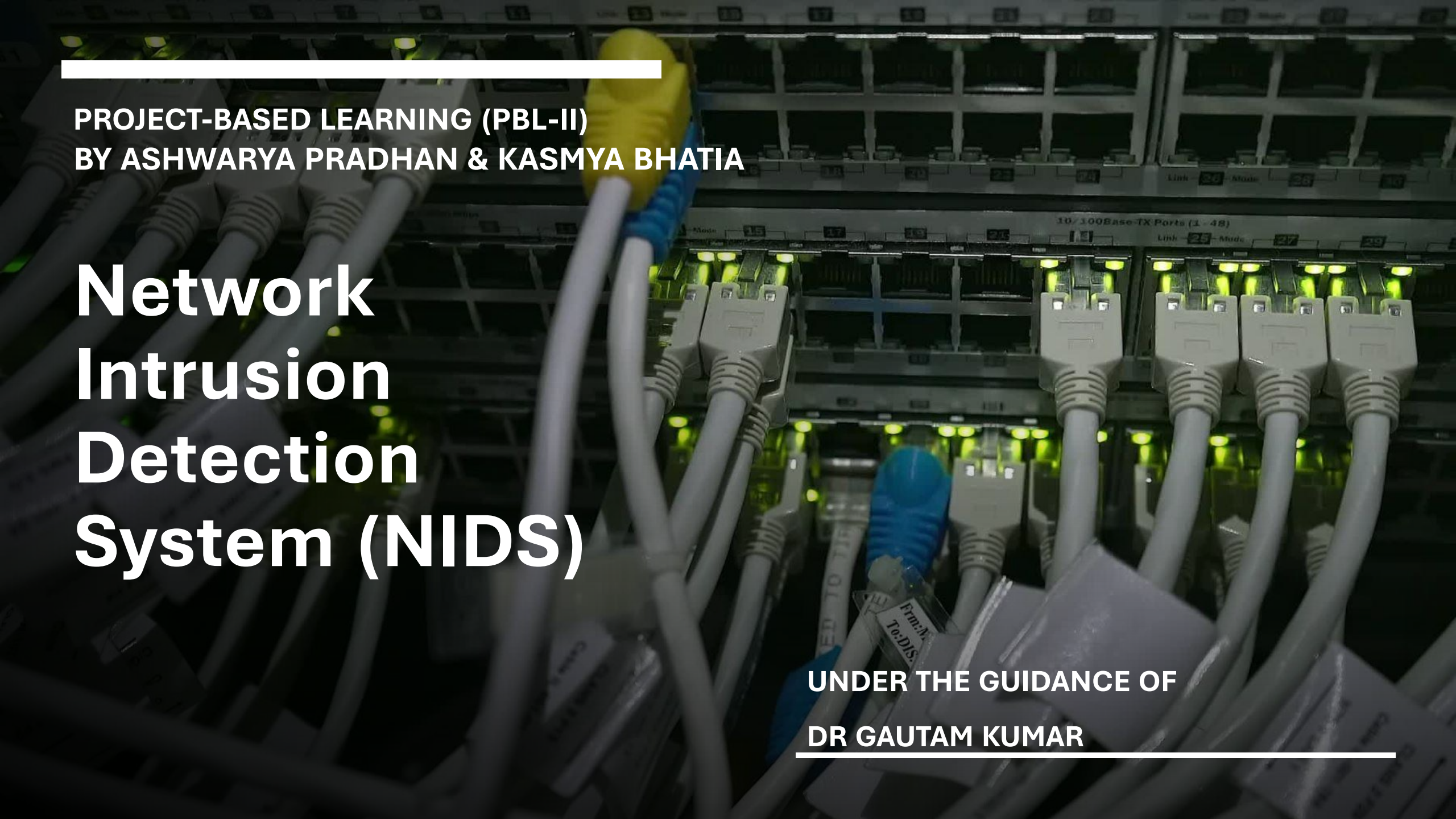


PROJECT-BASED LEARNING (PBL-II)

