Contents

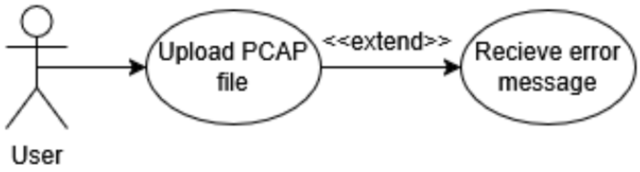
Epics and user stories

Epic One: PCAP Ingestion

User stories

1. As a user I want to upload a PCAP file so that I can analyse my network traffic.
2. As a **user** I want to know if the analysis is still in progress, so I am aware the app is working.

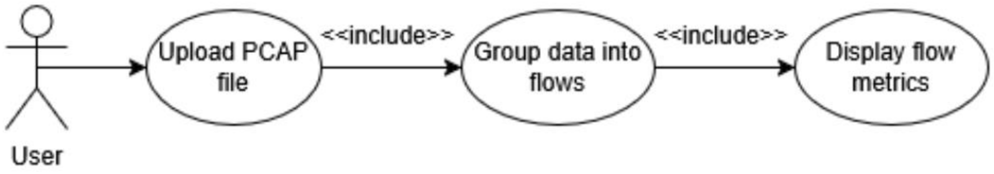
3. As a user I want to be told if the PCAP file is corrupted or invalid so that the analysis fails safely.
4. As a **user** I want to be able to re-upload a new file so that I can analyse multiple datasets.



Epic Two: Data extraction and processing

User stories

1. As a user I want network traffic to be grouped into flows so that I can understand patterns.
2. As a user I want to view flow metrics so that I can interpret users' network behaviour.
3. As a **user** I want to be notified if my PCAP file contains no valid IP packet data.
4. As a **user** I want the system to be able to process large files without freezing.



Epic Three: Dashboard UI

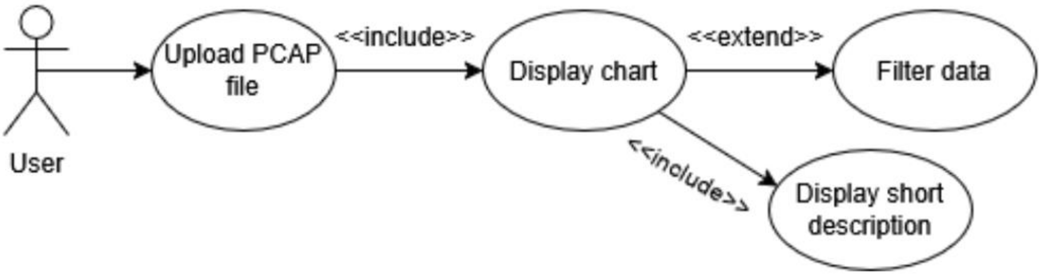
User stories

1. As a user I want a simple workflow so that I can easily analyse my data without advanced technical knowledge
2. As a user I want a clean and intuitive interface so that I can use the tool with minimal training
3. As a user I want the UI to function without crashing or freezing so that I can use the UI without hindrance.
4. As a user I want to see flow classifications in the dashboard so that I can understand traffic behaviour visually.
5. As a user I want anomalies to be highlighted in charts and tables so that I can easily spot them.
6. As a user I want to filter traffic by predicted class and anomaly status so that I can explore specific behaviours.

Epic Four: Data visualisation

User stories

1. As a user I want to be able to view interactive charts that show traffic size and flow statistics over time so that I can visually understand the data.
2. As a user I want to be able to filter the dashboard by protocol, time, and classification so that I can view specific activities.
3. As a user I want short descriptions for each chart so that I can understand what the data represents and means.
4. As a user I want summarised statistics so that I can make quick insights.



Epic Five: Documentation

User stories

1. As a user I want straightforward documentation so that I can understand how to get started.
2. As a user I want straightforward documentation of key network concepts so that I can understand the data and what it means
3. As a user I want documentation explaining how classifications are generated so that I understand system behaviour.
4. As a user I want documentation describing what the anomaly means and how it works so that I do not misinterpret results.

