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

Group ID: RP-24-25J-120

Research Logbook

Dassanayake E.D.

IT21192982

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

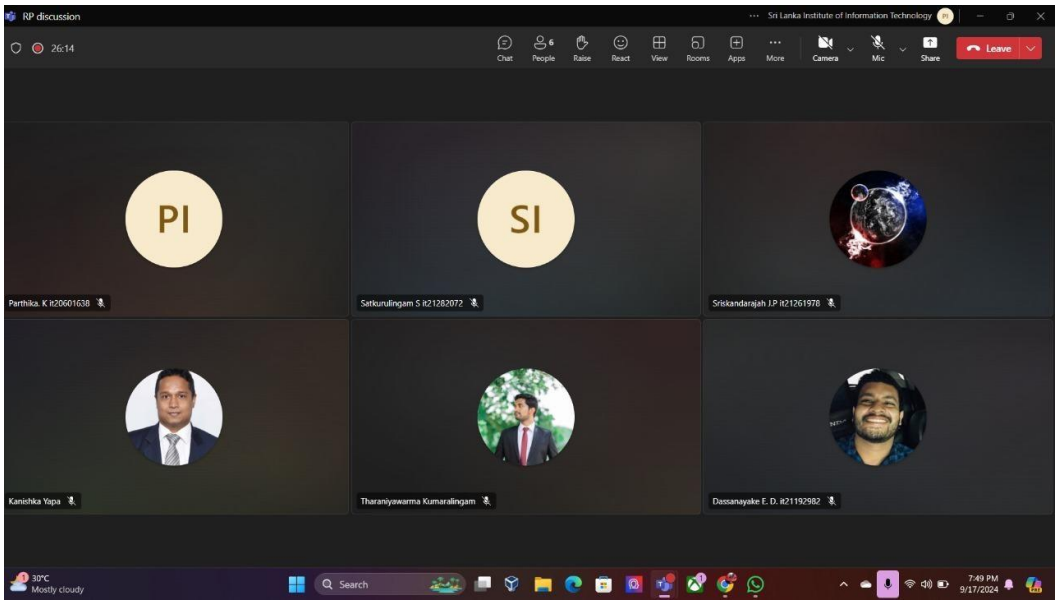
Sri Lanka Institute of Information Technology Sri Lanka

June 2025

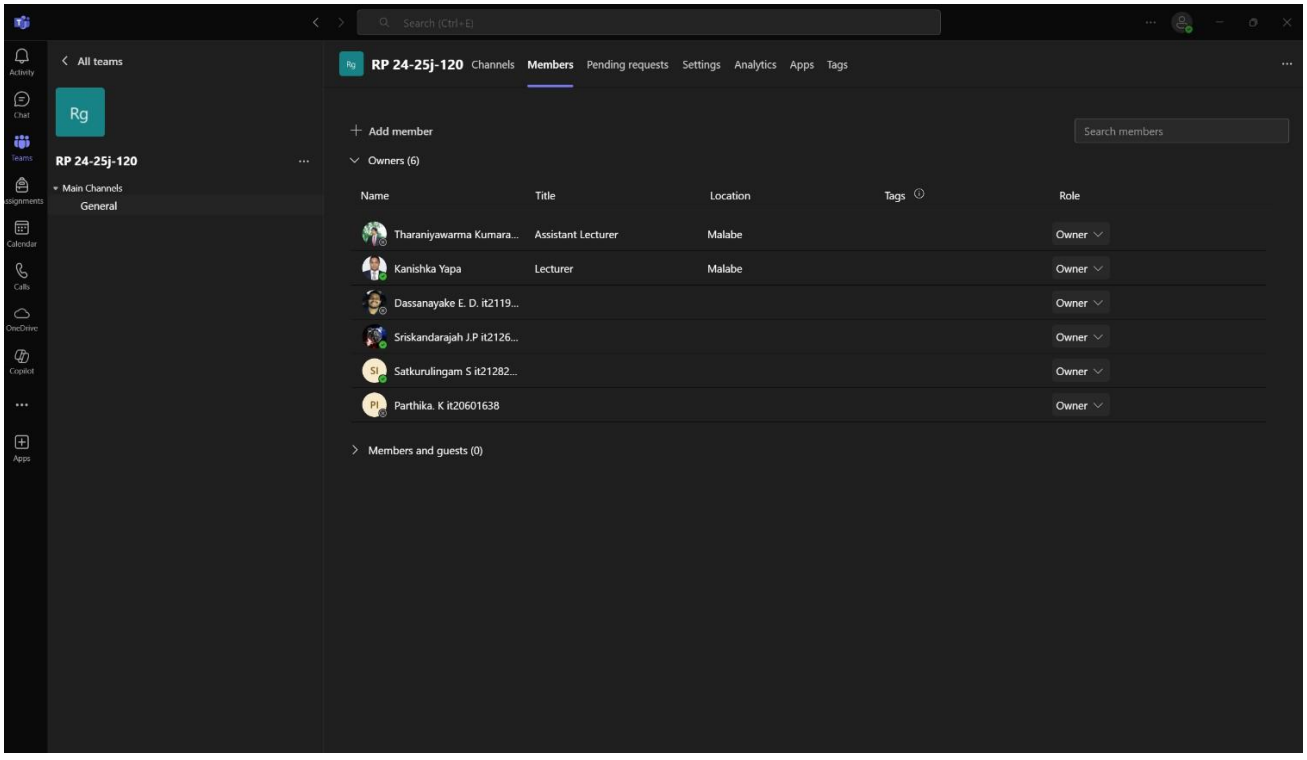
Tasks

► **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.

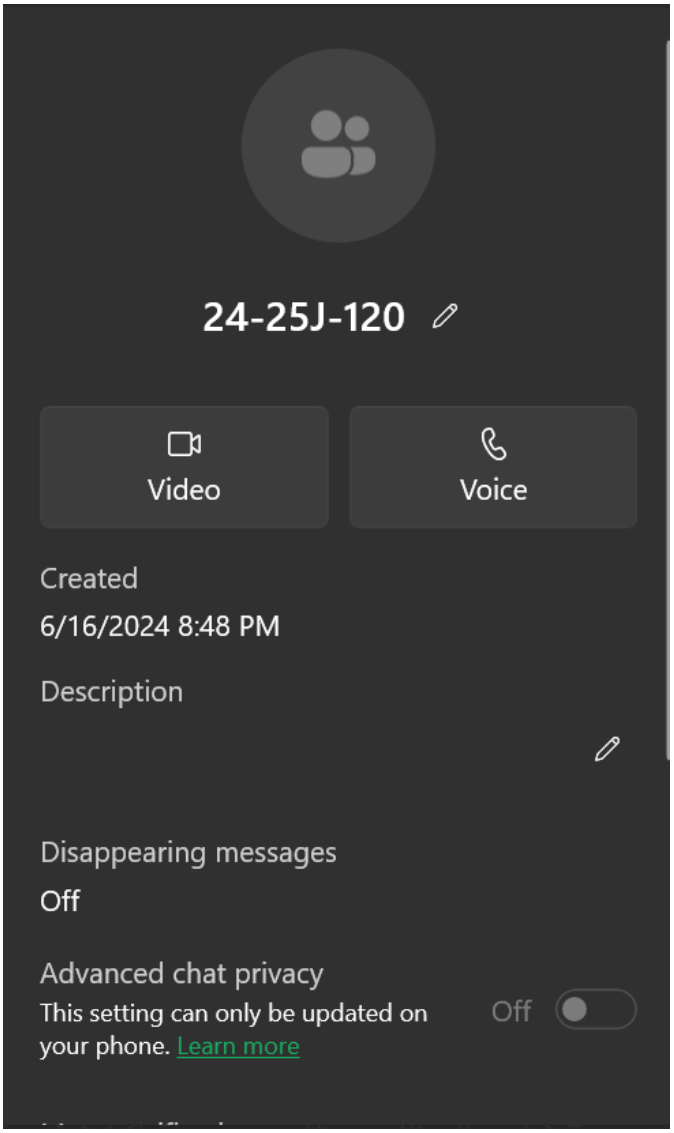


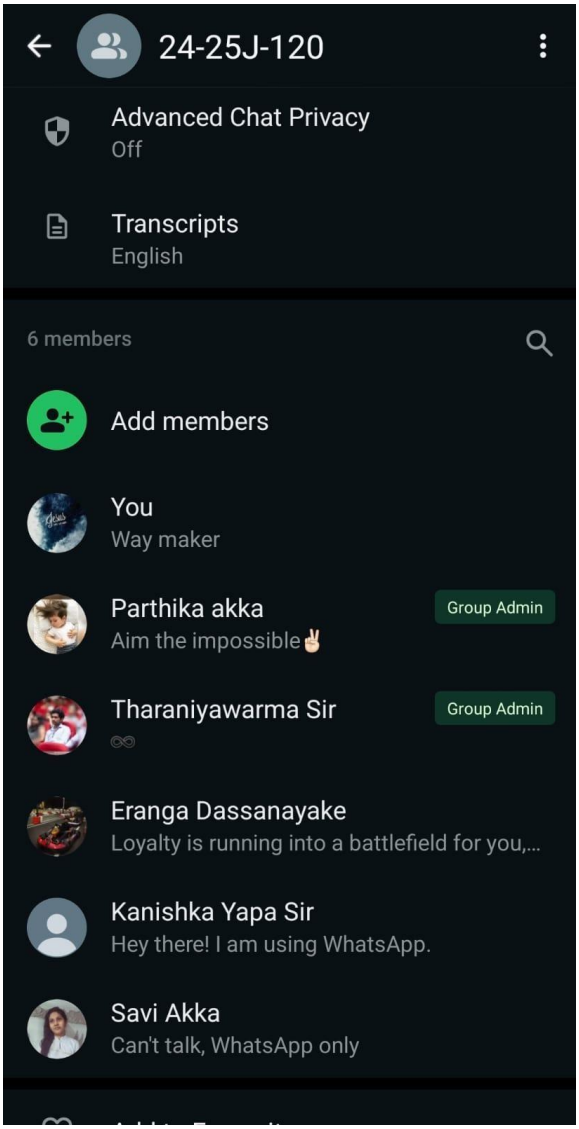
► **Created separate MS Teams channels for Conversation.**



► **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 proposed solution involves the development of an SDN-based Intelligent Intrusion Detection System (IIDS) that integrates machine learning for advanced threat detection and adaptive security policy enforcement. SDN-based Cybersecurity Information Collection and Management involves designing and developing a system for centralized and dynamic collection of cybersecurity-related information using SDN. It will integrate with network devices, sensors, and logs to enable real-time data collection and processing. SDN-based Cybersecurity Action and Response focuses on developing mechanisms to translate SDN outputs into actionable security responses, such as blocking malicious traffic or reconfiguring network paths. It aims to automate response strategies based on predefined security policies. Machine Learning-based Intrusion Detection Engine utilizes advanced machine learning algorithms to detect anomalies and potential threats in real-time. It involves data collection, preprocessing, model development, and integration with the SDN controller for continuous monitoring and analysis of network traffic. Adaptive Security Policy Enforcement Module develops a dynamic policy enforcement engine that adjusts security policies in real-time based on detected threats and network conditions. It ensures seamless integration with the SDN controller and the intrusion detection engine to provide a integrated and responsive security framework.

