# SDN-BASED INTELLIGENT INTRUSION DETECTION SYSTEM (IIDS) USING MACHINE LEARNING

Group ID: RP-24-25J-120


Research Logbook

Parthika. K

IT20601638

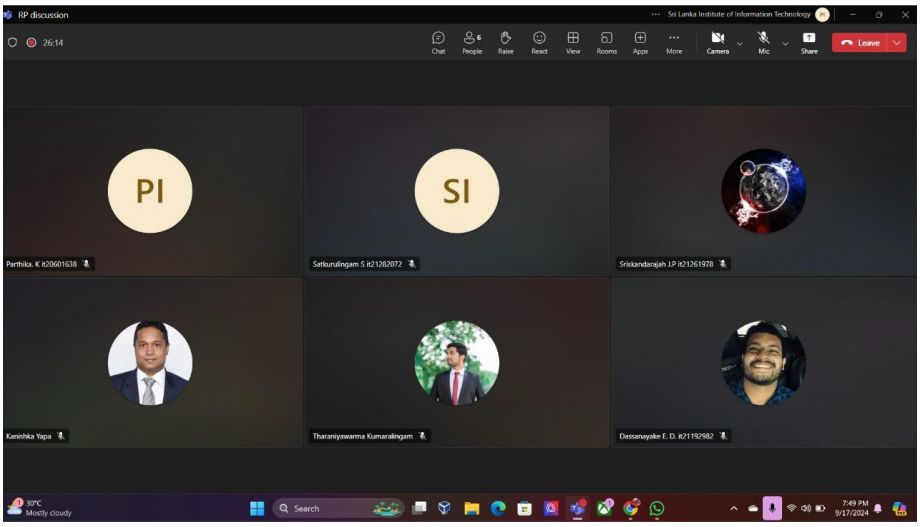
BSc (Hons) Degree in Information Technology Specialized in Cyber Security

Sri Lanka Institute of Information Technology Sri Lanka

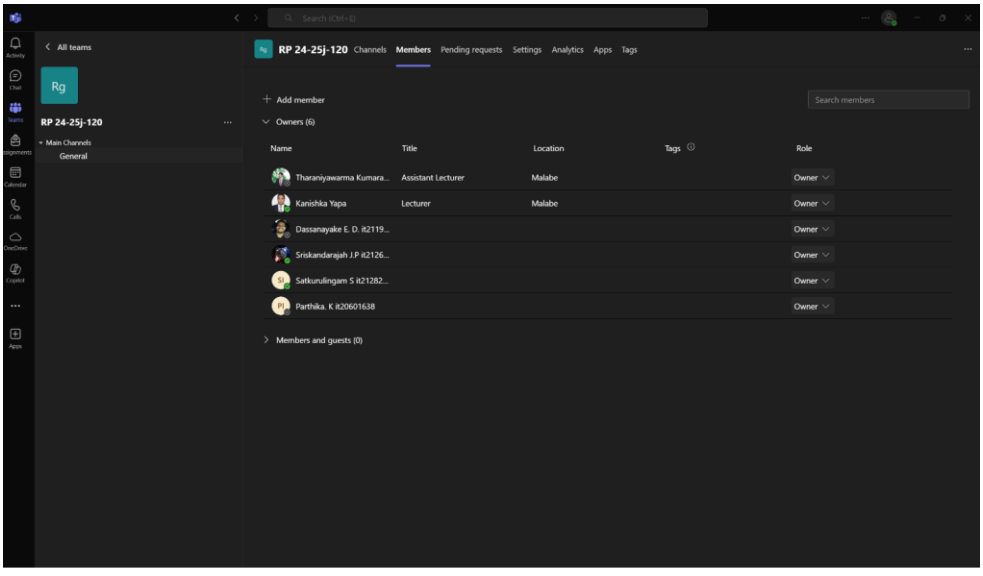
June 2025

## Tasks

- **Meeting with the supervisor to discuss the project topic for the first time.**
  - Physically meet the supervisor.
  - Discuss the research project topic area.
  - Get the supervisor's ideas about the research topic via Microsoft Teams.

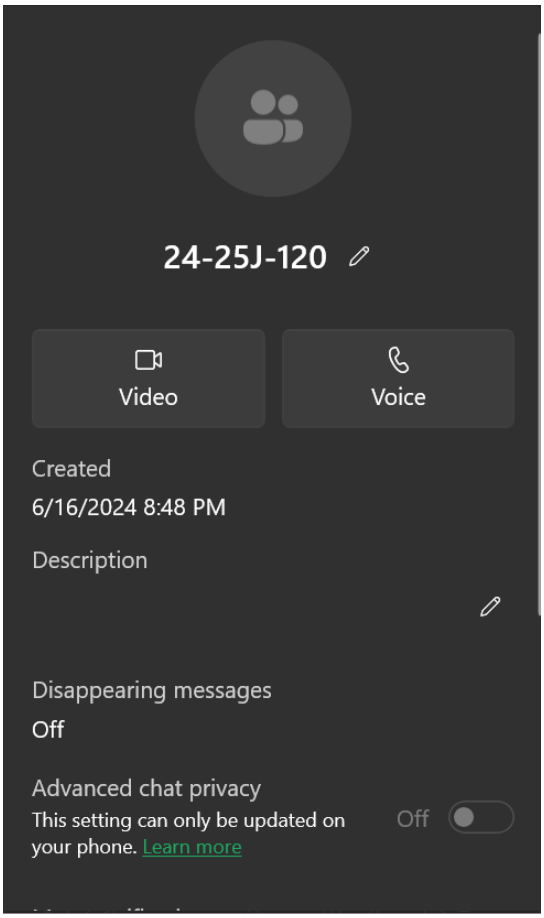


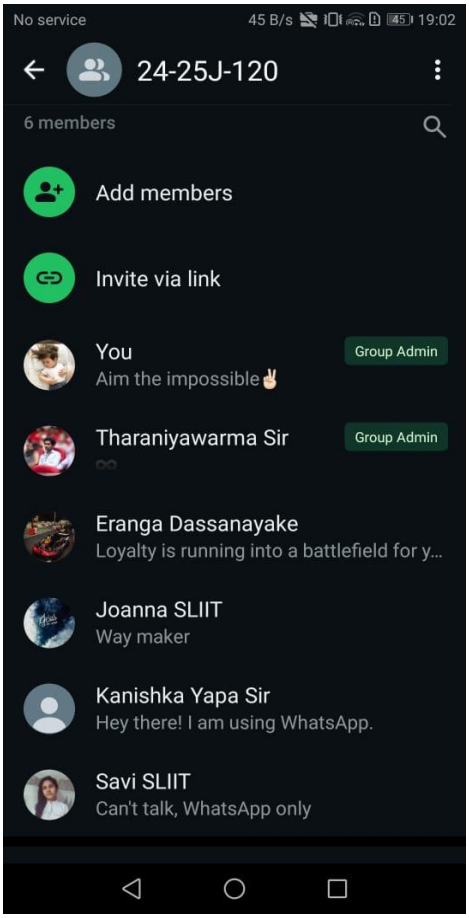

- **Create separate MS Teams channels for Conversations.**



## Task

- **Created the Research Team WhatsApp Group**
  - Discuss the research topic with team members.
  - Discuss the research problem.
  - Get the solution ideas with brainstorming sessions.
  - Identify the main solutions.
  - Assign tasks and conversation highlights.



► **Completed Task and Conversation highlights**.

- Creating the proposal document at supervisor request.
- Doing a literature review upon supervisor request.

The development of the SDN-based Intelligent Intrusion Detection System (IIDS) requires specialized expertise and knowledge in several domains: Software-Defined Networking (SDN): Understanding the principles, architecture, and operation of SDN, including SDN controllers, protocols, and network virtualization. Cybersecurity: In-depth knowledge of cybersecurity principles, threat landscapes, attack vectors, and defense mechanisms, particularly in network security. Machine Learning: Proficiency in developing and applying machine learning models for anomaly detection and pattern recognition. Network Traffic Analysis: Expertise in capturing, preprocessing, and analyzing network traffic data to identify features relevant to intrusion detection. System Integration and Development: Skills in integrating various modules and components, ensuring seamless communication and compatibility between the SDN controller, intrusion detection engine, and policy enforcement module. Performance Optimization: Ability to optimize algorithms and systems for real-time processing, high performance, and scalability.

Data Requirements- Network Traffic Data: Extensive datasets of network traffic, both normal and malicious, for training and testing machine learning models. Cybersecurity Threat Intelligence: Access to threat intelligence feeds and databases to update and refine detection models and security policies. System Logs and Event Data: Logs from network devices, servers, and security appliances to provide comprehensive visibility into network activities and potential threats. Simulated Attack Scenarios: Synthetic data representing different types of cyberattacks for testing the system's detection and response capabilities.

The development of the SDN-based Intelligent Intrusion Detection System (IIDS) requires specialized expertise and knowledge in several domains: Software-Defined Networking (SDN): Understanding the principles, architecture, and operation of SDN, including SDN controllers, protocols, and network virtualization. Cybersecurity: In-depth knowledge of cybersecurity principles, threat landscapes, attack vectors, and defense mechanisms, particularly in network security. Machine Learning: Proficiency in developing and applying machine learning models for anomaly detection and pattern recognition. Network Traffic Analysis: Expertise in capturing, preprocessing, and analyzing network traffic data to identify features relevant to intrusion detection. System Integration and Development: Skills in integrating various modules and components, ensuring seamless communication and compatibility between the SDN controller, intrusion detection engine, and policy enforcement module. Performance Optimization: Ability to optimize algorithms and systems for real-time processing, high performance, and scalability.

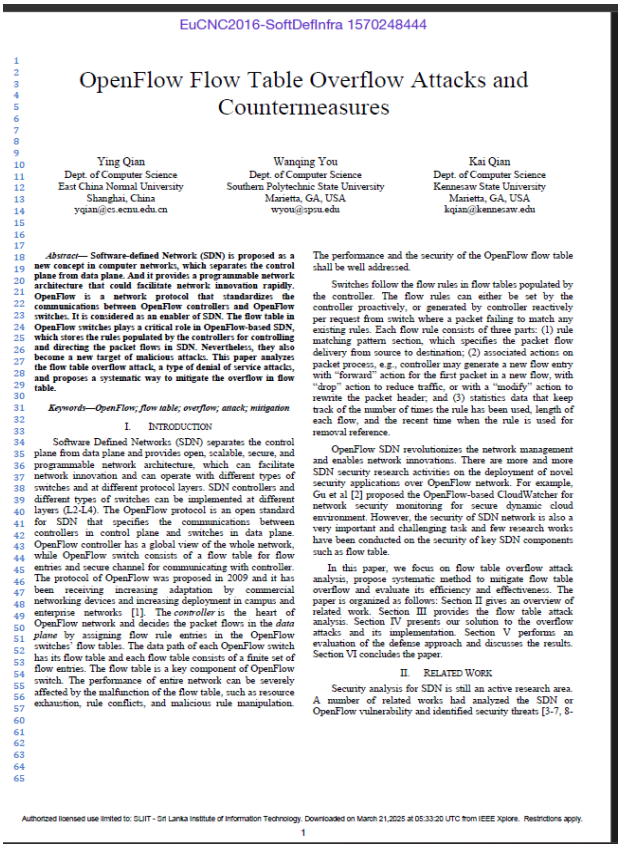
Data Requirements- Network Traffic Data: Extensive datasets of network traffic, both normal and malicious, for training and testing machine learning models. Cybersecurity Threat Intelligence: Access to threat intelligence feeds and databases to update and refine detection models and security policies. System Logs and Event Data: Logs from network devices, servers, and security appliances to provide comprehensive visibility into network activities and potential threats. Simulated Attack Scenarios: Synthetic data representing different types of cyberattacks for testing the system's detection and response capabilities.

## ► Proposed Machine Learning based Intrusion Detection Engine System

| Member Name | Sub Objective | Tasks | Novelty |
|---|---|---|---|
| Parthika.K | Develop a Machine Learning Based Intrusion Detection System to Find Flow table overflow attack | 1. **Data collection and preprocessing** Gather and prepare data to effectively train the machine learning model for detecting Flow Table overflow attacks.<br>2. **Develop machine learning model** Create and train a machine learning model to detect Flow table overflow attacks.<br>3. **Integration with SDN controller** Implement the trained model within the SDN controller to monitor and detect Flow table overflow attacks in real- time.<br>4. **Testing and optimization** The IDS's performance will be tested with various attack scenarios and traffic patterns to ensure accurate Flow table overflow detection. The system will be optimized, refined, and adjusted for improved efficiency and reliability. | **Adaptive Threat Response Mechanism**<br><br>This project introduces an innovative feature where the IDS doesn't just detect Flow table overflow attacks but also adapts its security measures in real-time. By analyzing ongoing network conditions and threat intelligence, the system can automatically adjust its defenses, making it more resilient to new and evolving threats. This dynamic and proactive approach to security is a key advancement over traditional IDS systems, which are often static and less responsive to changes in the threat landscape. |

► **Completed Task and Conversation Highlights**

- Determining the components for each member and discussing with the Supervisor.
- Fine tuning the scope for each component.
- Discussing the proposed components with co-supervisor.
- Find the Related research paper for individual SDN Component.
- Get a full idea of each research paper.
- Mark down the not covering SDN areas in these research papers.
- Identify the novelty parts of each individual component.
- Creating the Topic Assignment Form (TAF)
- Getting the approval from the Supervisor.



EuCNC2016-SoftDefInfra 1570248444

OpenFlow Flow Table Overflow Attacks and Countermeasures

Ying Qian
Dept. of Computer Science
East China Normal University
Shanghai, China
yqian@cs.ecnu.edu.cn

Wanqing You
Dept. of Computer Science
Southern Polytechnic State University
Marietta, GA, USA
wyou@spsu.edu

Kai Qian
Dept. of Computer Science
Kennesaw State University
Marietta, GA, USA
kqian@kennesaw.edu

*Abstract*— **Software-defined Network (SDN) is proposed as a new concept in computer networks, which separates the control plane from data plane. And it provides a programmable network architecture that could facilitate network innovation rapidly. OpenFlow is a network protocol that standardizes the communications between OpenFlow controllers and OpenFlow switches. It is considered as an enabler of SDN. The flow table in OpenFlow switches plays a critical role in OpenFlow-based SDN, which stores the rules populated by the controllers for controlling and directing the packet flows in SDN. Nevertheless, they also become a new target of malicious attacks. This paper analyzes the flow table overflow attack, a type of denial of service attacks, and proposes a systematic way to mitigate the overflow in flow table.**

***Keywords—OpenFlow; flow table; overflow; attack; mitigation***

## I. Introduction

Software Defined Networks (SDN) separates the control plane from data plane and provides open, scalable, secure, and programmable network architecture, which can facilitate network innovation and can operate with different types of switches and at different protocol layers. SDN controllers and different types of switches can be implemented at different layers (L2-L4). The OpenFlow protocol is an open standard for SDN that specifies the communications between controllers in control plane and switches in data plane. OpenFlow controller has a global view of the whole network, while OpenFlow switch consists of a flow table for flow entries and secure channel for communicating with controller. The protocol of OpenFlow was proposed in 2009 and it has been receiving increasing adaptation by commercial networking devices and increasing deployment in campus and enterprise networks [1]. The *controller* is the heart of OpenFlow network and decides the packet flows in the *data plane* by assigning flow rule entries in the OpenFlow switches' flow tables. The data path of each OpenFlow switch has its flow table and each flow table consists of a finite set of flow entries. The flow table is a key component of OpenFlow switch. The performance of entire network can be severely affected by the malfunction of the flow table, such as resource exhaustion, rule conflicts, and malicious rule manipulation. The performance and the security of the OpenFlow flow table shall be well addressed.

Switches follow the flow rules in flow tables populated by the controller. The flow rules can either be set by the controller proactively, or generated by controller reactively per request from switch where a packet failing to match any existing rules. Each flow rule consists of three parts: (1) rule matching pattern section, which specifies the packet flow delivery from source to destination; (2) associated actions on packet process, e.g., controller may generate a new flow entry with "forward" action for the first packet in a new flow, with "drop" action to reduce traffic, or with a "modify" action to rewrite the packet header; and (3) statistics data that keep track of the number of times the rule has been used, length of each flow, and the recent time when the rule is used for removal reference.

OpenFlow SDN revolutionizes the network management and enables network innovations. There are more and more SDN security research activities on the deployment of novel security applications over OpenFlow network. For example, Gu et al. [2] proposed the OpenFlow-based CloudWatcher for network security monitoring for secure dynamic cloud environment. However, the security of SDN network is also a very important and challenging task and few research works have been conducted on the security of key SDN components such as flow table.

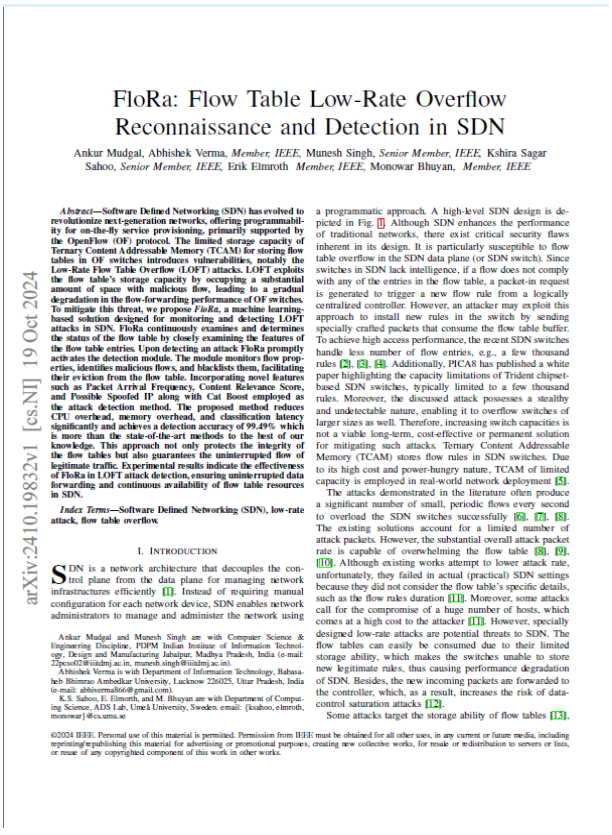
In this paper, we focus on flow table overflow attack analysis, propose systematic method to mitigate flow table overflow and evaluate its efficiency and effectiveness. The paper is organized as follows: Section II gives an overview of related work. Section III provides the flow table attack analysis. Section IV presents our solution to the overflow attacks and its implementation. Section V performs an evaluation of the defense approach and discusses the results. Section VI concludes the paper.

