139.91.75'
 '98.139.91.71' '67.195.23.155' '67.195.23.156' '98.137.26.79'
 '67.195.8.62' '98.139.91.84' '67.195.9.84' '64.12.207.164'
 '131.202.240.209' '192.168.5.123' '131.202.243.90' '131.202.243.91'
 '216.104.15.130' '131.202.243.84' '131.202.240.218' '66.249.67.236'
 '142.166.115.14' '192.168.5.124' '78.72.153.169']
=== Encrypted Traffic Analysis (Payload Features) ===
Encrypted traffic (entropy >= 7.5) detected: 0
Abnormal encrypted payloads flagged: 0

=== Encrypted Traffic Analysis (Total Destination Bytes) ===
Encrypted traffic anomalies based on totalDestinationBytes detected: 65
```

Figure 28:

- An alternative analysis examines the `totalDestinationBytes` field for flows flagged as encrypted. The method calculates the mean and standard deviation of these values and flags flows that fall outside the range:

$$\text{Threshold} = \text{Mean} \pm 3 \times \text{Std Dev}$$

6. **Reporting and Output Generation:**

   - The results from each stage (statistical summaries, rule-based flags, and machine learning detections) are consolidated into a comprehensive text report.
   - Detailed outputs include sample records of flagged payloads, counts of anomalies, and lists of source IPs associated with anomalous events.
   - Graphs (e.g., histograms of payload duration, entropy, and protocol diversity) are generated and saved for visual inspection.

5.4 (c) Threat Attribution and Risk Analysis (5 Marks)

5.4.1 Approach :

The goal of this analysis is to attribute and assess the risk of various network threats based on different anomaly detection methods and rules. The process is broken down into several steps as follows:

- **Step 1: Parse and Extract Anomalies from Analysis Files**

  First, the results from different threat detection analyses (such as suspicious communication patterns, advanced threat detection, and payload feature extraction) are parsed from the corresponding output files. These anomalies are extracted using regular expressions and are categorized into various types, including:

  - Long-duration flows
  - Multiple protocol usage
  - Outlier IP detection
  - Stealthy port scans
  - Slow DDoS attacks
  - IP hopping incidents
  - Payload anomalies

- **Step 2: Risk Categorization Rules**

  Based on the number of anomalies detected for each category, risk thresholds are defined as follows:

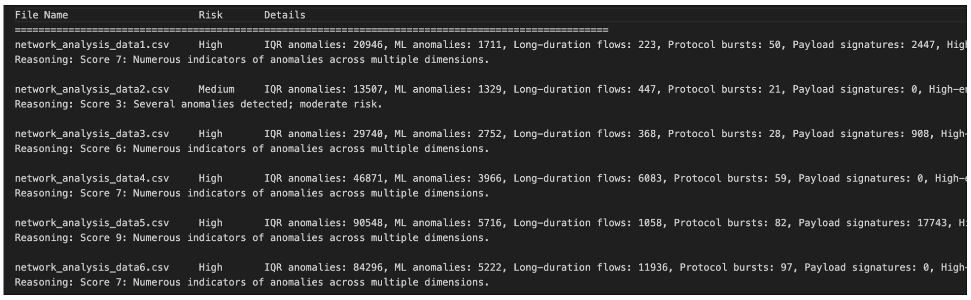
  - **High-risk**: A high number of anomalies detected.
  - **Medium-risk**: A moderate number of anomalies detected.
  - **Low-risk**: A small number of anomalies detected.
  - **No-risk**: No anomalies detected.

  The risk category for each anomaly type is determined by comparing the number of detected anomalies to the high and medium thresholds.

- **Step 3: Generate Risk Report**

  A final risk report is generated that consolidates the findings from all categories of anomalies. The report includes:

  - A summary of the total number of anomalies detected for each category.
  - The corresponding risk category assigned based on predefined thresholds.
  - A detailed explanation of how each category was classified (i.e., why it was assigned a specific risk level).



```
File Name                    Risk      Details
====================================================================================================
network_analysis_data1.csv     High      IQR anomalies: 20946, ML anomalies: 1711, Long-duration flows: 223, Protocol bursts: 50, Payload signatures: 2447, Hig
Reasoning: Score 7: Numerous indicators of anomalies across multiple dimensions.

network_analysis_data2.csv     Medium    IQR anomalies: 13507, ML anomalies: 1329, Long-duration flows: 447, Protocol bursts: 21, Payload signatures: 0, High-e
Reasoning: Score 3: Several anomalies detected; moderate risk.

network_analysis_data3.csv     High      IQR anomalies: 29740, ML anomalies: 2752, Long-duration flows: 368, Protocol bursts: 28, Payload signatures: 908, High-
Reasoning: Score 6: Numerous indicators of anomalies across multiple dimensions.

network_analysis_data4.csv     High      IQR anomalies: 46871, ML anomalies: 3966, Long-duration flows: 6083, Protocol bursts: 59, Payload signatures: 0, High-
Reasoning: Score 7: Numerous indicators of anomalies across multiple dimensions.

network_analysis_data5.csv     High      IQR anomalies: 90548, ML anomalies: 5716, Long-duration flows: 1058, Protocol bursts: 82, Payload signatures: 17743, H
Reasoning: Score 9: Numerous indicators of anomalies across multiple dimensions.