► **Proposed Machine Learning based Intrusion Detection Engine System**.

Dassanayake E. D.	Develop Machine Learning based Intrusion Detection Engine to find SQL Injection Attacks	**01. Model Development and Training** Develop and train a machine learning model using the preprocessed and feature-engineered dataset. Experiment with various algorithms, such as	**Adaptive Real-Time Detection** Integrating the machine learning model into a real-time environment with continuous feedback and retraining,
		Decision Trees, Random Forest, or Neural Networks, to determine which model performs best in identifying SQL injection attacks. Fine-tune hyperparameters to enhance the model's accuracy and generalization ability. **02. Real-Time SQL Injection Attack Detection** Implement the trained model in a real-time monitoring system capable of detecting SQL injection attacks as they occur. Integrate the model with a live database environment, where it continuously analyzes incoming SQL queries. Ensure the system is optimized for low latency, so it can effectively detect and block malicious queries before they execute. **03. Dataset Collection and Preprocessing** Collect a diverse dataset comprising legitimate SQL queries and SQL injection attack patterns. Preprocess the data to standardize query formats, remove noise, and label each	the system becomes more robust and adaptive, capable of detecting and responding to novel SQL injection techniques as they emerge. **Ensemble Learning for Robust Detection** Combining multiple machine learning models, the intrusion detection engine can achieve higher accuracy and reduced false positives, making it more reliable in real-world scenarios.
		query as benign or malicious. This will serve as the foundation for training and testing the machine learning models. **04. Feature Engineering and Selection** Identify key features that distinguish SQL injection attacks from legitimate queries. This may include query structure, presence of specific SQL keywords, patterns in string manipulations, and abnormal query length. Select the most impactful features through statistical methods or using algorithms like Recursive Feature Elimination (RFE).	

► 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.

2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)
March 17–18, 2021, Amity University Dubai, UAE

A CNN-BiLSTM based Approach for Detection of SQL Injection Attacks

Neel Gandhi
Department of Information and Communication Technology
Pandit Deendayal Energy University
Gandhinagar, Gujarat
neel.gict18@sot.pdpu.ac.in

Jaykumar Patel
Department of Information and Communication Technology
Pandit Deendayal Energy University
Gandhinagar, Gujarat
jay.pict18@sot.pdpu.ac.in

Rajdeepsinh Sisodiya
Department of Information and Communication Technology
Pandit Deendayal Energy University
Gandhinagar, Gujarat
rajdeep.sict18@sot.pdpu.ac.in

Nishant Doshi
Department of Computer Science And Engineering
Pandit Deendayal Energy University
Gandhinagar, Gujarat
nishant.doshi@sot.pdpu.ac.in

Shakti Mishra
Department of Computer Science And Engineering
Pandit Deendayal Energy University
Gandhinagar, Gujarat
shakti.mishra@sot.pdpu.ac.in

***Abstract*—Web application attacks concerned with Structured Query Language Injection(SQLI) have been a major threat in the field of cybersecurity.SQLI attacks majorly lead to leakage of user's data leading to data manipulation,updation and deletion in database management system. Traditional techniques used to prevent SQLI injections include rule-based matching and other related methods that are limited to a few number of SQL injections.Major concern regarding SQLI attacks relates to invention of new malicious SQL queries by hackers to perform SQLI attacks. The problem can be effectively dealt with use of machine learning algorithms for prediction of SQLI attacks.Paper proposes a hybrid CNN-BiLSTM based approach for SQLI attack detection. The proposed CNN-BiLSTM model had significant accuracy of 98% and superior performance compared to other machine learning algorithms. Also,paper presents a comparative study of different types of machine learning algorithms used for the purpose of SQLI attack detection. The study shows the performance of various algorithms based on accuracy, precision, recall, and F1 score with respect to proposed CNN-BiLSTM model in detection of SQL injection attacks.**

***Index Terms*—SQL Injection,CNN-BiLSTM,LSTM, Machine Learning**

I. Introduction

SQL injections are a major threat to web applications according to the statistics obtained from Open web application security project(OWSP) [1], SQL injection contributes a major percentage in vulnerabilities with respect to web applications. Another survey on SQL injection [2] proves that out of 300,000 attacks worldwide about 24.6% of them were SQL injection.Also, SQL injections are performed by attackers to steal, manipulate and update the existing information leading to leakage of user's web data. Detection of malicious SQL queries is difficult due to changing structure of SQL queries developed by hackers. SQL injection attacks are performed by insertion of malicious SQL queries into user input box to harm web application by extracting critical information from back-end database. In the past, SQL injections were detected by using blacklist filtering working on mechanism of regular expressions to blacklist some keywords or illegal statements.The limitation of this method is being restricted to a few number of SQL injections [3]. The trend of an increasing number of SQLI attacks is depicted by Fig1.

Fig. 1: Rise in number of SQLI attack worldwide

Hence, machine learning algorithms are used for the purpose of SQLI detection, in recent years, for the purpose of detecting new malicious SQL queries used by hackers for retrieving critical data. The paper presents hybrid CNN-Bi-LSTM based machine learning model for detection of SQL injection at-

978-1-6654-2921-4/21/$31.00 ©2021 IEEE

2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE) | 978-1-6654-2921-4/21$31.00 ©2021 IEEE | DOI: 10.1109/ICCIKE51210.2021.9410675

Second International Conference on Augmented Intelligence and Sustainable Systems (ICAISS 2023)
IEEE Xplore Part Number : CFP23CB2-ART ; ISBN : 979-8-3503-2579-9

An Analysis of AI-based SQL Injection (SQLi) Attack Detection

B. Brindavathi[1]
Department of Cyber Security in CSE
GMR Institute of Technology
Rajam, India
22341dak01@gmrit.edu.in

Aravind Karrothu[2]
Department of Information Technology
GMR Institute of Technology
Rajam, India
aravind.k@gmrit.edu.in

Chunduru Anilkumar[3]
Department of Information Technology
GMR Institute of Technology
Rajam, India
anilkumar.ch@gmrit.edu.in

***Abstract*—The SQL injection attack is a highly perilous vulnerability in the digital realm, especially for web pages, as recognized by the Open Web Application Security Project (OWASP) ranking. It is a type of code injection attacks. This kind of attacks basically breaches the virtual portion of the databases. Many web applications accept to store the user's private information (e.g., login credentials, credit card and other account details etc.,) in the database over the internet. The detection of SQL injection attack is going to be a tough task for everyone because of an attacker can deploy various new type of SQL injections in day-to-day life. There are many ways to create/detect SQLi attacks by using open-source tools such as Netsparker, SQLMap, jSQL Injection, Burp Suite, BBQSQL, Nessus, and etc., So, the researchers have required an innovative and never-before seen mechanisms for detecting these types of attacks. But still, there is a lack of knowledge for input validation. So, various authors have decided to improve the SQLi attack detection by using various artificial intellectual mechanisms like machine learning (ML) and deep learning (DL). Both ML and DL algorithms are solved many types of SQL injection attacks by using various classification or regression techniques. This study will focus on various recent methodologies that how to detect SQLi attacks along with which mechanism will provide better performance among all the existing working models.**

***Keywords*—Structured Query Language (SQL), SQL Injection (SQLi) attacks, Deep Learning (DL), Machine Learning (ML) and Open Web Application Security Project (OWASP)**

I. Introduction

In digital world, many of the organizations have used web-based applications for accessing the data over the Internet to perform different types of online communications. In the realm of web applications and websites, user-entered data during transactions is typically stored in a database. These databases, often relational in nature, are interacted with using a language known as Structured Query Language (SQL). Any web application accepts the user's data and produces SQL statements as output. The OWASP Top 10 is a widely recognized document aimed at raising awareness among developers and web application security professionals. It serves as a standard reference for identifying and understanding the most significant security risks that web applications face. The OWSP team released every year top 10 threat as per their analysis and the latest version have released on sept. 2021. The Figure 2 represents the severity of web applications related threats as per analysis from 2017 to 2021 [14]. In 2017 the injection attacks were in first position in terms of severity and then comes to 3rd position i.e., Injection attacks still in 3rd position due to high risk, so there is a need to prevent those risks with effective cost.

Fig.1. WEB Application architecture

Injection flaws, including SQL, NoSQL, OS, and LDAP injections, manifest when untrusted data is incorporated into a command or query sent to an interpreter. This enables malicious actors to exploit the interpreter and execute unintended commands or gain unauthorized access to data. According to the OWASP report analysis, the SQL injection attack ranks as one of the most prevalent web-based attacks and the third most significant web application security risk in 2021. This attacks mainly injects malicious SQL code into a web application, allowing the attacker to view or modify a database. An SQL injection (SQLi) attack, if successful, can lead to the unauthorized retrieval of sensitive data, such as passwords, credit card information, or personal user data. The below figure has explained the difference between malicious and genuine queries how they entered in the application.

