الجامعة السورية الخاصة
SYRIAN PRIVATE UNIVERSITY

An Intrusion Detection System in Sdn-enabled IoT Networks

Faculty of Informatics Engineering Department of
Systems Security and Computer Networking

Prepared by : Nagham Ashqar & Moustafa Seifo

1. Introduction

The Internet of Things (IoT) has transformed modern computing by enabling seamless connectivity between everyday devices — from smart home appliances to industrial sensors — facilitating automation, remote monitoring, and data-driven services. However, this rapid expansion also introduces profound security challenges, as the sheer scale and heterogeneity of IoT ecosystems expose numerous attack surfaces that adversaries can exploit. Many IoT devices are inherently resource-constrained and lack robust built-in security measures, resulting in vulnerabilities that traditional network defenses cannot effectively address. This combination of high connectivity and weak security posture makes IoT environments increasingly susceptible to cyberattacks such as distributed denial-of-service (DDoS), unauthorized access, and message manipulation, posing threats to confidentiality, integrity, and availability across critical infrastructures (e.g., smart utilities, healthcare systems). Contemporary research emphasizes that the conventional security techniques used in traditional networks are ill-equipped to meet the distinctive security needs of IoT systems due to their inability to adapt to dynamic traffic and protocol-specific behaviors inherent in IoT communications.[1]

Software-Defined Networking (SDN) has emerged as a promising architectural paradigm to enhance the security and manageability of complex network environments, including IoT. By decoupling the control plane from the data plane, SDN provides centralized network intelligence, global visibility across traffic flows, and programmable control over network behavior. This architectural flexibility allows security functions, such as anomaly detection and flow-based policy enforcement, to be abstracted from individual devices and handled centrally at the controller, enabling rapid responses to malicious activity. For IoT environments, SDN's centralized control enables scalable, protocol-agnostic management of diverse IoT traffic patterns and supports fine-grained enforcement of security policies, which are crucial for detecting and mitigating threats that originate from within or propagate across IoT networks. Studies have shown SDN's potential in providing improved visibility and control for security applications, including intrusion detection and access control, making it a suitable foundation for intelligent security solutions in IoT domains. [1]

Despite the architectural benefits provided by SDN, traditional intrusion detection systems — which often rely on static signatures or rule-based detection — are limited in their effectiveness in dynamic and heterogeneous IoT settings. Signature-based IDS cannot generalize to new or unknown attack variants and often fail to capture the subtle statistical differences between normal and malicious traffic in IoT protocols (such as MQTT). To address these limitations, the integration of machine learning (ML) techniques into IDS frameworks has been widely recognized in the literature as a powerful alternative. Machine learning–based IDS can model normal network behavior and detect deviations associated with attacks, enabling the identification of novel and evolving threats without relying on pre-existing signatures. By leveraging supervised and unsupervised learning algorithms, ML-based IDS can adapt to the diverse and voluminous traffic characteristic of IoT environments. The diverse applications of IoT, each with its unique data types and attack vectors, require IDS frameworks that are flexible and capable of providing comprehensive protection against a wide range of threats. [2]

In IoT communications, lightweight messaging protocols such as Message Queuing Telemetry Transport (MQTT) are commonly used due to their minimal overhead and suitability for resource-constrained devices. However, these protocols introduce additional complexity for security monitoring because traditional network

IDS tools are often not optimized for application-layer traffic semantics. The MQTTset dataset — which contains labeled benign and malicious MQTT traffic — provides an important benchmark for evaluating intrusion detection approaches designed specifically for MQTT-based IoT traffic. Using realistic IoT traffic patterns, including both normal operation and attack scenarios, enables machine learning models to learn discriminative features that reflect real-world behavior, improving the IDS's ability to detect threats in MQTT environments.[3] Evaluating IDS performance using such specialized real-world datasets is critical, as it ensures that developed models can generalize beyond synthetic or generic traffic datasets and operate effectively under realistic network conditions.