## II. Related Work

Security analysis for SDN is still an active research area. A number of related works had analyzed the SDN or OpenFlow vulnerability and identified security threats [3-7, 8-









FloRa: Flow Table Low-Rate Overflow Reconnaissance and Detection in SDN

Ankur Mudgal, Abhishek Verma, *Member, IEEE,* Munesh Singh, *Senior Member, IEEE,* Kshira Sagar Sahoo, *Senior Member, IEEE,* Erik Elmroth *Member, IEEE,* Monowar Bhuyan, *Member, IEEE*

***Abstract*—Software Defined Networking (SDN) has evolved to revolutionize next-generation networks, offering programmability for on-the-fly service provisioning, primarily supported by the OpenFlow (OF) protocol. The limited storage capacity of Ternary Content Addressable Memory (TCAM) for storing flow tables in OF switches introduces vulnerabilities, notably the Low-Rate Flow Table Overflow (LOFT) attacks. LOFT exploits the flow table's storage capacity by occupying a substantial amount of space with malicious flow, leading to a gradual degradation in the flow-forwarding performance of OF switches. To mitigate this threat, we propose *FloRa*, a machine learning-based solution designed for monitoring and detecting LOFT attacks in SDN. FloRa continuously examines and determines the status of the flow table by closely examining the features of the flow table entries. Upon detecting an attack FloRa promptly activates the detection module. The module monitors flow properties, identifies malicious flows, and blacklists them, facilitating their eviction from the flow table. Incorporating novel features such as Packet Arrival Frequency, Content Relevance Score, and Possible Spoofed IP along with Cat Boost employed as the attack detection method. The proposed method reduces CPU overhead, memory overhead, and classification latency significantly and achieves a detection accuracy of 99.49% which is more than the state-of-the-art methods to the best of our knowledge. This approach not only protects the integrity of the flow tables but also guarantees the uninterrupted flow of legitimate traffic. Experimental results indicate the effectiveness of FloRa in LOFT attack detection, ensuring uninterrupted data forwarding and continuous availability of flow table resources in SDN.**

***Index Terms*—Software Defined Networking (SDN), low-rate attack, flow table overflow.**

## I. Introduction

SDN is a network architecture that decouples the control plane from the data plane for managing network infrastructures efficiently [1]. Instead of requiring manual configuration for each network device, SDN enables network administrators to manage and administer the network using a programmatic approach. A high-level SDN design is depicted in Fig. 1. Although SDN enhances the performance of traditional networks, there exist critical security flaws inherent in its design. It is particularly susceptible to flow table overflow in the SDN data plane (or SDN switch). Since switches in SDN lack intelligence, if a flow does not comply with any of the entries in the flow table, a packet-in request is generated to trigger a new flow rule from a logically centralized controller. However, an attacker may exploit this approach to install new rules in the switch by sending specially crafted packets that consume the flow table buffer. To achieve high access performance, the recent SDN switches handle less number of flow entries, e.g., a few thousand rules [2], [3], [4]. Additionally, PICA8 has published a white paper highlighting the capacity limitations of Trident chipset-based SDN switches, typically limited to a few thousand rules. Moreover, the discussed attack possesses a stealthy and undetectable nature, enabling it to overflow switches of larger sizes as well. Therefore, increasing switch capacities is not a viable long-term, cost-effective or permanent solution for mitigating such attacks. Ternary Content Addressable Memory (TCAM) stores flow rules in SDN switches. Due to its high cost and power-hungry nature, TCAM of limited capacity is employed in real-world network deployment [5].

The attacks demonstrated in the literature often produce a significant number of small, periodic flows every second to overload the SDN switches successfully [6], [7], [8]. The existing solutions account for a limited number of attack packets. However, the substantial overall attack packet rate is capable of overwhelming the flow table [8], [9], [10]. Although existing works attempt to lower attack rate, unfortunately, they failed in actual (practical) SDN settings because they did not consider the flow table's specific details, such as the flow rules duration [11]. Moreover, some attacks call for the compromise of a huge number of hosts, which comes at a high cost to the attacker [11]. However, specially designed low-rate attacks are potential threats to SDN. The flow tables can easily be consumed due to their limited storage ability, which makes the switches unable to store new legitimate rules, thus causing performance degradation of SDN. Besides, the new incoming packets are forwarded to the controller, which, as a result, increases the risk of data-control saturation attacks [12].

Some attacks target the storage ability of flow tables [13],

Ankur Mudgal and Munesh Singh are with Computer Science & Engineering Discipline, PDPM Indian Institute of Information Technology, Design and Manufacturing Jabalpur, Madhya Pradesh, India (e-mail: 22pcse02@iiitdmj.ac.in, munesh.singh@iiitdmj.ac.in).

Abhishek Verma is with Department of Information Technology, Babasaheb Bhimrao Ambedkar University, Lucknow 226025, Uttar Pradesh, India (e-mail: abhivermal66@gmail.com).

K.S. Sahoo, E. Elmroth, and M. Bhuyan are with Department of Computing Science, ADS Lab, Umeå University, Sweden. email: {ksahoo, elmroth, monowar}@cs.umu.se

# Software-defined Networking: Challenges and Research Opportunities for Future Internet

Akram Hakiri[a,b], Aniruddha Gokhale[c], Pascal Berthou[a,b], Douglas C. Schmidt[c], Gayraud Thierry[a,b]

[a] CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France
[b] Univ de Toulouse, UPS, LAAS, F-31400 Toulouse, France
[c] Institute for Software Integrated Systems, Dept of EECS
Vanderbilt University, Nashville, TN 37212, USA

**Abstract**

Currently many aspects of the classical architecture of the Internet are etched in stone – a so called ossification of the Internet – which has led to major obstacles in IPv6 deployment and difficulty in using IP multicast services. Yet, there exist many reasons to extend the Internet, e.g., for improving intra-domain and inter-domain routing for high availability of the network, providing end-to-end connectivity for users, and allowing dynamic QoS management of network resources for new applications, such as data center, cloud computing, and network virtualization. To address these requirements, the next-generation architecture for the Future Internet has introduced the concept of Software-Defined Networking (SDN). At the core of this emerging paradigm is the separation and centralization of the control plane from the forwarding elements in the network as opposed to the distributed control plane of existing networks. This decoupling allows deployment of control plane software components (e.g., OpenFlow controller) on computer platforms that are much more powerful than traditional network equipment (e.g., switches/routers) while protecting the data and intellectual property of the vendors of such equipment.

A critical understanding of this emerging paradigm is necessary to address the multiple challenges in realizing the Future Internet and to resolve the ossification problem of the existing Internet. To address these requirements, this paper surveys existing technologies and the wide range of recent and state-of-the-art projects on SDN followed by an in-depth discussion of the major challenges in this area.

*Keywords:* Future Internet, Software-Defined Networking (SDN), OpenFlow, Network Function Virtualization, Forwarding Plane, Control Plane.

## 1. Introduction

### *1.1. The Need for a New Network Architecture.*

The capacity of the current Internet is rapidly becoming insufficient to cater to the large volumes of traffic patterns delivered by the new services and modalities (e.g., mobile devices and content, server virtualization, cloud services, big data), which is generated due to a large number of users, sensors and applications [1, 2].

Existing networks built with multiple tiers of static Ethernet switches arranged in a tree structure are ill-suited for the dynamic computing and storage needs of today's and future enterprise hyper-scale data centers, campuses, and carrier environments. Instead, new networking infrastructures are desired that will provide high performance, energy efficiency, and reliability. Moreover, they should improve the network speedup, scalability and robustness

*Email addresses:* hakiri@laas.fr (Akram Hakiri), a.gokhale@vanderbilt.edu (Aniruddha Gokhale), berthou@laas.fr (Pascal Berthou), d.schmidt@vanderbilt.edu (Douglas C. Schmidt), gayraud@laas.fr (Gayraud Thierry)



Expert Systems With Applications 237 (2024) 121400

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

# FTODefender: An efficient flow table overflow attacks defending system in SDN

Dan Tang [a], Zhiqing Zheng [a], Chao Yin [b], Bing Xiong [c], Zheng Qin [a], Qiuwei Yang [a,*]

[a] Hunan University, China
[b] School of Information Science and Technology from Jiujiang University, China
[c] Changsha University of Science and Technology, China

ARTICLE INFO

*Keywords:*
Criteria importance through intercriteria correlation
Low-rate flow table overflow attack
Light gradient boosting machine
Logistic regression
Software-defined networking

ABSTRACT

Software-Defined Networking (SDN) is a promising architecture that disentangles the control plane from the data plane. A mainstream southbound protocol for controller-to-switch communication in SDN is OpenFlow. In an OpenFlow-enabled network, SDN switches are connected to the logically centralized control plane, which issues control messages to direct packet forwarding and processing in the data plane. Given that the finite capacity of physical flow tables of switches, typically implemented with ternary content addressable memory (TCAM), attackers have the ability to carry out Denial-of-Service (DoS) attacks aimed at depleting the TCAM's resources and causing an overflow of the flow table. In this paper, we propose FTODefender, a method combining eviction and cut-off attack sources to detect Low-rate Flow Table Overflow (LFTO) attacks and mitigate this kind of attacks in time. FTODefender consists of two modules: *Detector* and *Mitigator*. *Detector* periodically examines the flow table and extract four specific detection features. It then employs the trained CRITIC weights to calculate the detection score based on these extracted features to verify if the LFTO attack has happened. *Mitigator* computes the features for each flow and utilizes the LightGBM-LR classification model to identify attack flows among all flows. Then, the module evicts the malicious rules to mitigate attacks. Finally, *Mitigator* counts the occurrence of each source IP address in the eviction list, determines the attacker IP based on a predefined threshold, and issues a flow rule to drop all packets originating from the attacker's IPs to sever the attack sources. Simulations show the effectiveness of FTODefender in mitigating the LFTO attacks, which proves that FTODefender is a practical solution to flow table overflow attacks.

## 1. Introduction

Software-defined networking (SDN) revolutionizes the network paradigm by introducing centralized policy management, direct programmability, programmable network traffic control, cross-platform and vendor neutrality through decoupling the control plane (CP) and the data plane (DP) (Xia, Wen, Foh, Niyato, & Xie, 2014). Owing to its scalability and flexibility (Polat, Türkoğlu, Polat, & Şengür, 2022), SDN has been widely adopted in data center networks (Abdelrahman et al., 2021). Different from the classical network framework, the DP devices of SDN only adhere to the orders from the CP to forward the data packets, leveraging routing and other complex network functions to the CP (Scaranti, Carvalho, Junior, Lloret, & Proença, 2022). In order to 2008). OpenFlow has emerged as the leading and widely recognized southbound interface that establishes the structure flow table to forward and process data packets. It provides users with direct access to network devices on the DP, enabling them to manipulate and control these devices directly (Tang, Zheng, Wang, Xiao and Yang, 2022).

While SDN has brought significant advantages through its innovative architecture, it is important to be aware of potential security risks associated with OpenFlow, which is a key component of SDN (Li, Meng, & Kwok, 2016). OpenFlow relies on flow tables as a pivotal component for storing crucial information for network configurations, making SDN switches a prime target for potential attacks. Additionally, since the network heavily relies on switches to receive and transmit control messages and implement flow rules based on commands from

SLIIT UNI
THE KNOWLEDGE UNIVERSITY

**IT4010 – Research Project - 2024**

**Topic Assessment Form**

**Project ID :** 24-25J-120

1. Topic (12 words max)

SDN-based Intelligent Intrusion Detection System (IIDS) using Machine Learning

2. Research group the project belongs to

**Computing Infrastructure and Security (CIS)**

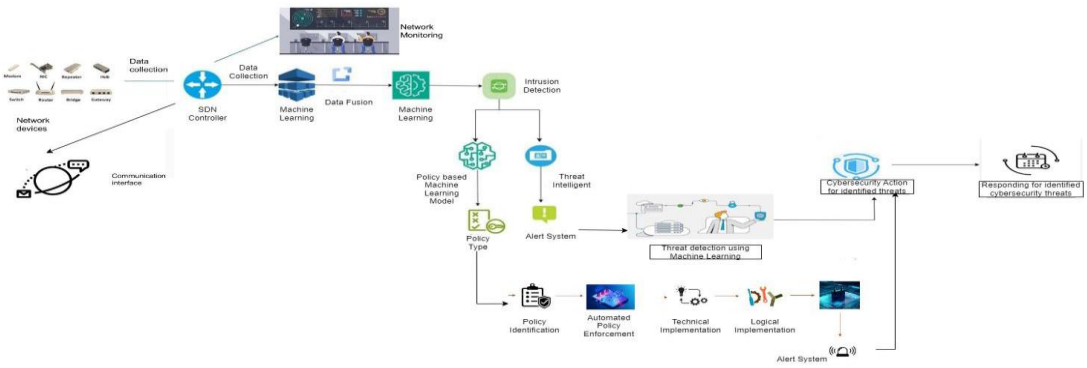
3. Research area the project belongs to

**Cyber Security (CS)**

4. If a continuation of a previous project:

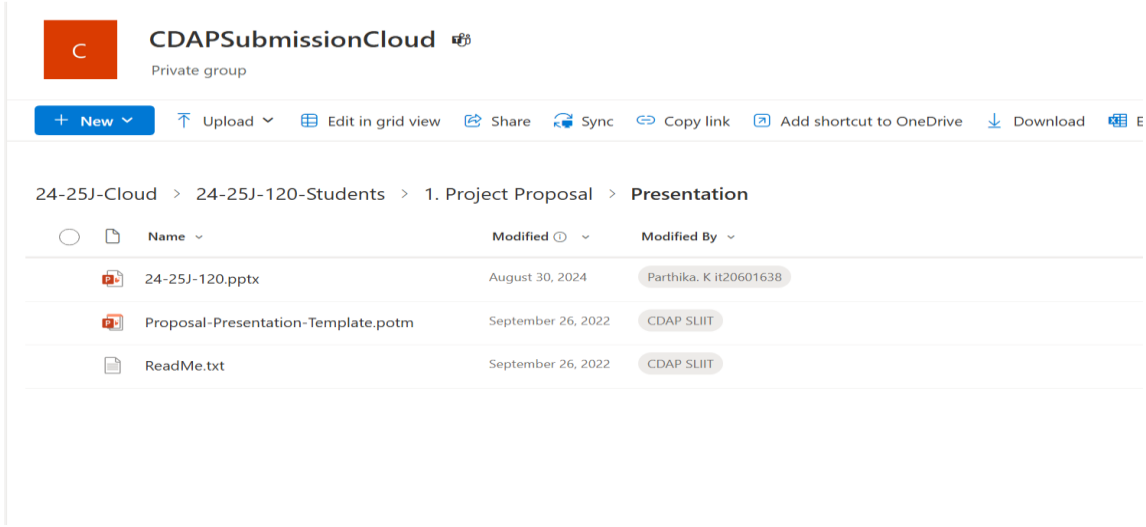
► **Complete Task and Conversation Highlights**

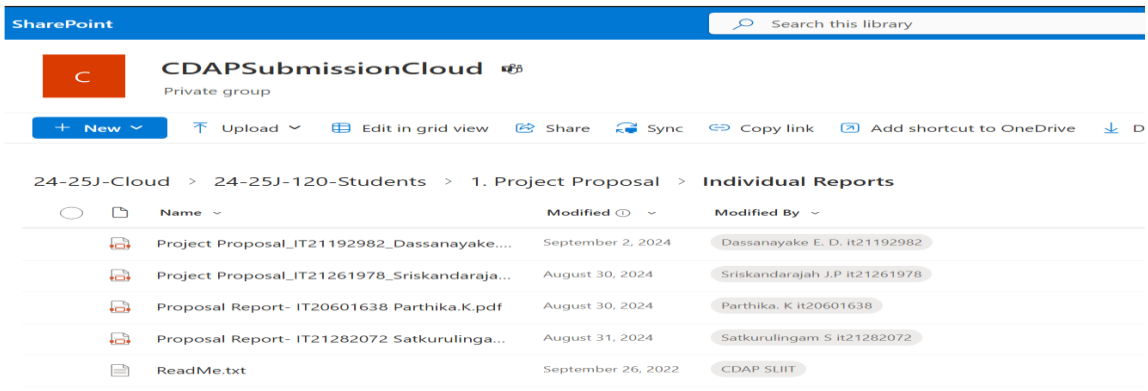
- Dividing the software components.
- Doing a thorough background investigation on each component.
- Creating the system architecture diagram of the proposed system.
- Discussing architecture with the Supervisor and Co-supervisor physically meeting.

