

K-9 Packet Sniffer



Engineering
& Computing

Fall 2024
Senior Design Project

Student: Sharek Rendon
Instructor: Seyedmasoud Sadjadi

Florida International University

Full Stack Developer

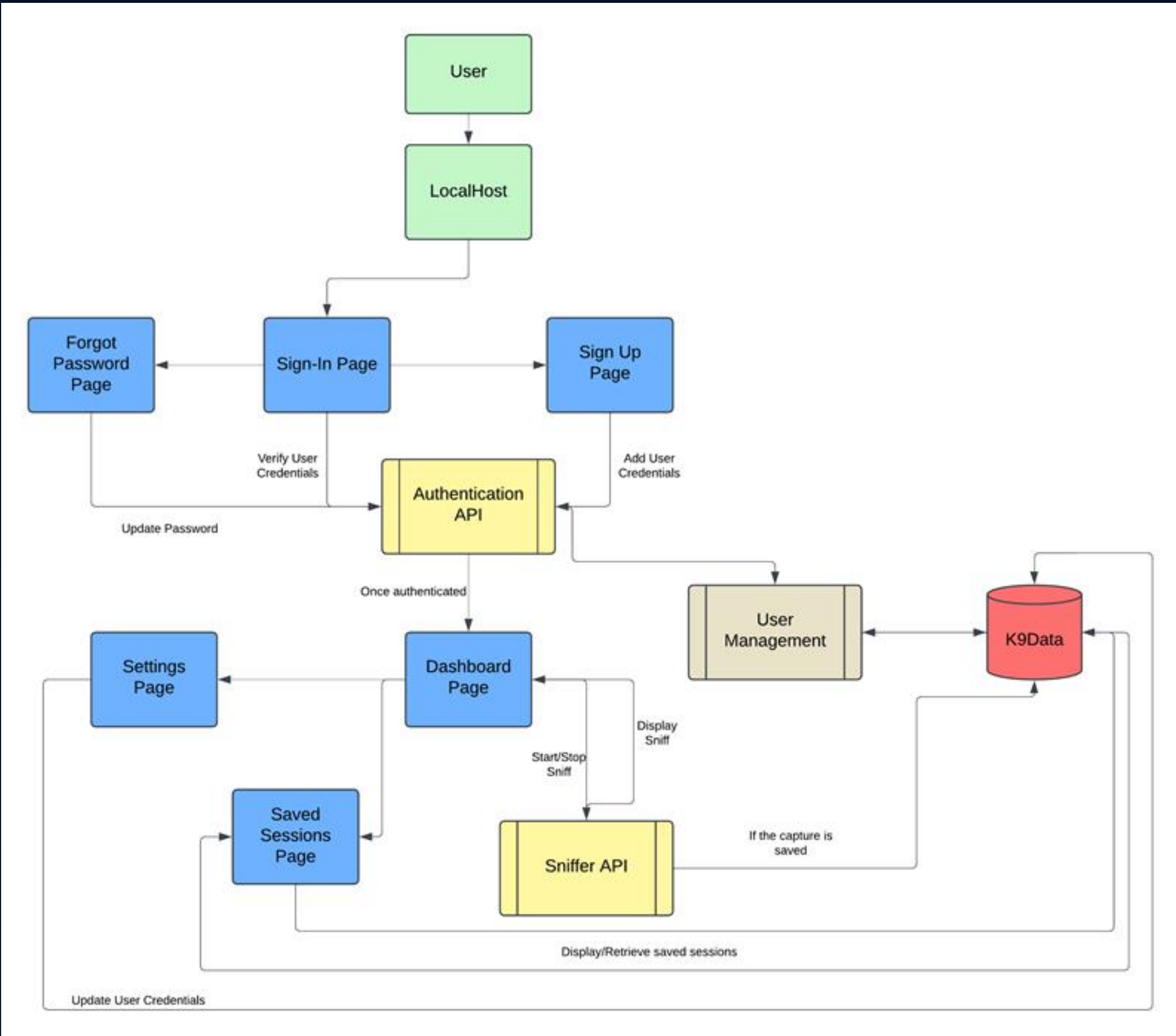
Problem:

Security professionals and network administrators depend on tools like Wireshark for analyzing traffic, yet these tools lack critical user session management. This creates the following challenges:

- **Data Security Risks:** Improper file management can result in sensitive information being mishandled, exposing the organization to potential breaches.
- **Limited User Accountability:** Without authentication systems, it's impossible to associate sessions with specific users, undermining access control.
- **Inefficient File Management:** The reliance on manual saving and organization of capture files increases the likelihood of errors or loss.