2. Problem Identification and Research Study

2.1 Problem identification

2.1.1 IoT Security Challenges

The Internet of Things (IoT) is growing explosively – by 2024 it reached tens of billions of devices – which greatly expands the attack surface of networks. Each new IoT device (sensors, home appliances, industrial controllers, etc.) can introduce vulnerabilities that attackers exploit to infiltrate entire systems. Many IoT nodes are highly resource-constrained and shipped with weak or outdated security (insufficient encryption/authentication, stale firmware). As a result, adversaries increasingly target such devices, enabling botnets and data breaches. For example, Mirai-style botnets have commandeered insecure cameras and routers, and recent attacks (e.g. ransomware on industrial systems, smart home hacks) demonstrate how exploited IoT devices can disrupt critical services. In short, the heterogeneity and poor security of IoT devices pose a severe cybersecurity problem: the uncoordinated, complex IoT environment enables attackers to compromise devices and move laterally, threatening privacy, availability, and safety across interconnected networks[4].

2.1.2 SDN-Based Centralized Control in IoT

Traditional IoT networks lack a unified management plane. Software-Defined Networking (SDN) offers a solution by decoupling the control plane from data forwarding, placing a central controller "brain" over the network. This centralized control is highly relevant to IoT: the SDN controller can maintain a global view of all IoT traffic and quickly enforce security policies. For example, SDN enables real-time network monitoring and anomaly detection across all links[1]. It also supports dynamic traffic management and fine-grained policy enforcement – the controller can isolate suspicious flows or quarantine devices on-the-fly[1]. By using SDN in an IoT setting, we gain unified visibility and programmability: we can push flow rules, collect statistics, and reconfigure the network centrally to mitigate attacks. In short, SDN's programmable centralized architecture makes it well-suited to manage the complex, heterogeneous IoT domain and respond to threats quickly[1][5].

2.1.3 ML-Based IDS vs. Traditional IDS

Conventional signature-based Intrusion Detection Systems rely on known attack signatures and must be constantly updated; they cannot recognize novel or obfuscated threats. In contrast, intelligent (machine learning–based) IDS can learn patterns of normal vs. malicious behavior and thus detect previously unseen attacks. ML-based IDS have two key advantages: they can adapt to evolving threats and they often achieve higher detection accuracy with fewer false alarms. For instance, state-of-the-art reviews note that ML-driven IDS "improves detection accuracy, adapts to new and evolving threats, identifies unknown threats, [and] reduces false positives" compared to static methods[6][2]. Anomaly-based ML systems build a model of normal IoT behavior and flag deviations, so they can catch zero-day exploits that signature systems miss[7][2]. In fact, modern SDN-IoT studies report significantly improved IDS performance when ML/DL techniques are integrated: one survey highlights that ML/DL models "significantly improved IDS accuracy" in SDN environments[7]. Overall, ML-based IDS are essential for IoT because they learn from data (including IoT-specific traffic patterns) and can generalize to new attack variants[7][2].

In response to the increasing security challenges faced by IoT networks, this project focuses on the design of an intrusion detection solution that addresses the limitations of traditional security mechanisms in such environments. The adoption of Software-Defined Networking enables centralized traffic monitoring and flexible control, which is particularly suitable for managing the scale and heterogeneity of IoT systems. To overcome the shortcomings of signature-based intrusion detection approaches, this work emphasizes machine learning–based techniques capable of detecting complex and previously unseen attack behaviors. Furthermore, the project concentrates on MQTT-based IoT communication, leveraging the MQTTset dataset to evaluate the proposed solution under realistic IoT traffic conditions. This focused combination of IoT security challenges, SDN-based control, machine learning–driven intrusion detection, and MQTT-aware traffic analysis forms the foundation of the proposed research.

2.2 Research Study

2.2.1 SDN in IoT

Researchers agree that SDN is a natural fit for IoT security. SDN's centralized controller can enforce network-wide policies in heterogeneous IoT deployments. For example, Al Hayajneh *et al.* (2020) proposed an SDN-based architecture to mitigate man-in-the-middle and other attacks in HTTP-based IoT systems, showing that the SDN controller enables rapid threat response and granular filtering[1]. However, the proposed architecture primarily focuses on HTTP-based communication and does not address lightweight IoT protocols such as MQTT, which limits its applicability to modern IoT environments.

