



*Florida International University
Knight Foundation School of Computing and Information Sciences*



Final Deliverable

As of: 12/04/2024

Project Title: K-9 Packet Sniffer

Team Members: Edward Medina, Sharek Rendon, Tiago Muller, Carlos Imery, Maritza Medina

Product Owner(s): Edward Medina

Instructor: Masoud Sadjadi



The MIT License (MIT)
Copyright (c) 2016 *Florida International University*

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.



Abstract

This document presents the information necessary to gain a thorough understanding of the K-9 Packet Sniffer. K-9 is a robust, user-friendly tool designed to monitor and analyze network traffic. It targets IT professionals, network engineers, cybersecurity specialists, and even casual users seeking to efficiently secure their networks. While tools like Wireshark are already widely available, K-9 seeks to address their limitations by prioritizing user-friendliness, security, organization, and streamlined workflows.

Introduction.....	4
Current System.....	5
Purpose of New System.....	7
User Stories.....	9
Implemented User Stories.....	9
Team Lead / Product Owner – Edward Medina.....	9
Frontend Developers – Maritza Medina & Sharek Rendon.....	9
Backend Developers – Tiago Muller, Edward Medina & Carlos Imery & Sharek Rendon.....	9
Database Developers – Carlos Imery & Edward Medina.....	10
Security Specialists – Tiago Muller.....	10
Pending User Stories.....	11
Team Lead / Product Owner – Edward Medina.....	11
Frontend Developers – Maritza Medina & Sharek Rendon.....	11
Backend Developers – Tiago Muller, Edward Medina & Carlos Imery.....	11
Database Developers – Carlos Imery & Edward Medina.....	12
Security Specialists – Tiago Muller.....	12
Project Plan.....	13
Hardware and Software Resources.....	15
Sprint Plannings.....	16
Sprint 1.....	16
Sprint 2.....	16
Sprint 3.....	16
Sprint 4.....	17



Sprint 5.....	17
Sprint 6.....	18
Sprint 7.....	18
System Design.....	19
Architectural Patterns.....	19
System and Subsystem Decomposition.....	19
Deployment Diagram.....	19
Design Patterns.....	19
System Validation.....	20
Glossary.....	21
Appendix.....	23
Appendix A: 1 - UML Diagram.....	23
Appendix B - Sniffer API Breakdown.....	24
Appendix B - User Interface Design.....	25
Appendix C - Sprint Review Reports.....	28
Sprint 5.....	30
Sprint 6.....	31
Sprint 7.....	32
Appendix D - User Manuals, Installation/Maintenance Document, Shortcomings/Wishlist Document and other documents.....	32
References.....	34

INTRODUCTION

The K-9 Packet Sniffer is a cutting-edge, user-friendly tool designed to revolutionize the way network traffic is monitored and analyzed. Whether you're an IT professional, network engineer, cybersecurity specialist, or an everyday user, K-9 provides a powerful yet intuitive solution to monitor, secure, and understand your network traffic with ease.

While tools like Wireshark have long been the industry standard for packet analysis, they can sometimes fall short in terms of organization, user accessibility, and workflow integration. We understand the critical need for tools that not only deliver in-depth insights into network activity but also simplify and streamline the user's workflow. This is where K-9 Packet Sniffer excels.



Our open-source platform bridges the gap between functionality and simplicity. With K-9, users can capture and monitor network traffic directly on their local devices, organize packet data seamlessly, and securely store packet captures—all within an integrated and intuitive system. Gone are the days of scattered downloads and cumbersome file management. K-9 offers a secure, centralized portal with robust user authentication, allowing users to review, manage, and download session data effortlessly in one convenient place.

The result is a comprehensive network analysis experience that combines powerful features with an emphasis on simplicity and security. Whether diagnosing complex network issues, enhancing your cybersecurity posture, or exploring traffic patterns for better performance, K-9 Packet Sniffer version 1.1 makes the process easier, more efficient, and more secure than ever.

Thank you for choosing K-9 Packet Sniffer. We're excited to bring you an exceptional tool that empowers your network monitoring and security efforts. With K-9, network analysis is not just a task—it's a streamlined, efficient, and user-focused experience.

Current System

1. Wireshark: The Industry Standard

Wireshark is arguably the most well-known network protocol analyzer, renowned for its comprehensive feature set and versatility. It allows users to capture live traffic, analyze packets at a granular level, and troubleshoot network issues effectively. Wireshark supports a wide range of protocols and provides detailed decoding capabilities, making it a go-to tool for network engineers and cybersecurity professionals.

However, despite its robust functionality, Wireshark has some limitations:

- ❖ **Steep Learning Curve:** The detailed interface and extensive feature set can be overwhelming for beginners or non-technical users.
- ❖ **Lack of Centralized Organization:** Packet captures (PCAPs) need to be manually organized and stored, often leading to scattered files across systems.
- ❖ **No Built-in Security for PCAPs:** Wireshark lacks a native, secure storage mechanism, making sensitive packet data vulnerable to unauthorized access.
- ❖ **Workflow Challenges:** While powerful, Wireshark doesn't inherently simplify workflows for repetitive tasks, requiring significant manual effort.



2. Tcpdump: The Command-Line Powerhouse

Tcpdump is a command-line tool for capturing and analyzing network traffic. It's lightweight, fast, and ideal for users comfortable with terminal-based workflows. Tcpdump is especially useful in environments where GUI-based tools like Wireshark are impractical or unavailable.

Despite its strengths, Tcpdump has notable drawbacks:

- ❖ **Minimal User Interface:** The lack of a graphical interface makes it inaccessible to non-technical users.
- ❖ **Limited Analysis Capabilities:** Tcpdump captures raw data and requires external tools for in-depth analysis.

3. SolarWinds Network Performance Monitor (NPM)

SolarWinds NPM provides network traffic monitoring and troubleshooting capabilities with a focus on performance metrics. It's more of an enterprise-grade solution that includes features like traffic visualization, bandwidth analysis, and alerting systems.

While effective for performance monitoring, NPM isn't tailored for deep packet-level analysis. Its high cost and focus on enterprise environments also make it less accessible to smaller teams or individual users.

4. Snort: Intrusion Detection and Prevention

Snort is an open-source intrusion detection and prevention system (IDS/IPS) that monitors network traffic for malicious activity. It uses rule-based analysis to identify threats in real time, making it a valuable tool for security-focused environments.

However, Snort isn't designed for general-purpose network traffic analysis. Its focus on intrusion detection means it lacks the breadth of protocol decoding and packet-level insights provided by tools like Wireshark.

