## **Prerequisites**

* A working **Open5GS 5G Core** setup (Kubernetes or Docker)
* Packet capture tools: tcpdump or Wireshark
* Python 3.8+ with pandas, scikit-learn, matplotlib, jupyter
* Optional: Kibana/Grafana for visualization

## **Table of Contents**

* 📍 Step 1: Capture and Extract Network Traffic
* 📊 Step 2: Preprocess Packet Data for AI Model
* 🧠 Step 3: Train an AI Model to Learn Normal Patterns
* 🚨 Step 4: Detect Anomalies and Visualize Results
* ✅ Deliverables
* 🧰 Tools Used

## **Step 1: Capture and Extract Network Traffic**

1. Capture Open5GS packet data:

sudo tcpdump -i any -w open5gs-traffic.pcap port 38412 or port 2152

2. Convert .pcap to CSV using tshark:

## **Step 2: Preprocess Packet Data for AI Model**

Create preprocess.py:

import pandas as pd

# Load dataset

df = pd.read\_csv("traffic.csv")

# Drop rows with missing values

df.dropna(inplace=True)

# Encode categorical fields

df['Protocol'] = df['Protocol'].astype('category').cat.codes

# Normalize values

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

df\_scaled = pd.DataFrame(scaler.fit\_transform(df.select\_dtypes(include='number')))

df\_scaled.to\_csv("processed\_traffic.csv", index=False)

## **Step 3: Train an AI Model to Learn Normal Patterns**

import pandas as pd

from sklearn.ensemble import IsolationForest

# Load preprocessed data

data = pd.read\_csv("processed\_traffic.csv")

# Train isolation forest

model = IsolationForest(contamination=0.05)

model.fit(data)

# Predict anomalies

data['anomaly'] = model.predict(data)

# Save results

data.to\_csv("anomaly\_results.csv", index=False)

## **Step 4: Detect Anomalies and Visualize Results**

Create visualize.py:

import pandas as pd

import matplotlib.pyplot as plt

results = pd.read\_csv("anomaly\_results.csv")

# Plot anomalies

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

plt.plot(results.index, results[0], label='Traffic')

anomalies = results[results['anomaly'] == -1]

plt.scatter(anomalies.index, anomalies[0], color='red', label='Anomaly')

plt.title("Detected Traffic Anomalies")

plt.legend()

plt.savefig("anomaly\_plot.png")

plt.show()

## **Deliverables**

* ✅ traffic.csv, processed\_traffic.csv, anomaly\_results.csv
* ✅ Python scripts for preprocessing, training, and visualization
* ✅ Screenshot or image of the anomaly plot (anomaly\_plot.png)
* ✅ Documentation or Jupyter notebook with workflow explanation

## **AI-Based Traffic Analysis for Open5GS**

**Objective:** Show students how to capture 5G core network traffic, process it, and detect anomalies using an AI model.

## **Step-by-Step Workflow**

### **🧱 Step 1: Environment Setup**

📍**Tools You'll Need (install these locally or in a VM like Ubuntu):**

| **Tool** | **Use** | **Install Command** |
| --- | --- | --- |
| **Open5GS** | **5G Core Network** | **Install Docs** |
| **UERANSIM** | **UE/gNB simulator** | **GitHub:** [**UERANSIM**](https://github.com/aligungr/UERANSIM) |
| **tcpdump** | **Traffic capture** | **sudo apt install tcpdump** |
| **tshark** | **Export to CSV** | **sudo apt install tshark** |
| **Python + Jupyter** | **Data science + demo** | **pip install jupyter pandas scikit-learn matplotlib** |

### **Step 2: Start Open5GS & UERANSIM**

Run Open5GS and UERANSIM normally and **initiate UE registration**.

In your Open5GS core (or AMF pod), run:

bash

sudo tcpdump -i any -w open5gs-traffic.pcap port 38412 or port 2152

Wait for registration + PDU session → stop capture (Ctrl+C).

**Step 3: Convert .pcap to CSV**

**tshark -r open5gs-traffic.pcap -T fields \**

**-e frame.time\_epoch -e ip.src -e ip.dst -e frame.len -e \_ws.col.Protocol \**

**-E header=y -E separator=, > traffic.csv**

**Output: A CSV of packet-level traffic data**

**Step 4: Run the AI Pipeline (in Terminal or Jupyter Notebook)**

**Preprocess**

python3 preprocess.py

**Train Model + Detect Anomalies**

python3 train\_model.py

**Visualize Anomalies**

**python3 visualize.py**

**Step 5: Deliver the Demo to Students**

* **Use Google Colab for Remote Labs**

**Upload:**

* **traffic.csv**
* **preprocess.py, train\_model.py, visualize.py**

### **Student Exercise Ideas**

* **Give them traffic.csv and have them tune the IsolationForest contamination level**
* **Ask them to identify which protocols trigger anomalies**
* **Replace IsolationForest with another model like KMeans or OneClassSVM**