Fig. 2. Normal-Malicious web application scenario

979-8-3503-2579-9/23/$31.00 ©2023 IEEE

2023 Second International Conference on Augmented Intelligence and Sustainable Systems (ICAISS) | 979-8-3503-2579-9/23$31.00 ©2023 IEEE | DOI: 10.1109/ICAISS58487.2023.10250505

2022 IEEE Global Conference on Computing, Power and Communication Technologies (GlobConPT).
India Habitat Centre, Lodhi Road, New Delhi, India. Sep 23-25, 2022

Detection of Cyber Attacks: XSS, SQLI, Phishing Attacks and Detecting Intrusion Using Machine Learning Algorithms

Aashutosh Bhardwaj[1]
Information Technology
International Institute of Information Technology,
Pune, India
aashutosh2899@gmail.com

Saheb Singh Chandok[2]
Information Technology
International Institute of Information Technology,
Pune, India
sahebchandok2001@gmail.com

Aniket Bagnawar[3]
Information Technology
International Institute of Information Technology,
Pune, India
bagnawaraniket66@gmail.com

Shubham Mishra[4]
Information Technology
International Institute of Information Technology,
Pune, India
shubhammishra4579@gmail.com

Dr. Deepak Uplaonkar[5]
Information Technology
International Institute of Information Technology,
Pune, India
uplaonkar@gmail.com

***Abstract*— Cyber-crime is spreading throughout the world, exploiting any type of vulnerability in the cloud computing platform. Ethical hackers are primarily concerned in identifying flaws and recommending mitigation measures. In the cyber security world, there is a pressing need for the development of effective techniques. The majority of IDS techniques used today are incapable of dealing with the dynamic and complex nature of cyber-attacks on computer networks. In cyber security, machine learning approaches have been utilized to handle important concerns such as intrusion detection, XSS, SQLI, and phishing detection. Machine learning approaches have been employed in order to detect the issues such as XSS, SQLI, Phishing attacks etc. In this study XSS attack is detected using CNN approach, SQLI attack is detected using Logistic Regression approach, phishing is detected using SVM approach. In addition to the above specified attacks: DTC, BNB, KNN approaches are employed to detect the intrusion in the system. As a result, CNN approach yields 98.59% accuracy for detecting XSS attacks, Logistic Regression approach yields 92.85% accuracy for SQLI, SVM approach yields 85.62% accuracy for phishing attacks. Approaches like DTC, BNB, KNN yields an accuracy of 99.47%, 90.67% and 99.16% respectively for detecting intrusions.**

***Keywords*—SQL Injection (SQLI), Cross-Site Scripting (XSS), Phishing Attacks, and Intrusion Detection Attack (IDS), Convolutional Neural Network (CNN), K Nearest Neighbor (KNN), Bernoulli Naïve Bayes (BNB), Support Vector Machine (SVM), Machine Learning (ML), Decision Tree Classifier (DTC) etc.**

I. Introduction

Physical things are now connected to cyber networks in the current era of information and communication technology, and these connections are referred to as "cyber-that aims to overwhelm the network and cause it to malfunction. Intelligence gathering efforts can be seen as a precursor to an imminent attack, as attackers usually precede the attack by scouting the target. That is, it forms the first phase of the attack. Therefore, the term exploit includes both reconnaissance and offensive activities, and the difference between the two is not always clear. Computer networks and the Internet have taken on significant importance in both our daily lives and our organizations. Malicious activity is becoming more and more common as we become more dependent on computers and communication networks. Network attacks are a major problem in today's communications environment. To ensure the reliable functionality of your network and the security of your user information, you need to monitor and analyze network traffic to detect malicious activity and attacks. Network threats have recently been detected using machine learning techniques. Network traffic can be analyzed to find patterns and similarities using machine learning algorithms. There is no requirement for manual analysis to derive attack patterns, unlike signature-based approaches. You may automatically create predictive models to find network assaults by using machine learning algorithms. Particularly as a result of the development of fifth-generation networks and artificial intelligence technologies in cybersecurity, new dangers and challenges to wireless communication systems have emerged. This system offers a summary of attack detection strategies that make use of deep learning methods. It introduces different applications that use deep learning structures and explains the fundamental network security and attack detection challenges. The focus is on attack detection systems based on different types of architectures, including: B. Auto encoders, hostile generation networks, recurrent neural networks, convolutional neural networks based on the

Computing, Power and Communication Technologies (GlobConPT) | 978-1-6654-9365-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBCONPT57482.2022.9938367

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)

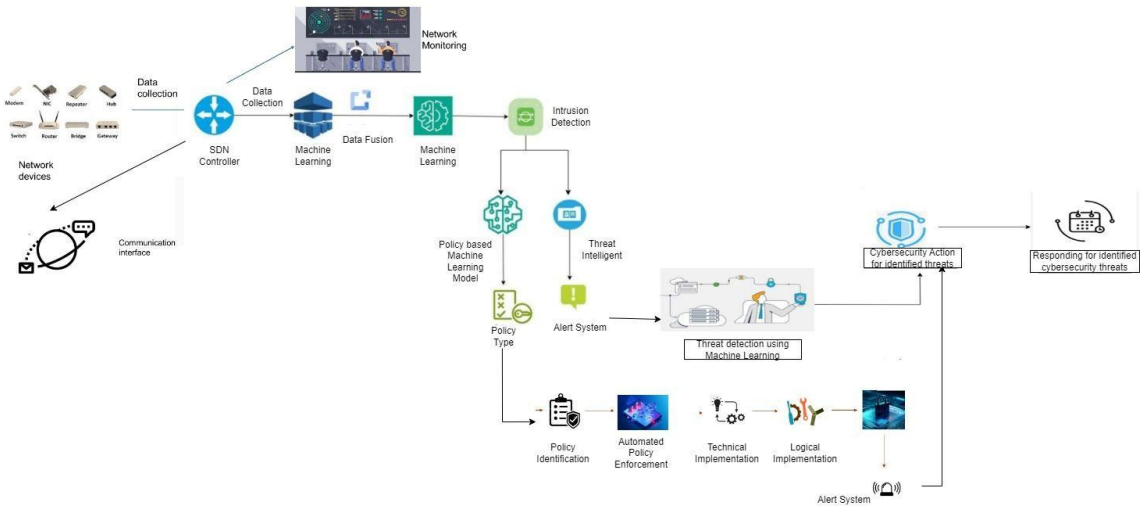
3. Research area the project belongs to

Cyber Security (CS)

4. If a continuation of a previous project:

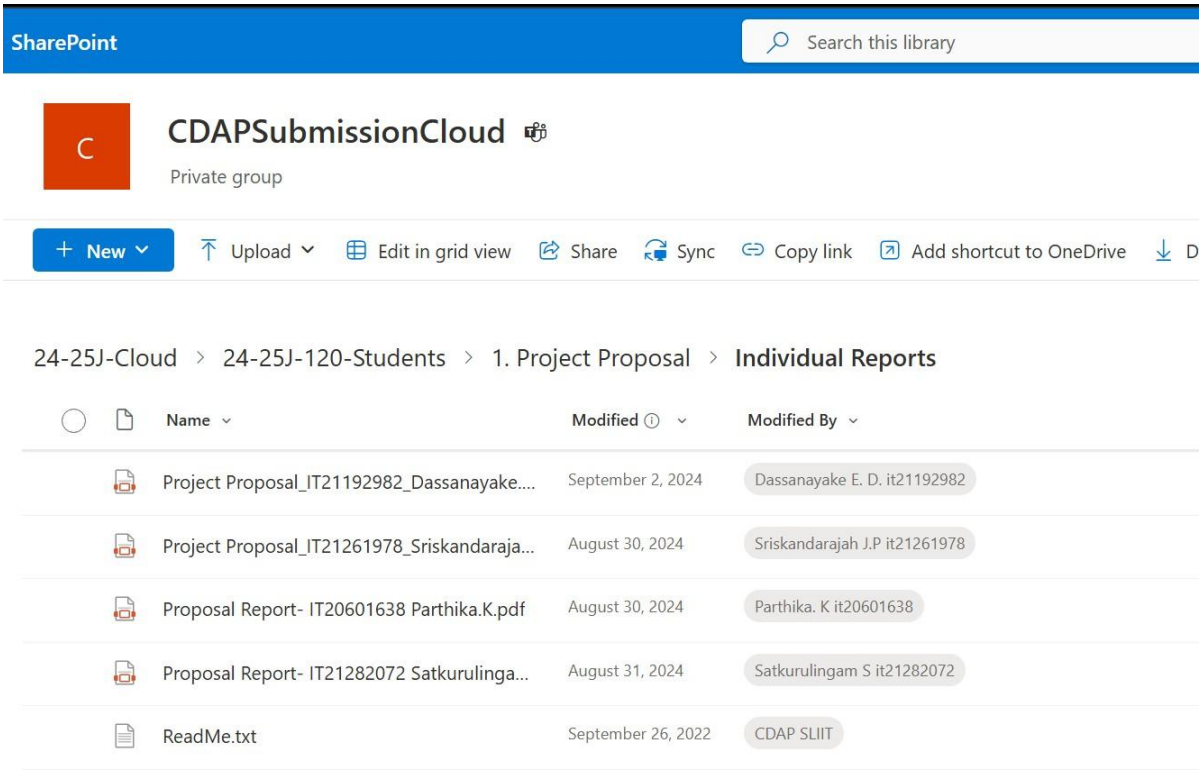
► Complete Task and Conversation Highlights

