NFC-IET, Multan

BSAI-2k24

Project Proposal

Project Title:

“IoT Intrusion Detection System”

Submitted by:

Moavia Amir (2k24_BSAI_72)

Muhammad Ramzan (2k24_BSAI_21)

Semester: 3rd

Submitted to: Prof. Khalid Mehmood Khan

Submitted Date: November 13, 2025

Abstract

The **IoT Intrusion Detection System (IDS)** is a lightweight, classroom-safe project designed to monitor IoT sensor networks and detect anomalous behaviours in real-time. With the rapid adoption of IoT devices in smart homes, laboratories, and industrial environments, security has become a critical concern. IoT devices are often resource-constrained and lack robust security measures, leaving them vulnerable to attacks such as **message-rate floods**, **byte-rate floods**, and **spoofing or tampering (MITM)**.

This project uses **ESP8266/NodeMCU sensors** to publish data to a local **MQTT broker**, with a Python-based IDS running on a laptop to monitor traffic, measure message and byte rates, and identify tampering attempts. Alerts are visualised on a simple **Flask web dashboard**, providing students with an intuitive view of sensor activity and detected anomalies.

The system is designed for **safe classroom demonstrations**, using isolated networks and owned devices to simulate attacks without risking external networks. Through this project, students gain practical experience in monitoring, detecting, and understanding IoT security threats, learning core information security concepts such as **traffic analysis, anomaly detection, and secure network monitoring**.

Additionally, the project lays the foundation for **future enhancements**, including automated mitigation, mobile dashboards, cloud integration, and support for multiple sensors. The IoT IDS demonstrates a hands-on, educational approach to IoT security, bridging the gap between theoretical knowledge and practical implementation in real-world networks.

Acronyms / Abbreviations

- **IDS** – Intrusion Detection System
- **MITM** – Man-in-the-Middle
- **MQTT** – Message Queuing Telemetry Transport
- **ESP** – ESP8266/NodeMCU Microcontroller

Introduction:

IoT devices are increasingly used in smart homes, labs, and industrial environments due to their convenience and automation capabilities. However, these devices are often resource-constrained and lack strong security measures, making them vulnerable to attacks such as message floods, large payloads, spoofing, and tampering. Such attacks can compromise data integrity, reduce availability, and even disrupt critical systems.

The **IoT Intrusion Detection System (IDS)** project demonstrates how a lightweight, local IDS can monitor IoT sensor traffic, detect anomalous behaviours in real-time, and alert users through a simple web dashboard. This project provides a practical, classroom-safe demonstration of **information security principles applied to IoT networks**, helping students understand attack detection, safe testing, and network monitoring.

Objectives:

The primary objectives of the IoT Intrusion Detection System project are:

1. Build a classroom-safe IDS that monitors MQTT traffic from ESP sensors.
2. Detect three core attack types:

    - ✓ **Message-rate flood** — high frequency of small messages.
    - ✓ **Byte-rate flood** — large payloads causing bandwidth spikes.
    - ✓ **Spoofing/Tampering (MITM)** — identity or data manipulation.

3. Provide a minimal web UI showing live sensor data and alert status.
4. Ensure the demonstration is safe, limited to owned/isolated networks, and reproducible.
5. Introduce students to practical information security techniques for IoT networks.

Scope

In-Scope:

- Monitoring MQTT traffic from ESP8266/NodeMCU sensors.
- Detecting the three attack types: message-rate flood, byte-rate flood, and spoofing/tampering.
- Displaying real-time alerts on a local web dashboard.
- Demonstrating attacks safely using a second ESP or mobile device on a controlled network.

Out-of-Scope:

- Deployment on public or university networks.
- Integration with cloud IoT platforms (except optional future extension).
- Advanced automated mitigation or blocking of attacks beyond alerts.

Functional Requirements

1. ESP sensors must publish JSON payloads to MQTT topics at a normal rate (~1 msg/sec).
2. IDS must subscribe to all sensor topics and calculate:

    - ✓ Messages per second (msg_rate)
    - ✓ Bytes per second (byte_rate)

3. IDS must raise alerts if thresholds are exceeded:

    - ✓ Message-rate flood alert (msg_rate > 30 msg/s)
    - ✓ Byte-rate flood alert (byte_rate > 4000 B/s)

4. IDS must detect spoofed device IDs or tampered messages.
5. Web dashboard must display:

- ✓ Sensor readings (temperature)
- ✓ System status (Normal / Alert / Attack type)

Non-Functional Requirements

- **Performance:** IDS updates must occur in real-time (max 200 ms latency).
- **Reliability:** Alerts must be consistent and repeatable during demonstrations.
- **Security:** The system must operate only on a private, isolated network.
- **Usability:** Dashboard should be simple to read and operate for classroom demonstration.
- **Portability:** ESP firmware and Python scripts should run on any compatible laptop/ESP setup.