BY ASHWARYA PRADHAN & KASMYA BHATIA

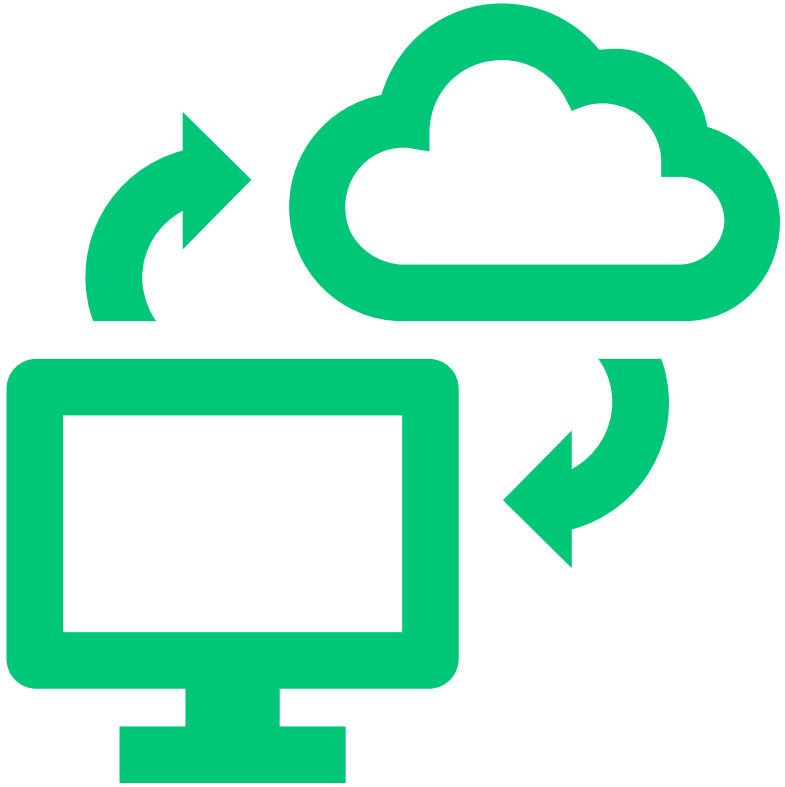
Network Intrusion Detection System (NIDS)

UNDER THE GUIDANCE OF

DR GAUTAM KUMAR

Introduction

- In today’s digital world, organizations & individuals rely on connected systems for vital operations. This increases exposure to cyber threats, making security essential. Even a small vulnerability can lead to widespread disruption and data loss.
- Network Intrusion Detection Systems (NIDS) plays a key role by continuously monitoring network traffic in real time to detect malicious activity, policy violations, and breaches acting as an early warning against advanced threats.



Project Overview

Problem Statement

The exponential growth of digital connectivity has significantly increased cybersecurity threats.

Organizations require real-time solutions to identify and mitigate intrusions efficiently.

Objective

To design and implement a Python-based Network Intrusion Detection System capable of real-time traffic monitoring.

Objectives of NIDS

STRENGTHEN NETWORK SECURITY

IMPROVE ACCURACY IN THREAT DETECTION

PROVIDE INSTANT ALERTS AND QUICK RESPONSE

ENSURE FLEXIBILITY AND SEAMLESS INTEGRATION

Functioning of NIDS

Comprehensive Traffic Analysis - Continuous inspection of all inbound and outbound network communications

Advanced Threat Detection - Identification of malicious patterns and anomalous behavior that evade signature-based tools

Real-Time Alerting - Immediate notification of potential security incidents for rapid response