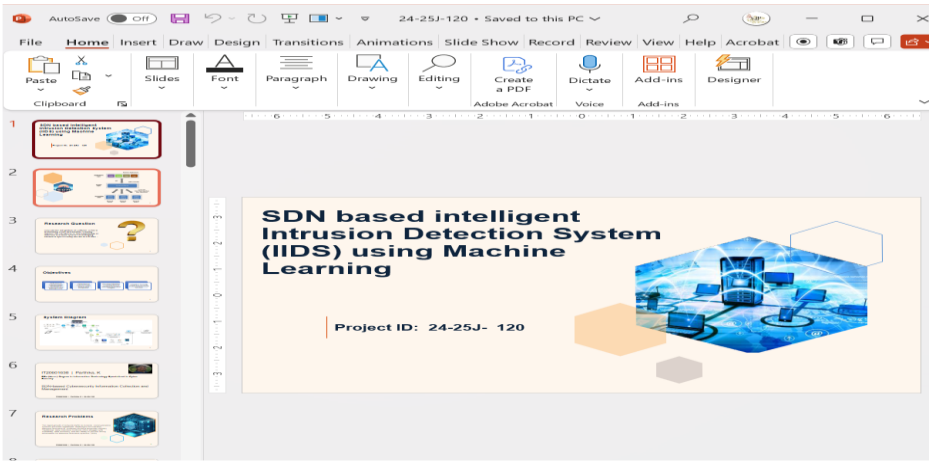


► **Complete Task and Conversation Highlights.**

- Finalizing the components and getting ready for the progress presentation.
- Discussion the project with the supervisor before the proposal presentation.

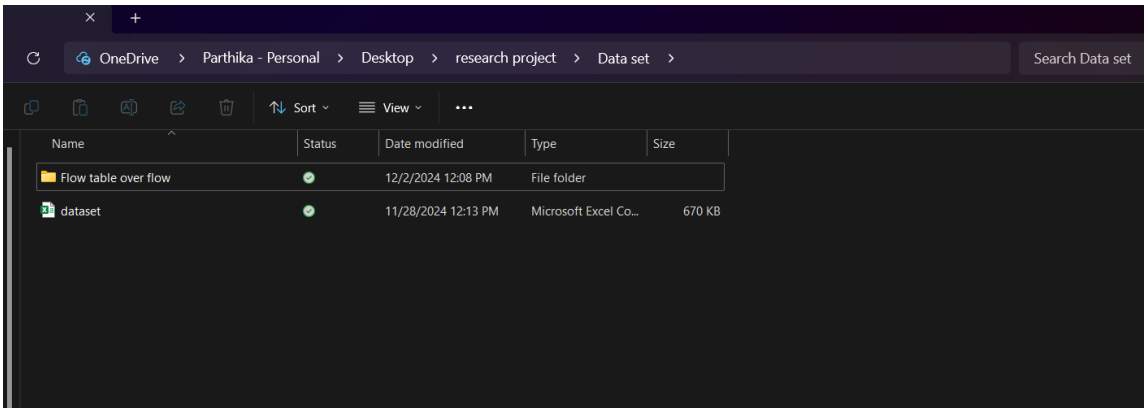


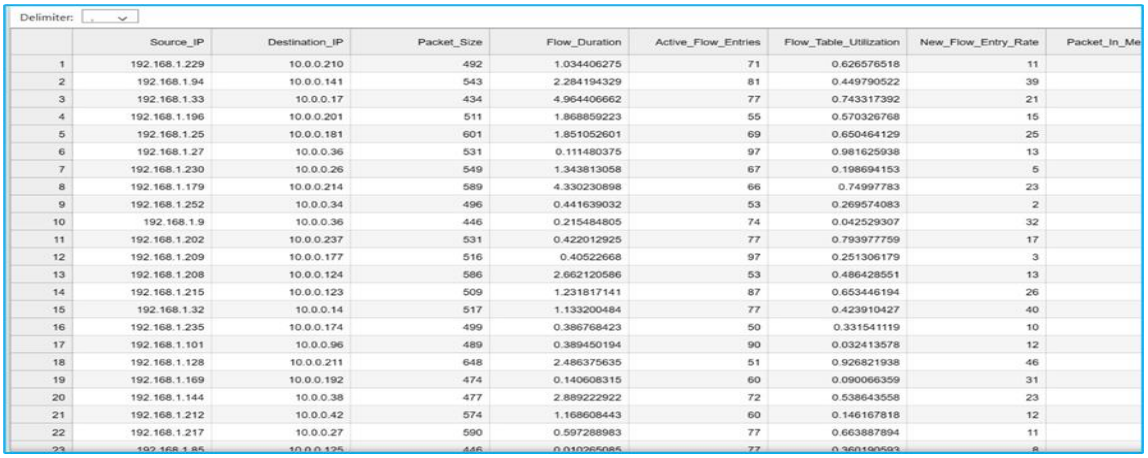



## ► Completed Task and Conversation Highlights

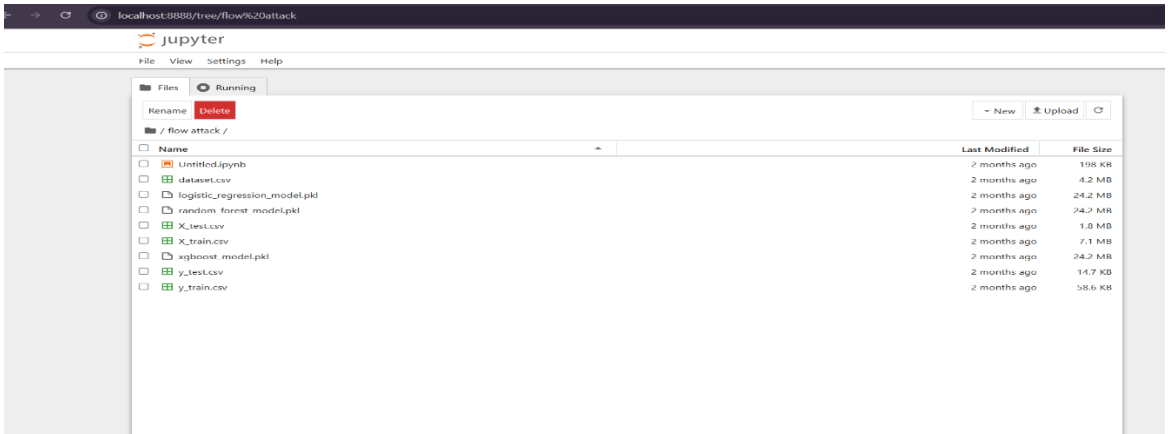
- Finding the sample dataset until SDN system develop.
- Discussing with the co-supervisor the potential model and its accuracy and which model we should proceed with for the prediction.



Delimiter: ,

| | Source_IP | Destination_IP | Packet_Size | Flow_Duration | Active_Flow_Entries | Flow_Table_Utilization | New_Flow_Entry_Rate | Packet_In_Me |
|---|---|---|---|---|---|---|---|---|
| 1 | 192.168.1.229 | 10.0.0.210 | 492 | 1.034406275 | 71 | 0.626576518 | 11 | |
| 2 | 192.168.1.94 | 10.0.0.141 | 543 | 2.284194329 | 81 | 0.449790522 | 39 | |
| 3 | 192.168.1.33 | 10.0.0.17 | 434 | 4.964406662 | 77 | 0.743317392 | 21 | |
| 4 | 192.168.1.196 | 10.0.0.201 | 511 | 1.868859223 | 55 | 0.570326768 | 15 | |
| 5 | 192.168.1.25 | 10.0.0.181 | 601 | 1.851052601 | 69 | 0.650464129 | 25 | |
| 6 | 192.168.1.27 | 10.0.0.36 | 531 | 0.111480375 | 97 | 0.981625938 | 13 | |
| 7 | 192.168.1.230 | 10.0.0.26 | 549 | 1.343813058 | 67 | 0.198694153 | 5 | |
| 8 | 192.168.1.179 | 10.0.0.214 | 589 | 4.330230898 | 66 | 0.74997783 | 23 | |
| 9 | 192.168.1.252 | 10.0.0.34 | 496 | 0.441639032 | 53 | 0.269574083 | 2 | |
| 10 | 192.168.1.9 | 10.0.0.36 | 446 | 0.215484805 | 74 | 0.042529307 | 32 | |
| 11 | 192.168.1.202 | 10.0.0.237 | 531 | 0.422012925 | 77 | 0.793977759 | 17 | |
| 12 | 192.168.1.209 | 10.0.0.177 | 516 | 0.40522668 | 97 | 0.251306179 | 3 | |
| 13 | 192.168.1.208 | 10.0.0.124 | 586 | 2.662120586 | 53 | 0.486428551 | 13 | |
| 14 | 192.168.1.215 | 10.0.0.123 | 509 | 1.231817141 | 87 | 0.653446194 | 26 | |
| 15 | 192.168.1.32 | 10.0.0.14 | 517 | 1.133200484 | 77 | 0.423910427 | 40 | |
| 16 | 192.168.1.235 | 10.0.0.174 | 499 | 0.386768423 | 50 | 0.331541119 | 10 | |
| 17 | 192.168.1.101 | 10.0.0.96 | 489 | 0.389450194 | 90 | 0.032413578 | 12 | |
| 18 | 192.168.1.128 | 10.0.0.211 | 648 | 2.486375635 | 51 | 0.926821938 | 46 | |
| 19 | 192.168.1.169 | 10.0.0.192 | 474 | 0.140608315 | 60 | 0.090066359 | 31 | |
| 20 | 192.168.1.144 | 10.0.0.38 | 477 | 2.889222922 | 72 | 0.538643558 | 23 | |
| 21 | 192.168.1.212 | 10.0.0.42 | 574 | 1.168608443 | 60 | 0.146167818 | 12 | |
| 22 | 192.168.1.217 | 10.0.0.27 | 590 | 0.597288983 | 77 | 0.663887894 | 11 | |
| 23 | 192.168.1.85 | 10.0.0.125 | 446 | 0.010265085 | 77 | 0.360190593 | 8 | |



```python
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report

# Train Logistic Regression model
log_reg = LogisticRegression(max_iter=1000, random_state=42)
log_reg.fit(X_train, y_train)

# Make predictions
y_pred_log_reg = log_reg.predict(X_test)

# Evaluate model
accuracy_log_reg = accuracy_score(y_test, y_pred_log_reg)
print(f"Logistic Regression Accuracy: {accuracy_log_reg:.4f}")
print("\nClassification Report:\n", classification_report(y_test, y_pred_log_reg))
```

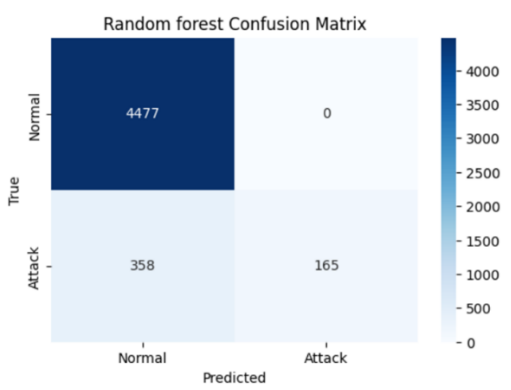
```
Logistic Regression Accuracy: 0.8954
```

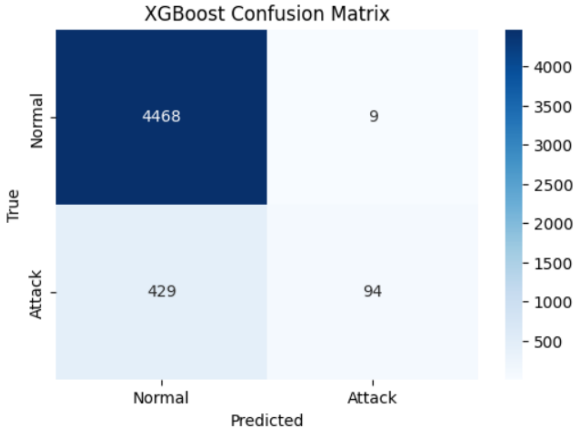
```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report

# Initialize the model
model = RandomForestClassifier(n_estimators=100, random_state=42)

# Train the model
model.fit(X_train, y_train.values.ravel())  # Use .ravel() to avoid shape issues

# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Model Accuracy: {accuracy:.4f}")
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

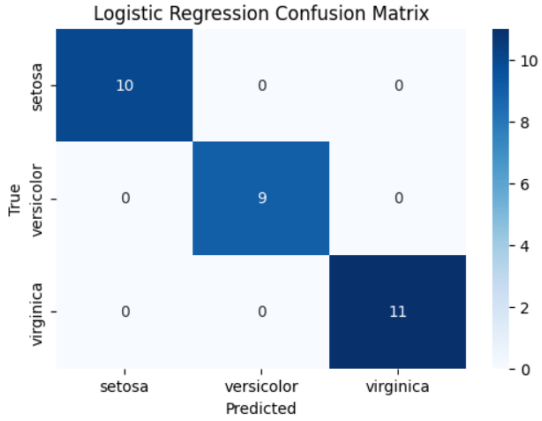
```
Model Accuracy: 0.9284
```

```python
from xgboost import XGBClassifier

# Initialize and train the XGBoost model
xgb_model = XGBClassifier(eval_metric="logloss", random_state=42)
xgb_model.fit(X_train, y_train)

# Make predictions
y_pred_xgb = xgb_model.predict(X_test)

# Evaluate the model
accuracy_xgb = accuracy_score(y_test, y_pred_xgb)
print(f"Model Accuracy with XGBoost: {accuracy_xgb:.4f}")
print("\nClassification Report:\n", classification_report(y_test, y_pred_xgb))
```