Epic Six: Code Maintainability

User stories

1. As a developer I want the code to be modular so that I can update individual components without breaking others.

Epic Seven: Performance and Error Handling

User stories:

1. As a developer I want data extraction and validation to be efficient so that large PCAP files do not incur long delays.

2. As a developer I want the dashboard application to display a readable, user-friendly error if the application crashes.
3. As a developer I want to conduct tests for each pipeline stage so that changes can be validated.

Epic Eight: User Action Classification (ML)

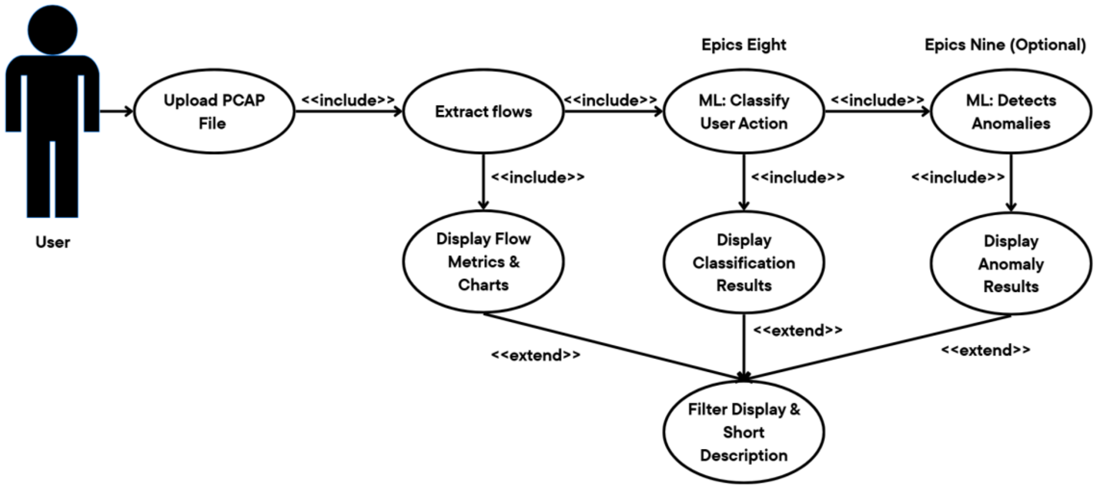
User Stories:

1. As a user I want network flows to be automatically classified so that I can understand user behaviour.
2. As a user I want each flow to have a label so that I can see what type of activity it represents.
3. As a user I want classification to run automatically after PCAP upload so that I do not need to trigger it manually.

Epic Nine: Anomaly Detection (ML – optional)

User Stories:

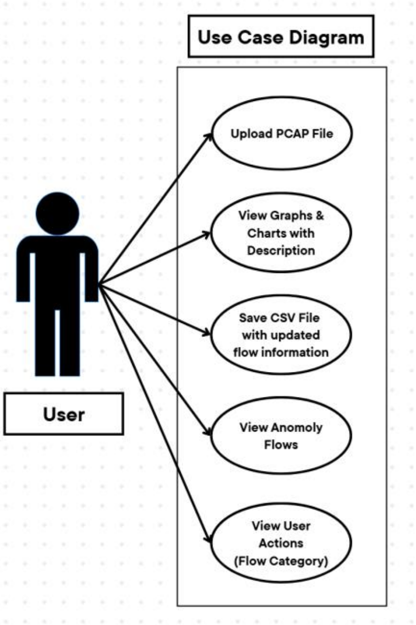
1. As a user I want anomalous flows to be clearly marked so that they stand out from normal traffic.
2. As a user I want the system to detect unusual or suspicious network behaviour so that I can identify potential threats or errors.