Requirements:

- JavaScript
- CSS
- React
- Python
- Scapy
- SQL



Current System:

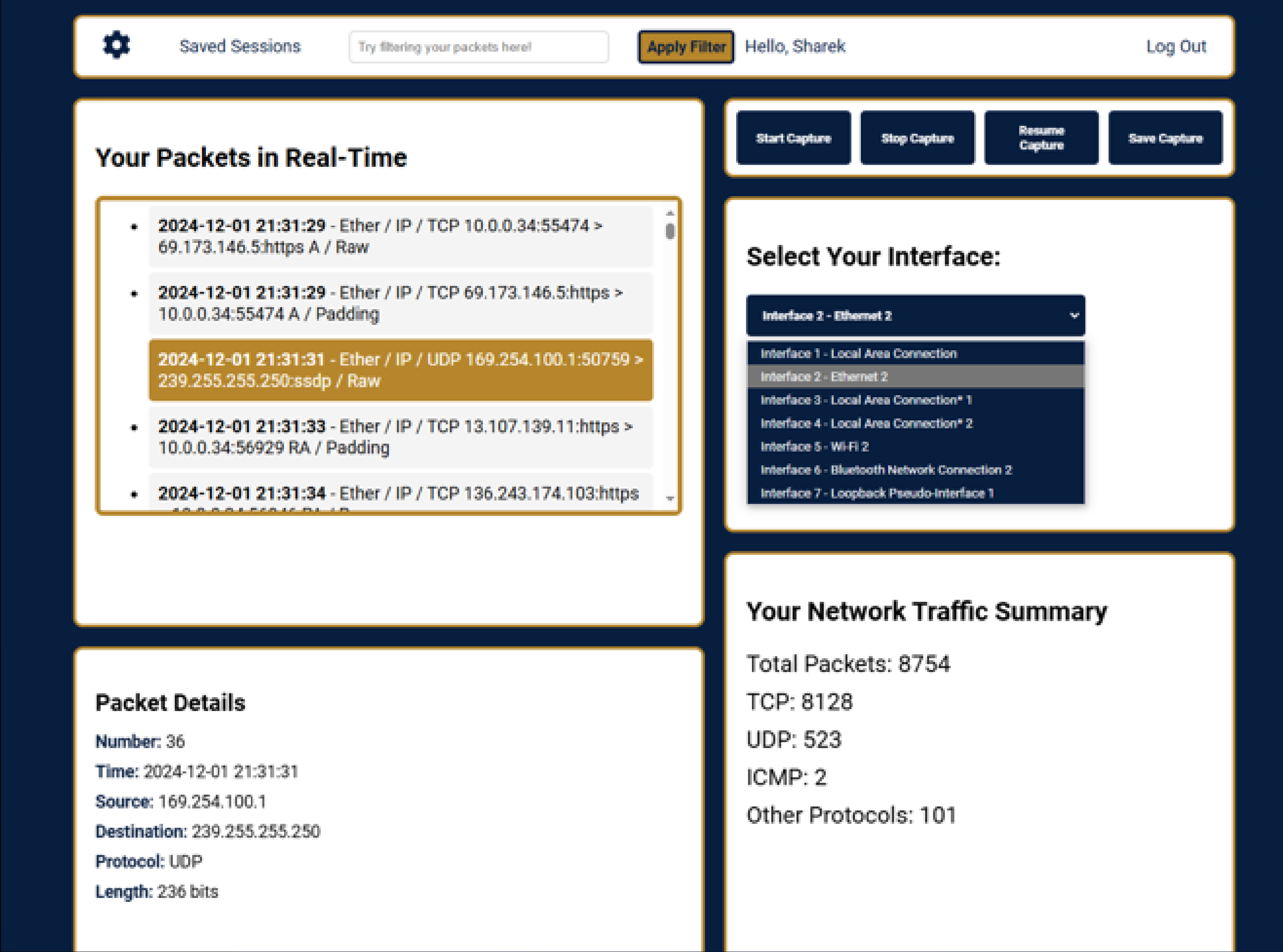
Tools in Use:

- Popular tools like Wireshark empower users to capture and inspect network traffic in real-time, offering robust packet analysis capabilities.