```
Model Accuracy with XGBoost: 0.9124
```

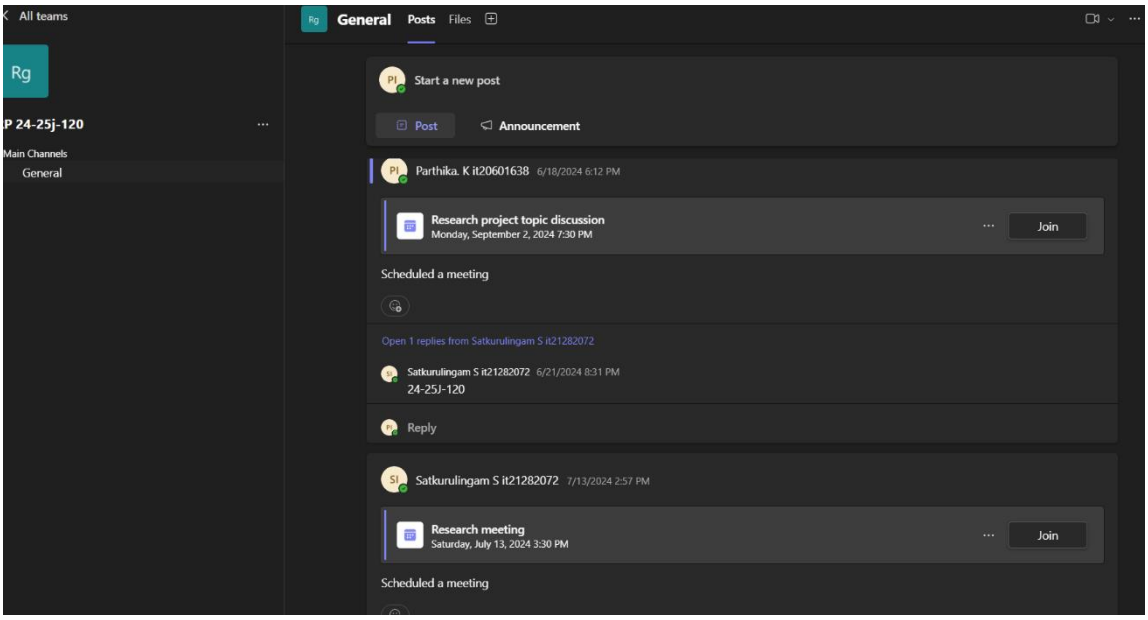


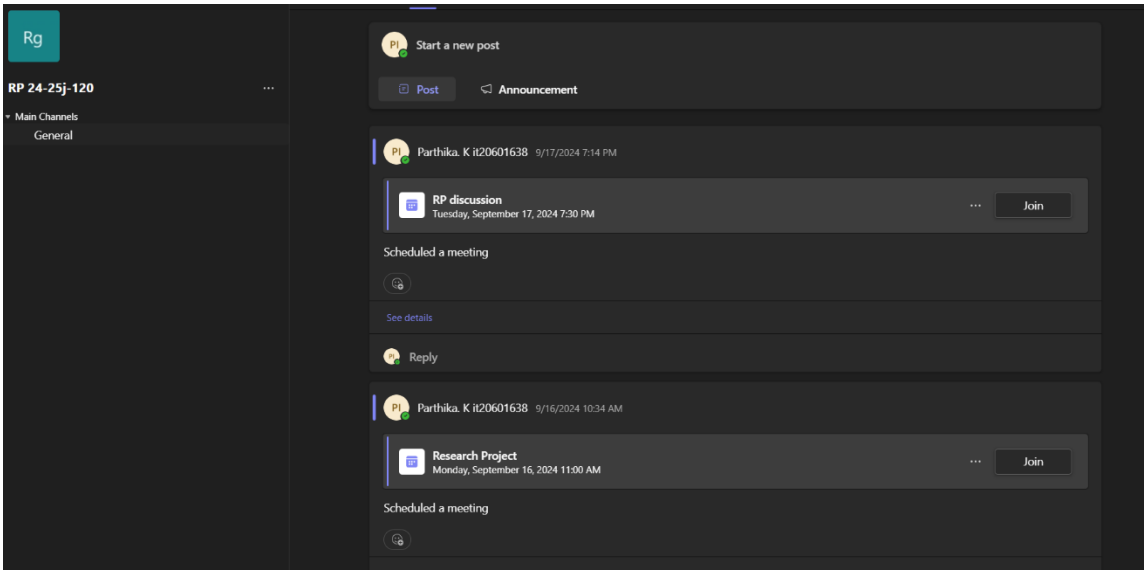



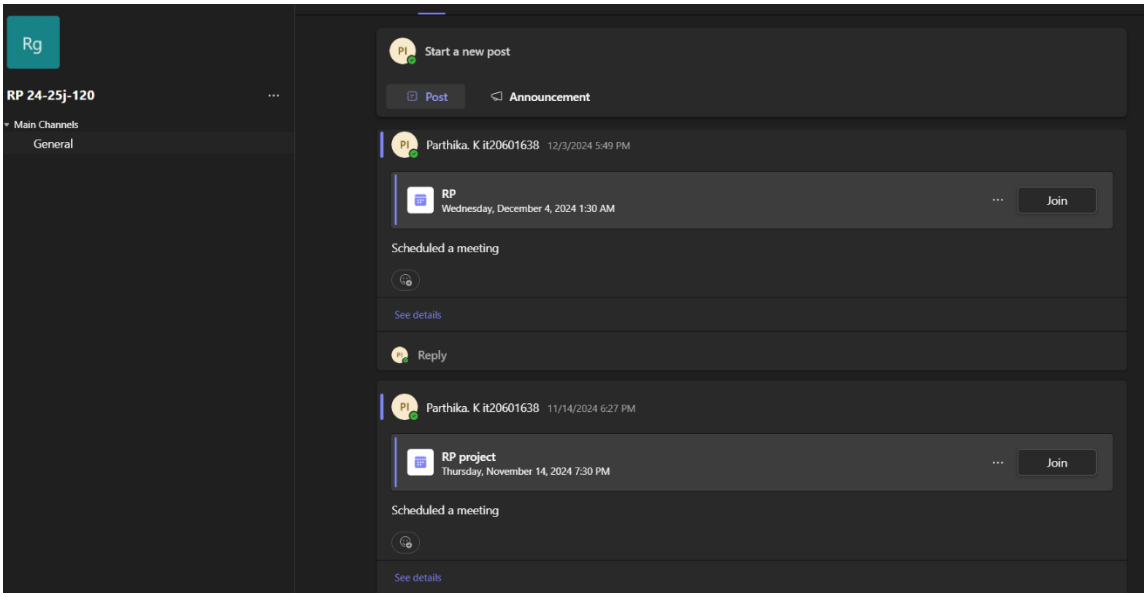



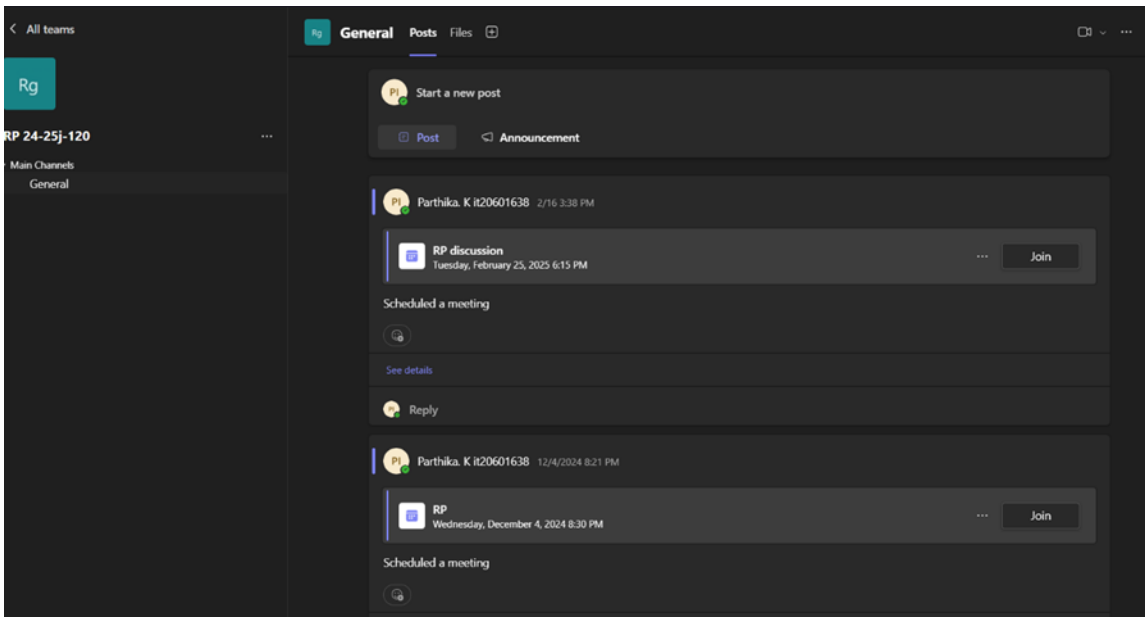

► **Complete Tasks and Conversation Highlights**

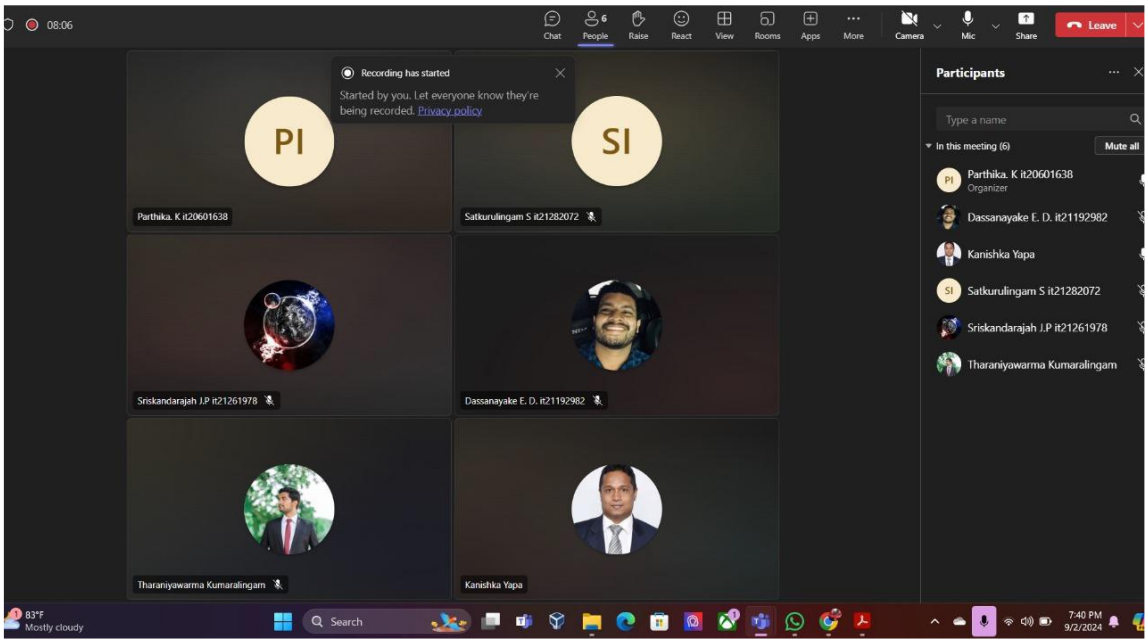
- Meeting with the research team and deciding the implementation milestone on Microsoft Teams.




Rg
RP 24-25j-120
Main Channels
General
Start a new post
Post
Announcement
Parthika. K it20601638 9/17/2024 7:14 PM
RP discussion
Tuesday, September 17, 2024 7:30 PM
Join
Scheduled a meeting
See details
Reply
Parthika. K it20601638 9/16/2024 10:34 AM
Research Project
Monday, September 16, 2024 11:00 AM
Join
Scheduled a meeting

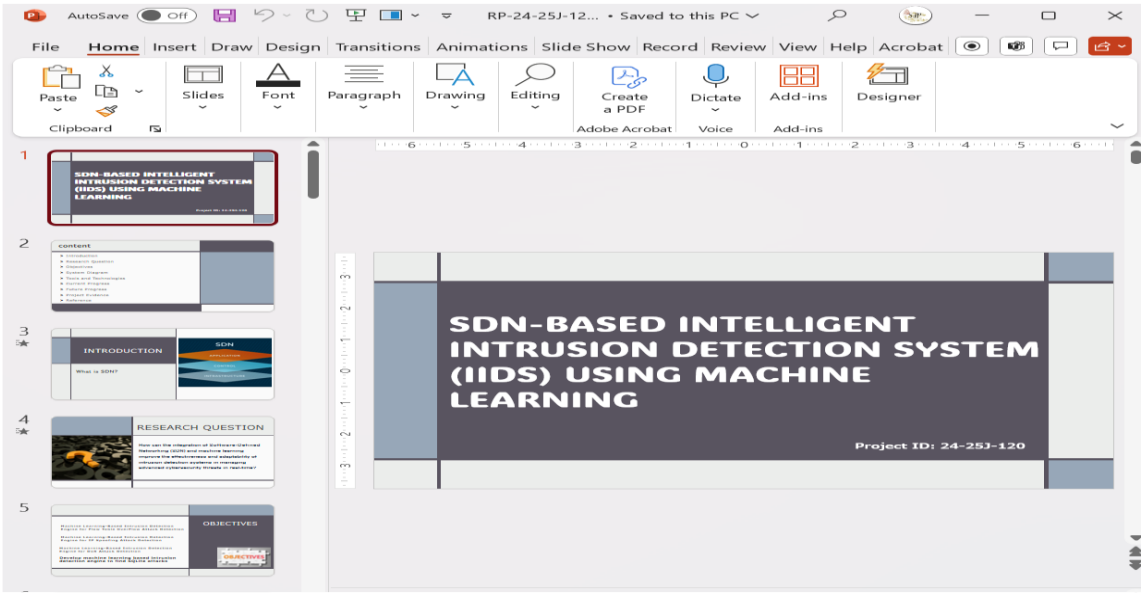

Rg
RP 24-25j-120
Main Channels
General
Start a new post
Post
Announcement
Parthika. K it20601638 12/3/2024 5:49 PM
RP
Wednesday, December 4, 2024 1:30 AM
Join
Scheduled a meeting
See details
Reply
Parthika. K it20601638 11/14/2024 6:27 PM
RP project
Thursday, November 14, 2024 7:30 PM
Join
Scheduled a meeting
See details

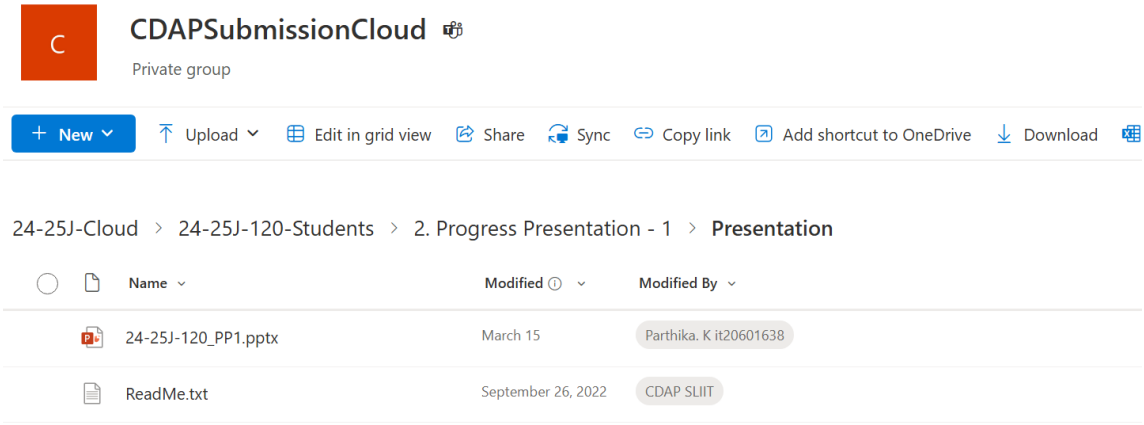

All teams
Rg
RP 24-25j-120
Main Channels
General
General
Posts
Files
Start a new post
Post
Announcement
Parthika. K it20601638 2/16 3:38 PM
RP discussion
Tuesday, February 25, 2025 6:15 PM
Join
Scheduled a meeting
See details
Reply
Parthika. K it20601638 12/4/2024 8:21 PM
RP
Wednesday, December 4, 2024 8:30 PM
Join
Scheduled a meeting




► **Completed Tasks and Conversation Highlights**

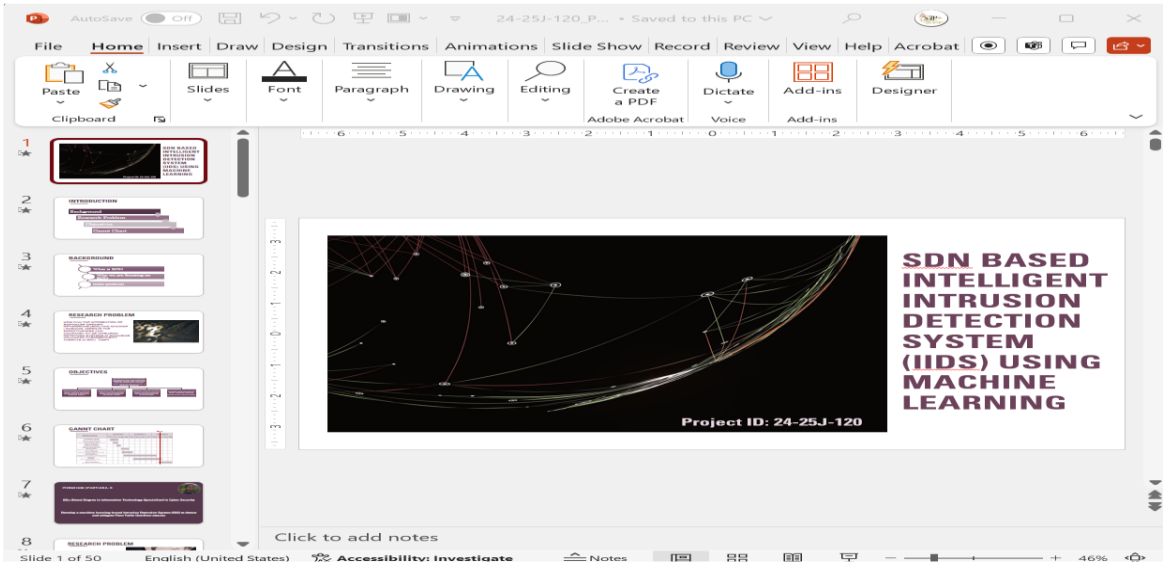
- Prepare for Progress Presentation 1 (PP1).
- Creating the presentation.
- Finalizing the Projects.
- Communication with the supervisor after finalizing the project.

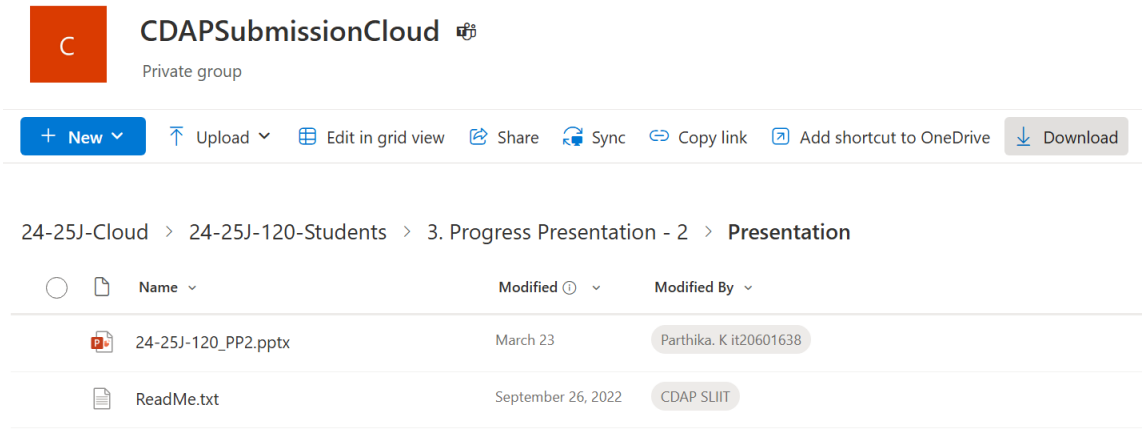





► **Completed Task and Conversation Highlights**

- Prepare for Progress Presentation 2 (PP2).
- Creating the presentation.
- Finalizing the Projects.
- Communication with the supervisor after finalizing the project.






## ► Completed Task and Conversation Highlights

- Started writing the research paper.
- Exploring the IEEE standards and word tools.
- Communicating with supervisor and getting the supervisor feedback

---

# SDN-based Intelligent Intrusion Detection System (IIDS) using Machine Learning

Parthika.K
*Faculty of Computing*
*Cyber Security Specilization*
*Sri Lanka Institute of Information Technology*
Malabe, Sri Lanka
kparthika@gmail.com

Satkurulingam.S
*Faculty of Computing*
*Cyber Security Specilization*
*Sri Lanka Institute of Information Technology*
Malabe, Sri Lanka
savithurisatkurulingam@gmail.com

Sriskandarajah J.P
*Faculty of Computing*
*Cyber Security Specilization*
*Sri Lanka Institute of Information Technology*
Malabe, Sri Lanka
srijoanna0@gmail.com

Dassanayake E.D
*Faculty of Computing*
*Cyber Security Specilization*
*Sri Lanka Institute of Information Technology*
Malabe, Sri Lanka
erangadassanayake15@gmail.com

Kanishka Prajeewa Yapa
*Department of Computer Systems Engineering*
*Sri Lanka Institute of Information Technology*
Malabe, Sri Lanka
kanishka.y@sliit.lk

Tharaniyawarma.K
*Department of Computer Systems Engineering*
*Sri Lanka Institute of Information Technology*
Malabe, Sri Lanka
tharaniyawarma.k@sliit.lk

***Abstract*— The increasing network complexity needs Software-Defined Networking (SDN) as a key solution to establish effective management systems through dynamic control mechanisms. SDN network infrastructure encounters four main security threats including Denial of Service (DoS), Flow Table Overflow, SQLite and Topology Poisoning attacks. This paper presents IIDS as an SDN-based intelligent intrusion detection system that operates with machine learning schemes to identify threats during real-time interactions. A dynamic system links the SDN controller with machine learning models for network traffic analysis to detect anomalies. The testing of the proposed method generates results for accuracy while also measuring precision and recall along with F1-score values. Starting from optimized attack detection the implemented technology is confirmed as an effective security framework for SDN networks because of its precise outcome.