University of Colorado **Boulder**

# Project Proposal

Abhishek Koppa & Abhishek Nadgir

*ECEN 5623, Real-Time Embedded Systems*

*April 9th, 2025*

# Table of Contents

# Contents

# List of Figures

# Project Overview

This project aims to develop a lightweight Intrusion Detection System (IDS) on a multi-core Raspberry Pi 4 using the Data Plane Development Kit (DPDK). The system captures Ethernet traffic in real-time, processes packets to detect malicious behavior, and logs security-related events. It is designed with a Real-Time Embedded Systems (RTES) focus, featuring rate-monotonic scheduling (RMS) for concurrent threads on shared cores.
DPDK APIs are used to enable high-speed packet capture and memory-efficient inter-thread communication using ring buffers. The IDS system architecture uses multiple threads mapped to specific CPU cores to ensure parallelism and reduce processing delay.

## High Level Requirements

- Hardware Platform
  - 1 × Raspberry Pi 4B
  - 1 × LED (connected via GPIO)
- Operating System: Raspberry Pi OS (Debian-based)
- Key features:
  - Real-time Ethernet packet capture using DPDK
  - Multi-core threaded architecture with per-core responsibilities
  - Detection of suspicious network patterns
  - RMS-scheduled concurrent services (LED feedback, logger)
  - Logging to structured CSV format
- Real-Time Requirements (Tasks and Deadlines):

| | Task | Thread Name | Core # | Period | Deadline | RM Priority | Description |
|---|---|---|---|---|---|---|---|
| 0 | RT1 | Data Logger | 3 | 100 ms | 100 ms | 1 (Lowest) | Logs received and flagged packet info to CSV |
| 1 | RT2 | LED Blinker | 3 | 50 ms | 50 ms | 2 | Blinks LED based on current threat level |
| 2 | RT3 | Sequencer | 3 | 20 ms | 20 ms | 3 (Highest) | Ensures periodic release of logger and LED tasks |
| 3 | RT4 | Packet Reception Thread | 1 | 50 ms | 50 ms | N/A | Captures incoming packets and stores them in ring buffer |
| 4 | RT5 | Packet Processing and Intrusion Detection | 2 | 100 ms | 50 ms | N/A | Processes packets from buffer and detects threats |
| 5 | NRT1 | Linux OS and Kernel | 0 | Always Running | Always Running | N/A | Linux kernel and operating system tasks |

Figure 1: Real-time task breakdown

  - All threads run on a single core to enforce RM scheduling constraints.
  - Worst-Case Execution Time (WCET) for each task will be profiled and used in utilization tests (e.g., Liu and Layland bound for 3 tasks ≈ 0.78 total CPU utilization).

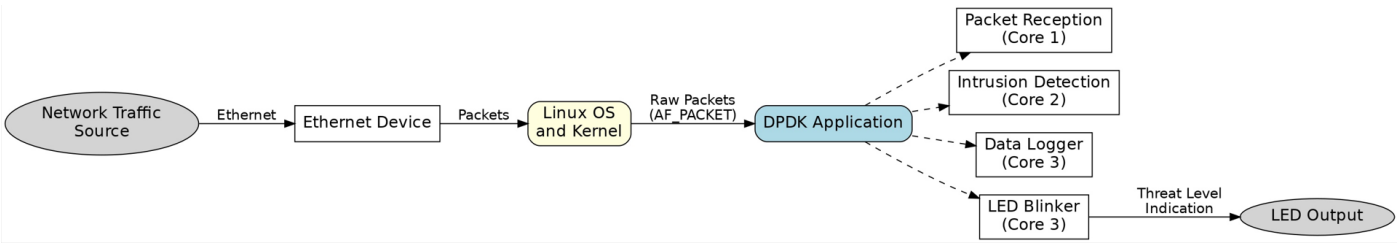


Figure 2: Simplified System Architecture

# Must-Have Minimum System Requirements

- **Real-Time Packet Capture** using DPDK (rte_eth_rx_burst)
- **Multi-core Thread Assignment:**
  - Core 0: Linux OS and Kernel
  - Core 1: Packet Reception Thread
  - Core 2: Packet Processing and Intrusion Detection
  - Core 3: Data Logger, LED Blinker, and Sequencer (all RMS scheduled)
- **Ring Buffer Communication** using DPDK rte_ring
- **CSV Logging** of packets and intrusion alerts
- **LED Feedback** using GPIO to indicate system threat level
- **Rate Monotonic Scheduling (RMS)** implemented for independent services on Core 3

# Good-to-Have Target Requirements

- **Support for basic detection rules** (e.g., port scans, malformed packets, SYN floods)
- **Timestamped logging** with severity level and packet details
- **Filter out suspicious traffic** based on port/IP
- **Configuration file** for enabling/disabling detection rules

# Stretch Goals (Extra Features)

- **Integration with Real Traffic** Sources instead of dummy packet generators (e.g., using iperf, real LAN traffic, or mirroring)
- **Live visual dashboard** to display logs and metrics

# Subsystem Summary

- **Packet Capture Subsystem:**
  - Captures incoming Ethernet packets at high speed
- **Intrusion Detection Subsystem:**
  - Processes incoming packets
  - Analyzes packet headers and content for suspicious or malicious activity
  - Identifies and flags threats based on defined security criteria
- **Data Logging Subsystem:**

  - Records packet data including source/destination addresses, protocol, and threat classification to persistent storage

- **System Status Indicators:**
  - Provides visual indication of system threat status
    * Normal operation: Slow blinking LED
    * Threat detected: Rapid blinking LED
    * General data activity: Separate continuously active LED

# Test Plan Approach

- **Functional Testing:**
  - Verify packet capture, processing, and logging via real or simulated traffic
  - Validate LED behavior and log accuracy against known intrusion scenarios
- **Real-Time Constraints Testing:**
  - Measure and log packet detection latency from packet reception to threat detection
  - Measure and analyze end-to-end detection latency, optionally using synchronized hardware and software profiling between transmitting and receiving devices
  - Verify that real-time deadlines and system response times are consistently met
- **Stress Testing:**
  - Saturate packet reception rate to assess system stability
  - Monitor and record system behavior under peak load conditions, including buffer management
- **Scenario Testing:**
  - Test system response to benign traffic versus specifically crafted malicious packets
  - Ensure only genuinely malicious traffic triggers alerts

# Proposed Schedule

| Task | Responsible | Target Date |
|---|---|---|
| Set up DPDK environment and verify packet capture on RPi | Abhishek N | April 18 |
| Implement ring buffer logic to enqueue and dequeue packets across threads | Abhishek K | April 19 |
| Build packet processing logic to extract headers and detect basic patterns | Abhishek N | April 20 |
| Develop RMS-based sequencer to periodically invoke logger and LED tasks | Abhishek K | April 21 |
| Implement LED blinking thread with different threat levels | Abhishek N | April 22 |
| Implement CSV logger to record packet info and intrusion alerts | Abhishek N | April 23 |
| Integrate all threads (Rx, Detector, Logger, LED) across cores | Abhishek K | April 24 |
| Simulate intrusion scenarios using Python scripts and verify detection | Abhishek K | April 25 |
| System testing, performance evaluation, and final cleanup | Both Abhisheks | April 27 |