Epic Ten: ML Integration & Model

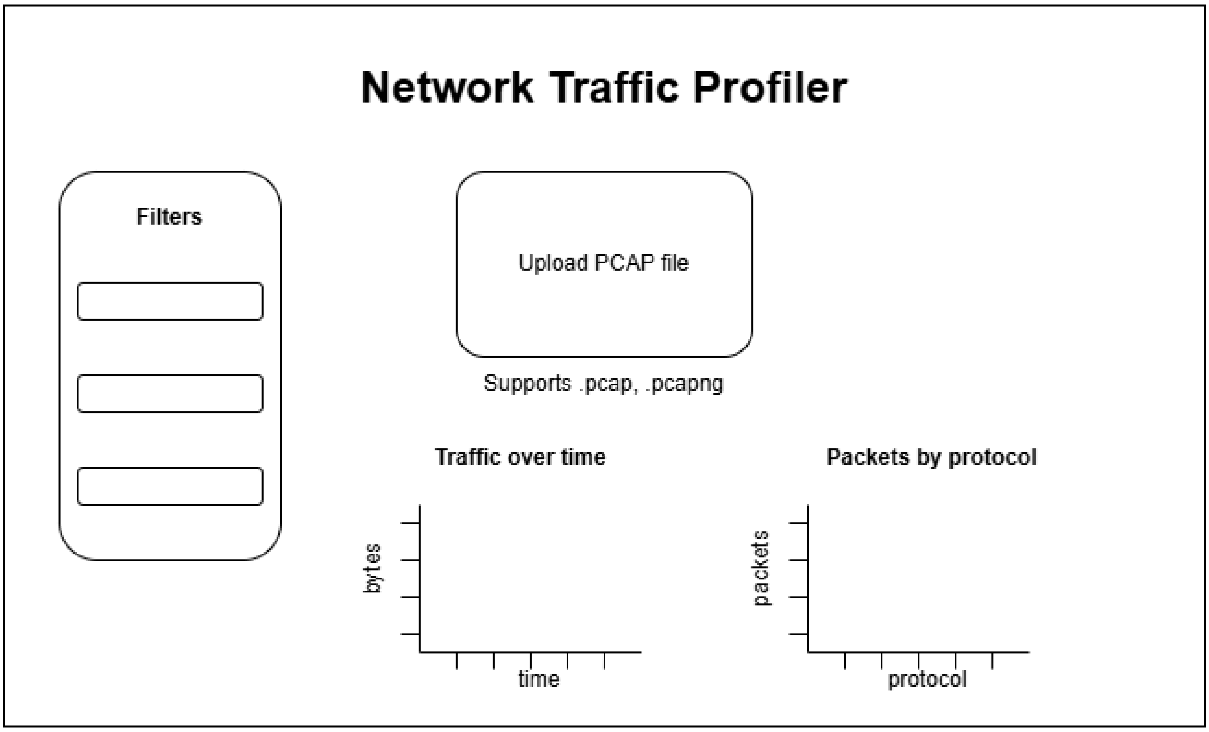
1. As a developer I want ML inference to run efficiently so that the dashboard updates without long delays.
2. As a developer I want to handle model errors gracefully so that the system does not crash if prediction fails.