**

**Keywords—cyber security, software defined networking, intrusion detection system, machine learning**

## I. Introduction

By implementing Software-Defined Networking (SDN) operators can execute dynamic management operations combined with automated system management tasks for their network infrastructure [1]. Multiple security threats can occur through SDN's central management architecture because it exposes itself to various cyber-attacks. Network resilience becomes unsustainable due to difficulties with implementing protection measures for new security threats within the current obtain live policy controls that improve network protection. Studies present evidence that SDN Technology integration with machine learning achieves successful threat detection solutions by using intelligent intrusion systems which improve security protection for network infrastructure.

## II. Literature Review

Studies performed by technologists demonstrate IDS technology as a solution to enhance SDN security while dynamic network administration needs advanced threat management solutions [3]. Traditional IDS operations heavily depend on signature detection thus their ability to detect modern cyber threats remains low [4]. Machine learning within IDS technology speeds up performance and adjusts to new threats because it detects unknown attacks by analyzing patterns and statistical deviations [5]. The application of machine learning in IDS achieves superior performance speed as well as adaptability through its ability to detect unknown threats through anomaly detection and pattern recognition capabilities. Numerous researchers have focused on using multivariate statistical analysis to detect SDN attacks, as this approach enhances network security detection capabilities. By analyzing multiple variables simultaneously, this technique improves anomaly detection, identifying potential security threats more accurately. Researchers have explored various statistical models to enhance intrusion detection systems, making them more effective in mitigating security risks within SDN environments [6]. The evaluation of Flow Table

🌟 **Looking to Publish Your Research and Meet RP Deadlines?**

🍂 **ICAC 2025 is the Opportunity!**

If you're searching for a **reputed and impactful venue** to publish your research paper and meet your Research Project (RP) deadlines, **ICAC 2025** is ready to support you.

📅 **Date:** Wednesday, **21st May 2025**
🕢 **Time:** 7.30 PM
🎙️ **Speaker:** *Dr. Prasanna Sumathipala*

🔗 **Zoom Link:** https://zoom.us/j/94450599762?pwd=gxr3Zp8abKaS8kRN5mlA0san4sDRMZ.1


## OBJECTIVES

Create a platform for networking and collaborative research and development activities among national and international researchers.

Promote advanced research culture among academic and cooperate communities around the globe.

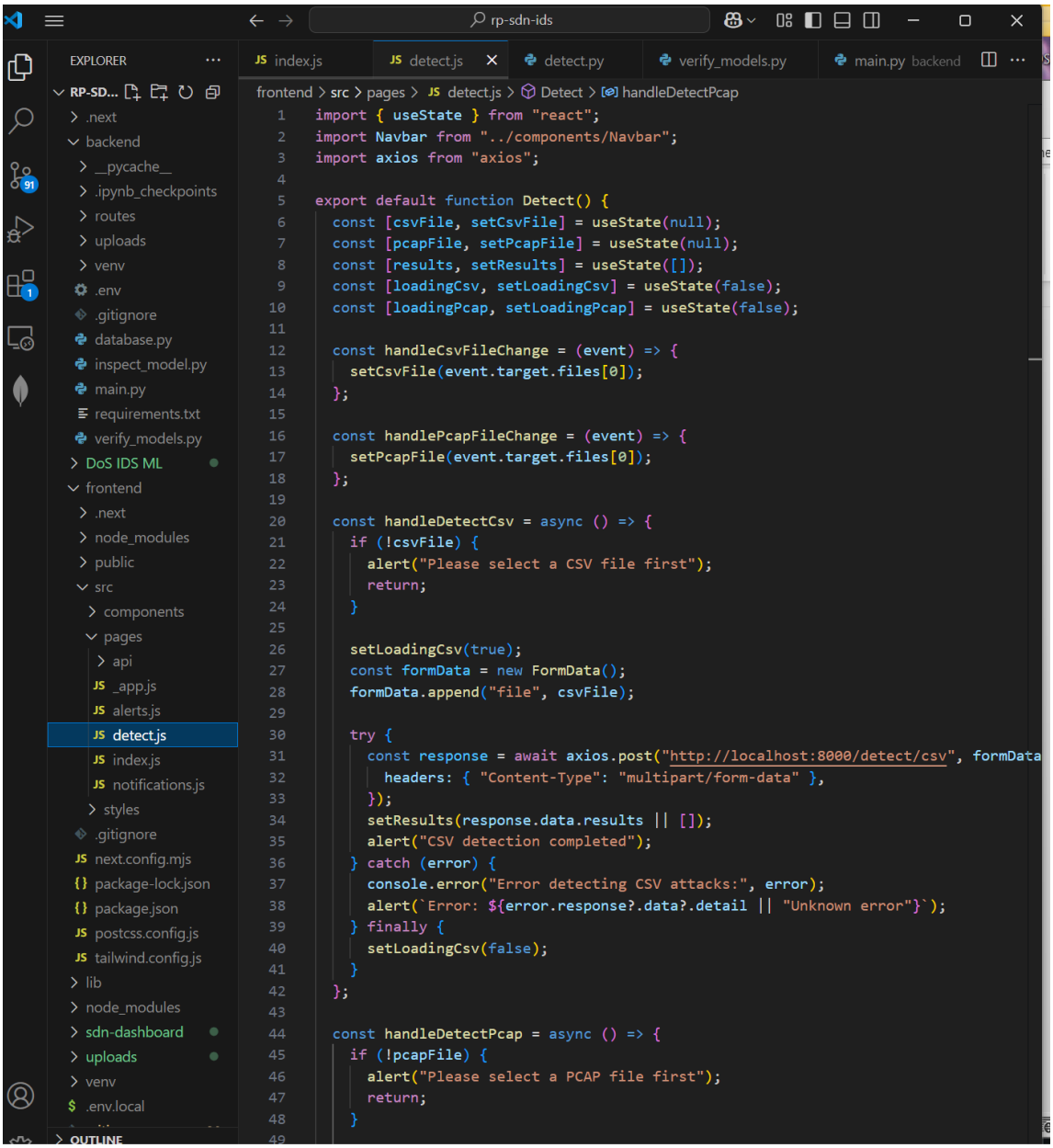
Highlight the achievements of local and international researchers in the field of computing.

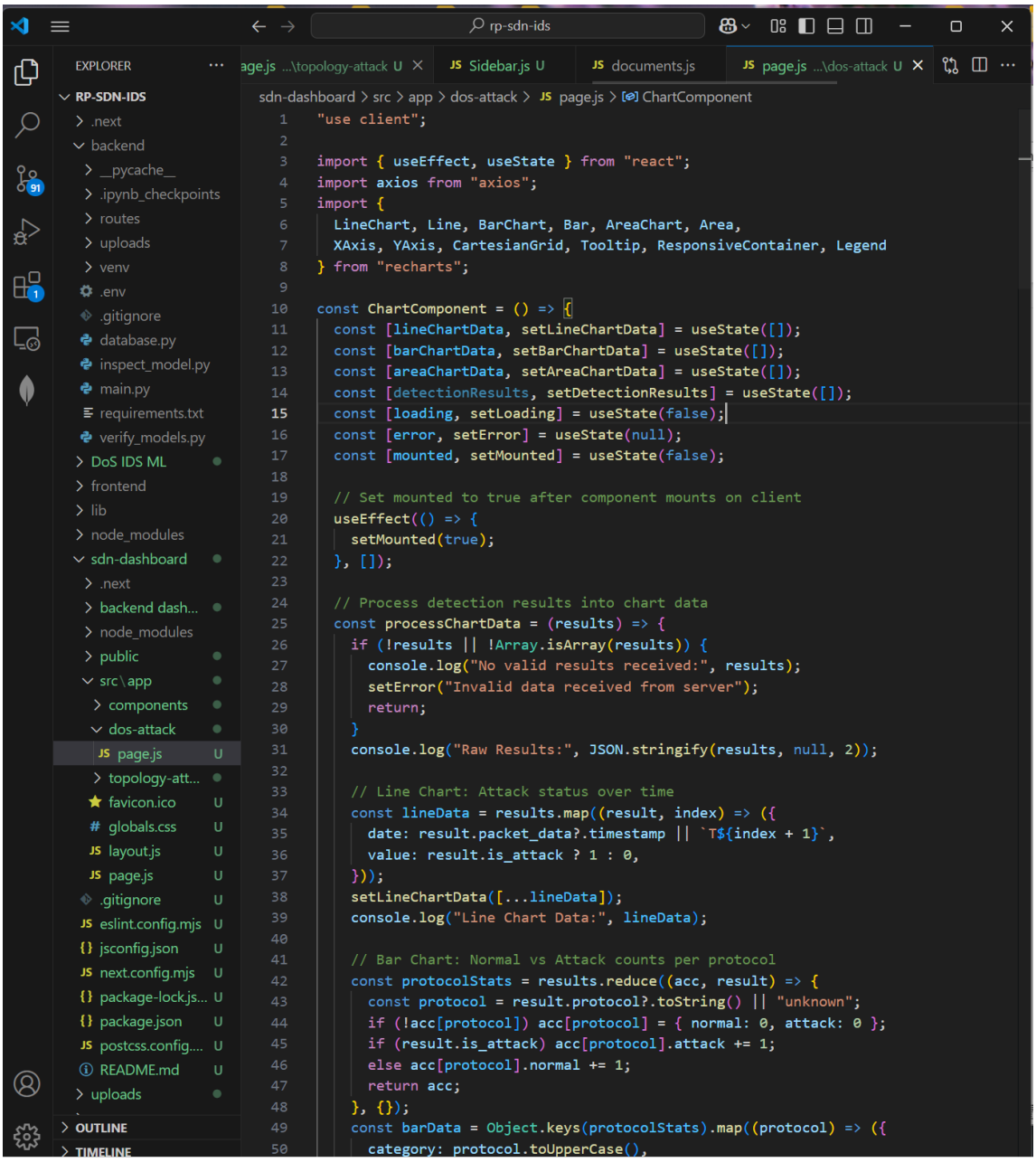
Foster groundbreaking innovations in computing by integrating Artificial Intelligence, Quantum Computing, and emerging technologies to solve complex challenges, drive cross-disciplinary advancements, and shape the future of industries worldwide.

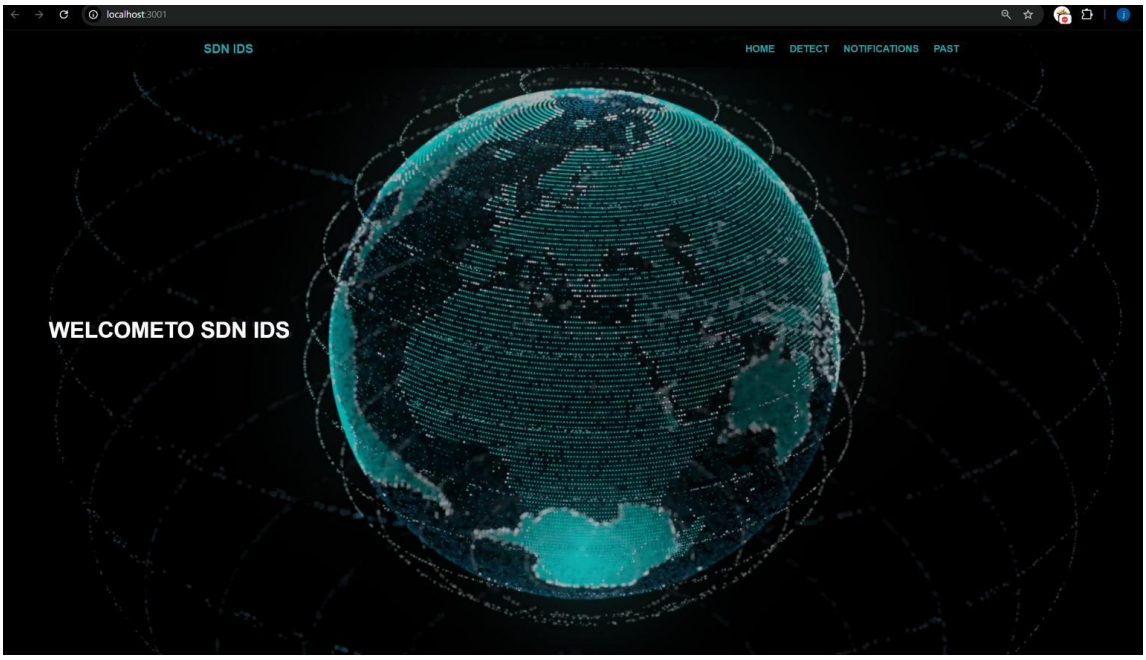
| | |
|---|---|
| Paper Submission - Open | 05th May 2025 Submit Now |
| Paper Submission - Close | 20th August 2025 |
| Acceptance Notification | 20th October 2025 |
| Camera Ready & Registration - Open | 20th October 2025 |
| Camera Ready & Registration - Close | 10th November 2025 |
| Conference | 09th and 10th December 2025 |

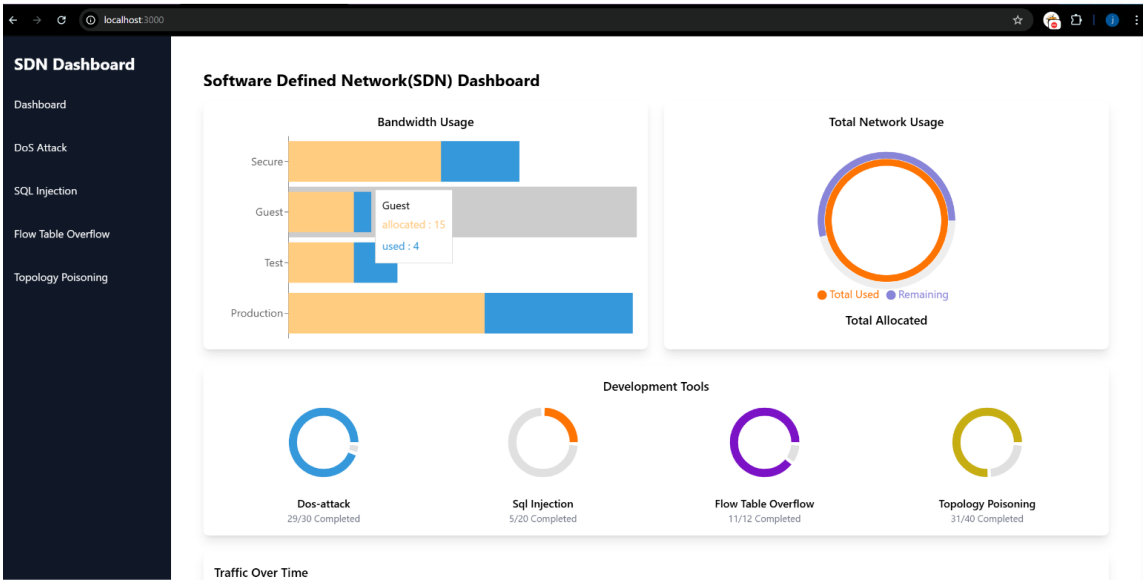
► **Completed Tasks and Conversation Highlights**

- Creating the front-end of the application.
- Integration of all the components.
- Discussing the supervisor's suggestions




rp-sdn-ids
EXPLORER
RP-SDN-IDS
.next
backend
__pycache__
.ipynb_checkpoints
routes
uploads
venv
.env
.gitignore
database.py
inspect_model.py
main.py
requirements.txt
verify_models.py
DoS IDS ML
frontend
lib
node_modules
sdn-dashboard
.next
backend dash...
node_modules
public
src\app
components
dos-attack
page.js
topology-att...