5. NetFlow Analyzers



NetFlow-based tools such as PRTG Network Monitor and ManageEngine NetFlow Analyzer provide high-level traffic insights, focusing on flow data rather than packet-level analysis. These tools are great for understanding network usage trends and detecting anomalies.

The limitation here is granularity: NetFlow analyzers don't capture individual packets, so they're unsuitable for diagnosing low-level network issues or understanding detailed traffic behavior.

Purpose of New System

The purpose of the K-9 Packet Sniffer system is to address the challenges and limitations present in existing network traffic analysis tools by providing a comprehensive, user-friendly, and secure solution. While tools like Wireshark, tcpdump, and others offer powerful capabilities, they often require steep learning curves, lack secure data management features, and create fragmented workflows. K-9 Packet Sniffer is designed to fill these gaps by combining robust functionality with simplicity and accessibility.

Key objectives of the K-9 Packet Sniffer system include:

- ❖ Simplifying Network Traffic Analysis
 - K-9 prioritizes ease of use, making it accessible to both technical and non-technical users. By offering an intuitive interface, streamlined workflows, and automation for common tasks, K-9 empowers users to focus on their analysis rather than struggling with complex configurations or command-line inputs.
- ❖ Providing Secure and Centralized Data Management
 - Unlike traditional tools that leave users managing scattered packet capture files, K-9 offers a secure portal where captured data can be organized, stored, and accessed in a centralized manner. With built-in user authentication and secure storage protocols, K-9 ensures the safety and confidentiality of sensitive network data.
- ❖ Delivering a Seamless User Experience
 - K-9 integrates powerful packet capturing and analysis capabilities with a modern approach to data management. Users can monitor, capture, and review network



traffic directly on their local systems and efficiently access or download data through the secure portal—all in one cohesive platform.

❖ Enhancing Accessibility for a Broader Audience

- While many existing tools cater to advanced users, K-9 aims to lower the entry barrier for everyday users, small businesses, and aspiring cybersecurity professionals. By providing an open-source, community-driven platform, K-9 makes high-quality network traffic analysis more widely available.

❖ Fostering Collaboration and Scalability

- By creating a system that organizes and secures network data, K-9 facilitates better collaboration within IT teams. Its modular and open-source architecture ensures scalability, allowing future enhancements and integrations to meet evolving user needs.

❖ Offering an Open-Source Alternative

- The K-9 Packet Sniffer serves as an open-source alternative to costly enterprise solutions, offering a powerful and free-to-use platform for individuals and teams who may not have the resources to invest in expensive tools.

In essence, the K-9 Packet Sniffer system seeks to redefine how network traffic is monitored and analyzed, providing users with a solution that is not only powerful and efficient but also secure and accessible. With this system, we aim to empower users to take control of their network data while fostering a more streamlined and collaborative cybersecurity environment.



USER STORIES

The following section provides the detailed user stories that were implemented in this iteration of the K-9 Packet Sniffer project. These user stories served as the basis for the implementation of the project's features. This section also shows the user stories that are to be considered for future development.

Implemented User Stories

Team Lead / Product Owner – Edward Medina

Integrated User Stories:

1. *As a Product Owner*, I want to define and prioritize features for K9 Packet Sniffer to ensure the product meets user needs.
 2. *As a Team Lead*, I want to review all team deliverables to ensure consistency and alignment with project goals.
 3. *As a Product Owner*, I want to coordinate sprints and team meetings to keep the project on track.
-

Frontend Developers – Maritza Medina & Sharek Rendon

Integrated User Stories:

1. *As a Frontend Developer*, I want to design an intuitive and responsive user interface so users can easily navigate the platform.
 2. *As a Frontend Developer*, I want to implement a dashboard where users can view packet capture summaries for quick analysis.
 3. *As a Frontend Developer*, I want to integrate user authentication UI components to ensure secure access to the platform.
-

Backend Developers – Tiago Muller, Edward Medina & Carlos Imery & Sharek Rendon

**Integrated User Stories:**

1. *As a Backend Developer*, I want to build APIs to handle packet capture uploads and retrievals securely.
 2. *As a Backend Developer*, I want to implement a robust system for managing user sessions to prevent unauthorized access.
 3. *As a Backend Developer*, I want to develop functionality to process and parse captured packets for detailed analysis.
 4. *As a Backend Developer*, I want to develop a user authentication process for ease of logging in and saving packet captures, as well as, for security reasons.
-

Database Developers – Carlos Imery & Edward Medina**Integrated User Stories:**

1. *As a Database Developer*, I want to design a secure schema for storing user data and packet capture metadata.
 2. *As a Database Developer*, I want to ensure efficient queries for retrieving packet capture details to enhance platform performance.
 3. *As a Database Developer*, I want to implement database encryption to safeguard sensitive user information.
-

Security Specialists – Tiago Muller**Integrated User Stories:**

1. *As a Security Specialist*, I want to implement role-based access control (RBAC) to ensure appropriate user permissions.
2. *As a Security Specialist*, I want to conduct penetration testing to identify and address potential vulnerabilities in the platform, such as CWE Common Weakness Enumeration.
3. *As a Security Specialist*, I want to implement secure communication protocols (e.g., HTTPS, SSL) for all data transmissions.
4. *As a Security Specialist*, I want to create a risk matrix based on the vulnerabilities in the software and create a Disaster Recovery and Business Continuity plan



Pending User Stories

Initially, we had devised for the sniffer to be able to connect/be executed on a router so that we may gain access to all the packets available on a network, not just the packets between the machine executing our program and the router. However, this has yet to be implemented and is possibly being scrapped, as the development for this idea met complications along the way.

Team Lead / Product Owner – Edward Medina

Pending User Stories:

1. *As a Product Owner*, I need to create detailed user acceptance criteria to verify the functionality of each feature before deployment.
 2. *As a Team Lead*, I want to compile a comprehensive user manual to guide end-users in using the tool.
-

Frontend Developers – Maritza Medina & Sharek Rendon

Integrated User Stories:

Pending User Stories:

1. *As a Frontend Developer*, I want to develop a real-time visualization tool to display live traffic data in an easy-to-understand format.
 2. *As a Frontend developer*, I need to add multi-language support to enhance accessibility for a global audience.
-

Backend Developers – Tiago Muller, Edward Medina & Carlos Imery

Pending User Stories:

1. *As a Backend Developer*, I need to integrate packet filtering capabilities to allow users to analyze specific traffic patterns.
 2. *As a Backend Developer*, I want to implement data compression for stored packet captures to optimize storage space.
-



Database Developers – Carlos Imery & Edward Medina

Pending User Stories:

1. *As a Database Developer*, I need to create backup and recovery mechanisms to prevent data loss in case of system failure.
 2. *As a Database Developer*, I want to develop a feature for archiving old packet captures automatically to maintain database performance.
-

Security Specialists – Tiago Muller

Pending User Stories:

1. *As a Security Specialist*, I need to integrate multi-factor authentication (MFA) to strengthen user account security.
2. *As a Security Specialist*, I want to set up continuous monitoring for detecting and responding to security threats in real-time.