3. As a developer I want to save and reuse trained models so that I do not need to retrain them every time.

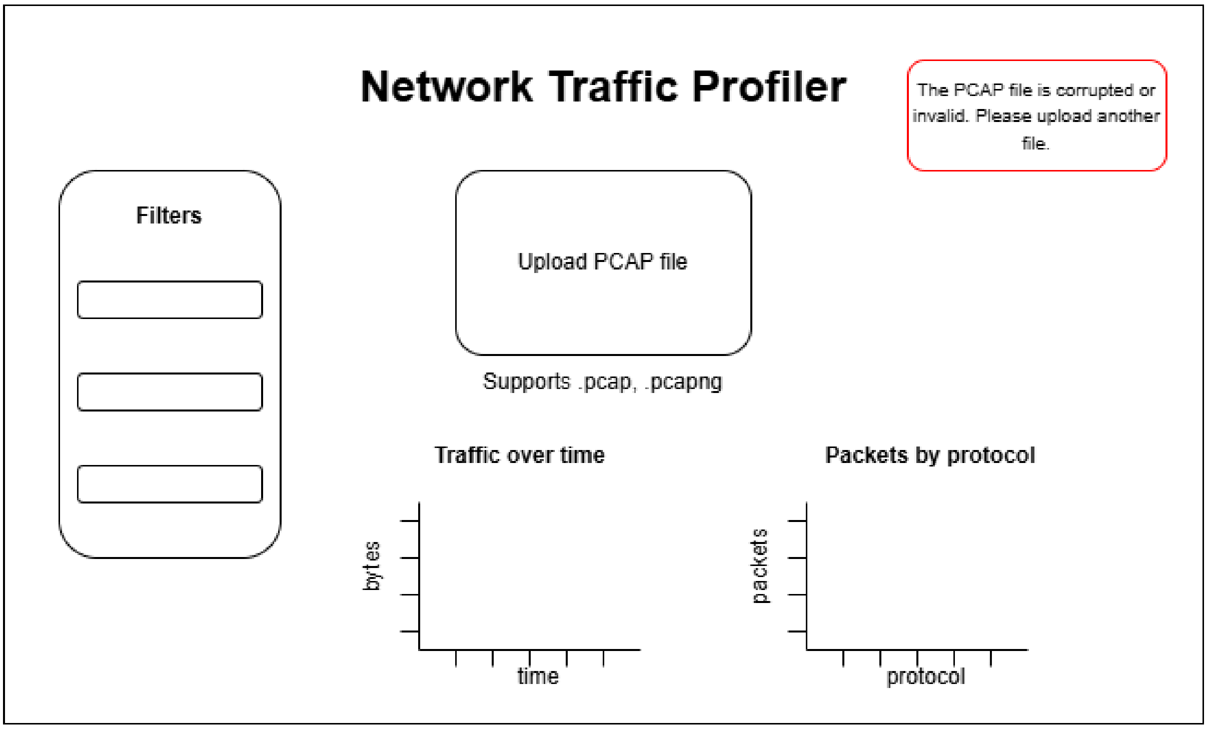

Use Case Diagram
User
Upload PCAP File
View Graphs & Charts with Description
Save CSV File with updated flow information
View Anomoly Flows
View User Actions (Flow Category)

Wireframes

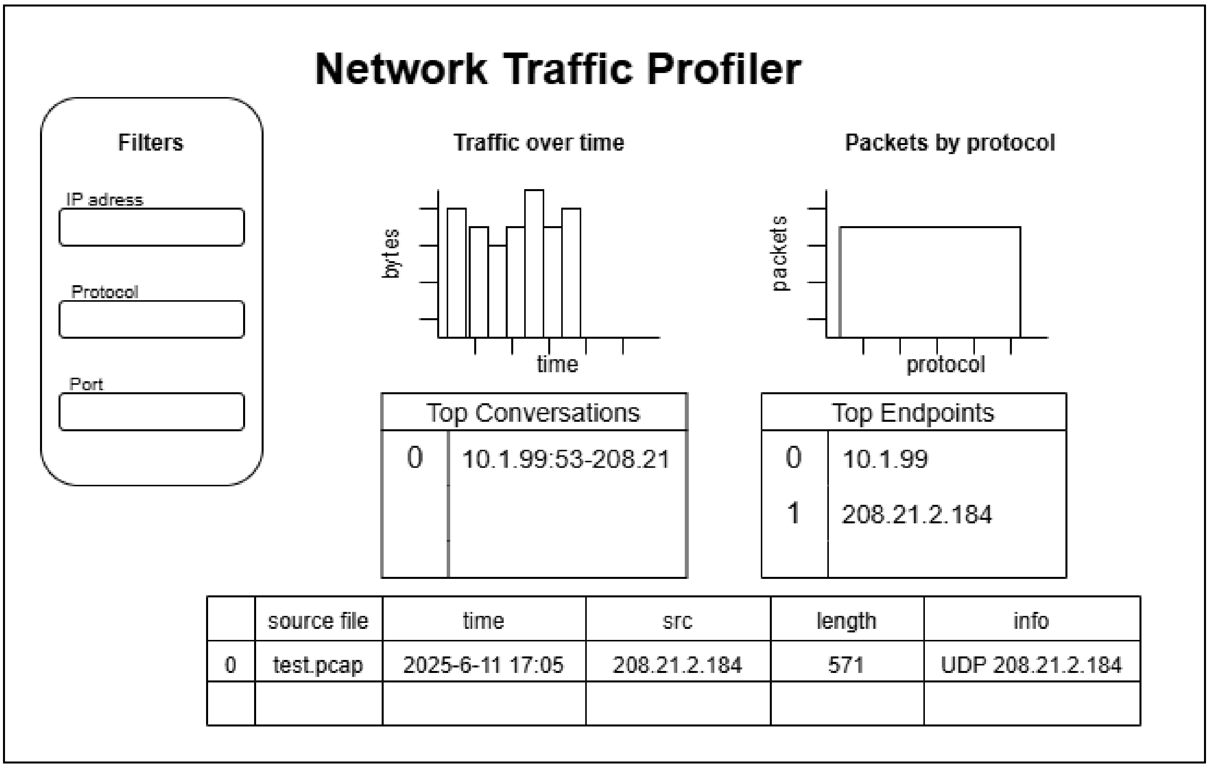
Wireframe 1 – Upload Screen (Empty State)