favicon.ico
globals.css
layout.js
page.js
.gitignore
eslint.config.mjs
jsconfig.json
next.config.mjs
package-lock.js...
package.json
postcss.config....
README.md
uploads
OUTLINE
TIMELINE
page.js ...\topology-attack U
Sidebar.js U
documents.js
page.js ...\dos-attack U
sdn-dashboard > src > app > dos-attack > page.js > ChartComponent
"use client";
import { useEffect, useState } from "react";
import axios from "axios";
import {
LineChart, Line, BarChart, Bar, AreaChart, Area,
XAxis, YAxis, CartesianGrid, Tooltip, ResponsiveContainer, Legend
} from "recharts";
const ChartComponent = () => {
const [lineChartData, setLineChartData] = useState([]);
const [barChartData, setBarChartData] = useState([]);
const [areaChartData, setAreaChartData] = useState([]);
const [detectionResults, setDetectionResults] = useState([]);
const [loading, setLoading] = useState(false);
const [error, setError] = useState(null);
const [mounted, setMounted] = useState(false);
// Set mounted to true after component mounts on client
useEffect(() => {
setMounted(true);
}, []);
// Process detection results into chart data
const processChartData = (results) => {
if (!results || !Array.isArray(results)) {
console.log("No valid results received:", results);
setError("Invalid data received from server");
return;
}
console.log("Raw Results:", JSON.stringify(results, null, 2));
// Line Chart: Attack status over time
const lineData = results.map((result, index) => ({
date: result.packet_data?.timestamp || `T${index + 1}`,
value: result.is_attack ? 1 : 0,
}));
setLineChartData([...lineData]);
console.log("Line Chart Data:", lineData);
// Bar Chart: Normal vs Attack counts per protocol
const protocolStats = results.reduce((acc, result) => {
const protocol = result.protocol?.toString() || "unknown";
if (!acc[protocol]) acc[protocol] = { normal: 0, attack: 0 };
if (result.is_attack) acc[protocol].attack += 1;
else acc[protocol].normal += 1;
return acc;
}, {});
const barData = Object.keys(protocolStats).map((protocol) => ({
category: protocol.toUpperCase(),


localhost:3001
SDN IDS
HOME
DETECT
NOTIFICATIONS
PAST
WELCOMETO SDN IDS


localhost:3000
SDN Dashboard
Dashboard
DoS Attack
SQL Injection
Flow Table Overflow
Topology Poisoning
Software Defined Network(SDN) Dashboard
Bandwidth Usage
Secure
Guest
Test
Production
Guest
allocated : 15
used : 4
Total Network Usage
Total Used
Remaining
Total Allocated
Development Tools
Dos-attack
29/30 Completed
Sql Injection
5/20 Completed
Flow Table Overflow
11/12 Completed
Topology Poisoning
31/40 Completed
Traffic Over Time

► **Completed Tasks and Conversation Highlights**

- Complete Individual Thesis Reports.
- Creation Group Thesis Reports.

SDN-BASED INTELLIGENT INTRUSION DETECTION SYSTEM (IIDS) USING MACHINE LERANING

Parthika. K

(IT20601638)

BSc.(Hons) in Information Technology

Specializing in Cyber Security

Department of Information System Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

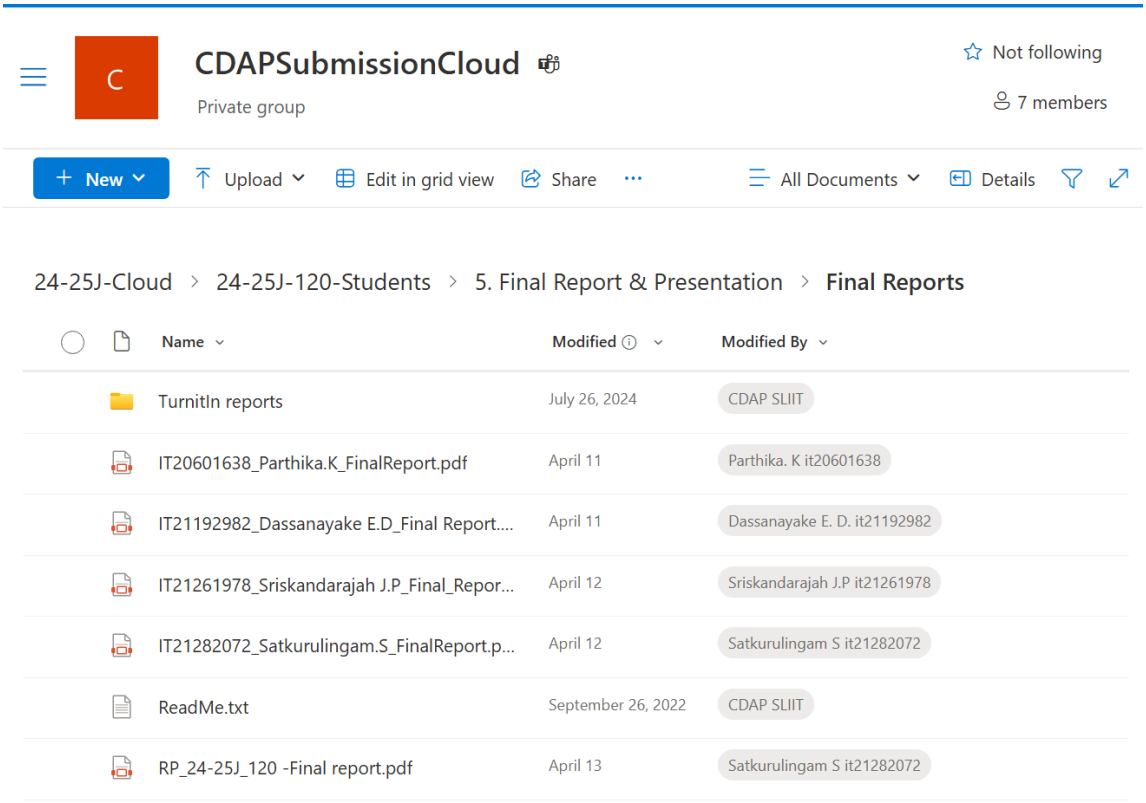


**DECLARATION**

I declare that this is my own work, and this dissertation does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text. Also, I hereby grant to Sri Lanka Institute of Information Technology the non-exclusive right to reproduce and distribute my dissertation in whole or part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

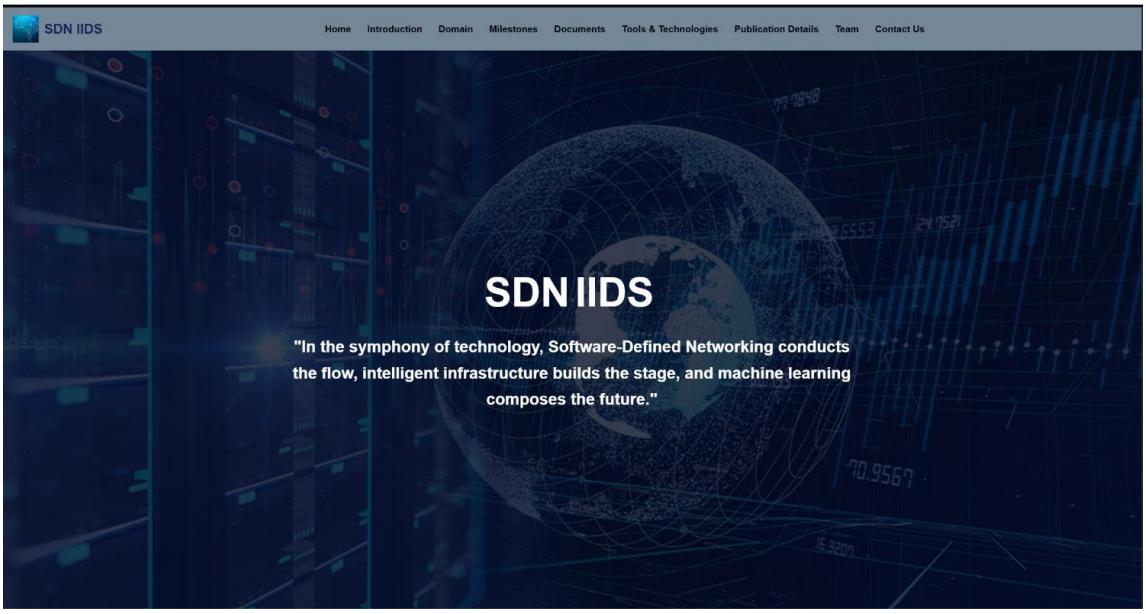
Signature: Date: 11/04/2025

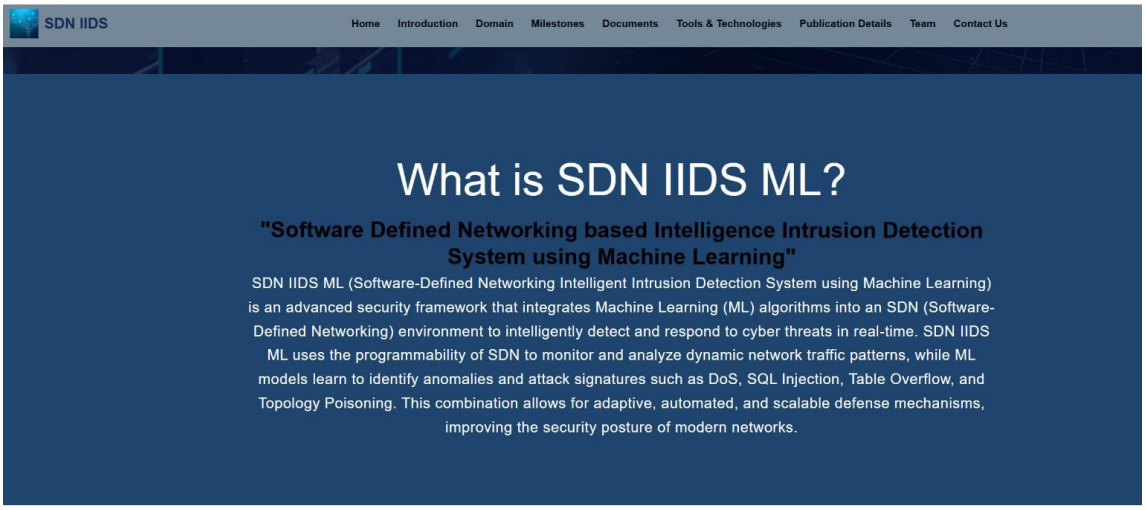
Signature of the Supervisor: Date:

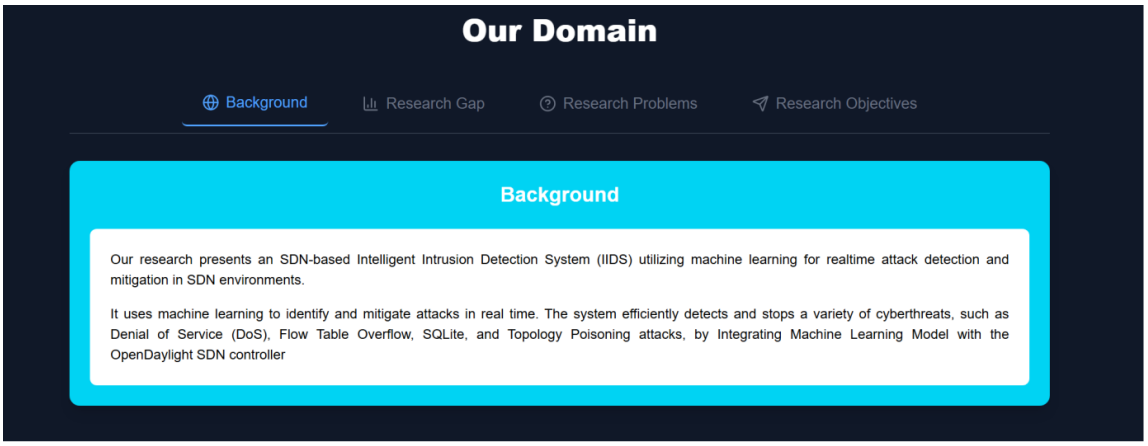

C
CDAPSubmissionCloud
Private group
Not following
7 members
New
Upload
Edit in grid view
Share
All Documents
Details
24-25J-Cloud
24-25J-120-Students
5. Final Report & Presentation
Final Reports
Name
Modified
Modified By
TurnitIn reports
July 26, 2024
CDAP SLIIT
IT20601638_Parthika.K_FinalReport.pdf
April 11
Parthika. K it20601638
IT21192982_Dassanayake E.D_Final Report....
April 11
Dassanayake E. D. it21192982
IT21261978_Sriskandarajah J.P_Final_Repor...
April 12
Sriskandarajah J.P it21261978
IT21282072_Satkurulingam.S_FinalReport.p...
April 12
Satkurulingam S it21282072
ReadMe.txt
September 26, 2022
CDAP SLIIT
RP_24-25J_120 -Final report.pdf
April 13