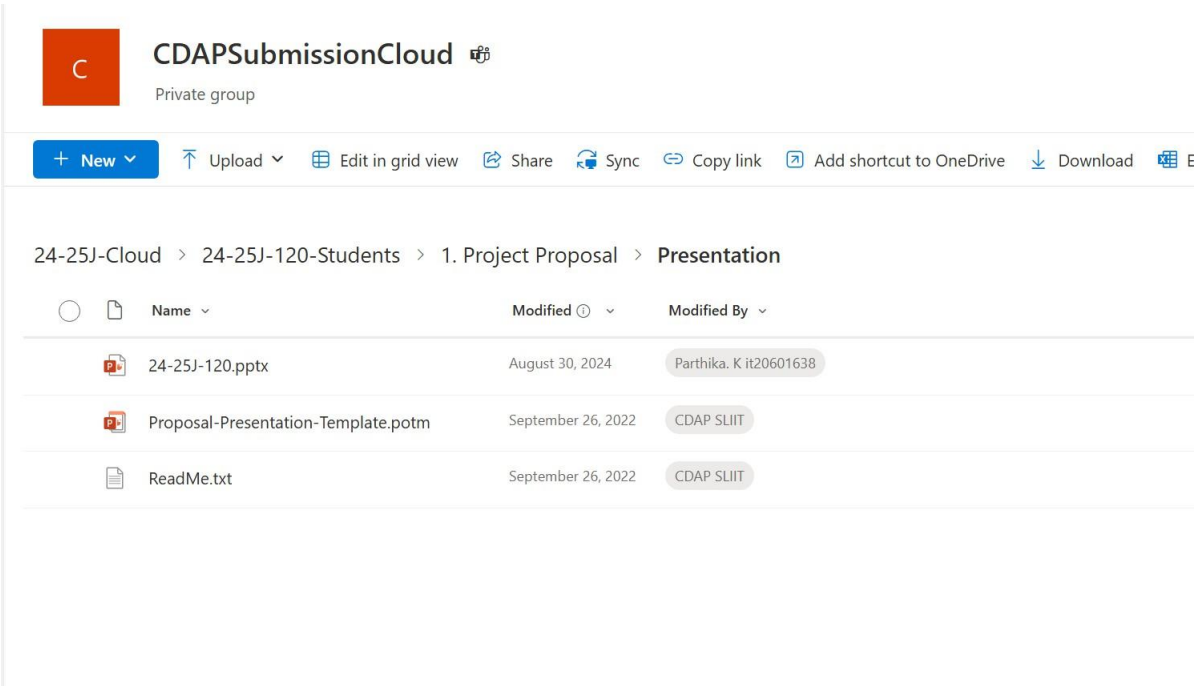
- Dividing the software components.
- Doing a thorough background investigation on each component.
- Creating the system architecture diagram of the proposed system.
- Discussing architecture with the Supervisor and Co-supervisor physically meeting.