Wireframe 2 – Upload Error State



Wireframe 3 – Dashboard (Full data view)



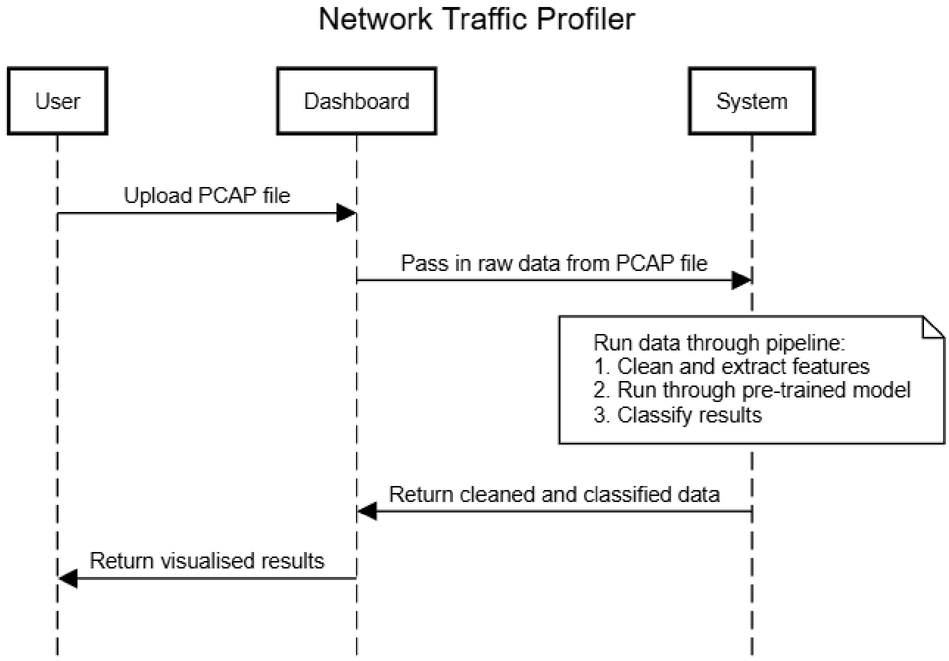
Top Conversations	
0	10.1.99:53-208.21

Top Endpoints	
0	10.1.99
1	208.21.2.184

	source file	time	src	length	info
0	test.pcap	2025-6-11 17:05	208.21.2.184	571	UDP 208.21.2.184