Satkurulingam S it21282072

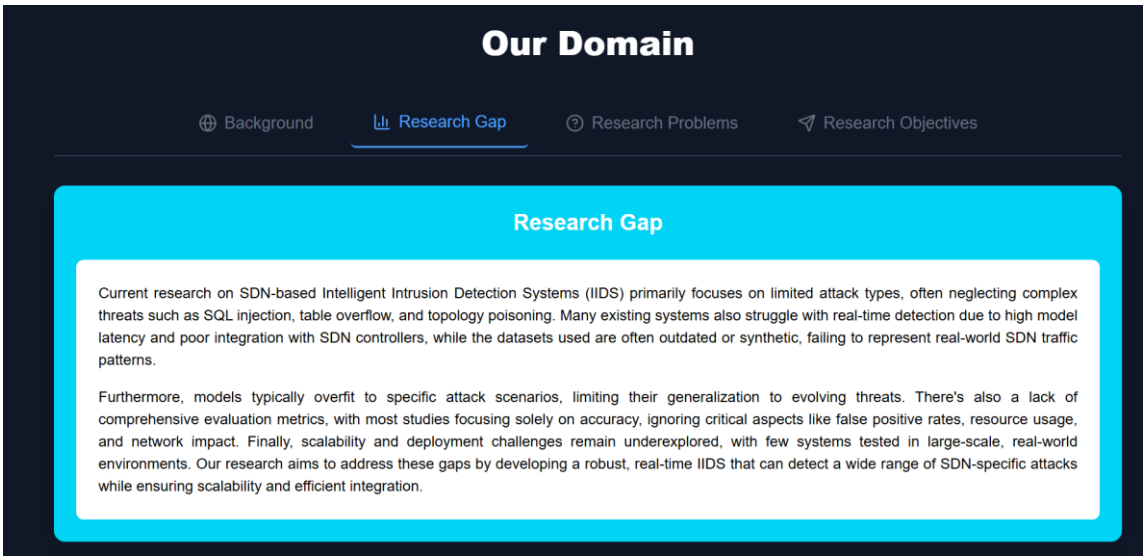
► **Completed Tasks and Conversation Highlights**

- Create a website for the solution




Our Domain
Background
Research Gap
Research Problems
Research Objectives
Background
Our research presents an SDN-based Intelligent Intrusion Detection System (IIDS) utilizing machine learning for realtime attack detection and mitigation in SDN environments.
It uses machine learning to identify and mitigate attacks in real time. The system efficiently detects and stops a variety of cyberthreats, such as Denial of Service (DoS), Flow Table Overflow, SQLite, and Topology Poisoning attacks, by Integrating Machine Learning Model with the OpenDaylight SDN controller


Our Domain
Background
Research Gap
Research Problems
Research Objectives
Research Gap
Current research on SDN-based Intelligent Intrusion Detection Systems (IIDS) primarily focuses on limited attack types, often neglecting complex threats such as SQL injection, table overflow, and topology poisoning. Many existing systems also struggle with real-time detection due to high model latency and poor integration with SDN controllers, while the datasets used are often outdated or synthetic, failing to represent real-world SDN traffic patterns.
Furthermore, models typically overfit to specific attack scenarios, limiting their generalization to evolving threats. There's also a lack of comprehensive evaluation metrics, with most studies focusing solely on accuracy, ignoring critical aspects like false positive rates, resource usage, and network impact. Finally, scalability and deployment challenges remain underexplored, with few systems tested in large-scale, real-world environments. Our research aims to address these gaps by developing a robust, real-time IIDS that can detect a wide range of SDN-specific attacks while ensuring scalability and efficient integration.

# Our Domain

Background | Research Gap | Research Problems | Research Objectives

## Research Problems

### Flow Table Overflow

Table overflow attacks in SDN target the limited flow table capacity of switches, overwhelming them with excessive flow entries. Existing detection methods either rely on static thresholds or reactive strategies that are ineffective under adaptive attack patterns. There is a critical need for proactive, intelligent detection models that can recognize subtle anomalies in flow dynamics and prevent table saturation without impacting legitimate traffic.

### Topology Poisoning

Topology poisoning attacks exploit the dynamic nature of SDN by injecting false topology information, leading to incorrect routing decisions and network disruption. Existing detection approaches often rely on static rules or topology snapshots, which fail to adapt to rapidly changing network states. There is a pressing need for ML-based systems that can learn normal topology patterns, detect deviations in real time, and safeguard the network from such attacks.

### Denial of Service

Current SDN-based IDS systems often focus on detecting common DoS attacks like UDP floods but fail to effectively identify protocol-specific threats such as SNMP and DNS amplification attacks in real time. The lack of protocol-aware models and comprehensive datasets limits the system's ability to distinguish between normal traffic and sophisticated DoS patterns, leading to high false positives and delayed mitigation in dynamic SDN environments.

### SQL injection

SQL injection attacks in SDN environments are under-researched, as most studies focus on web applications. In SDN, malicious SQL queries can target northbound APIs or management systems, causing misconfigurations or unauthorized data access. The lack of tailored ML models for SQLi in SDN and the absence of real-time detection frameworks create a significant vulnerability, necessitating research into robust SQLi detection mechanisms for SDN control layers.

# Our Domain

Background | Research Gap | Research Problems | Research Objectives

## Research Objectives

### Main Objective

Our main objective is to develop SDN-based Intelligent Intrusion Detection System (IIDS) utilizing machine learning for real-time attack detection and mitigation in SDN environments