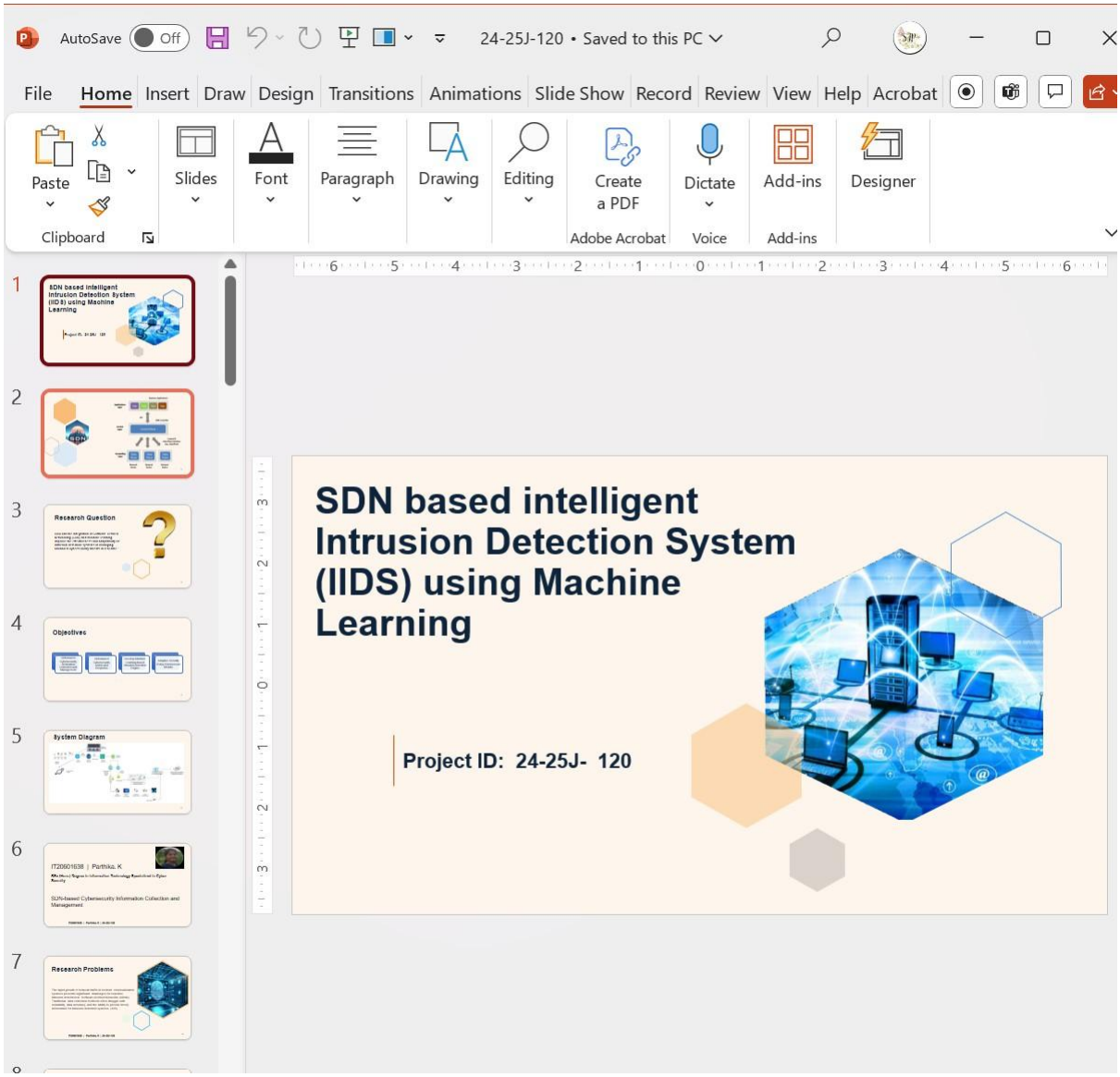


► **Complete Task and Conversation Highlights.**

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

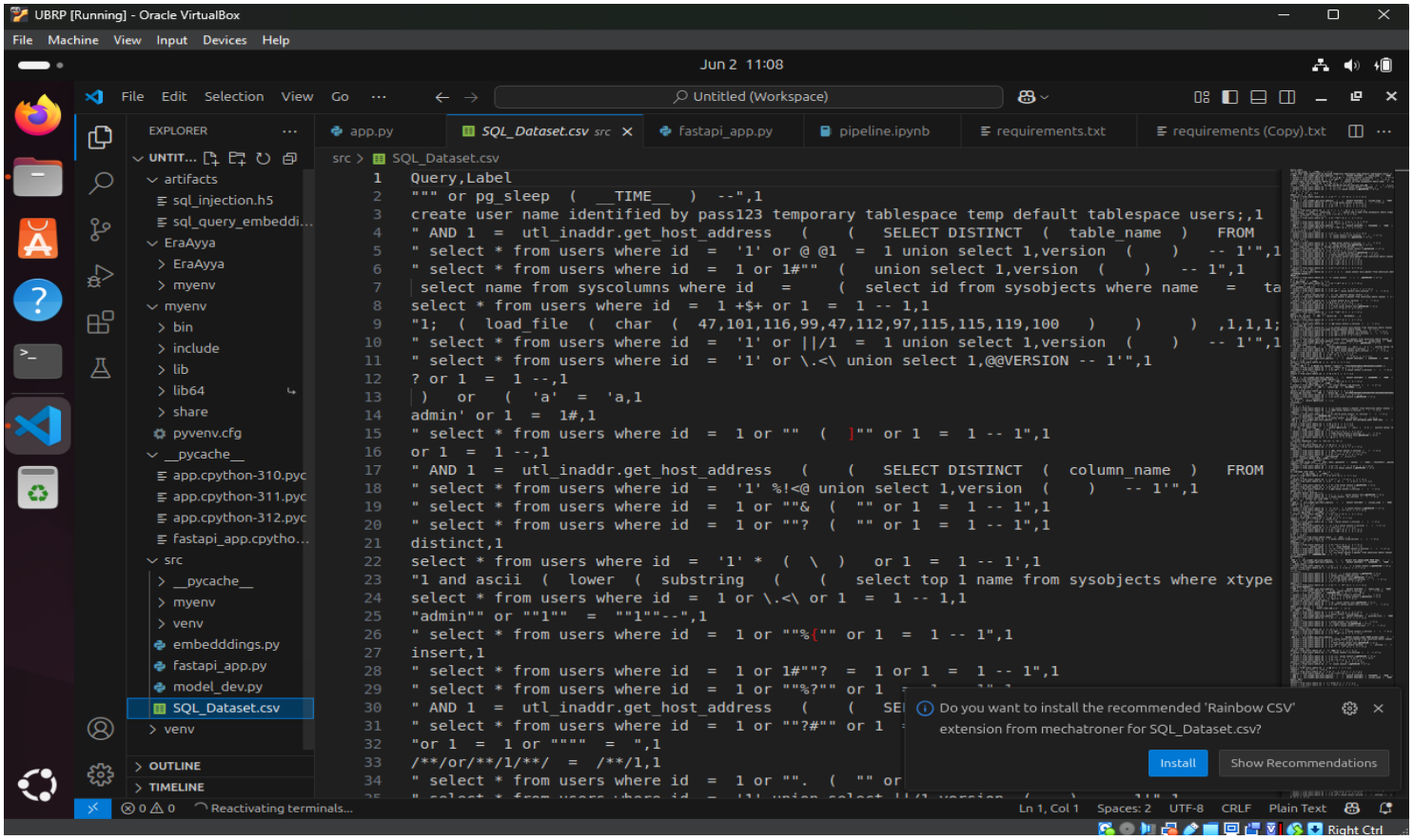


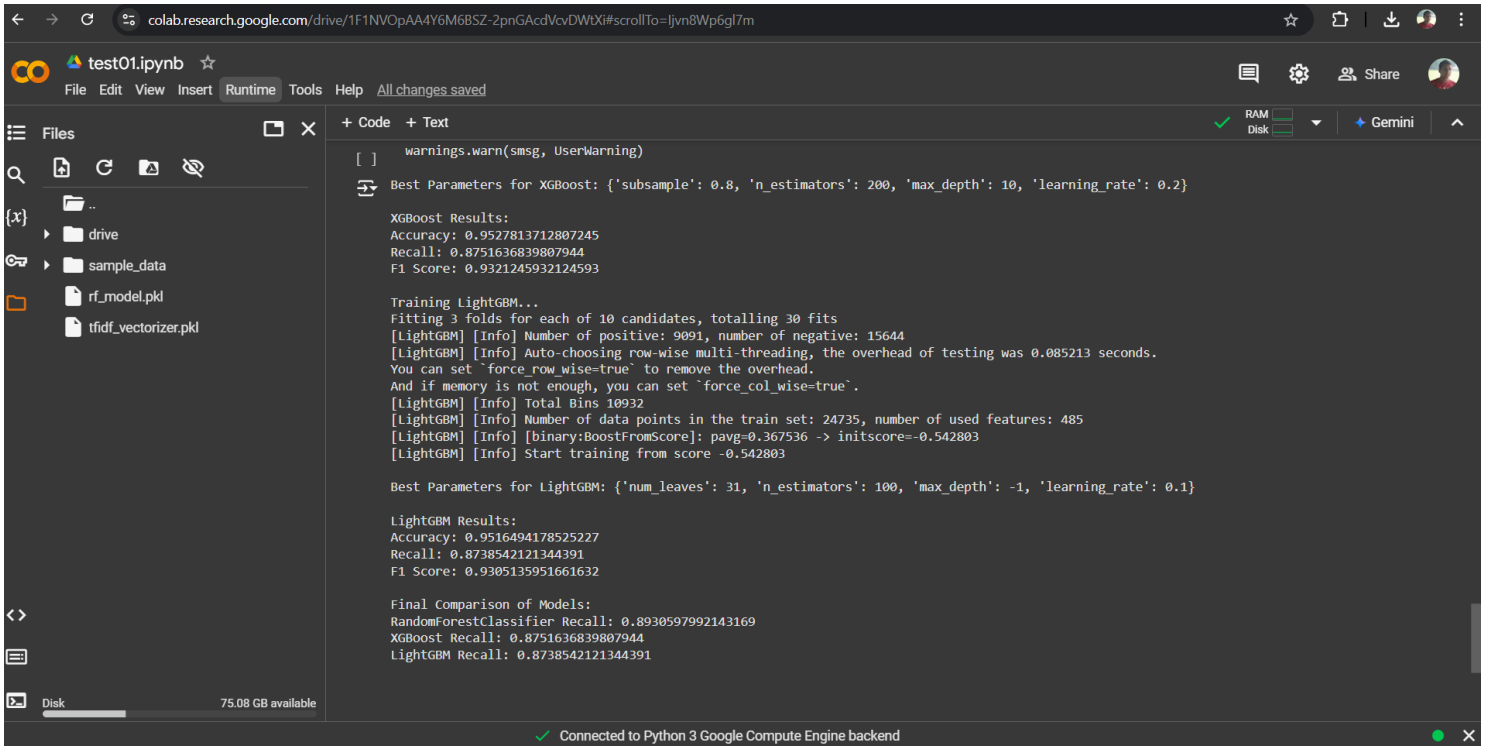



AutoSave Off
24-25J-120 • Saved to this PC
File Home Insert Draw Design Transitions Animations Slide Show Record Review View Help Acrobat
Paste
Clipboard
Slides
Font
Paragraph
Drawing
Editing
Create a PDF
Adobe Acrobat
Dictate
Voice
Add-ins
Add-ins
Designer
SDN based intelligent Intrusion Detection System (IIDS) using Machine Learning
Project ID: 24-25J- 120

► 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.





+ Code + Text

```
print("Recall:", recall_score(y_test, y_pred))
print("F1 Score:", f1_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Step 6: Save the Model for Integration
with open("rf_model.pkl", "wb") as file:
    pickle.dump(model, file)

# Save the TF-IDF Vectorizer for later use
with open("tfidf_vectorizer.pkl", "wb") as file:
    pickle.dump(tfidf, file)

print("Model and vectorizer saved successfully!")
```

```
Accuracy: 0.9568240620957309
Precision: 0.9903100775193798
Recall: 0.8921868179834134
F1 Score: 0.9386911595866819

Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.99      0.97      3893
           1       0.99      0.89      0.94      2291

    accuracy                           0.96      6184
   macro avg       0.97      0.94      0.95      6184
weighted avg       0.96      0.96      0.96      6184

Model and vectorizer saved successfully!
```

colab.research.google.com/drive/1F1NVOpAA4Y6M6BSZ-2pnGAcdVcvDWtXi#scrollTo=wNLu8_FvBw_L

test01.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Share

Files

drive
MyDrive
Certificates Scans
Colab Notebooks
EY Ernst & Young
SLIIT RP
Modified_SQL_Data...
SLIIT, Sampath
19950426.pdf
Advent of Cyber 2023 T...
Application Form Online...
Application Form Online...
CISCO Dr.Lakmal Rupa...
Certificate of Medical Ex...
Cover Letter Eranga Das...

Disk 75.08 GB available

+ Code + Text

RAM Disk Gemini

```
print("dataset shape:",data.shape )
```

```
dataset shape: (30919, 2)
```

```
# Step 1: Load Dataset
data = pd.read_csv('/content/drive/MyDrive/SLIIT RP/Modified_SQL_Dataset.csv')
```

```
print(data.head())
```

```
                                               Query  Label
0                 " or pg_sleep  (  __TIME__  ) --      1
1  create user name identified by pass123 tempora...      1
2   AND 1  =  utl_inaddr.get_host_address  (    ...      1
3   select * from users where id  =  '1' or @ @1 ...      1
4   select * from users where id  =  1 or 1#"  ( ...      1
```

