**Network Packet Sniffer with Alert Messages**

**Introduction :**

Network security is paramount in today’s digital landscape, where cyber threats such as unauthorized access, data breaches, and malicious activities pose significant risks. Real-time network traffic monitoring is essential for identifying suspicious behavior and enabling swift responses to potential threats. The Network Packet Sniffer with Alert System is a Python-based solution designed to capture, analyze, and visualize network traffic while detecting anomalies and sending email alerts. By integrating an intuitive graphical user interface (GUI) and real-time analytics, this project aims to enhance network security, providing administrators with actionable insights to mitigate risks effectively.

**Abstract :**

This report describes the development of a real-time network traffic monitoring system built with Python and Scapy. The system captures network packets, extracts key data (e.g., IP addresses, ports, protocols), and stores it in an SQLite database for efficient analysis. Anomaly detection, based on predefined rules such as unusual port activity or high traffic volumes, triggers email alerts via the SMTP protocol using the email.mime.text module. A Tkinter-based GUI and a Dash-powered dashboard with Plotly and Matplotlib visualizations enable real-time traffic monitoring. Multithreading ensures seamless packet capture and processing. The project delivers enhanced network visibility, automated threat detection, and interactive data insights, making it a valuable tool for network security management.

**Tools Used :**

**Python:** Core programming language for scripting and logic implementation.

**Scapy:** For packet capturing and analysis, enabling real-time sniffing of network interfaces.

**email.mime.text:** To construct and send alert emails via SMTP.

**Dash and Plotly:** For creating interactive web-based dashboards and visualizations.

**Pandas:** For data manipulation and analysis of captured traffic data.

**SQLite3:** Lightweight database for storing packet details (e.g., IP addresses, ports, protocols).

**Tkinter:** For building a desktop GUI with live-tracking features.

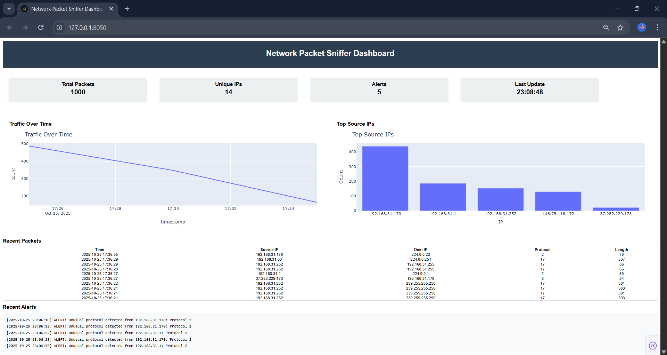
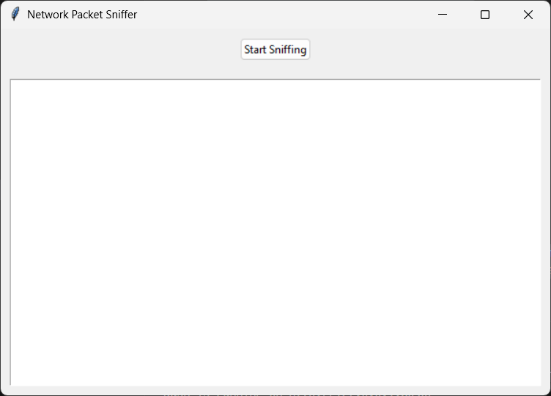
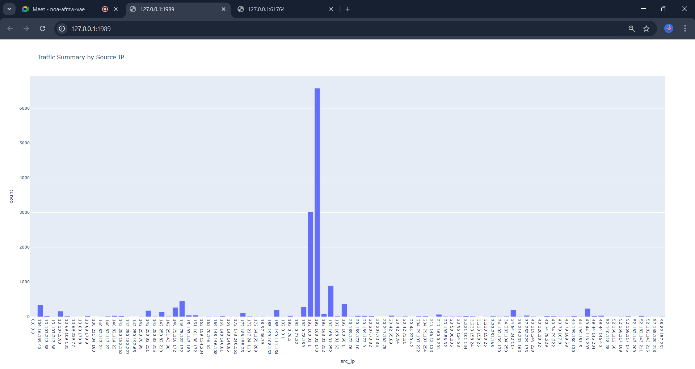
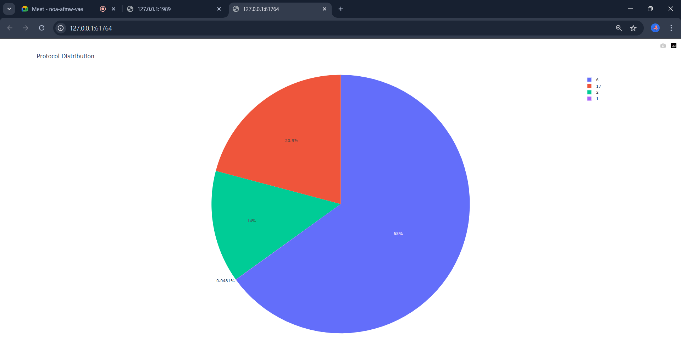
**Threading:** To handle concurrent tasks like packet capture, analysis, and GUI updates without blocking.

**Matplotlib:** For generating static and dynamic plots of traffic statistics.

**Steps :**

* **Environment Setup:**
* Installed required libraries (scapy, pandas, matplotlib, dash, plotly, tkinter, etc.).
* Configured the environment with administrator privileges for packet capture.
* Set up an SQLite database to store packet information (source/destination IP, port, protocol).
* **Packet Capture and Analysis:**
* Used Scapy to capture network packets in real-time.
* Parsed packets to extract relevant fields (source/destination IP, port, protocol).
* Implemented logic to detect anomalies (high packet rates, blacklisted IPs) using predefined rules.
* **Data Storage:**
* Stored captured packet data in an SQLite database for persistence and querying.
* Utilized Pandas for efficient data processing and filtering.
* **Alert System:**
* Configured the email.mime.text module to send email alerts when anomalies are detected.
* Integrated anomaly detection logic to trigger alerts based on thresholds or patterns.
* **Visualization and GUI:**
* Developed a Tkinter-based GUI to display live packet information and system status.
* Used Matplotlib for static graphs (e.g., traffic volume over time).
* Built an interactive dashboard with Dash and Plotly for real-time traffic visualization.
* **Multithreading:**
* Implemented threading to handle simultaneous packet capture, analysis, and GUI updates without performance bottlenecks**.**
* **Testing and Validation:**
* Tested the system on a local network to verify packet capture accuracy and anomaly detection.
* Validated email alerts and ensured visualizations accurately reflected traffic patterns.
* Optimized performance to handle high-traffic scenarios.

**Conclusion :**

The Network Packet Sniffer with Alert System is a robust tool for real-time network monitoring and threat detection. By leveraging Python, Scapy, and SQLite, it efficiently captures and stores network packet data, while Pandas and visualization libraries (Matplotlib, Dash, Plotly) provide clear insights into traffic patterns. The Tkinter GUI and multithreading enhance user interaction and system performance, respectively. The email alert system ensures timely notifications of suspicious activities, making the tool valuable for network administrators. This project demonstrates the integration of multiple technologies to create a scalable, user-friendly solution for network security, with potential for further enhancements like advanced machine learning-based anomaly detection.