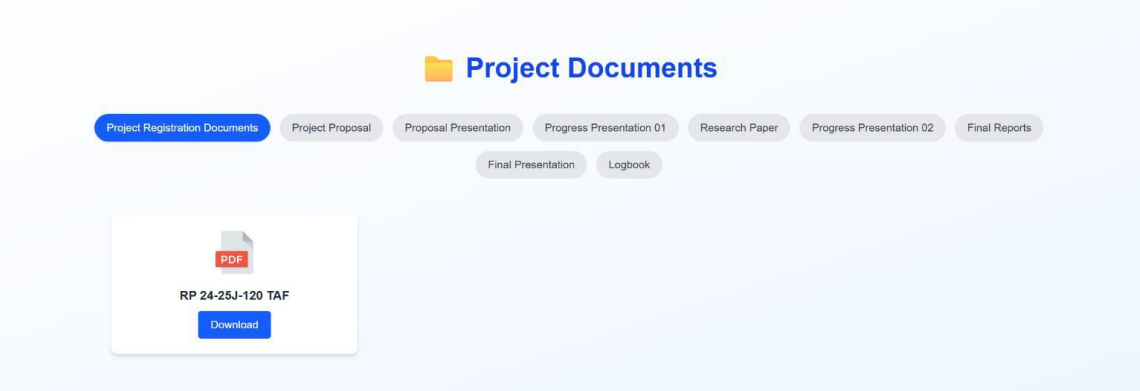
### Specific Objectives

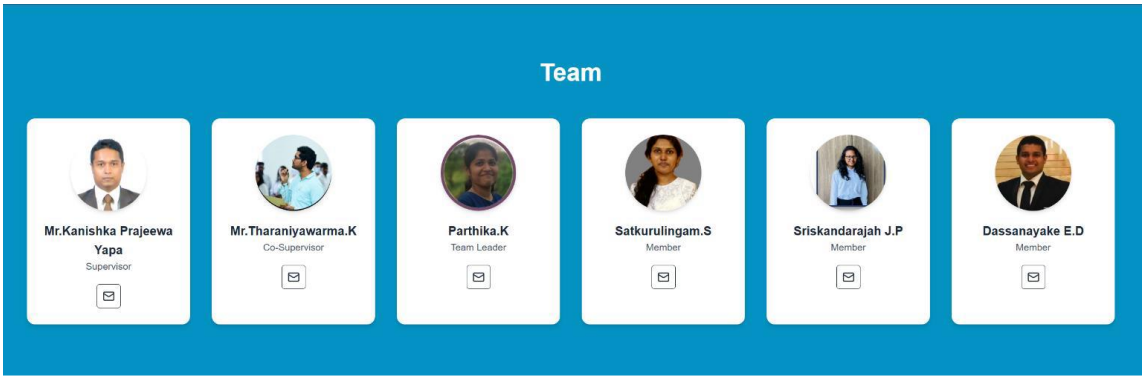
**1.** Develop machine learning based intrusion detection engine to find Flow Table Overflow attacks

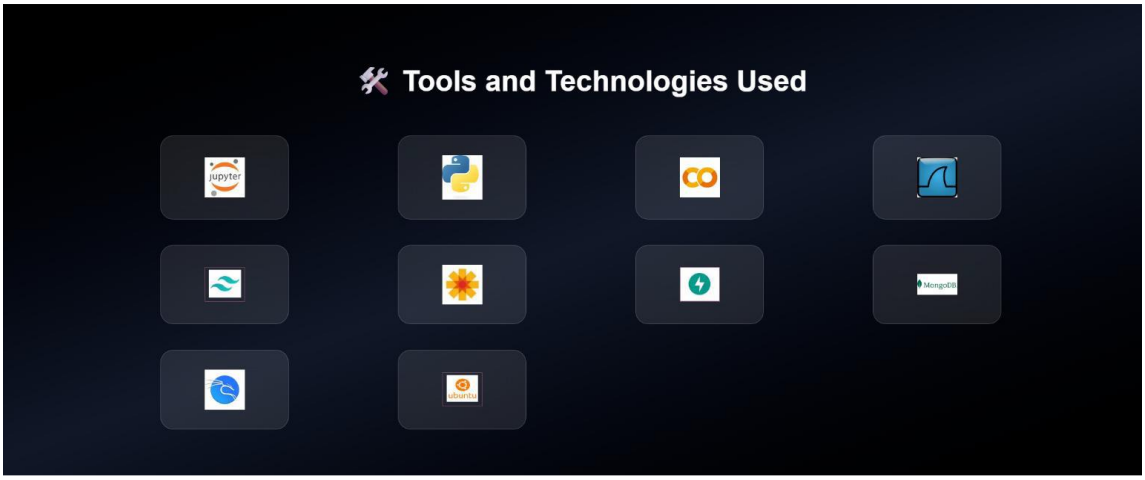
**2.** Develop machine learning based intrusion detection engine to find Topology Poisoning attacks

**3.** Develop machine learning based intrusion detection engine to find Denial of Service attacks

**4.** Develop machine learning based intrusion detection engine to find SQLite attacks

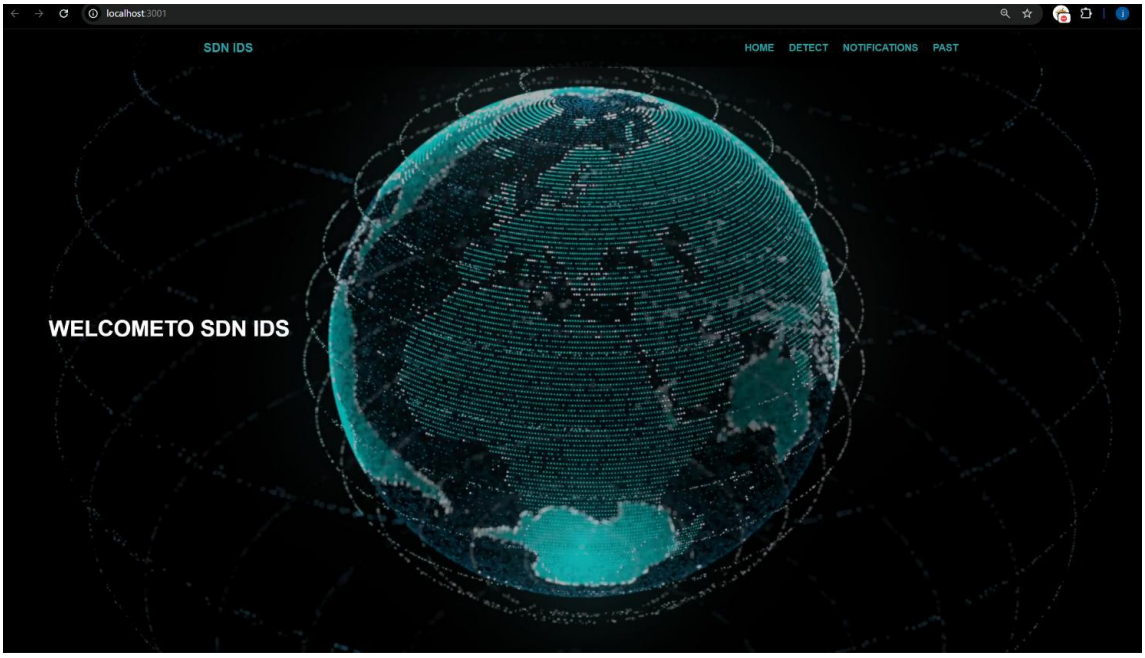

Project Documents
Project Registration Documents
Project Proposal
Proposal Presentation
Progress Presentation 01
Research Paper
Progress Presentation 02
Final Reports
Final Presentation
Logbook
PDF
RP 24-25J-120 TAF
Download

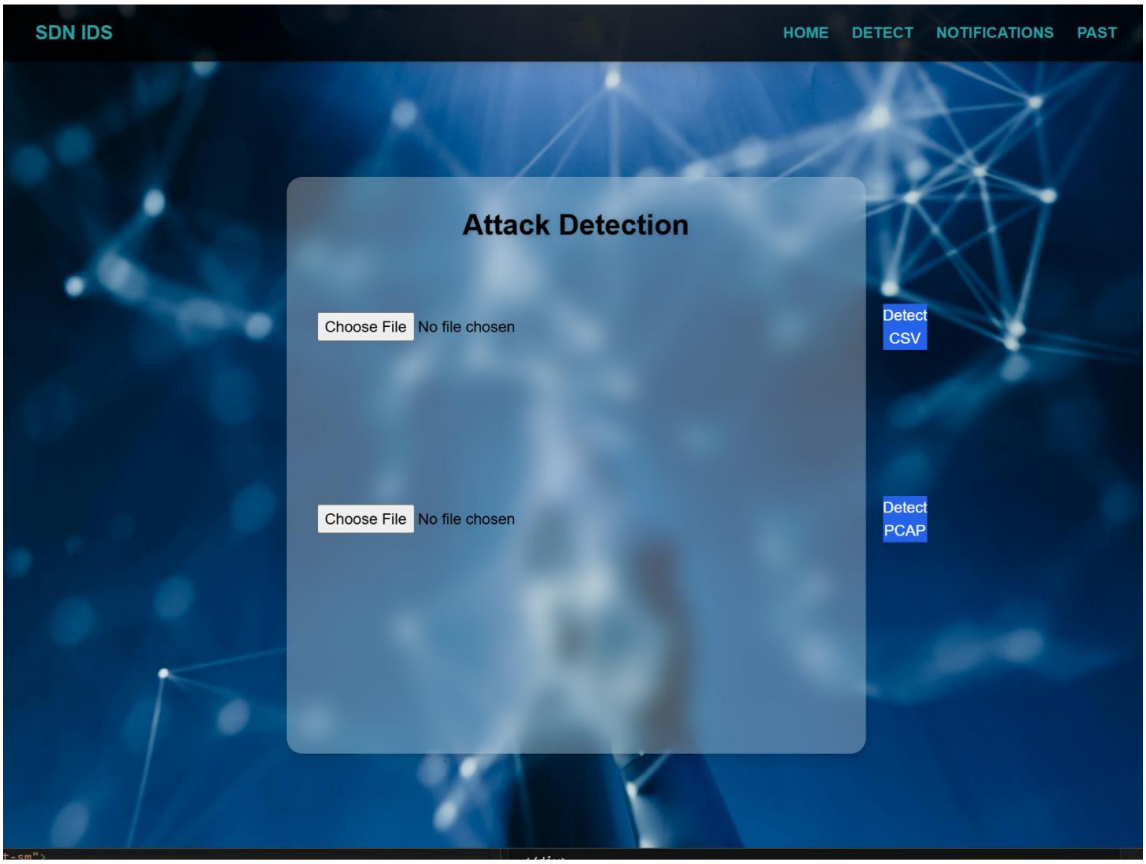

Team
Mr.Kanishka Prajeewa Yapa
Supervisor
Mr.Tharaniyawarma.K
Co-Supervisor
Parthika.K
Team Leader
Satkurulingam.S
Member
Sriskandarajah J.P
Member
Dassanayake E.D
Member

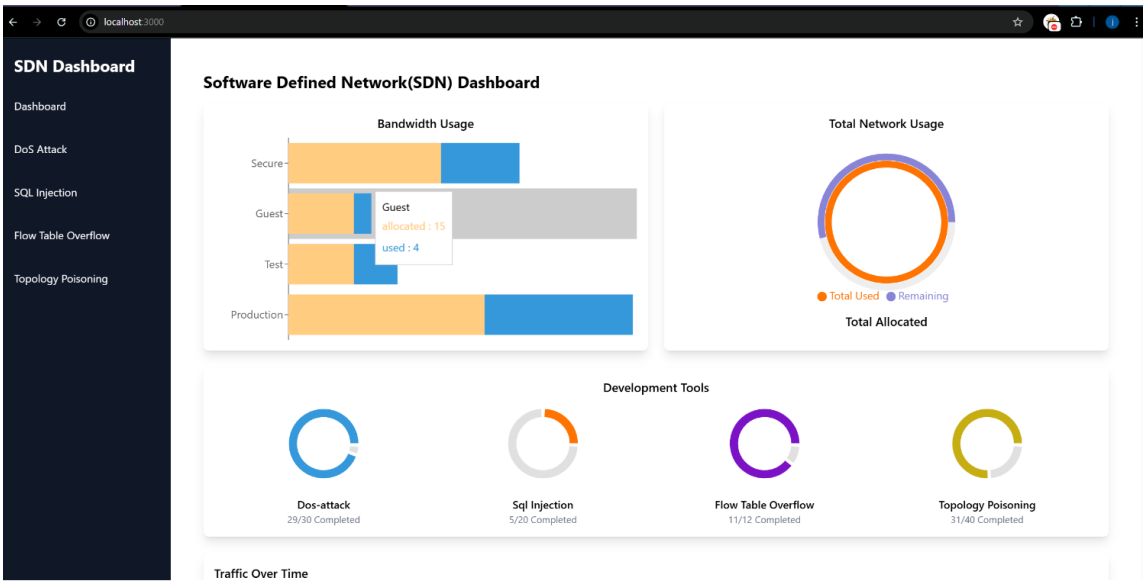

Tools and Technologies Used

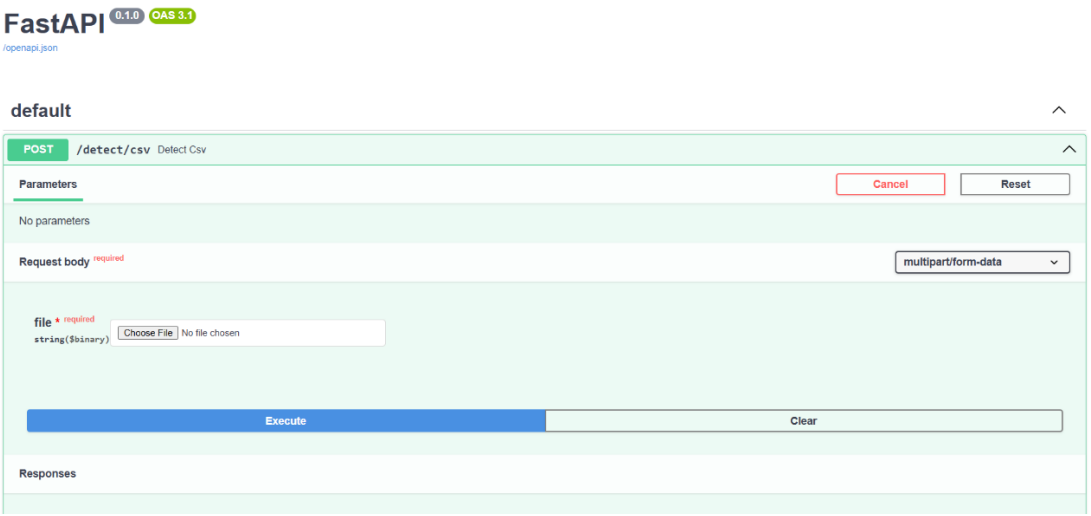
► **Completed Tasks and Conversation Highlights**

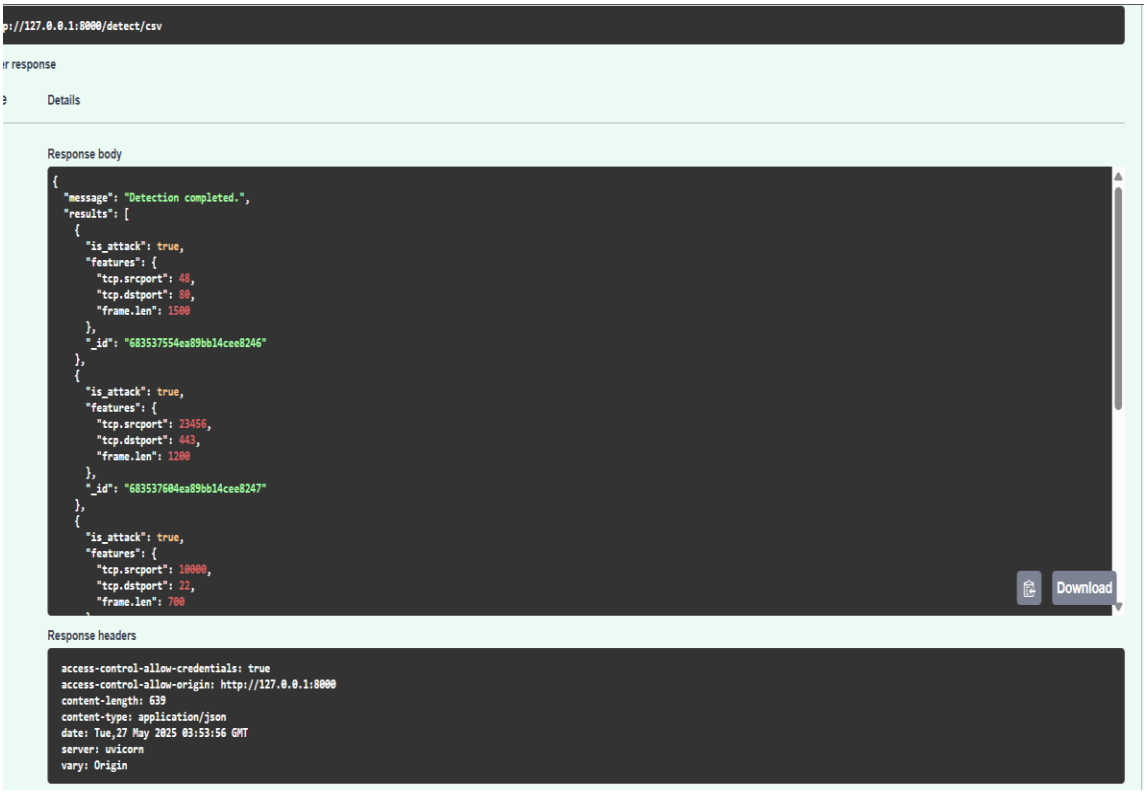
- Final Research Project Product

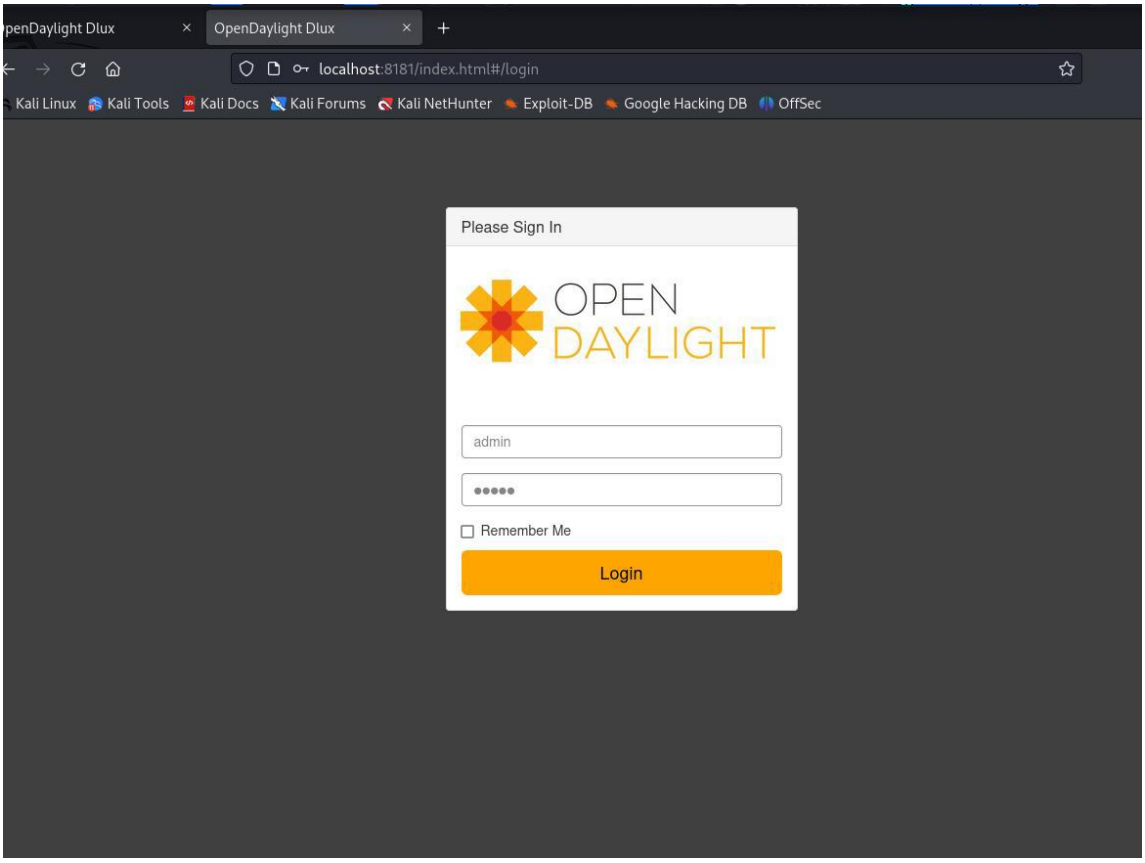


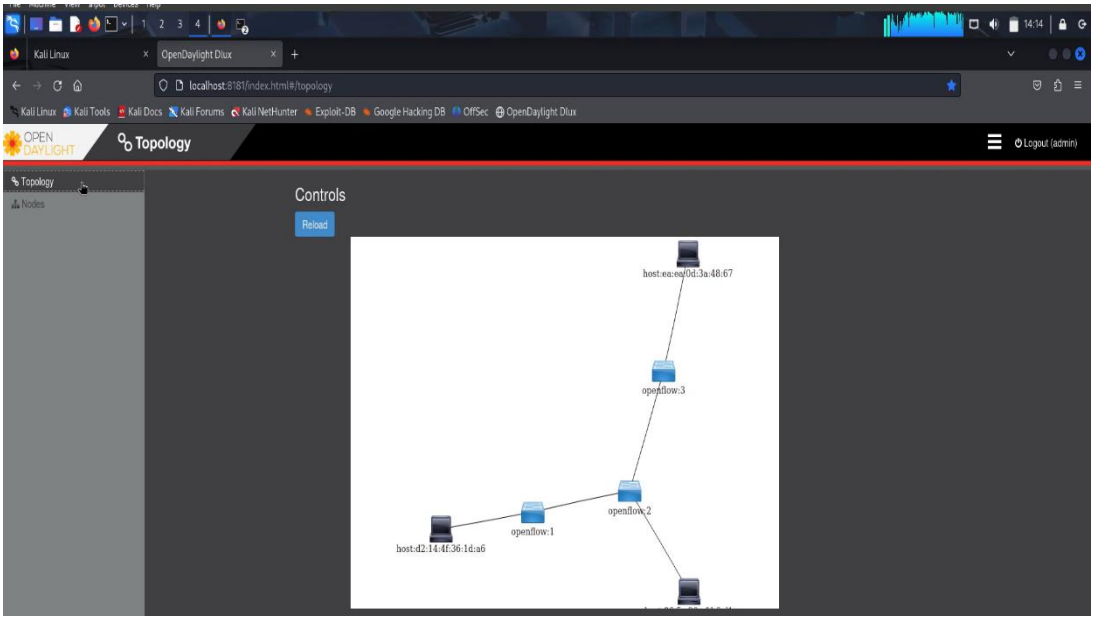



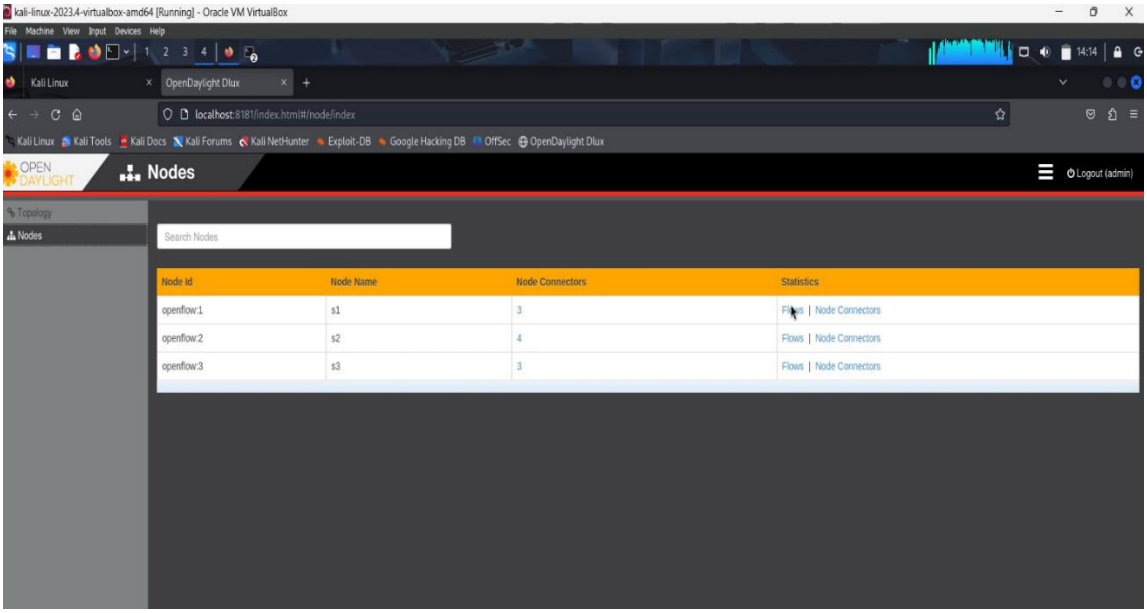

localhost:3000
SDN Dashboard
Dashboard
DoS Attack
SQL Injection
Flow Table Overflow
Topology Poisoning
Software Defined Network(SDN) Dashboard
Bandwidth Usage
Secure
Guest
Test
Production
Guest
allocated : 15
used : 4
Total Network Usage
Total Used
Remaining
Total Allocated
Development Tools
Dos-attack
29/30 Completed
Sql Injection
5/20 Completed
Flow Table Overflow
11/12 Completed
Topology Poisoning
31/40 Completed
Traffic Over Time

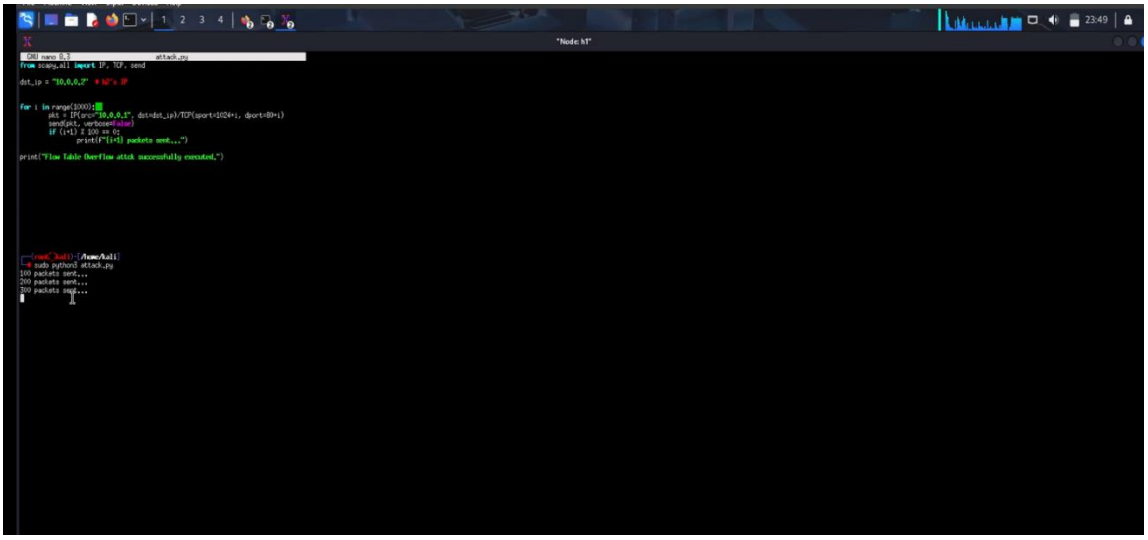

FastAPI
0.1.0
OAS 3.1
/openapi.json
default
POST
/detect/csv
Detect Csv
Parameters
Cancel
Reset
No parameters
Request body
required
multipart/form-data
file
required
string($binary)
Choose File
No file chosen
Execute
Clear
Responses


p://127.0.0.1:8000/detect/csv
r response
Details
Response body
{
"message": "Detection completed.",
"results": [
{
"is_attack": true,
"features": {
"tcp.srcport": 48,
"tcp.dstport": 80,
"frame.len": 1500
},
"_id": "683537554ea89bb14cee8246"
},
{
"is_attack": true,
"features": {
"tcp.srcport": 23456,
"tcp.dstport": 443,
"frame.len": 1200
},
"_id": "683537604ea89bb14cee8247"
},
{
"is_attack": true,
"features": {
"tcp.srcport": 10000,
"tcp.dstport": 22,
"frame.len": 700
Download
Response headers
access-control-allow-credentials: true
access-control-allow-origin: http://127.0.0.1:8000
content-length: 639
content-type: application/json
date: Tue,27 May 2025 03:53:56 GMT
server: uvicorn
vary: Origin


penDaylight Dlux
OpenDaylight Dlux
localhost:8181/index.html#/login
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
Please Sign In
OPEN
DAYLIGHT
admin
Remember Me
Login

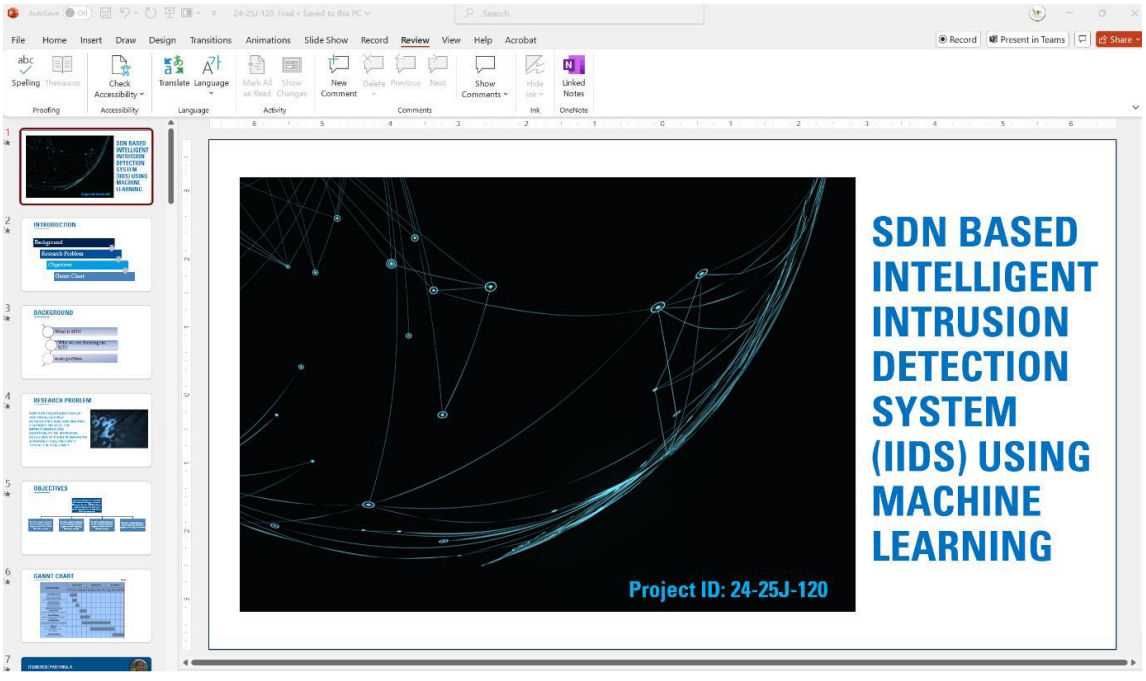

Kali Linux
OpenDaylight Dlux
localhost:8181/index.html#/topology
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec OpenDaylight Dlux
OPEN DAYLIGHT
Topology
Logout (admin)
Topology
Nodes
Controls
Reload
host:ea:ea:0d:3a:48:67
openflow:3
openflow:2
openflow:1
host:d2:14:4f:36:1d:a6

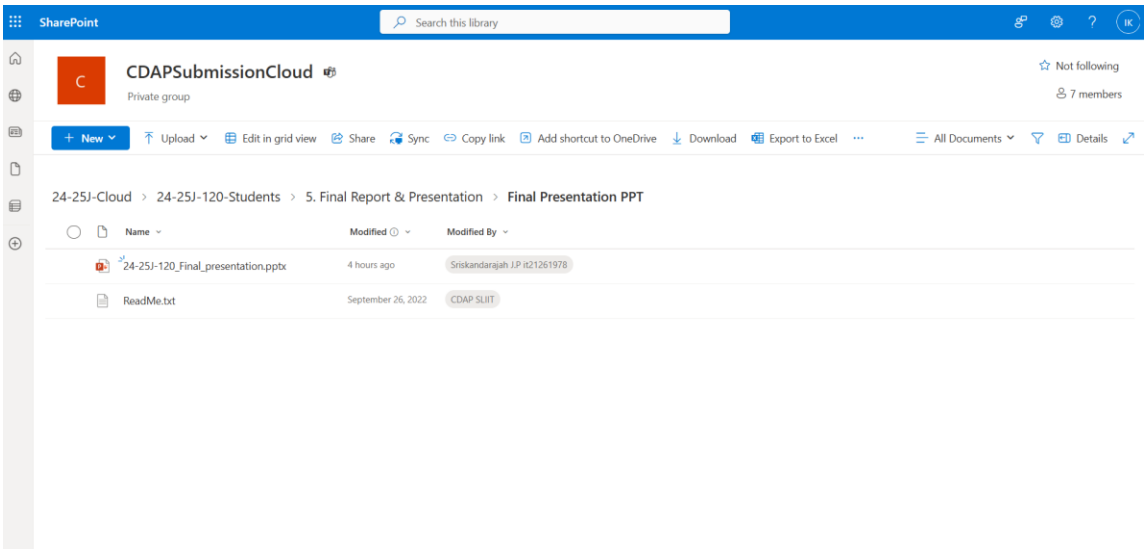

kali-linux-2023.4-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Kali Linux
OpenDaylight Dlux
localhost:8181/index.html#/node/index
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec OpenDaylight Dlux
OPEN DAYLIGHT
Nodes
Logout (admin)
Topology
Nodes
Search Nodes
Node Id
Node Name
Node Connectors
Statistics
openflow:1
s1
3
Flows | Node Connectors
openflow:2
s2
4
Flows | Node Connectors
openflow:3
s3
3
Flows | Node Connectors




► **Completed Task and Conversation Highlights**

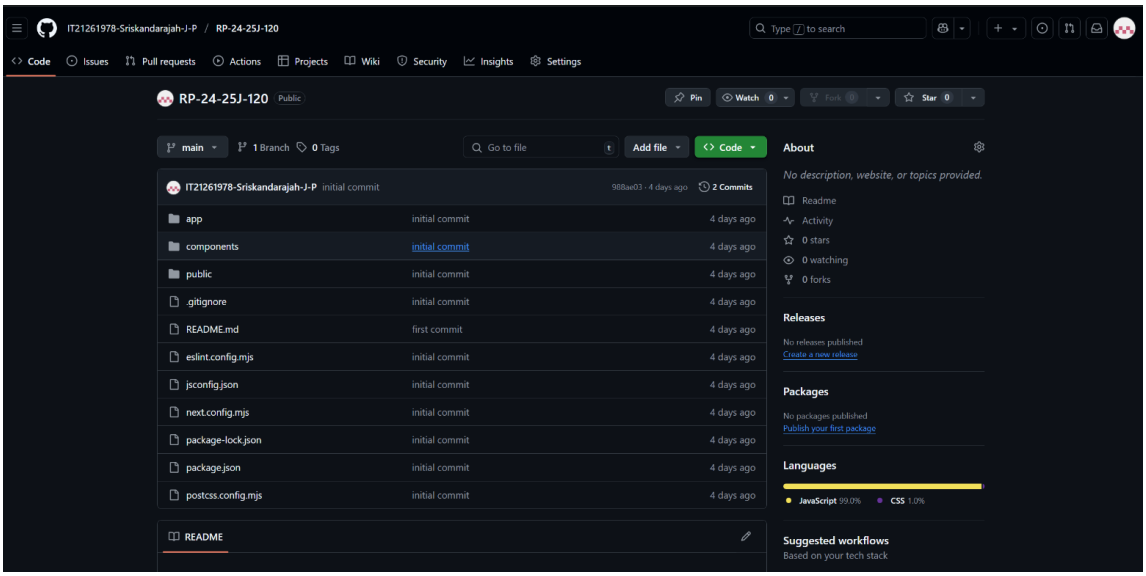
- Prepare for Final Presentation
- Creating the presentation.






## ► Completed Task and Conversation Highlights

- Commit and push the website codes in GitHub before deploying



► **Completed Task and Conversation Highlights**

- Deploy the website using vercel.