Similarly, recent surveys note that SDN "provides centralized control, dynamic traffic management, [and] comprehensive network visibility" in IoT networks, which jointly enhance security[4]. In practice, SDN controllers like RYU can act as the network "brain" for IoT: they collect flow statistics from switches/emulated IoT links (e.g., via Mininet) and apply policies globally. These features (centralized monitoring, on-demand flow rules, isolation between network segments) allow SDN-enabled IoT systems to detect anomalies and quarantine threats more effectively than legacy networks[4][1]. Also, a representative and well-structured SDN-based security architecture for IoT is proposed by Karmakar et al. (2020), where the SDN controller acts as a policy decision authority for the entire IoT network [8]. In their architecture, IoT devices (sensors and actuators) connect to the network through OpenFlow-enabled gateways, which are controlled by a centralized SDN controller. The controller maintains a global view of the network topology, device identities, and traffic flows, enabling it to enforce fine-grained security policies dynamically. On the other hand, while the architecture effectively establishes centralized policy enforcement and device-level access control, it does not incorporate intelligent intrusion detection mechanisms capable of analyzing traffic behavior or identifying sophisticated network-based attacks. Recent studies have also explored SDN-based architectural enhancements for IoT environments. For example, Bedhief et al. proposed an SDN–Docker-based IoT architecture that leverages centralized SDN control and containerized services to improve scalability, flexibility, and traffic management, while providing a suitable foundation for deploying advanced security mechanisms in IoT networks [13].

2.2.2 ML-based IDS in SDN-IoT

The integration of machine learning (ML) / deep learning (DL) with Software-Defined Networking (SDN) has become a major research trend for securing IoT environments, because SDN enables centralized traffic visibility and programmable enforcement at the controller, which are ideal for deploying data-driven IDS functions. Recent work shows that ML/DL models can be embedded as SDN security applications that analyze flow statistics (e.g., packet counts, durations, rates) gathered from OpenFlow switches and then trigger mitigation rules when attacks are detected. For example, an SDN-enabled IoT IDS implemented using Mininet and a Ryu controller demonstrates how lightweight statistical flow features can be extracted and classified using ML algorithms (e.g., SVM, MLP, KNN) to detect botnet-related anomalies in IoT traffic [10]. In parallel, broader IDS frameworks propose multi-stage or multi-attack ML-based systems that compare multiple classifiers (e.g., RF, SVM, LR, KNN, XGBoost) and emphasize that algorithm selection strongly affects false alarms and generalization across datasets [11]. Beyond intrusion detection, SDN-IoT research also leverages ML for traffic intelligence and classification, showing that ML can be applied at the SDN control layer using a small set of time-based features (useful even under encryption), while comparing feature-selection techniques (e.g., SFS vs. SHAP) and standard classifiers (RF, DT, KNN) [12]. More recent studies reinforce the comparative picture: classical ML methods such as Random Forest often remain competitive because they can achieve strong accuracy with lower computational cost, while sequential DL models such as LSTM can better capture temporal dependencies but may exhibit higher false positives depending on the dataset and feature quality [14].

Recent work has proposed ML/DL-based IDS for SDN-enabled IoT networks, often evaluated in simulation or using benchmark IoT/SDN datasets. Table 1 compares five representative studies (2020–2025) by the attacks they model, IDS deployment, ML/DL methods, data used, mitigation strategy, and key advantages/limitations.

Study (Year)	Attacks Simulated	IDS Placement	Algorithms (Dataset)	IDS Type	Mitigation Approach	Pros	Cons
[16]	DoS, Probe, U2R, R2L	SDN controller	Hybrid Feature Selection + LightGBM (NSL-KDD)	Machine Learning (Hybrid)	None (detection only)	Very high accuracy (~98%)[1]; fast training	No mitigation mechanism; only evaluated on NSL-KDD (non-IoT)
[17]	DoS, DDoS, Port scanning, Fuzzing, OS fingerprinting	Mininet (OpenFlow SDN)	LSTM, CNN, DNN, SVM (Custom SDN-IoT datasets DS1, DS2)	Deep Learning	None (detection only)	Multi-class detection (≈97% accuracy); tested on two realistic SDN-IoT datasets	High computational overhead (limits real-time use); no mitigation
[18]	DDoS, MiTM (ARP spoofing)	Distributed (controller + switches)	BBSC (Belief-Based Secure Correlation, in SDN testbed)	Hybrid (encryption + flow rules)	Yes – encrypted data and flow rerouting to trusted switches	Real-time prevention of DDoS/MiTM using SDN flow control	scope limited to DDoS and ARP-based MitM
14	MiTM (ARP/DNS spoofing, session hijack)	Offline (simulated dataset)	LSTM, Random Forest, SVM (Custom IoT traffic dataset)	Machine/Deep Learning	None (detection only)	High detection rates (RF 94%, LSTM 92%); detailed MitM taxonomy	No SDN deployment or live testing; no active mitigation strategy

Table 1- A comparative table for recent works

The comparative table highlights recent machine learning-based IDS implementations in SDN-enabled IoT environments, showcasing the diversity in attack scenarios, detection algorithms, and deployment strategies. It underscores the research trend toward controller-based ML detection but also reveals a key limitation: many studies focus solely on offline detection, lacking real-time mitigation. This comparison helps justify the design choices in our project, particularly the emphasis on integrated detection and response within a live SDN-IoT testbed.