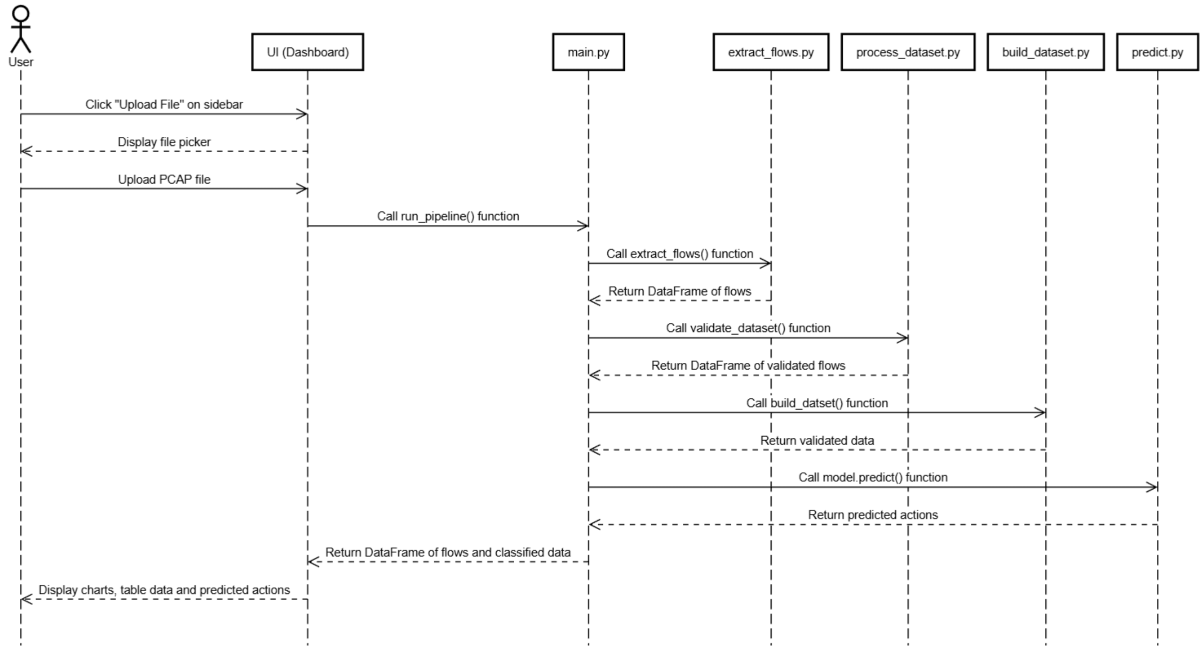
Sequence Diagrams

Sequence Diagram 1 – Simplified System Architecture


Network Traffic Profiler
User
Dashboard
System
Upload PCAP file
Pass in raw data from PCAP file
Run data through pipeline:
1. Clean and extract features
2. Run through pre-trained model
3. Classify results
Return cleaned and classified data
Return visualised results

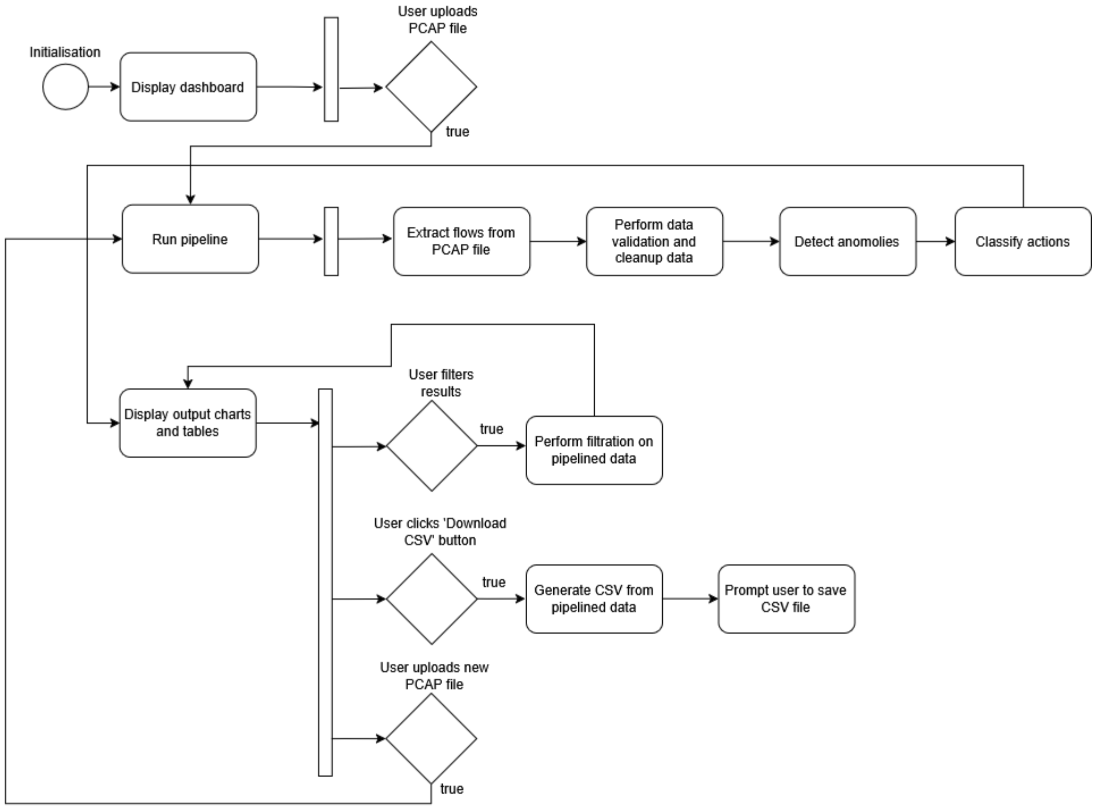
Sequence Diagram 2 – Detailed System Architecture