System Design

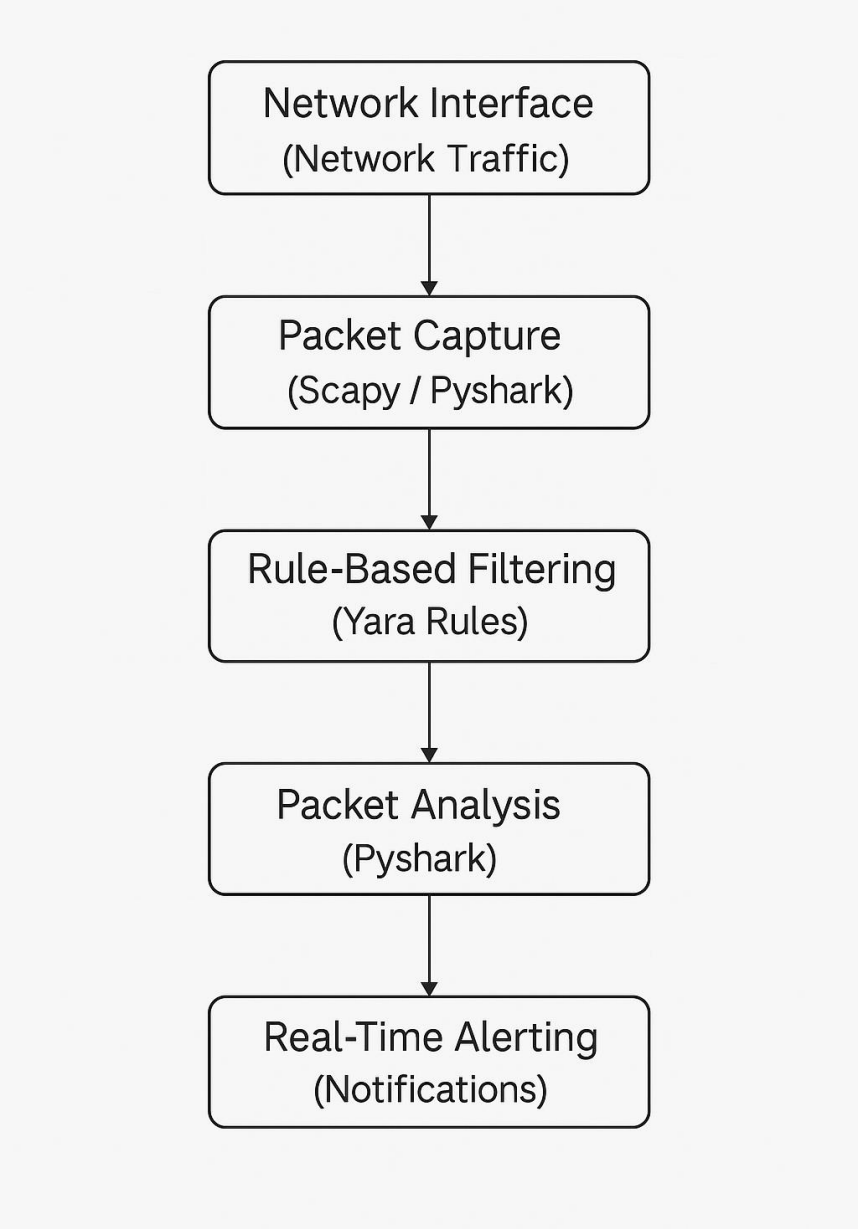
Main Components of the system:

- **Packet Capture Module:** Collects real-time network traffic using Scapy & Pyshark.
- **Filtering Module:** Applies YARA rules to filter out suspicious packets.
- **Analysis Module:** Extracts key information like IPs, ports, and payload for examination.
- **Alert Module:** Displays or logs warnings when a threat is detected.

System Architecture

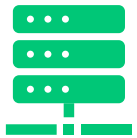
The system architecture diagram illustrates a **modular** design leveraging open-source tools, where each core component is responsible for a specific function, allowing for clear separation of concerns.

This modular design ensures flexibility and scalability, as each component can evolve and be maintained independently, making the system more adaptable to future security needs.



Tools & Libraries Used

Python 3: Core programming language for building the NIDS system.