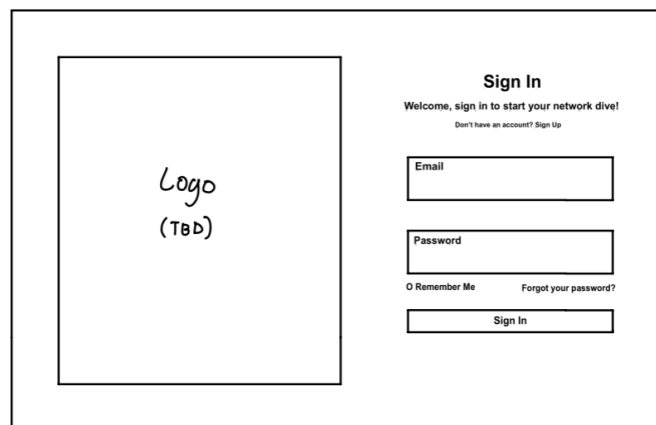


PROJECT PLAN

This section describes the planning that went into the realization of this project. This project incorporated the agile development techniques and as such required the sprints to be planned. These sprint plans are detailed in the section. This section also describes the components, both software and hardware, chosen for this project.

Roles planned for at the beginning of the development process consisted of:

- ❖ Team Lead / Product Owner – Edward Medina
- ❖ Frontend Developers – Maritza Medina & Sharek Rendon
- ❖ Backend Developers – Tiago Muller & Edward Medina & Carlos Imery & Sharek Rendon
- ❖ Database Developers – Carlos Imery & Edward Medina
- ❖ Security Specialists – Tiago Muller



*Initial prototype of what the log-in page would look like

The original plan was to have two different tools; a Packet Sniffer and a Network Scanner. We later decided to go with only the Packet Sniffer but also integrate parts of a Network Scanner. Primarily the Packet Sniffer was going to include Packet capture & Analysis by capturing and analyzing network packets. Also capturing packets from different network interfaces and decoding various protocols, filtering & sorting packets, as well as setting filters for capturing specific types/ Providing data visualization with visual representations for the user. The Network Scanner was originally planned to focus on network discovery, detecting and enumerating all devices on a network. As well as Port Scanning, scanning and identifying open ports on target devices, including service and OS detection.



Our initial debate was deciding whether the software was to be run on Linux or Windows. While debating the possibility of developing a packet sniffer on Windows, Linux is typically the preferred environment for networking and security-related programming. Ultimately deciding on Windows due to its ease of development and familiarity with OS. Another decision was determining access to low-level networking, tool availability, open-source community, performance and stability, and straightforward security permissions.

Utilizing a three-tier framework approach:

- ❖ Presentation
 - Graphical user interface (GUI)
- ❖ Application
 - Core functionality (processes user inputs, applies business rules, and performs calculations or data processing) in Python
- ❖ Data
 - Data storage and retrieval (interacts with databases, file systems, or other data sources to read, write, and update data) with SQL & PHP



Hardware and Software Resources

The following is a list of all hardware and software resources that were used in this project:

- ❖ JavaScript
 - <https://www.javascript.com/>
- ❖ Python
 - <https://www.python.org/>
- ❖ CSS
 - <https://code.visualstudio.com/docs/languages/css>
- ❖ HTML
 - <https://code.visualstudio.com/docs/languages/html>
- ❖ Node.js
 - <https://nodejs.org/>
- ❖ SQL
 - <https://www.mysql.com/>
- ❖ React
 - <https://react.dev/>
- ❖ Scapy
 - <https://scapy.net/>
- ❖ Pycharm
 - <https://www.jetbrains.com/pycharm/>
- ❖ IntelliJ
 - <https://www.jetbrains.com/idea/>



Sprint Plannings

Sprint 1

During our Sprint Planning meeting, we estimated our team's velocity to be mid-high, acknowledging our busy schedules but highlighting our strong communication and ability to work collaboratively. Together, we prioritized key user stories for this sprint, focusing on researching the fundamentals of packet sniffers, exploring existing tools, planning our development approach, and drafting an interactive prototype for the user interface. We also aimed to test basic functionality and investigate SQL database integration for traffic logging and anomaly detection. Each of us committed to specific tasks that align with our strengths, ensuring we build a solid foundation to keep the project on track and achieve our goals efficiently.

Sprint 2

During our Sprint Planning meeting for the K-9 Packet Sniffer project, we set a mid-high velocity and focused on three key user stories: front-end development, back-end functionality, and database integration. For the front end, we are creating a user interface with authentication features, a real-time packet dashboard, and notifications. The back end will handle user authentication, packet capture, filtering, and real-time processing, while the database setup includes cloud storage, data integration between components, and session management. Maritza and Sharek are leading front-end efforts, with Sharek also focusing on backend and database integration. Tiago is managing backend development, Edward is ensuring the database captures inputs effectively, and Carlos is implementing a cloud-based SQL database while sharing Wireshark resources to support the team. Together, we aim to solidify the project's core functionality during this sprint.

Sprint 3

During our Sprint Planning meeting for the K-9 Packet Sniffer project, we reaffirmed a mid-high velocity and outlined three main user stories for the sprint. For the Front End, we are building a user authentication interface, including login/logout, registration, and password reset, along with a dashboard to display real-time packet data, traffic summaries, and notifications. In the Back End, we are implementing user authentication, packet capture, filtering, and processing, and enabling our software to connect with router components via SSH for direct deployment and



communication. For the Database, we are setting up cloud storage, ensuring smooth data flow between the front and back ends, and managing user sessions with tracking and timeout features. Each of us is tackling specific areas: Maritza and Sharek are working on the front end, while Sharek is also reviewing backend and database processes. Tiago is developing backend functionalities, Edward is focusing on database integration, and Carlos is exploring router-based software deployment and contributing to database development. This sprint, we aim to enhance the project's core systems and overall functionality.

Sprint 4

During our Sprint Planning meeting for the K-9 Packet Sniffer project, we confirmed a mid-high velocity and prioritized three main user stories. For the Front End, we plan to develop a user authentication interface with login/logout forms, user registration, and password reset, alongside a dashboard displaying real-time packet data, traffic summaries, and alerts. In the Back End, we will implement user authentication, packet capture and filtering, real-time processing, and enable router integration via SSH for deployment and communication. For the Database, we aim to establish cloud storage, ensure data flow between components, and manage user sessions with tracking and timeouts. Each of us is contributing to these goals: Maritza is focusing on the front end and learning key languages, while Tiago is handling backend functionality. Edward is ensuring seamless database integration, Carlos is working on router-based deployment and database development, and Sharek is focusing on front-end development while supporting backend efforts. Together, we aim to deliver significant progress on core features this sprint.