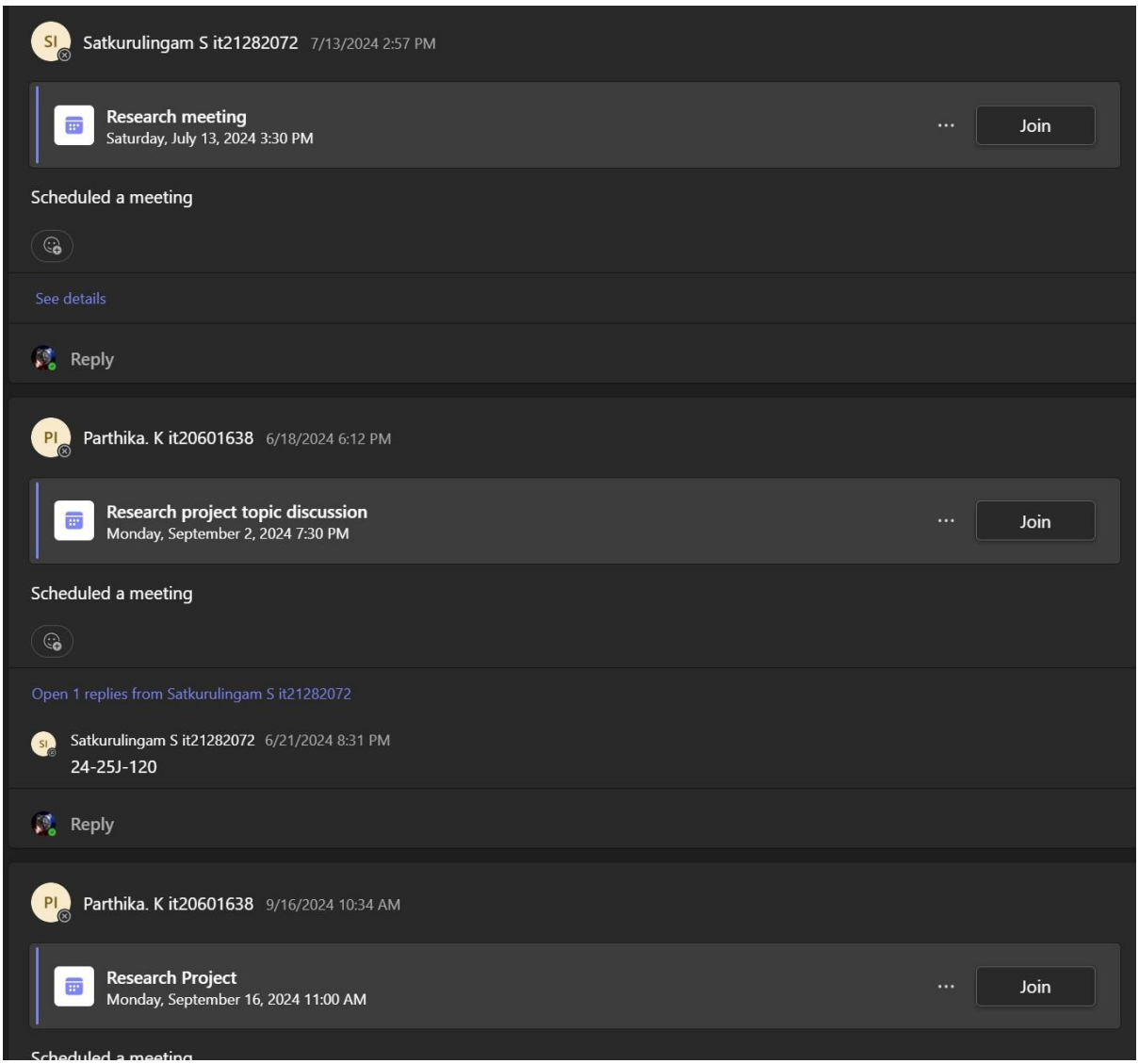
```
print(data.info())
```

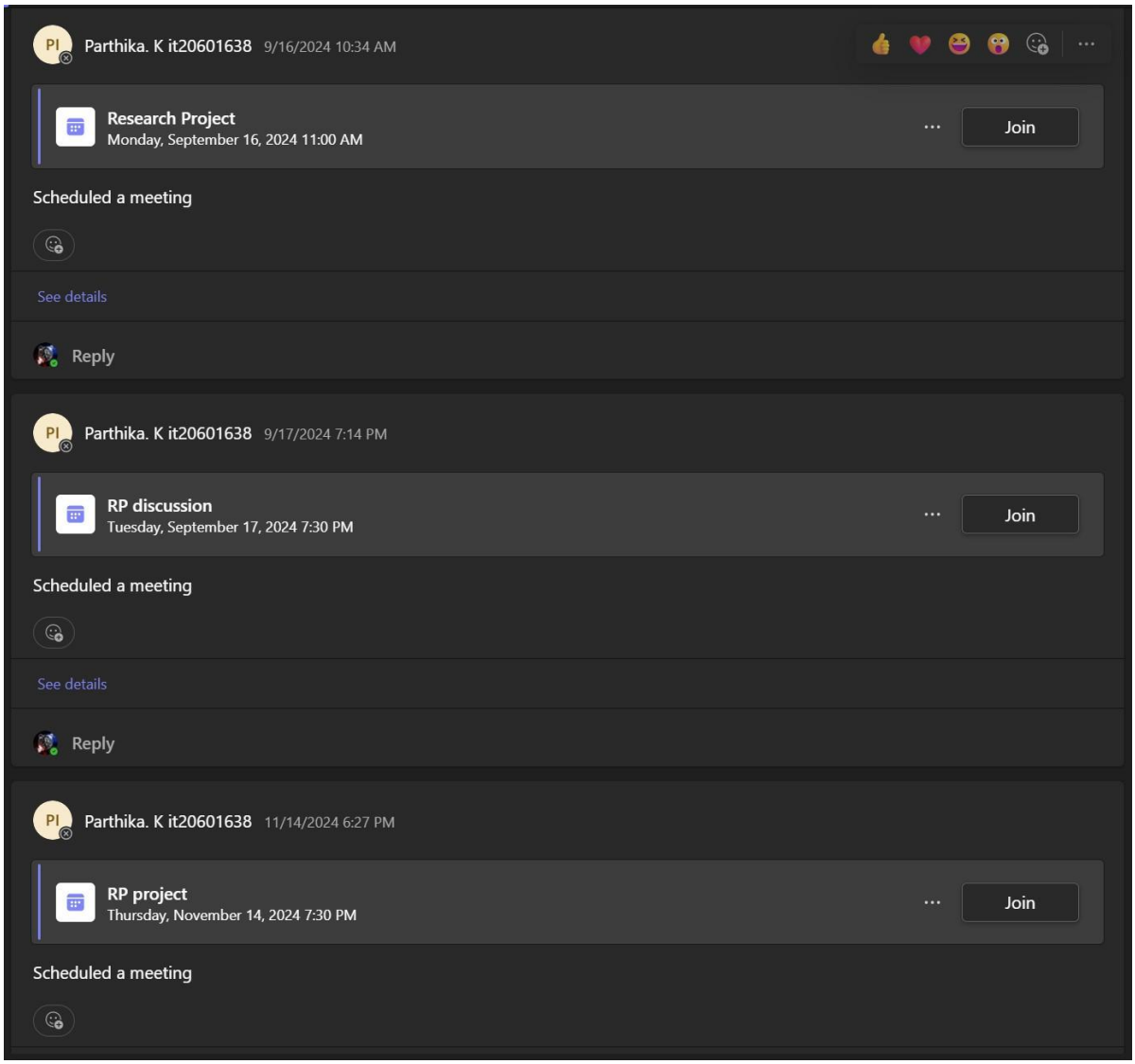
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30919 entries, 0 to 30918
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Query   30919 non-null  object
 1   Label   30919 non-null  int64
dtypes: int64(1), object(1)
memory usage: 483.2+ KB
None
```

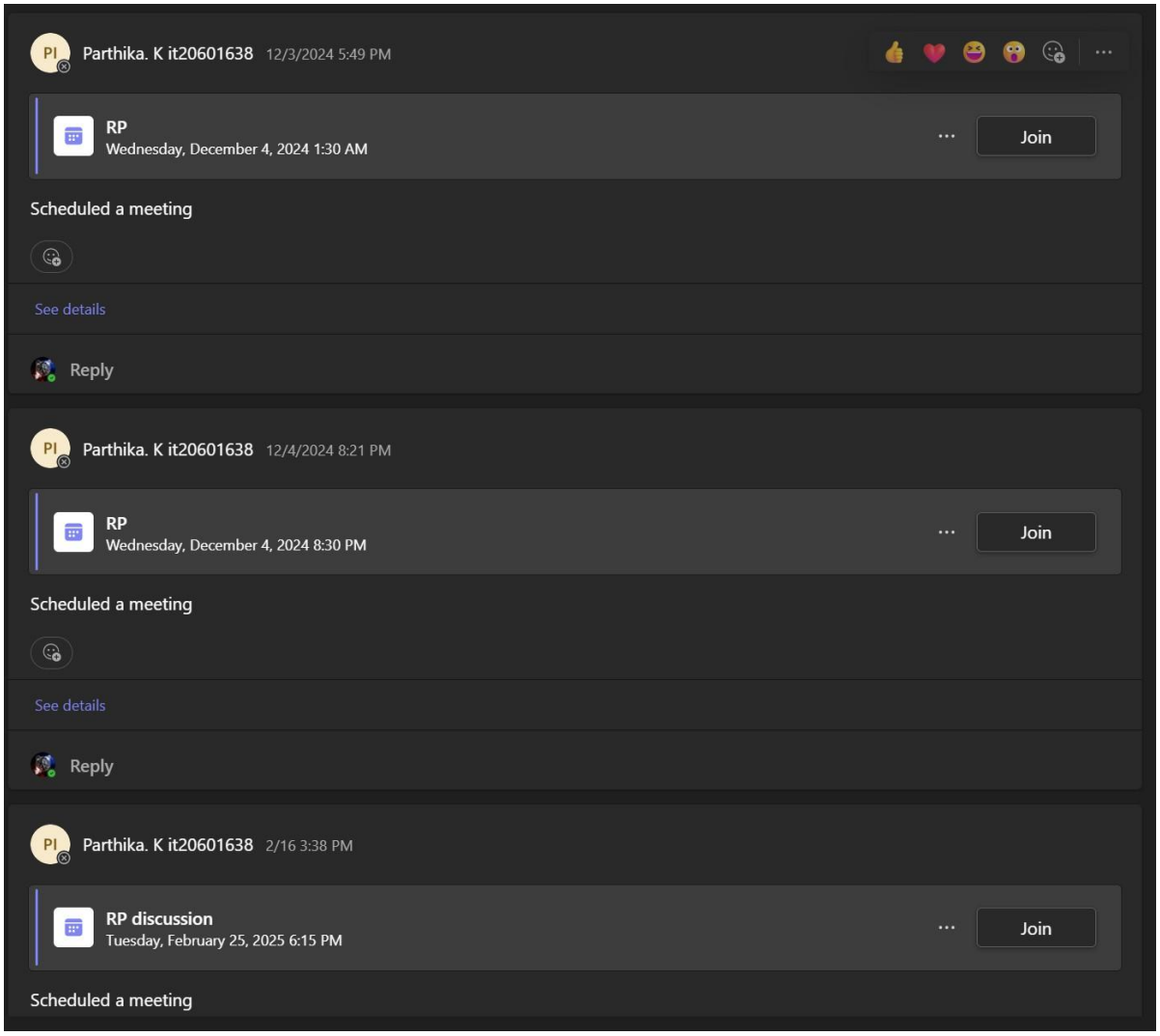
0s completed at 10:28 AM

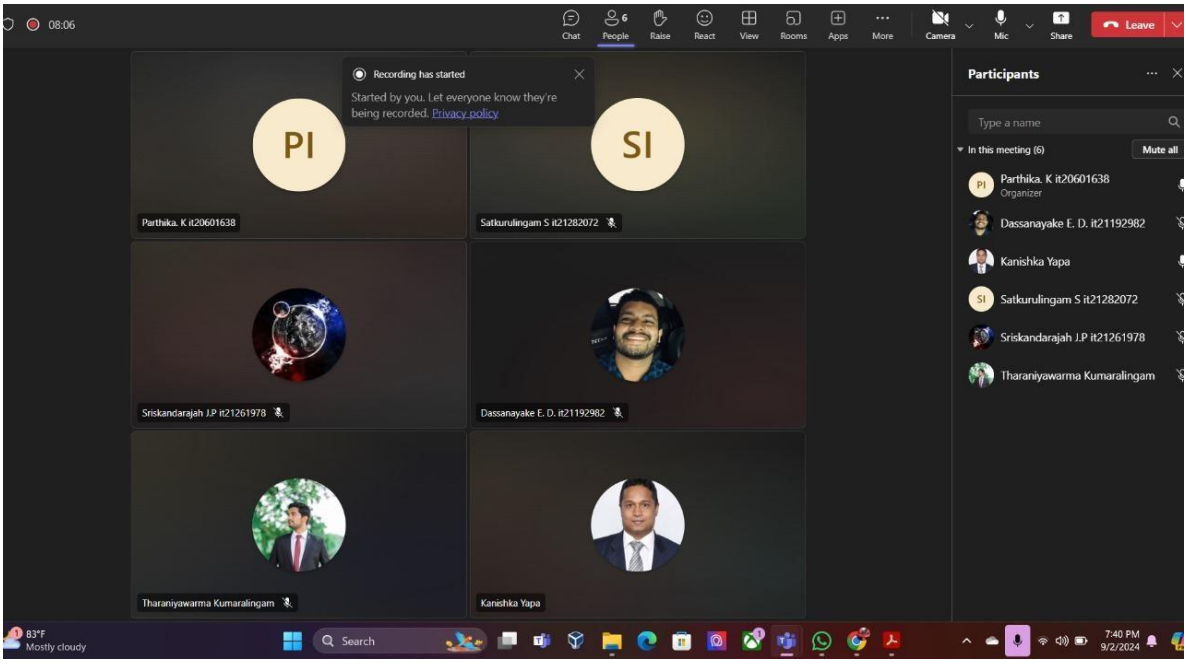
► Complete Tasks and Conversation Highlights

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



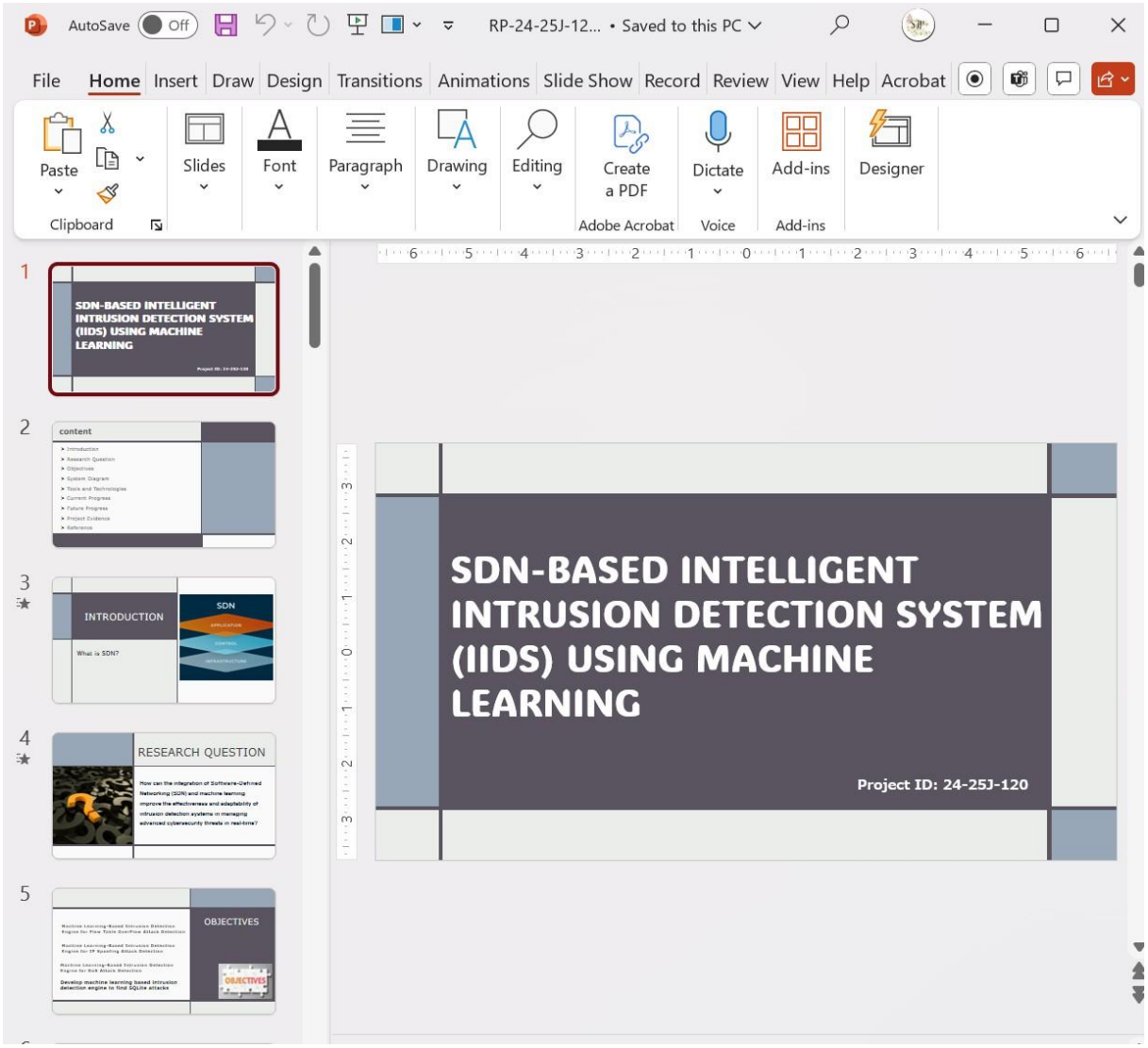

PI
Parthika. K it20601638 9/16/2024 10:34 AM
Research Project
Monday, September 16, 2024 11:00 AM
Join
Scheduled a meeting
See details
Reply
PI
Parthika. K it20601638 9/17/2024 7:14 PM
RP discussion
Tuesday, September 17, 2024 7:30 PM
Join
Scheduled a meeting
See details
Reply
PI
Parthika. K it20601638 11/14/2024 6:27 PM