2.2.3 Datasets Used

Researchers use a mix of general-purpose and IoT-specific datasets to train and test IDS models. Older intrusion datasets like KDD Cup 1999 and NSL-KDD (and derivatives) are still cited in literature, but they lack modern IoT characteristics[3]. More recent IDS collections include UNSW-NB15 and the CIC-IDS series (CIC-IDS2017, CSE-CIC-IDS2018, CICDDoS2019), which contain up-to-date attack types. Crucially for IoT, specialized datasets have been developed: for example, BoT-IoT (UNSW) and N-BaIoT contain IoT botnet traffic (Mirai, Bashlite, etc.) in smart devices[3][9]; TON_IoT focuses on Industrial IoT sensor data[9]; IoT-23 is a labeled flow dataset of IoT malware traffic[3]. In the SDN-IoT context, the InSDN dataset captures real SDN controller traffic and is frequently used[9]. For MQTT-based environments, datasets like MQTT-IoT-IDS2020 and the MQTTset (introduced above) provide MQTT message flows with attacks[3][15]. In short, the literature reports using combinations of legacy (KDD, NSL), network IDS (UNSW-NB15, CIC), and IoT-specific (BoT-IoT, N-BaIoT, TON_IoT, MQTTset, etc.) datasets to cover the variety of normal and malicious IoT traffic encountered in practice[3][9].

2.3 Research gap

Despite the extensive research conducted on intrusion detection in IoT and SDN-based networks, several limitations remain evident in existing studies. A significant portion of the literature focuses on generic network traffic or traditional IoT protocols, while MQTT-specific intrusion detection is still underexplored, despite its widespread adoption in IoT environments. Additionally, many proposed IDS solutions rely on datasets that do not accurately represent real MQTT communication patterns or overlook the practical constraints of deploying IDS mechanisms within SDN-controlled networks. Furthermore, existing approaches often emphasize detection accuracy without sufficiently addressing the feasibility of real-time deployment, centralized monitoring, and protocol-aware feature extraction. Also, there is a lack of fully integrated, real-time emulation frameworks that combine detection and mitigation in a closed control loop under realistic IoT traffic conditions. These limitations highlight the need for an IDS that combines SDN-based centralized control, machine learning–driven detection, and MQTT-aware traffic analysis, evaluated using realistic datasets. This research gap motivates the proposed project, which aims to address these shortcomings through the design and implementation of a practical and scalable IDS for MQTT-based IoT networks.

2.4 Conclusion

Based on the literature reviewed, it is evident that integrating machine learning-based IDS within SDN-enabled IoT networks offers a promising direction for improving IoT security. However, most existing works focus on offline detection without real-time deployment or mitigation capabilities. Our project addresses this gap by designing and implementing a live SDN-IoT environment using RYU, Mosquitto, and the MQTTset dataset, where the IDS operates in real-time to detect and mitigate attacks at the controller level—bridging the divide between theoretical models and practical, deployable security solutions.

3. System Design and Planning

3.1 Functional and Non-Functional Requirements

We derive the system requirements from the problem definition and IoT/SDN context. Functional requirements (FR) specify what the system must do, while non-functional requirements (NFR) constrain how it should perform. Table 2 summarizes the key requirements. For example, the system must capture and inspect MQTT traffic in real time and flag anomalous or malicious flows (FR). It must automatically alert administrators and reconfigure the switch to isolate or drop detected attacks, while allowing normal MQTT publish/subscribe traffic to continue. Non-functional requirements include real-time performance (e.g. processing latency under ~1 s) and high detection accuracy (to minimize false positives). The system should scale to typical IoT deployments (supporting dozens of hosts) and run on commodity hardware (using Python and open-source tools) for ease of development and deployment. Compliance with standard security practices (e.g. secure MQTT and OpenFlow channels) is also ensured.