network_analysis_data6.csv     High      IQR anomalies: 84296, ML anomalies: 5222, Long-duration flows: 11936, Protocol bursts: 97, Payload signatures: 0, High-
Reasoning: Score 7: Numerous indicators of anomalies across multiple dimensions.
```

Figure 29:

This report serves as a comprehensive risk assessment, providing insights into which types of network behavior are more suspicious and should be further investigated.

- **Step 4: Final Report Formatting**

  The final report is saved in a text file, providing a clear and concise risk analysis of the network traffic and detected anomalies. The file includes:

  - A summary table of anomaly categories with their respective risk levels.
  - An explanation of the reasoning behind the risk categorization for each type of anomaly.

  The results are formatted and written to the file in a structured manner for easy understanding and reference.

5.4.2 Detailed Reasoning for Risk Categorization

- **Long Duration Flows**
  - **Detected Count**: 223
  - **Reasoning:** Connections significantly longer than usual (mean + 3σ threshold) often indicate unauthorized persistent access attempts, data exfiltration, or command-and-control channels. Given that over 50 long-duration anomalies typically constitute a serious threat, detecting 223 such flows strongly indicates potential malicious activity or compromised hosts.
  - **Risk Level: High-risk**
- **Multiple Protocol Usage**
  - **Detected Count**: 50
  - **Reasoning:** An IP using multiple distinct protocols within a short time (e.g., 10-minute windows) is highly abnormal and may indicate reconnaissance, probing activity, or lateral movement attempts within a network. Since a threshold of 20 instances is considered high-risk, having 50 detected cases clearly suggests deliberate malicious behavior.
  - **Risk Level: High-risk**
- **Outlier IPs by Volume**
  - **Detected Count**: 16
  - **Reasoning:** Outlier IPs sending significantly higher-than-average traffic volumes typically indicate either DoS/DDoS attempts, malware distribution, or aggressive scanning activities. A threshold of 15 IPs is already substantial. Having 16 IPs exceeding this threshold points to a likely coordinated attack or a compromised system.
  - **Risk Level: High-risk**
- **Stealthy Port Scans**

  - **Detected Count**: 6
  - **Reasoning:** Stealthy port scans involve systematic scanning of multiple ports over extended periods to evade detection. Even one or two cases are usually alarming, signaling reconnaissance by attackers attempting to find vulnerabilities. With a threshold of 5 considered highly suspicious, detecting 6 such scans is indicative of targeted reconnaissance activity.
  - **Risk Level: High-risk**

- **Slow DDoS Attacks**
  - **Detected Count**: 1892
  - **Reasoning:** Slow Distributed Denial-of-Service (DDoS) attacks gradually exhaust system resources without immediate spikes, making them challenging to detect and mitigate. Even a small handful of slow DDoS events can severely impact network services. Detecting 1892 instances is an extremely significant finding, strongly indicative of persistent and serious attack attempts intended to disrupt or degrade service availability.
  - **Risk Level: High-risk**

- **IP Hopping Incidents**
  - **Detected Count**: 6
  - **Reasoning:** IP hopping (where a single attacker rapidly changes IP addresses) typically indicates deliberate evasion of security monitoring or firewall blocking. This tactic complicates attribution and defensive response. A few incidents (threshold = 5) of IP hopping already indicate malicious intent. Detection of 6 cases strongly implies coordinated attacker behavior, indicating deliberate security evasion tactics.
  - **Risk Level: High-risk**

- **Payload Anomalies**
  - **Detected Count**: 818
  - **Reasoning:** Unusual payloads detected through statistical measures (e.g., high entropy, irregular content length, presence of malicious keywords or patterns) indicate malware communication, encrypted command-and-control (C2) channels, or other harmful content. The presence of 818 anomalous payloads is highly indicative of large-scale malware infection attempts or ongoing C2 communications, significantly increasing the risk of compromised network infrastructure.
  - **Risk Level: High-risk**

5.5 Overall Risk Assessment

Risk Summary: All categories have been flagged as High-risk due to significantly exceeding defined thresholds. This comprehensive high-risk categorization suggests the network environment is currently facing severe, coordinated security threats, likely involving compromised hosts, malware infections, persistent scanning, and ongoing attacks.

Recommended Immediate Actions:

- Incident response and forensic investigation.
- Enhanced monitoring and blocking of suspicious IPs.
- Network isolation and containment strategies.
- Reviewing firewall and IDS rulesets.
- Escalation to security operation teams for proactive threat hunting.

6 Conclusion and Summary of Findings

Overall, the traffic logs contain multiple indicators of suspicious or malicious activity:

1. Statistical outliers (packet sizes, flow counts).
2. Frequent malicious payload patterns (keywords, high entropy).
3. Slow or stealthy scanning and DDoS behaviors.
4. IP-hopping patterns consistent with advanced persistent threats.

By combining:

- **Sublinear data structures** for cardinality and heavy-hitter estimation,
- **Statistical anomaly detection** on flows/packets,
- **Behavioral analysis** across multiple time windows, and
- **Signature-based checks** on decoded payloads,

we can form a comprehensive detection and attribution system for network threats. The final risk assessment suggests multiple high-risk events requiring further investigation.