Progress Report - Web-Based Packet Analyzer

Project Title

Web-Based Packet Analyzer

Team Members

Braden Wilson, Conor Culpepper, Josa Rhodes, Suud Shomade, Hunter Hammontree, Wesley White

Meeting Platform

Discord

Project Overview

The Web-Based Packet Analyzer is a lightweight, browser-accessible tool that captures live network traffic using Python and visualizes it in real time. It allows users to monitor incoming and outgoing packets, filter them by protocol or IP address, and optionally view traffic summaries with visual charts. The project focuses on usability, performance, and education - especially for students or those new to network analysis.

Tools & Technologies

- Python - backend logic and packet parsing
- Flask - serves the web application
- Flask-SocketIO - enables real-time WebSocket communication
- Scapy - captures and interprets packets
- HTML/CSS/JavaScript - frontend user interface
- Chart.js (optional) - visualizes network traffic by protocol

Team Roles & Contributions

- Braden Wilson - Frontend developer; responsible for UI layout, styling (dark mode), and integrating live updates with JavaScript.
- Conor Culpepper - Backend developer; handled Flask-SocketIO integration and WebSocket event handling.
- Josa Rhodes - Packet processing lead; worked on parsing packets with Scapy and formatting them for

frontend use.

- Suud Shomade - Project coordinator and documentation lead; managed weekly goals, progress tracking, and this report.
- Hunter Hammontree - Data visualization; working on integrating Chart.js for protocol breakdown charts.
- Wesley White - Systems integration and testing; verifying cross-platform compatibility and handling error cases in real-time capture.

Progress & Milestones

- Project planning & setup: Complete
- Flask web server & Socket.IO setup: Complete
- Scapy packet capture integration: Complete
- Real-time display in web interface: Complete
- Search/filter feature: In Progress
- Chart.js protocol visualization: In Progress
- Thread-safe capture & performance: In Progress
- Final testing/documentation: Upcoming

Team Communication

We meet on Discord 2-3 times per week for code reviews, debugging, and task planning. Team members also collaborate via GitHub Issues and a shared Notion page for documentation. We have a strong rhythm of shared responsibility and clear communication.

Challenges Faced

- Thread Management: Ensuring that the packet sniffing thread does not block the main Flask app.
- Packet Volume: Filtering large volumes of data in real time has introduced performance bottlenecks.
- Frontend Sync: Ensuring the live table updates without glitches during high-traffic bursts.

Next Steps

- Finalize advanced filter functionality (regex, protocol type, port range)

Progress Report - Web-Based Packet Analyzer

- Polish UI with improved packet detail hover or popup modals
- Complete Chart.js integration with interactive toggles
- Begin full code testing across different browsers and OS
- Create user guide and demo video for final presentation