Requirement Type	Requirement
Functional	The system must monitor MQTT publish/subscribe traffic in real time.
	The system must detect malicious activity (e.g. spoofed or flood messages) using machine-learning analysis of the traffic.
	The system must generate alerts/logs and instruct the SDN switch to block or isolate flows identified as attacks.
	The system must allow legitimate publisher–subscriber communication to proceed unhindered.
Non-Functional	Detection latency must be very low (on the order of seconds or less) to allow immediate mitigation.
	Accuracy should be high (e.g. ≥95% in distinguishing benign vs. attack traffic) to minimize false alarms.
	The system must operate on standard Linux-based hosts (using Python) and interoperate with MQTT and SDN standards.
	The design should be modular and maintainable (with clear controller applications and logs).

Table 2 – System Requirements

3.2 System Architecture Design

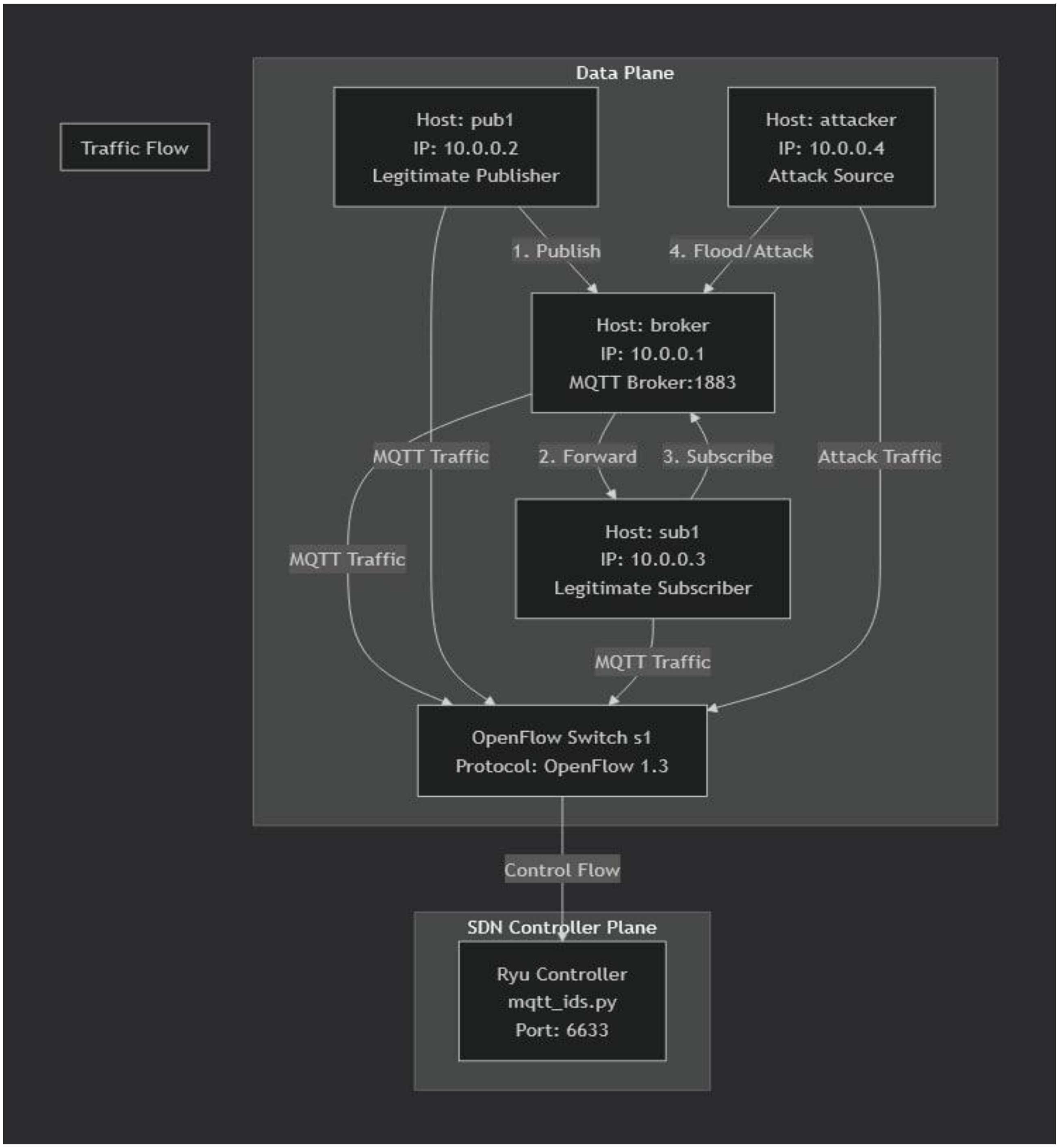

Figure 1 - High Level System Architecture

The architecture (Figure 1) comprises an MQTT publisher, subscriber, attacker host, and a central MQTT broker, all connected via an OpenFlow-enabled switch. The Ryu SDN controller (running our custom mqtt_ids.py app) sits logically above the switch. The publisher and subscriber simulate IoT sensors communicating via the broker; the attacker injects malicious MQTT traffic (e.g. false data or flooding messages). All network packets traverse the OpenFlow switch, which forwards flows based on the controller's rules. The Ryu controller periodically polls the switch for flow statistics and applies an integrated ML-based IDS model. As network traffic passes through, the ML classifier in Ryu evaluates each flow: if the behavior matches known attack patterns, the controller dynamically pushes a new flow rule (or drop rule) to the switch, thereby isolating the malicious traffic. Otherwise, the switch forwards packets normally (e.g. publisher → broker → subscriber). In this way, legitimate MQTT communications remain uninterrupted while attacks are detected and mitigated in real time. The centralized SDN control plane enables this global view and rapid response: the Ryu app serves as "central intelligence," using OpenFlow to monitor switch traffic and reconfigure network behavior based on ML inference.

3.3 Technology Stack Selection

We select open-source tools well-suited for SDN, IoT, and ML development, justifying each choice by capability and community support:

- Python 3 (Programming Language): Python is used for all software components (Ryu apps, data processing, ML). It has extensive libraries for network protocols (e.g. paho-mqtt), SDN (Ryu is Python-based), and machine learning (scikit-learn, TensorFlow). Notably, Ryu itself is implemented in Python, and we use Python to script the Mininet topology and controller logic. This choice simplifies integration and leverages a large developer ecosystem.