Network Traffic Profiler System Architecture


User
UI (Dashboard)
main.py
extract_flows.py
process_dataset.py
build_dataset.py
predict.py
Click "Upload File" on sidebar
Display file picker
Upload PCAP file
Call run_pipeline() function
Call extract_flows() function
Return DataFrame of flows
Call validate_dataset() function
Return DataFrame of validated flows
Call build_datset() function
Return validated data
Call model.predict() function
Return predicted actions
Return DataFrame of flows and classified data
Display charts, table data and predicted actions

Activity diagram

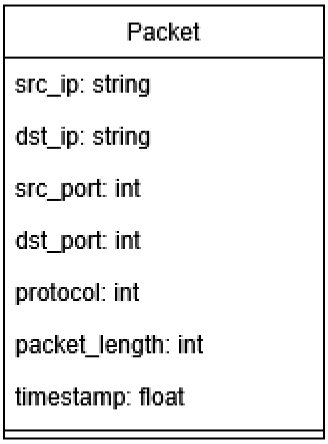
The diagram below shows the flow of the web application from initialisation.


Initialisation
Display dashboard
User uploads PCAP file
true
Run pipeline
Extract flows from PCAP file
Perform data validation and cleanup data
Detect anomolies
Classify actions
Display output charts and tables
User filters results
true
Perform filtration on pipelined data
User clicks 'Download CSV' button
true
Generate CSV from pipelined data
Prompt user to save CSV file
User uploads new PCAP file
true

Class diagrams

PCAP data extraction

The below ‘Packet’ diagram represents the data that makes up a single packet when extracted from a PCAP file. The ‘Flow’ diagram represents aggregated packet data that together makes up a network flow. Finally, the ‘Features’ diagram represents data extracted from PCAP files used for training the action classification ML.



Packet

src_ip: string

dst_ip: string

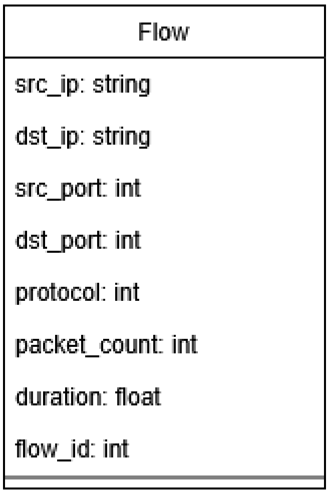
src_port: int

dst_port: int

protocol: int

packet_length: int

timestamp: float



Flow

src_ip: string

dst_ip: string

src_port: int

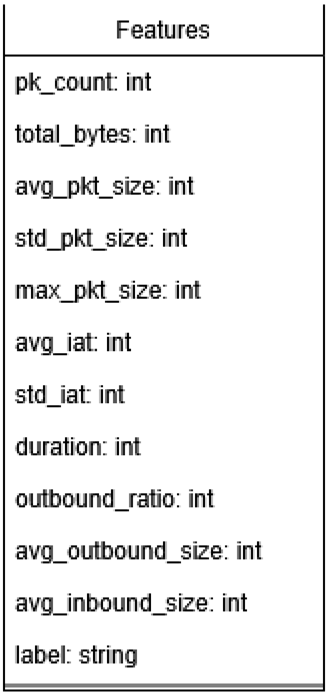
dst_port: int

protocol: int

packet_count: int

duration: float

flow_id: int



Features

pk_count: int

total_bytes: int

avg_pkt_size: int

std_pkt_size: int

max_pkt_size: int

avg_iat: int

std_iat: int

duration: int

outbound_ratio: int

avg_outbound_size: int

avg_inbound_size: int

label: string