Sprint 5

During our Sprint Planning meeting for the K-9 Packet Sniffer project, we discussed our team's velocity and remained confident that we could maintain regular communication and meet the project goals. The product owner outlined several high-priority user stories for the next sprint. For the Front End, we will focus on building a user management interface with profile settings, session management, and role-based access controls. Additionally, we will prioritize session management and capture control, including start/stop/resume capture features, multiple interfaces, and capture statistics. We will also work on a packet inspection interface with packet details, protocol breakdowns, and filtering controls.

For the Back End, we will develop user authentication and authorization features, enhance the packet capture engine to support different protocols and application layer detection, and improve packet filtering logic. We will also focus on enabling router component integration



via SSH or switch rerouting for packet information sharing. The database will involve setting up cloud storage and ensuring smooth data flow from the front to the back end. We will also manage user authentication, session tracking, and timeout features. Each team member has specific responsibilities: Maritza will tackle front-end and back-end tasks while learning Python, JavaScript, and CSS; Tiago will handle back-end development, focusing on authentication and integration with the front end; Edward will ensure database inputs are properly handled; Carlos will assist with SQL database development and investigate making the sniffer run locally on routers; and Sharek will help with dashboard creation and back-end functionality.

Sprint 6

During the Sprint Planning Meeting for the Packet Sniffer (K-9) project, the team, consisting of Carlos Imery, Maritza Medina, Edward Medina, Tiago Muller, and Sharek Rendon, discussed the velocity of the team, estimated it to be mid-high. Despite having busy schedules, the team can meet regularly and maintain stable communication to accomplish tasks. The product owner chose the following prioritized user stories for the next sprint: The Front End includes tasks related to user management, session management, capture control, packet inspection/analysis, saved sessions, and settings. The Back End focuses on user authentication, packet capture and handling, and the application running on router components, such as enabling SSH connection and installing software in router components. The database covers cloud storage setup, connecting front-end data to the back end, and session management, including session tracking and timeout.

Each team member indicated their contributions: Maritza will focus on verifying the functionality of both the code and the user interface, ensuring that everything works as expected from a usability perspective. Tiago will focus on penetration testing to resolve security concerns and ensure the program's security and stability. Edward will help develop the user authentication portion of the software, ensuring that data from both the front and back end are saved in the database. Carlos will assist with SQL database tasks, including optimizing queries and managing data storage while contributing to the logo design and collaborating with Maritza and Sharek on the saved captures interface. Sharek will help with the front-end creation of the dashboard and integrate packet filtering, traffic summary, and packet detail features with the back end.

Sprint 7



For the final Sprint, our team's shared goal for this sprint was to compile and organize all necessary progress, documents, and programs for the class presentation. Each member contributed to the creation of posters, presentations, and supporting documents, ensuring that every aspect of the project—technical details, front-end designs, back-end functionality, database, and program—was well-documented and prepared for delivery. The final deliverables included a comprehensive presentation that showcased key features such as user authentication, database integration, front-end design, audit logs, and packet sniffer functionality. Our team ensured that all information was accurately represented and aligned for the class presentation, effectively showcasing the full scope of their project.



SYSTEM VALIDATION

To ensure the accuracy of our packet sniffer, we validate the captured packets by comparing them against well-established packet-capturing methods, such as Wireshark and tcpdump. Both Wireshark and tcpdump are industry-standard tools that provide robust packet analysis and capture features. The validation process involves the following steps:

- 1. Packet Comparison: After capturing a set of packets with our sniffer, we replicate the same capture using Wireshark or tcpdump in the same network environment. We then compare the packets in both captures, checking for consistency in packet headers, payloads, and timing.
- 2. Timestamp Matching: We ensure that the timestamps for packets captured by both our sniffer and Wireshark/tcpdump are closely aligned, accounting for any potential network latency or minor discrepancies.
- 3. Protocol Identification: We verify that the protocols identified by our sniffer match those reported by Wireshark and tcpdump. This includes verifying higher-level protocols such as HTTP, FTP, and DNS, as well as lower-level protocols like TCP, UDP, and ARP.
- 4. Packet Integrity: Using packet analysis tools like Wireshark’s packet inspector, we check that the integrity of the packets is maintained, ensuring that no data loss or corruption occurs during the capture process.
- 5. Error Checking: We use checksums and other verification methods to confirm that the captured packet data is consistent between our sniffer and the reference tools.

By cross-referencing the results from our sniffer with trusted tools like Wireshark and tcpdump, we ensure that our packet capture process is both accurate and reliable, providing valuable insights for further analysis and development.

Example WireShark to Sniffer.py comparison: (note that source ports differ due to the captures being run at separate instances from one another).

65 2.304367		10.0.0.65		128.119.245.12		TCP		769 63464 → 80 [PSH, ACK] Seq=1 Ack=1 Win=513 Len=715						
Frame 65: 769 bytes on wire (6152 bits), 769 bytes captured (6152 bits) on interface \Device\NPF{...}								0030 02 01 17 6d 00 00 50 4f 53 54 20 2f 77 69 72 65P O S T /wire						
Ethernet II, Src: Intel dc:d5:d4 (98:8d:46:dc:d5:d4), Dst: ARRISGroup_e3:f3:b0 (a8:70:5d:e0:40:40)								0040 73 68 61 72 6b 2d 6c 61 62 73 2f 6c 61 62 33 2d ... shark-la bs/lab3-						
Internet Protocol Version 4, Src: 10.0.0.65, Dst: 128.119.245.12								0050 31 2d 72 65 70 6c 79 2e 68 74 6d 20 48 54 50 ... 1-reply. htm HTTP						
Transmission Control Protocol, Src Port: 63464, Dst Port: 80, Seq: 1, Ack: 1, Len: 715								0060 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 67 61 69 61 ... /1.1- Ho st: gaia						
Data (715 bytes)								0070 2e 63 73 2e 75 6d 61 73 73 2e 65 64 75 0d 0a 43cs.umass.edu- C						
Data [..]: 504f5354202f77697265736861726b2d6c6162732f6c6162332d312d7265706c792e68746d2048								0080 6f 6e 6e 65 63 74 69 6f 6e 3a 20 6b 65 65 70 2d ... onnectio n: keep-						
[Length: 715]								0090 61 6c 69 76 65 0d 0a 43 6f 6e 74 65 6e 74 2d 4c ... alive- C ontent-L						
Captured Packets:														
#	Source IP	Dest IP	Src Port	Dest Port	Protocol	Seq Number	Ack Number	Flags	Window Size	Checksum	Length	Binary	Hexadecimal	ASCII
1	10.0.0.35	10.0.0.67	N/A	N/A	ARP	N/A	N/A	N/A	N/A	N/A	42	N/A	N/A	N/A