- Ryu SDN Controller: Ryu is an open-source, Python-based SDN controller framework that fully implements the OpenFlow protocol. It is chosen for its ease of use and active community. The Ryu framework allows us to install custom applications (mqtt_ids.py) to collect flow statistics and modify switch rules programmatically. Its Python APIs match our developer tools.
- Mininet Network Emulator: Mininet is used to emulate the SDN network on a single Linux host. It quickly instantiates virtual hosts (publisher, subscriber, attacker) and Open vSwitch instances, all connected under the control of Ryu. Mininet is widely adopted in SDN research for its lightweight virtualization of network topologies[4]. We deploy Mininet on an Ubuntu VM to create the IoT topology.
- Mosquitto MQTT Broker: The Mosquitto broker provides the standard MQTT publish/subscribe service. It is lightweight, open-source, and implements the MQTT protocol used by our IoT devices. Publisher and subscriber hosts communicate via Mosquitto to exchange messages, and the attacker also targets this broker. As described in [34], Mosquitto is commonly used for MQTT messaging in SDN-IoT studies
- Machine Learning Libraries: We use Python's scikit-learn library for training and running the IDS models on the MQTTset data. Scikit-learn is a common choice for ML prototyping and was specifically used in prior MQTT security research. It provides mature implementations of classifiers (e.g. Random Forest, SVM) for offline training on the dataset, and these models are then loaded into the Ryu app for live inference. Training is done offline on the MQTTset dataset, which contains both benign and attack traffic for MQTT; this dataset was created exactly for ML-based IDS in IoT.
- Supporting Tools: We employ standard utilities for testing and analysis: Wireshark or tcpdump to capture packets, hping3 to simulate network traffic including flood attacks, and Git for version control. These choices are consistent with SDN/IoT projects and help validate and document system behavior.

3.4 Project Planning and Timeline

We outline a high-level schedule (Gantt-style) for the remaining work. Table 3 breaks the project into phases, estimated durations, and major tasks. The plan allocates time for continued literature review, system design (this phase), implementation of each module, testing of the integrated system, and final documentation. As shown, we expect roughly 12–13 weeks of work, divided as follows:

Phase	Duration	Tasks
Literature Review	3 weeks	Search and survey recent IDS/SDN/IoT literature; refine system requirements.
System Design	2 weeks	Create detailed architecture diagrams; specify requirements (this phase). Select tools/technologies and prepare development environment.
Implementation	3-4 weeks	Develop Ryu controller application (mqtt_ids.py); configure Mininet topology with MQTT hosts; code ML data pipeline and training.
Testing & Evaluation	2 weeks	Generate MQTT traffic (benign and attacks); verify flow monitoring in Ryu; evaluate detection accuracy and latency; tune model.
Documentation	1-2 weeks	Write final report and prepare presentation, compile code documentation and user manual.