Methodology

- **Development Approach:** Iterative/Agile approach with weekly milestones.
- **Implementation Steps:**
  1. Set up Mosquitto MQTT broker on laptop.
  2. Program ESP sensors to publish normal and attack traffic.
  3. Develop Python-based IDS to monitor msg_rate, byte_rate, and spoofing/tampering.
  4. Create a simple Flask web UI for real-time alerts.
  5. Test normal operation, then simulate attacks safely using ESP or phone.
- **Testing & Validation:** Run three attack scenarios in isolation and confirm IDS alerts in real-time.

Timeline and Milestone

Weeks	Tasks
1	Setup Mosquitto MQTT broker & ESP environment
2	Program ESP sensor to publish normal data
3	Develop Python IDS (msg_rate & byte_rate detection)
4	Implement spoofing/tampering detection
5	Integrate all components and run controlled demos

Ethics & Safety Statement:

- All demonstrations will be performed on **owned and isolated networks** only.
- No attacks will be executed on public or institutional networks.
- Students must not attempt MITM or flooding attacks on devices they do not own.
- All data collected is for demonstration only; no personal data will be used.

Working Principle:

- **Sensor Data Publishing:**
  - The ESP sensor publishes JSON messages containing temperature readings to a designated MQTT topic (e.g., `home/sensor/temp`) at a normal rate (~1 message per second).
- **IDS Monitoring:**
  - The laptop runs the MQTT broker and subscribes to all sensor topics.
  - The Python-based IDS continuously measures:
    - **Message rate (msg/sec)** per topic
    - **Byte rate (bytes/sec)** per topic
- **Attack Detection:**
  - If the **message rate** exceeds the threshold → IDS raises a **message-rate flood alert**.
  - If the **byte rate** exceeds the threshold → IDS raises a **byte-rate flood alert**.
  - If a device ID or payload is tampered with → IDS raises a **spoof/tamper alert**.
- **Alert Notification:**
  - Alerts are displayed in real-time on the Flask web dashboard with the type of attack and affected topic.
  - The dashboard shows sensor readings, system status (*Normal / Alert / Attack type*), and timestamps.
- **Safe Demonstration:**
  - Attacks are simulated using a second ESP or a mobile device (with MQTT app/Termux) on an isolated network.
  - No external or public network devices are involved, ensuring classroom safety.

Expected Outcomes:

After completing this project, the following outcomes are expected:

1. **Functional IDS Prototype:**
    - A working Intrusion Detection System that monitors IoT sensor traffic in real-time.
    - Correctly detects message-rate floods, byte-rate floods, and spoofing/tampering attempts.
2. **Web Dashboard Visualization:**
    - Live display of sensor readings, system status, and alerts for different attack types.
3. **Classroom-Safe Demonstration:**
    - Simulated attacks executed safely on isolated devices.
    - Clear demonstration of IDS responses to abnormal network activity.
4. **Hands-On Learning:**
    - Understanding practical implementation of information security in IoT networks.
    - Demonstration of real-time monitoring, alerting, and simple mitigation concepts.
5. **Documentation & Reporting:**
    - Complete code repository for ESP sketches and Python scripts.
    - Screenshots, logs, and demonstration videos showing the detection of each attack type.

Future Enhancements:

- Add automated mitigation (block/throttle suspicious devices).
- Integrate with cloud platforms (ThingSpeak/Blynk) for remote monitoring.
- Support additional sensors (motion, humidity, etc.).
- Build a mobile app for alerts and dashboard.
- Add MQTT authentication/encryption for stronger security.
- Test scalability with multiple sensors and topics.

References:

1. Mosquitto MQTT Broker Documentation — https://mosquitto.org/documentation/

2. Paho MQTT Python Client Documentation — https://www.eclipse.org/paho/

3. NodeMCU & ESP8266 Programming Guides — https://nodemcu.readthedocs.io/

Conclusion:

The **IoT Intrusion Detection System** project demonstrates a practical, classroom-safe approach to monitoring IoT networks and detecting attacks such as message floods, large payloads, and tampering. By combining ESP sensors, Python-based IDS, and a real-time web dashboard, the system provides hands-on learning for information security principles. This project underscores the importance of protecting IoT systems and lays the foundation for future enhancements.