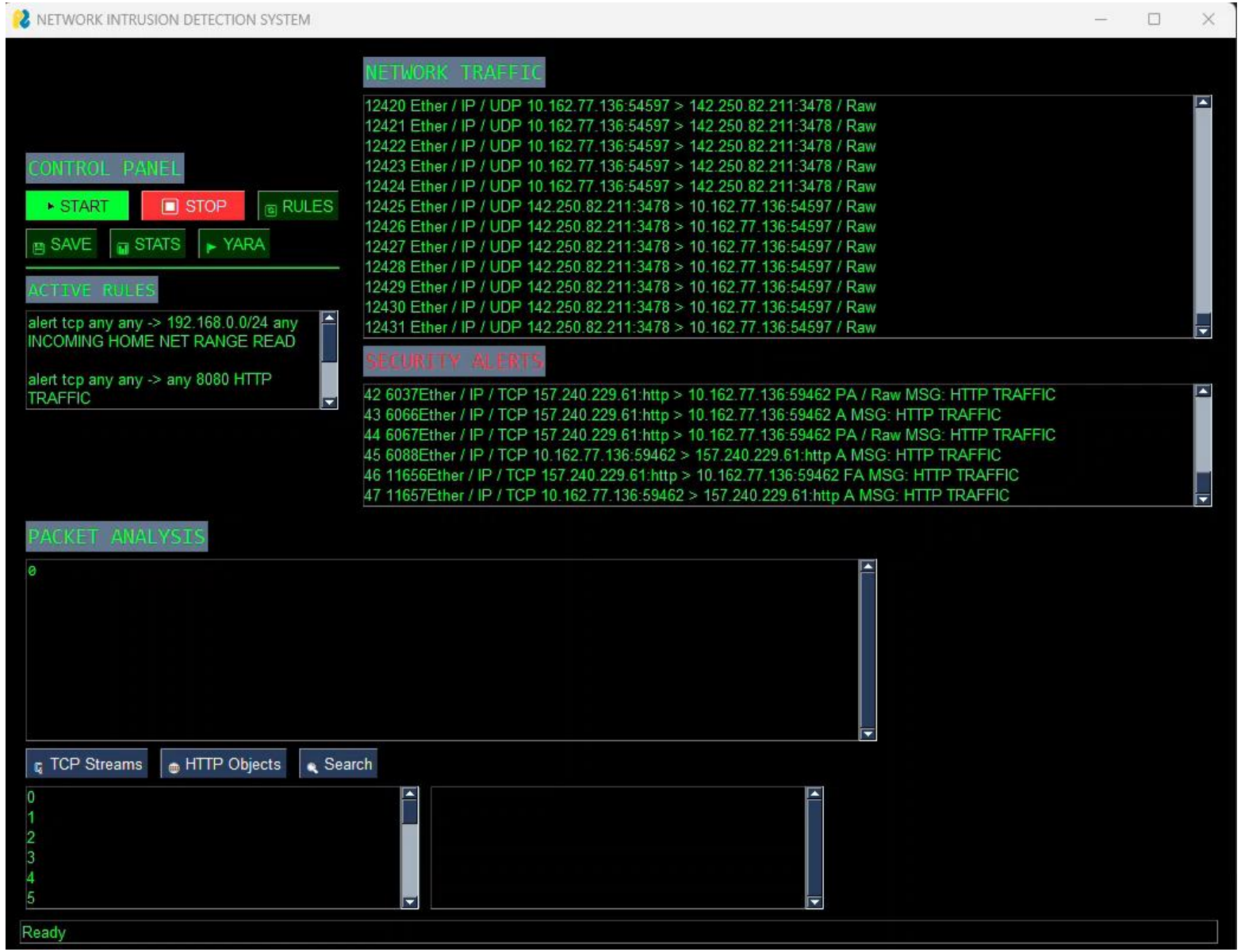
Scapy: Captures and crafts network packets at a low level.

Pyshark: Parses and analyzes packets using Wireshark's capabilities.

YARA: Applies rule-based pattern matching for threat detection.

SimplePyLibrary: Provides a GUI interface for user interaction.