Table 3 - Our Work Schedule

References

1. Mishra, S. R., Shanmugam, B., Yeo, K. C., & Thennadil, S. (2025). SDN-Enabled IoT Security Frameworks—A Review of Existing Challenges. *Technologies*, *13*(3), 121.
2. Rahman, M. M., Shakil, S. A., & Mustakim, M. R. (2025). A survey on intrusion detection system in IoT networks. *Cyber Security and Applications, 3,* 100082.
3. Vaccari, I., Chiola, G., Aiello, M., Mongelli, M., & Cambiaso, E. (2020). MQTTset, a New Dataset for Machine Learning Techniques on MQTT. *Sensors*, *20*(22), 6578.
4. Sebestyen, H., Popescu, D. E., & Zmaranda, R. D. (2025). *A literature review on security in the Internet of Things: Identifying and analysing critical categories*. Computers, 14(2), 61.
5. Al Hayajneh, A., Bhuiyan, M. Z. A., & McAndrew, I. (2020). *Improving Internet of Things (IoT) security with software-defined networking (SDN)*.
6. Berhili, M., Chaieb, O., & Benabdellah, M. (2024). *Intrusion detection systems in IoT based on machine learning: A state of the art*. Procedia Computer Science, 251, 99–107.
7. Ataa, M. S., Sanad, E. E., & El-khoribi, R. A. (2024). *Intrusion detection in software defined network using deep learning approaches*. Scientific Reports, 14, 29159.
8. Karmakar, K. K., Varadharajan, V., Nepal, S., & Tupakula, U. (2021). *SDN-enabled secure IoT architecture*. IEEE Internet of Things Journal, 8(8).
9. Dhirar, H., & Hamad, A. (2025). *Comparative evaluation of a novel IDS dataset for SDN-IoT using deep learning models against InSDN, BoT-IoT, and ToN-IoT*. *Measurement: Digitalization, 4*, 100015.
10. Ahmed, Md. Rayhan; Islam, salekul; Shatabda, Swakkhar; Islam, A. K. M. Muzahidul; Robin, Md. Towhidul Islam (2021): *Intrusion Detection System in Software-Defined Networks Using Machine Learning and Deep Learning Techniques –A Comprehensive Survey.* TechRxiv. Preprint.
11. Ferrão, T., Manene, F., & Ajibesin, A. A. (2023). *Multi-Attack Intrusion Detection System for Software-Defined Internet of Things Network*. Computers, Materials & Continua, 75(3), 4985–5007.
12. Owusu, A. I., & Nayak, A. (2020). *An intelligent traffic classification in SDN-IoT: A machine learning approach*. In 2020 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom). IEEE.
13. Bedhief, I., Kassar, M., & Aguili, T. (2023). *Empowering SDN-Docker based architecture for Internet of Things heterogeneity*. Journal of Network and Systems Management, 31, 14.
14. Ali, M. A., & Al-Sharafi, S. A. H. (2025). *Intrusion detection in IoT networks using machine learning and deep learning approaches for MitM attack mitigation*. Discover Internet of Things, 5, 48.
15. Khan, M. A., Khan, M. A., Jan, S. U., Ahmad, J., Jamal, S. S., Shah, A. A., Pitropakis, N., & Buchanan, W. J. (2021). *A deep learning-based intrusion detection system for MQTT enabled IoT*. Sensors, 21(21), 7016.
16. Logeswari G., Bose S., Anitha T. (2023). *An Intrusion Detection System for SDN Using Machine Learning*. *Intelligent Automation & Soft Computing*, 35(1), 867–880.
17. Chaganti R., Suliman W., Ravi V., Dua A. (2023). *Deep Learning Approach for SDN-Enabled Intrusion Detection System in IoT Networks*. *Information*, 14(1):41.
18. Cherian M.M., Varma S.L. (2022). *Mitigation of DDOS and MiTM Attacks using Belief Based Secure Correlation in SDN-IoT Networks*. *IJCNIS*, 14(1), 52–68.