GLOSSARY

1. **MIT License**
A permissive open-source license that allows users to use, copy, modify, merge, publish, distribute, sublicense, and sell software while providing it "as is" without warranty.
2. **Packet Sniffer**
A tool that monitors and captures network traffic, enabling the analysis of data packets traveling across a network.
3. **Wireshark**
A widely used network protocol analyzer for detailed packet-level analysis and troubleshooting.
4. **Tcpdump**
A command-line packet analysis tool used to capture and filter network traffic.
5. **Packet Capture (PCAP)**
A standard format for storing captured network packets.
6. **Graphical User Interface (GUI)**
The visual interface through which users interact with software.
7. **Session Management**
A process for managing user interactions within a system, including logging in, logging out, and tracking activities.
8. **SQL (Structured Query Language)**
A programming language used for managing and manipulating relational databases.
9. **Open-Source Software**
Software with source code that is freely available for modification, distribution, and use.
10. **Role-Based Access Control (RBAC)**
A security mechanism that restricts system access based on user roles.
11. **Authentication**
The process of verifying the identity of a user before granting access to a system.
12. **Protocol**
A set of rules governing the exchange or transmission of data between devices in a network.
13. **Encryption**
The process of encoding data to prevent unauthorized access.
14. **Intrusion Detection System (IDS)**
Software or hardware that monitors a network for malicious activities or policy violations.
15. **Disaster Recovery and Business Continuity Plan**
A strategy to restore critical systems and operations after a disruption or disaster.
16. **Three-Tier Framework**
An architecture model dividing a system into three layers: presentation (UI), application (logic), and data (storage).
17. **Scapy**
A Python library used for interactive packet manipulation and analysis.
18. **Router Integration**
The process of connecting and deploying tools or systems directly on a network router.
19. **Sprint**
A set period during which specific work must be completed and made ready for review in agile project management.
20. **UML (Unified Modeling Language)**
A standardized modeling language used to visualize system designs.
21. **Checksum**
A value used to verify the integrity of data transmitted over a network.
22. **SHA-256**
A cryptographic hash function producing a 64-character fixed hash value, commonly used for secure data storage.
23. **Cloud Storage**
A service for storing data on remote servers accessed via the internet.
24. **Real-Time Processing**
The immediate processing of data as it is received.