System Design:

- Limitations in Practice:
- Disorganized Workflow: Capture files must be managed manually, introducing inefficiencies and increasing the likelihood of data mismanagement.
 - No Role-Based Security: Open access to session files means anyone with system access can modify or view them, creating serious security and compliance concerns.
 - Scaling Problems: As the number of capture files grows, managing them becomes impractical, particularly in collaborative or enterprise environments.
 - Barriers to Entry: The interface is not beginner-friendly, with advanced features often hidden behind complex and unintuitive workflows.

Dashboard:



Tools Used:



Full Stack Contribution:

- In our project to develop a packet sniffer, each team member played a vital role in achieving the final product. I personally contributed to its functionality and development by mapping the back-end functions, such as the Python packet sniffer file, to the dashboard's capture features, ensuring smooth integration between the data processing and user interface.
- Using the packet filtering template created by my group, I developed a filtering function in Java that allows users to analyze specific data in real-time. I also created a Python script to scan available interfaces and developed an interactive display for packet details and network traffic summary.
- Additionally, I connected our database server to enable the capture and storage of packets generated by our K-9 software. These contributions were made possible through a combination of Java, Python, and CSS.

Database:

session_id	timestamp	source_ip	destination_ip	protocol	packetlength	packet_data
26	2024-12-01 21:31:29.000	10.0.0.34	162.159.134.234	TCP	54	Ether / IP / TCP 10.0.0.34:56296 > 162.159.134.234:htt...
26	2024-12-01 21:31:29.000	10.0.0.34	20.127.250.238	TCP	55	Ether / IP / TCP 10.0.0.34:55664 > 20.127.250.238:htt...
26	2024-12-01 21:31:29.000	10.0.0.34	20.127.250.238	TCP	66	Ether / IP / TCP 20.127.250.238:https > 10.0.0.34:5566...
26	2024-12-01 21:31:29.000	10.0.0.34	69.173.146.5	TCP	55	Ether / IP / TCP 10.0.0.34:55474 > 69.173.146.5:https...
26	2024-12-01 21:31:29.000	69.173.146.5	10.0.0.34	TCP	60	Ether / IP / TCP 69.173.146.5:https > 10.0.0.34:55474...
26	2024-12-01 21:31:31.000	169.254.100.1	239.255.255.250	UDP	236	Ether / IP / UDP 169.254.100.1:50759 > 239.255.255.2...
26	2024-12-01 21:31:33.000	13.107.139.11	10.0.0.34	TCP	60	Ether / IP / TCP 13.107.139.11:https > 10.0.0.34:56929...
26	2024-12-01 21:31:34.000	136.243.174.103	10.0.0.34	TCP	98	Ether / IP / TCP 136.243.174.103:https > 10.0.0.34:569...
26	2024-12-01 21:31:34.000	10.0.0.34	136.243.174.103	TCP	102	Ether / IP / TCP 10.0.0.34:56946 > 136.243.174.103:htt...
26	2024-12-01 21:31:34.000	136.243.174.103	10.0.0.34	TCP	60	Ether / IP / TCP 136.243.174.103:https > 10.0.0.34:569...
26	2024-12-01 21:31:34.000	239.255.255.250	10.0.0.104	UDP	554	Ether / IP / UDP 10.0.0.104:ssdp > 239.255.255.250:ss...
26	2024-12-01 21:31:34.000	10.0.0.104	239.255.255.250	UDP	538	Ether / IP / UDP 10.0.0.104:ssdp > 239.255.255.250:ss...
26	2024-12-01 21:31:34.000	104.18.162.67	10.0.0.34	TCP	79	Ether / IP / TCP 104.18.162.67:https > 10.0.0.34:56831...
26	2024-12-01 21:31:34.000	10.0.0.34	104.18.162.67	TCP	83	Ether / IP / TCP 10.0.0.34:56831 > 104.18.162.67:https...
26	2024-12-01 21:31:34.000	104.18.162.67	10.0.0.34	TCP	60	Ether / IP / TCP 104.18.162.67:https > 10.0.0.34:56831...
26	2024-12-01 21:31:35.000	13.107.139.11	10.0.0.34	TCP	60	Ether / IP / TCP 13.107.139.11:https > 10.0.0.34:56927...
26	2024-12-01 21:31:35.000	10.0.0.104	239.255.255.250	UDP	402	Ether / IP / UDP 10.0.0.104:ssdp > 239.255.255.250:ss...

Acknowledgement

The material presented in this poster is based upon the work supported by Sharek Rendon. I am thankful for the help that I received from my group members, Edward Medina, Carlos Imery, Maritza Medina and Tiago Muller