RP project
Thursday, November 14, 2024 7:30 PM
Join
Scheduled a meeting

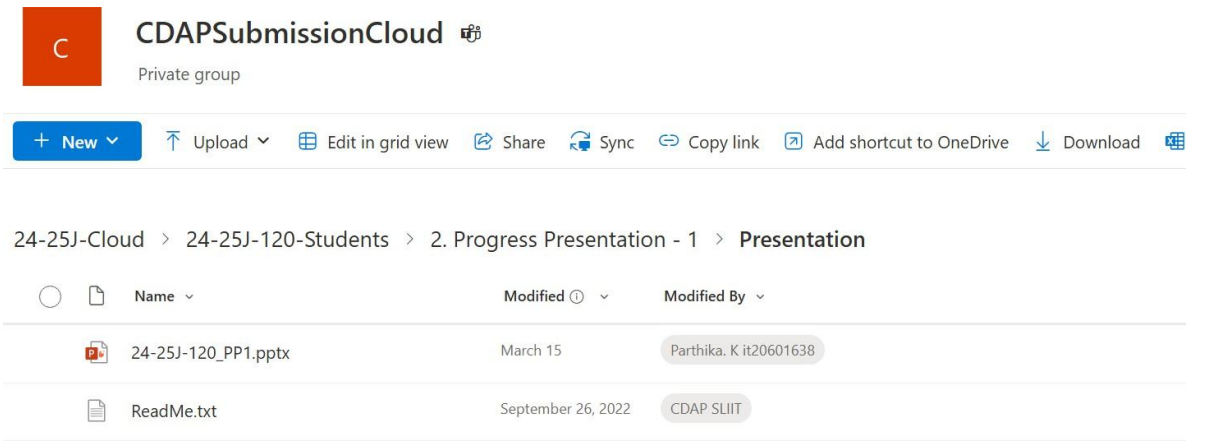

PI
Parthika. K it20601638 12/3/2024 5:49 PM
RP
Wednesday, December 4, 2024 1:30 AM
Join
Scheduled a meeting
See details
Reply
PI
Parthika. K it20601638 12/4/2024 8:21 PM
RP
Wednesday, December 4, 2024 8:30 PM
Join
Scheduled a meeting
See details
Reply
PI
Parthika. K it20601638 2/16 3:38 PM
RP discussion
Tuesday, February 25, 2025 6:15 PM
Join
Scheduled a meeting


08:06
Chat
People
Raise
React
View
Rooms
Apps
More
Camera
Mic
Share
Leave
Recording has started
Started by you. Let everyone know they're being recorded. Privacy policy
PI
SI
Parthika. K it20601638
Satkurulingam S it21282072
Sriskandarajah J.P it21261978
Dassanayake E. D. it21192982
Tharaniyawarma Kumaralingam
Kanishka Yapa
Participants
Type a name
In this meeting (6)
Mute all
Parthika. K it20601638
Organizer
Dassanayake E. D. it21192982
Kanishka Yapa