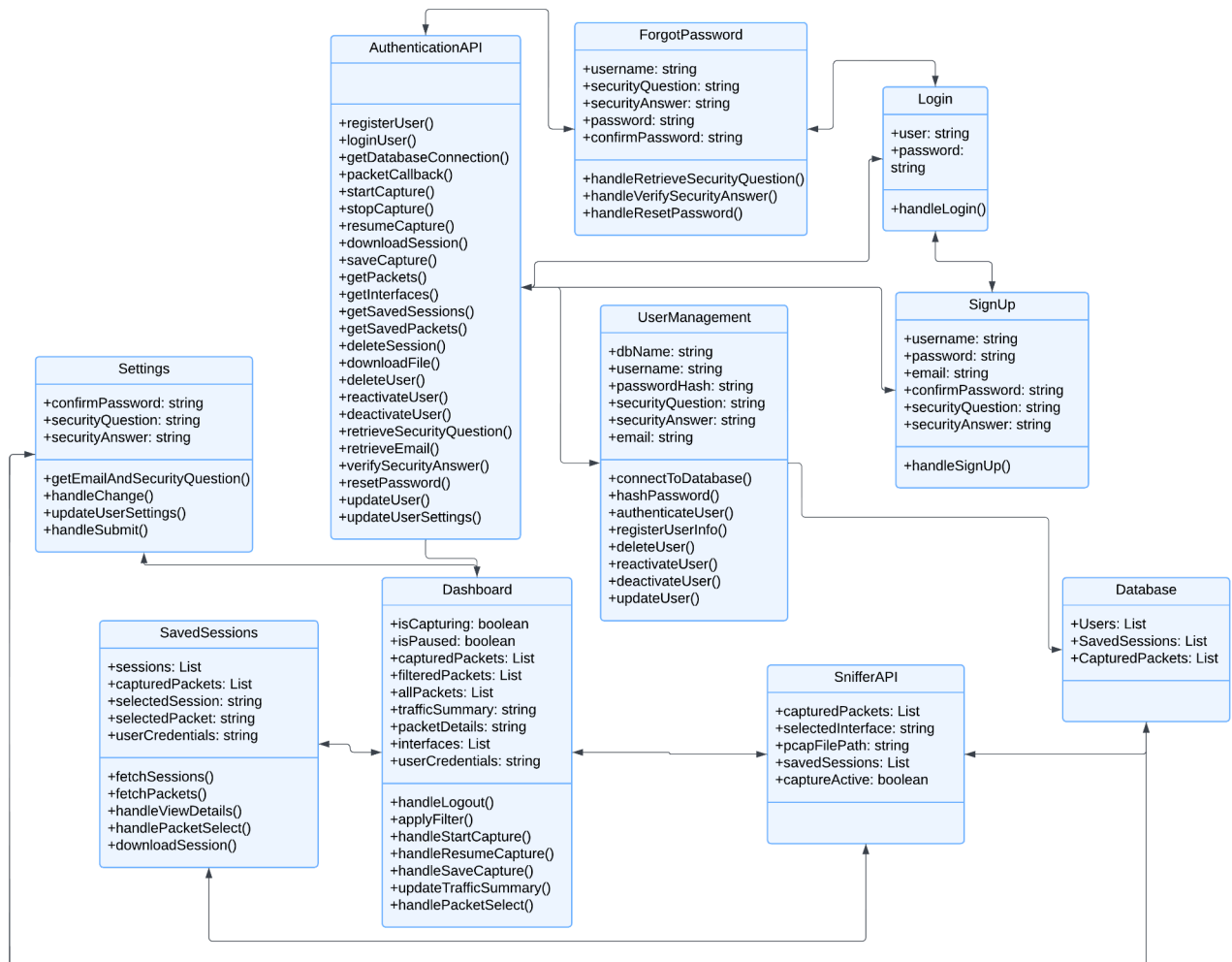


25. **Secure Socket Layer (SSL)**
A standard security protocol for establishing encrypted links between a server and a client.
26. **NetFlow**
A network protocol for collecting IP traffic information to monitor and analyze network usage.
27. **Modular Architecture**
A system design approach that divides software into smaller, independent modules that can be developed and maintained separately.
28. **Low-Level Networking**
Operations and programming that involve direct interaction with network interfaces and protocols.
29. **Anomaly Detection**
The identification of unusual patterns or behaviors in data that may indicate security risks or issues.
30. **Packet Filtering**
A process used to analyze and control the flow of data packets based on pre-defined criteria.
31. **Penetration Testing**
A security exercise in which a simulated cyberattack is performed to identify vulnerabilities in a system.
32. **Multifactor Authentication (MFA)**
A security system that requires multiple forms of verification to gain access to a resource.
33. **NetFlow Analyzers**
Tools that provide high-level traffic insights using flow data instead of packet-level analysis.
34. **Subsystem Decomposition**
Breaking down a system into smaller, manageable components or subsystems for design and implementation.
35. **Validation**
The process of verifying that a system or its components meet the intended requirements and specifications.
36. **Business Rules**
The logic and rules defined to guide processes within a system or application.
37. **Appendix**
Supplementary material at the end of a document, providing additional details and references.
38. **Namespace**
A structured container for identifiers, such as variable or function names, ensuring they are unique within a program or system.
39. **Scalability**
The capability of a system to handle increased loads or expand its functionality without performance loss.
40. **Network Interface**
A hardware or software interface that connects a device to a network



APPENDIX

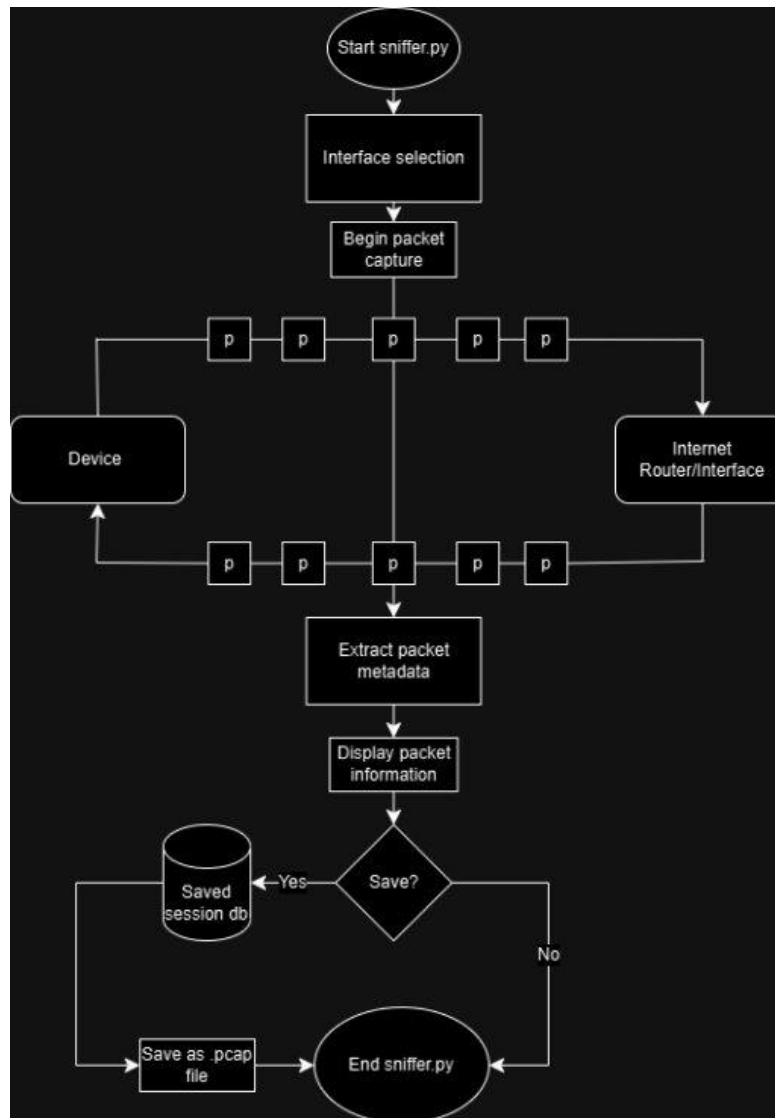
Appendix A: 1 - UML Diagram





Appendix B - Sniffer API Breakdown

Sniffer.py (before front-end integration):





Appendix C - User Interface Design

The image displays two screenshots of the K-9 Packet Sniffer user interface, both featuring a dark blue background and the K9 logo at the top center.

Sign In Screen:

- Header: K9 logo.
- Title: **Sign In**
- Welcome message: "Welcome! Please sign-in below to begin sniffing."
- Link: "Don't have an account? [Sign Up](#)"
- Username field: "Enter your username"
- Password field: "Enter your password"
- Remember me checkbox: ☐ Remember me
- Forgot password link: [Forgot your password?](#)
- Sign In button: A gold button with a lock icon and the text "Sign In".

Sign Up Screen:

- Header: K9 logo.
- Title: **Sign Up**
- Welcome message: "We would love to have you join our sniffer community! Sign-up below to begin."
- Form fields: Full Name, Username, Email, Password, Confirm Password, Select a security question (dropdown), and Answer to security question.
- Sign Up button: A gold button with the text "Sign Up".
- Disclaimer: "By clicking the 'Sign Up' button, you are creating an account, and you agree to the [Terms of Use](#)."
- Back to Sign In link: [Back to Sign In](#)



Forgot Password

Username

Retrieve Security Question

Back to Sign In

 Saved Sessions Apply Filter Hello, Mid456 Log Out

Your Packets in Real-Time

- 2024-12-03 23:28:24 - Ether / ARP who has 192.168.1.76 says 192.168.1.253
- 2024-12-03 23:28:24 - Ether / ARP who has 192.168.1.77 says 192.168.1.253
- 2024-12-03 23:28:24 - Ether / ARP who has 192.168.1.83 says 192.168.1.253
- 2024-12-03 23:28:24 - Ether / ARP who has 192.168.1.94 says 192.168.1.198 / Padding
- 2024-12-03 23:28:24 - Ether / ARP who has 192.168.1.86

Start Capture Stop Capture Resume Capture Save Capture

Select Your Interface:

Interface 4 - Wi-Fi

Current Interface: Wi-Fi



2024-12-03 23:28:24 - Ether / ARP who has 192.168.1.94 says 192.168.1.198 / Padding