Implementation

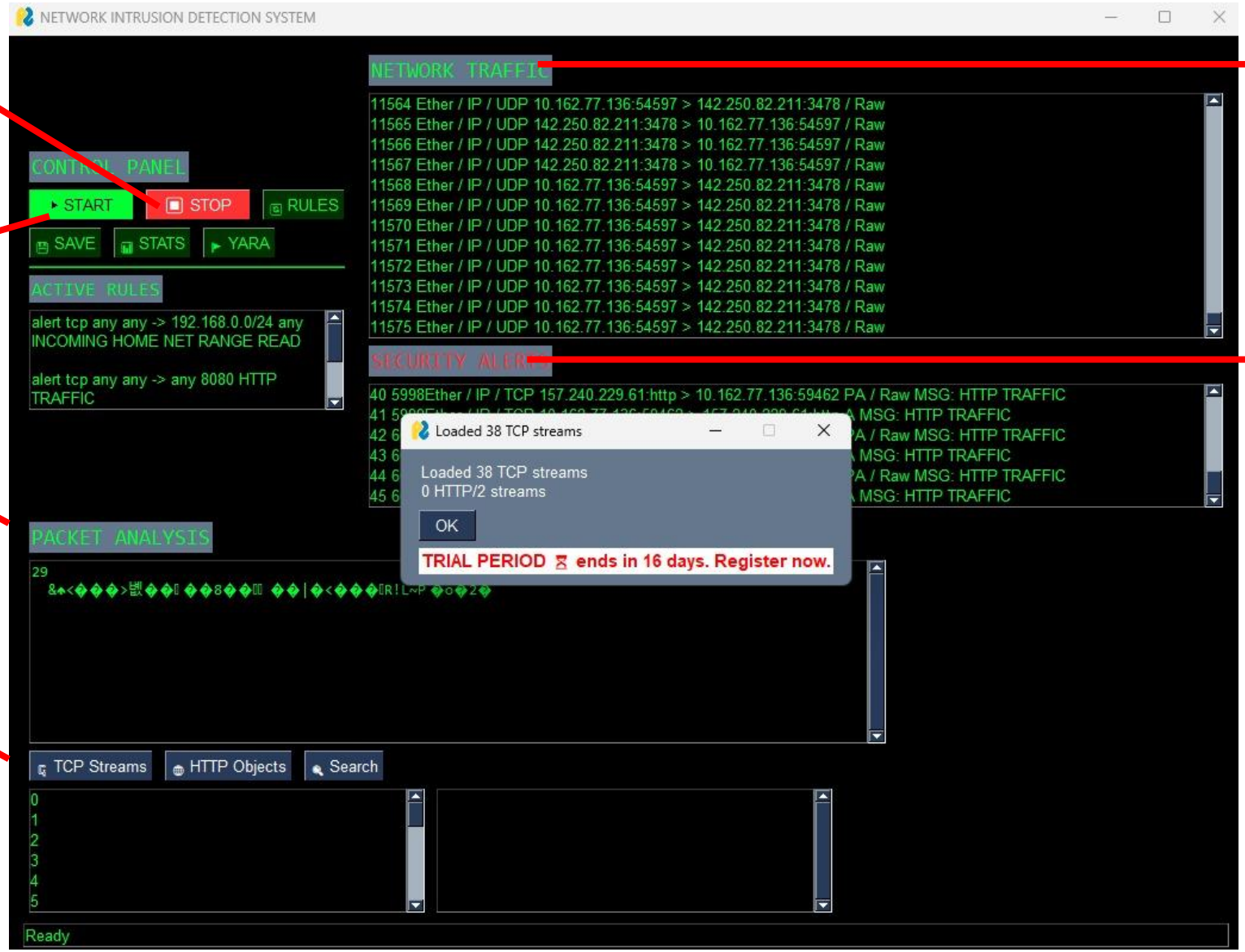


Stops the ongoing packet capture

Initiates packet capture on the selected network interfaces

Displays detailed payload information

Reconstructs and lists all TCP streams



Shows all captured packets in summary form

Displays packets that triggered detection rules

Use Case Scenarios

REAL-TIME MONITORING: CAPTURES AND ANALYZES NETWORK TRAFFIC LIVE.

MODULAR DESIGN: EASY TO UPDATE OR EXPAND DETECTION RULES AND FEATURES.

CUSTOM RULE APPLICATION: SUPPORTS YARA-BASED THREAT DETECTION.

CROSS-ENVIRONMENT USAGE: SUITABLE FOR HOMES, ENTERPRISES, AND EDUCATION.

LIGHTWEIGHT & EXTENSIBLE: DESIGNED FOR SCALABILITY AND LOW RESOURCE USE.

Opportunities for Enhancements

ML Integration: Add anomaly detection using machine learning.

More Protocols: Support HTTP, DNS, FTP, etc.

GUI Dashboard: Build with SimplePyLibrary for real-time visuals.

Log Management: Add structured storage and search features.

Auto Rule Updates: Sync latest YARA rules from online sources.

Cross-Platform Support: Run on Linux, Windows, and cloud setups.

Conclusion

- Successfully built a Python-based NIDS for real-time detection of network threats using Scapy, Pyshark, and YARA.
- Gained deep insights into network protocols, intrusion detection techniques, and packet-level analysis.
- Demonstrated the feasibility of a modular, customizable, and lightweight NIDS for diverse use cases.
- Project serves as a foundation for future enhancements such as ML integration and advanced protocol support.