2024-12-03 23:28:24 - Ether / ARP who has 192.168.1.94

Packet Details

Number: 4
Time: 2024-12-03 23:28:24
Source: 192.168.1.198
Destination: 192.168.1.94
Protocol: ARP
Length: 52 bits

Your Network Traffic Summary

Total Packets: 169
TCP: 0
UDP: 0
ICMP: 0
Other Protocols: 169

Settings

Username:

Email:

Password:

Confirm Password:

Security Question:

Security Answer:



Saved Sessions

Hello, Mid456

• Session 1 - 12/2/2024 - 862 packets

Download

Back To Dashboard



Appendix D - Sprint Review Reports

Sprint 1

The Research and Planning phase meeting took place from 8:45 P.M. to 9:00 P.M. with attendees Edward Medina, Sharek Rendon, Carlos Imery, Maritza Medina, and Tiago Muller. The team discussed the following focus areas for the initial phase of the project, with each member outlining their contributions:

- **Edward Medina** will lead the team in researching best practices for user authentication and database integration.
- **Sharek Rendon** will investigate tools and technologies to improve the user interface, specifically focusing on user-friendly design and potential Java integration.
- **Carlos Imery** will research enhancing the packet sniffer's features, such as filtering and error handling while experimenting with router port switching.
- **Maritza Medina** will explore options for creating clean, user-friendly Packet Inspection/Session Management interfaces, evaluating the best approach for the application's user experience.
- **Tiago Muller** will research the implementation of Audit Logs and Filters, focusing on the use of Python for back-end development and integration strategies.

The team identified the primary research areas and planned their next steps for the project's development.

Sprint 2

The Sprint 2 Review meeting took place from 8:45 P.M. to 9:00 P.M. with attendees Edward Medina, Sharek Rendon, Carlos Imery, Maritza Medina, and Tiago Muller. After a show-and-tell presentation, the following user stories were accepted by the product owners, with team members committing to their tasks:

- **Edward Medina** will continue working on the development of the user authentication and database integration, ensuring that inputs from the front and back end are saved within the database.
- **Sharek Rendon** will focus on improving the user interface of the application, and explore the integration of a program into the existing prototype.
- **Carlos Imery** will assist with the packet sniffer development and research, ensuring that the system functions correctly with the local database setup, rather than the initially planned online service.



- **Maritza Medina** will finish the User Authentication Interface and begin work on the dashboard, using JavaScript/CSS based on the prototype, team feedback, and Wireshark.
- **Tiago Muller** will focus on the back-end development for user authentication, specifically working on password reset, user registration, and login/logout functions.

The team is aligned on their responsibilities for Sprint 3, building on their progress from the previous sprint and moving forward with database development, front-end enhancements, and user authentication features.

Sprint 3

The Sprint 3 Review meeting took place from 8:45 P.M. to 9:00 P.M. with attendees Edward Medina, Sharek Rendon, Carlos Imery, Maritza Medina, and Tiago Muller. After a show-and-tell presentation, the product owners accepted the implementation of the following user stories, and the team confirmed their focus for the next sprint:

- **Edward Medina** will continue working on user authentication and database integration, ensuring that data from the front and back end is correctly saved within the database.
- **Sharek Rendon** will focus on the front-end development, continuing to improve the user interface and exploring how to implement Java into the prototype.
- **Carlos Imery** will assist in the development of the packet sniffer and further investigate the local database setup, ensuring proper functionality with the software running on an internet router.
- **Maritza Medina** will work on the user management pages and dashboard, using JavaScript/CSS and considering feedback from the prototype and Wireshark to ensure a clean and user-friendly interface.
- **Tiago Muller** will focus on developing the back-end features for audit logs and filters, using Python to build these aspects for integration in future sprints.

The team is aligned on their tasks and objectives for Sprint 4, with a focus on completing the user interface, strengthening back-end features, and continuing database development.

Sprint 4

The Sprint 4 Review meeting took place from 8:45 P.M. to 9:00 P.M. with attendees Edward Medina, Sharek Rendon, Carlos Imery, Maritza Medina, and Tiago Muller. After a show-and-tell presentation, the product owners accepted the implementation of the following user stories, with team members confirming their commitment to the tasks:



- **Edward Medina** will focus on user authentication and database development for Sprint 4, ensuring that front and back-end inputs are saved in the database.
- **Sharek Rendon** continues working on front-end development, particularly enhancing the user interface and integrating Java into the prototype.
- **Carlos Imery** will improve the packet sniffer's back end by adding packet filtering, solar coding, error messages, and better protocol control.
- **Maritza Medina** will focus on Packet Inspection/Session Management using JavaScript/CSS, following feedback from the prototype and Wireshark.
- **Tiago Muller** will work on developing Audit Logs and Filters on the back end, using Python for future integration.

Sprint 5

The Sprint 5 Review meeting took place from 8:45 P.M. to 9:00 P.M. with attendees Edward Medina, Sharek Rendon, Carlos Imery, Maritza Medina, and Tiago Muller. Sprint 5 served as a prelude to the final phase of the project (Sprint 6), with the team laying the groundwork for the upcoming tasks.

- **Edward Medina** set up the necessary databases and began creating the foundational pages for the program, ensuring the system's data storage and structure were properly configured for the following sprint.
- **Tiago Muller** assisted with debugging, identifying and resolving issues in the code, and began preparing the penetration testing (pentest) steps to ensure the security of the application.
- **Sharek Rendon** and **Carlos Imery** continued with their previous tasks, with Sharek focusing on integrating the front-end and back-end components and Carlos working on SQL database optimization, as well as further developing the saved captures interface.
- **Maritza Medina** helped optimize code procedures, fine-tuned the user interface, and worked on enhancing the front-end functionality to improve the overall user experience.

This sprint allowed the team to solidify the core components of the application, preparing for Sprint 6, where the focus would shift to finalizing functionality, security testing, and ensuring seamless integration.

Sprint 6

The Sprint 6 Review meeting took place from 8:45 P.M. to 9:00 P.M. with attendees Edward Medina, Sharek Rendon, Carlos Imery, Maritza Medina, and Tiago Muller. In this sprint, each



team member focused on specific tasks to ensure the project's functionality, security, and integration.

- **Edward Medina** focused on ensuring there were no errors in the code functionality, thoroughly testing and reviewing the code to guarantee smooth operations across all modules.
- **Tiago Muller** conducted penetration testing (pen testing) to identify vulnerabilities within the application, ensuring the software's security and stability.
- **Maritza Medina** worked on verifying the proper functioning of both the code and the user interface, ensuring that everything operated as expected from a usability perspective.
- **Sharek Rendon** was dedicated to integrating the front-end and back-end components, ensuring smooth communication between the user interface and the underlying system.
- **Carlos Imery** worked on SQL database tasks, including optimizing queries and managing data storage, as well as contributing to the logo design and creating the saved captures interface alongside Edward and Sharek.

The team made significant progress on both functional and security aspects of the project, collaborating to address different critical components and ensuring all parts were ready for final implementation and presentation.

Sprint 7

The Sprint Review meeting for the final sprint took place from 8:45 P.M. to 9:00 P.M. with attendees Edward Medina, Sharek Rendon, Carlos Imery, Maritza Medina, and Tiago Muller. During this sprint, the team shared a common goal of compiling and organizing all progress, documents, and programs necessary for the class presentation. Everyone worked together to ensure that all aspects of the project—technical details, front-end designs, back-end functionality, and packet sniffer development—were thoroughly documented and prepared for delivery.

Each team member contributed to the creation of posters, presentations, and supporting documents. The final deliverables included a comprehensive presentation reflecting user authentication, database integration, front-end design, audit logs, and packet sniffer features. The team worked collaboratively to ensure all information was accurately represented and ready for class presentation, aligning efforts to showcase the full scope of their project.



Appendix E - User Manuals, Installation/Maintenance Document, Shortcomings/Wishlist Document and other documents

Setup SQL Guide:

Download SQL2019-SSEI-Expr.exe

<https://www.microsoft.com/en-US/download/details.aspx?id=101064>

Download SSMS

Connect with:

Authentication: Windows Authentication

Then follow Queries:

```
CREATE DATABASE K9Data;
```

Then within K9Data Run:

```
CREATE TABLE Users (  
    Username NVARCHAR(50) PRIMARY KEY,           -- Username as a unique  
    identifier  
    Email NVARCHAR(255) NOT NULL UNIQUE,         -- Email must be unique  
    FullName NVARCHAR(100) NOT NULL,             -- Full name  
    PasswordHash NVARCHAR(64) NOT NULL,         -- SHA-256 produces a  
    64-character hash  
    IsActive BIT NOT NULL DEFAULT 1,            -- Active status (1 =  
    True, 0 = False)  
    SecurityQuestion NVARCHAR(255) NOT NULL,    -- Security question  
    SecurityAnswer NVARCHAR(64) NOT NULL        -- Security answer hashed  
    with SHA-256  
);
```

```
CREATE TABLE SavedSessions (  
    SessionID INT IDENTITY(1,1) PRIMARY KEY,    --
```



```

Unique session identifier
    SessionDate DATE NOT NULL,           --
Date of extraction
    UserName NVARCHAR(50) NOT NULL,      --
Username as a foreign key
    SessionNumber INT NOT NULL,          --
Session number
    CapturedPackets INT NOT NULL,        --
Number of captured packets
    CONSTRAINT FK_SavedSessions_Users FOREIGN KEY (UserName)
        REFERENCES Users(Username)      --
Define the foreign key constraint
    ON DELETE CASCADE                    --
Optional: cascade delete when a user is removed
    ON UPDATE CASCADE,                   --
Optional: cascade update if username changes
    CONSTRAINT UQ_SavedSessions UNIQUE (UserName, SessionNumber) --
Ensure unique sessions per user
)

CREATE TABLE CapturedPackets (
    id INT PRIMARY KEY IDENTITY,
    session_id INT,
    timestamp DATETIME,
    source_ip VARCHAR(50),
    destination_ip VARCHAR(50),
    protocol VARCHAR(10),
    packetlength INT,
    FOREIGN KEY (session_id) REFERENCES SavedSessions(SessionID)
);

ALTER TABLE CapturedPackets
ADD packet_data TEXT;

```



Appendix F - API & React Setup To Begin The Program

Both the Authentication and Sniffer API must be running for the software to operate as intended.

This could be done as seen below:

Sniffer.API

```
PS C:\Users\james\Documents\K9\demo-project> python .\sniffer_api.py
* Serving Flask app 'sniffer_api'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://10.0.0.232:5000
Press CTRL+C to quit
```

Authentication.API

```
PS C:\Users\james\Documents\K9\demo-project\src> python .\authentication_api.py
* Serving Flask app 'authentication_api'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5001
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 102-899-521
```

Then you can start the program

```
PS C:\Users\james\Documents\K9\demo-project\src> npm start

> demo-project@0.1.0 start
> react-scripts start

(node:20176) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' option.
(Use 'node --trace-deprecation ...' to show where the warning was created)
(node:20176) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' option.
Starting the development server...
Compiled successfully!

You can now view demo-project in the browser.

Local:      http://localhost:3000
On Your Network:  http://127.0.0.1:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

REFERENCES



Scapy. *Scapy Documentation*. Read the Docs, <https://scapy.readthedocs.io/en/latest/>. Accessed 3 Dec. 2024.

Saeed, Mohammad. "Packet Sniffing in Python (Windows)." *Stack Overflow*, 18 Oct. 2010, <https://stackoverflow.com/questions/462439/packet-sniffing-in-python-windows>. Accessed 3 Dec. 2024.

Alix, Pierre. "Scapy Raw Sockets." *Stack Overflow*, 2 Feb. 2014, <https://stackoverflow.com/questions/23060928/scapy-raw-sockets>. Accessed 3 Dec. 2024.

Microsoft. *SQL Server Documentation*. Microsoft, <https://learn.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver16>. Accessed 3 Dec. 2024.

Microsoft. *SQL Server Management Studio (SSMS) Documentation*. Microsoft, <https://learn.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver16>. Accessed 3 Dec. 2024.

W3Schools. *CSS Reference*. W3Schools, <https://www.w3schools.com/cssref/index.php>. Accessed 3 Dec. 2024.

MDN Web Docs. *JavaScript*. Mozilla, <https://developer.mozilla.org/en-US/docs/Web/JavaScript>. Accessed 3 Dec. 2024.

Wireshark. *Wireshark Documentation*. Wireshark, <https://www.wireshark.org/docs/>. Accessed 3 Dec. 2024.

Wireshark. *Wireshark*. GitHub, <https://github.com/wireshark/wireshark>. Accessed 3 Dec. 2024.

Node.js. *Node.js API Documentation*. Node.js, <https://nodejs.org/docs/latest/api/>. Accessed 3 Dec. 2024.

Wu, Zhihao. "Simple Distributable Desktop App with Python and Web." *Medium*, 19 Nov. 2020, <https://medium.com/@zhihaowu208/simple-distributable-desktop-app-with-python-and-web-abc0665feb3>. Accessed 3 Dec. 2024.

PragimTech. "Install ReactJS." *PragimTech*, <https://www.pragimtech.com/blog/reactjs/install-reactjs/>. Accessed 3 Dec. 2024.