Satkurulingam S it21282072
Sriskandarajah J.P it21261978
Tharaniyawarma Kumaralingam
83°F
Mostly cloudy
Search
7:40 PM
9/2/2024

► Completed Tasks and Conversation Highlights

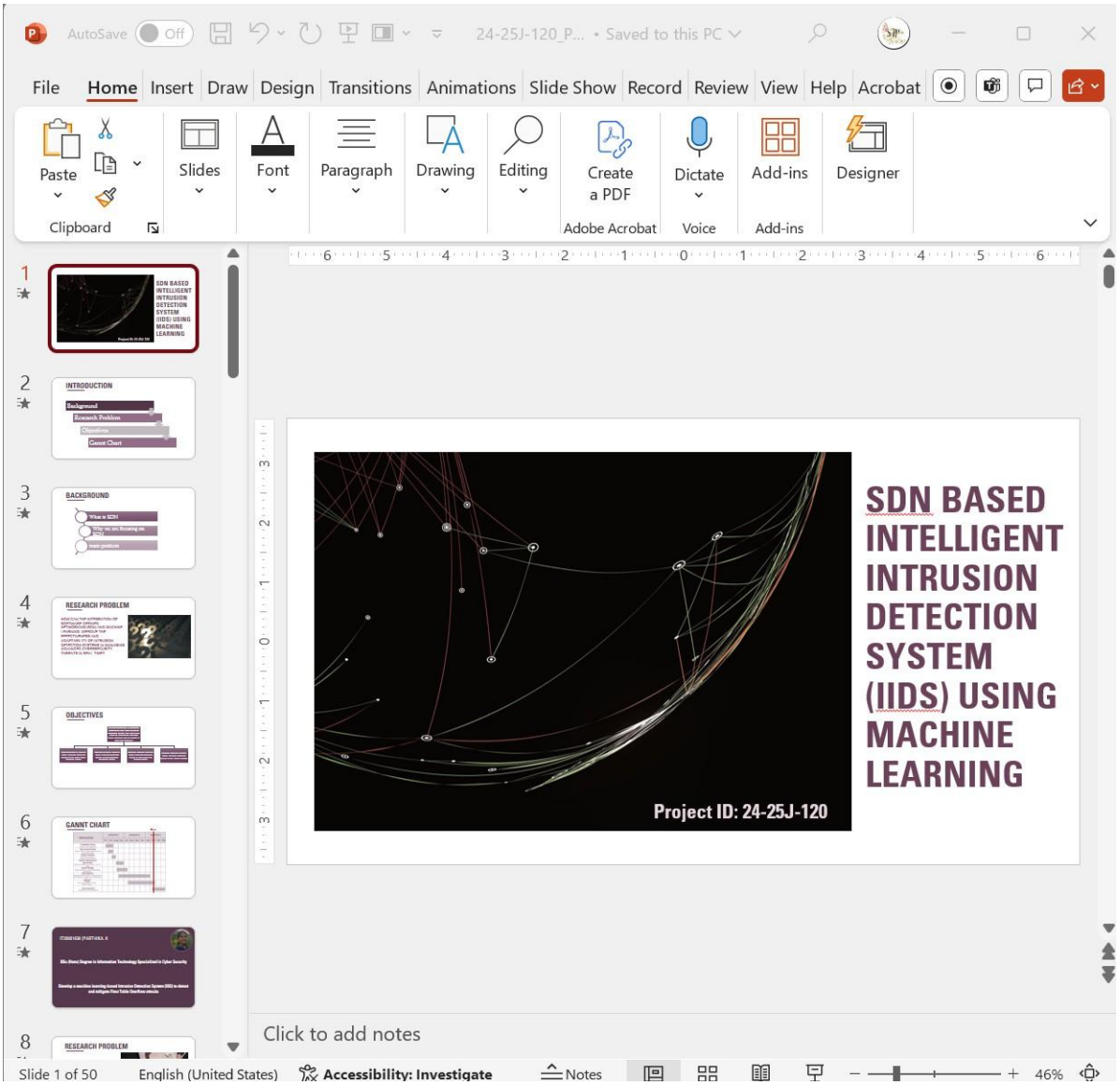
- Prepare for Progress Presentation 1 (PP1).
- Creating the presentation.
- Finalizing the Projects.
- Communication with the supervisor after finalizing the project.

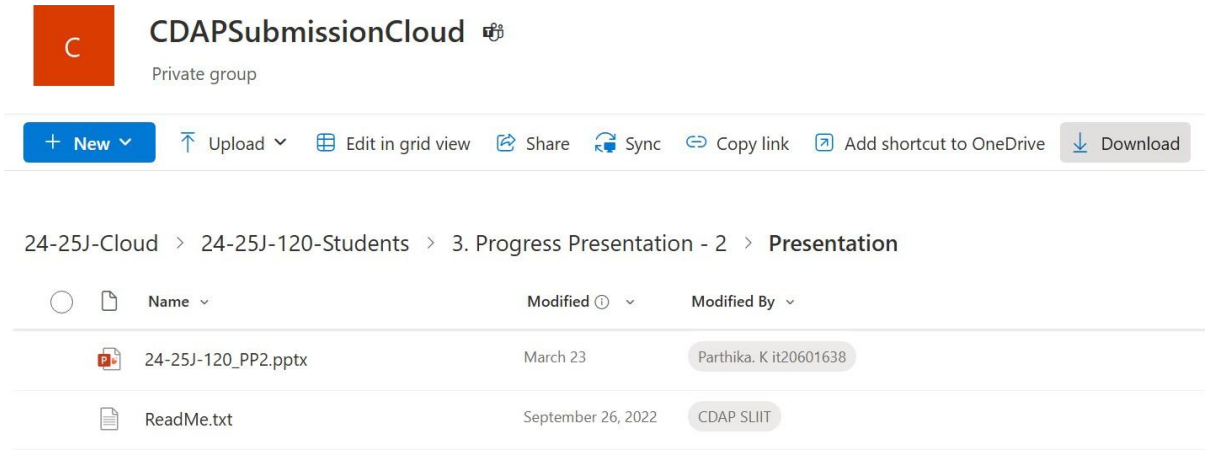




► **Completed Task and Conversation Highlights**

- 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.

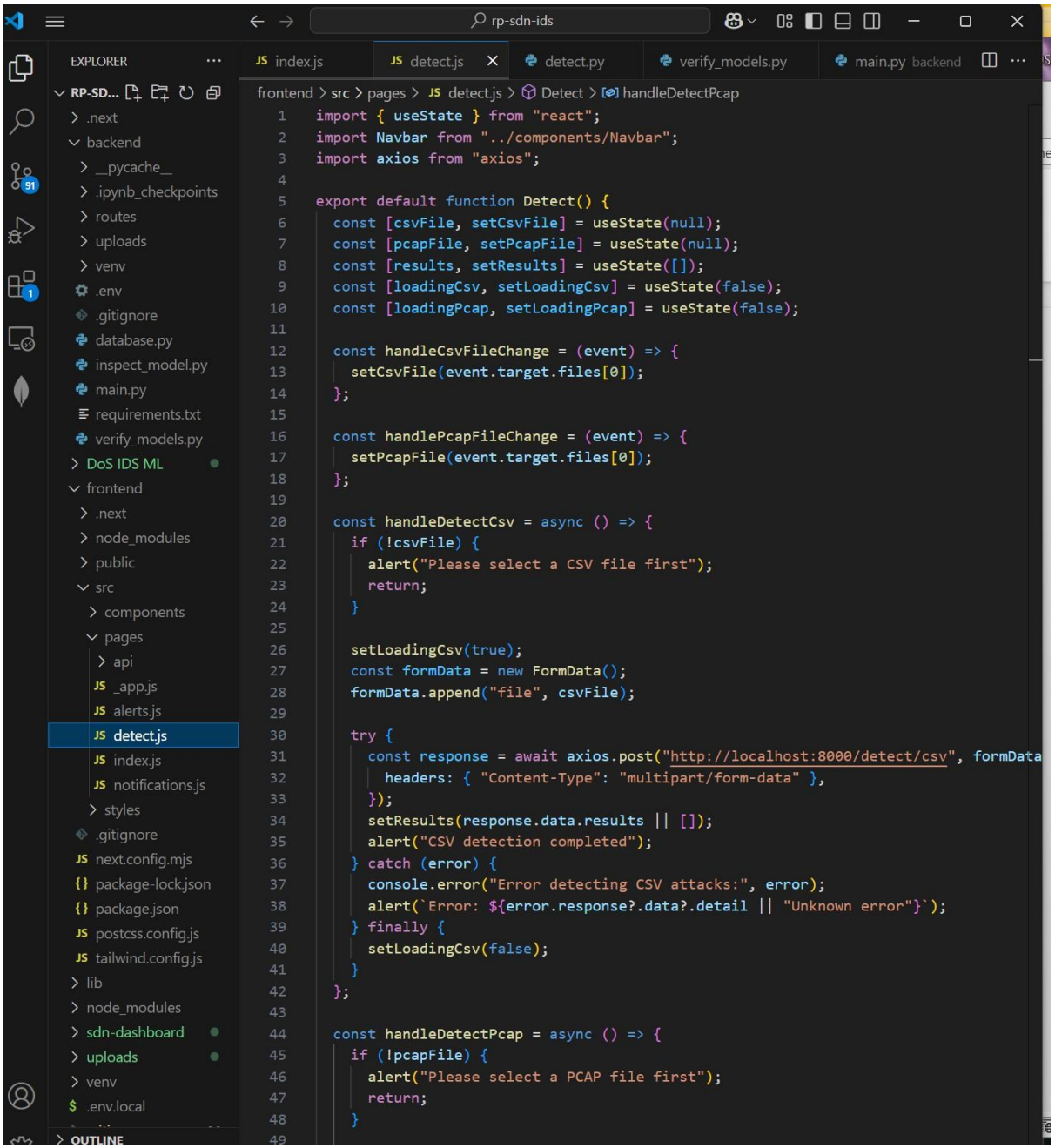
Highlight the achievements of local and international researchers in the field of computing.

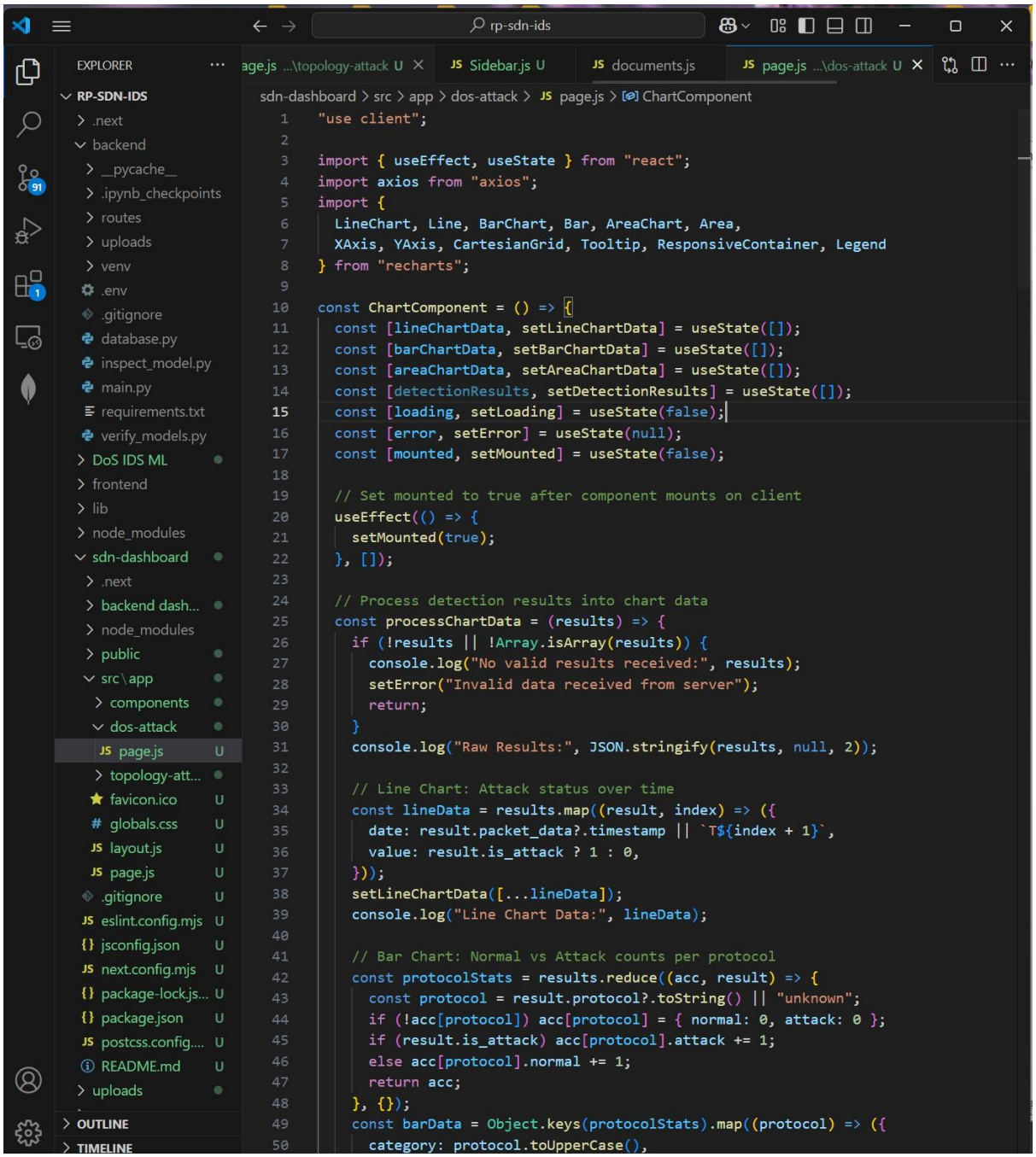
Foster groundbreaking innovations in computing by integrating Artificial Intelligence, Quantum Computing, and emerging technologies to solve complex challenges, drive cross-disciplinary advancements, and shape the future of industries worldwide.

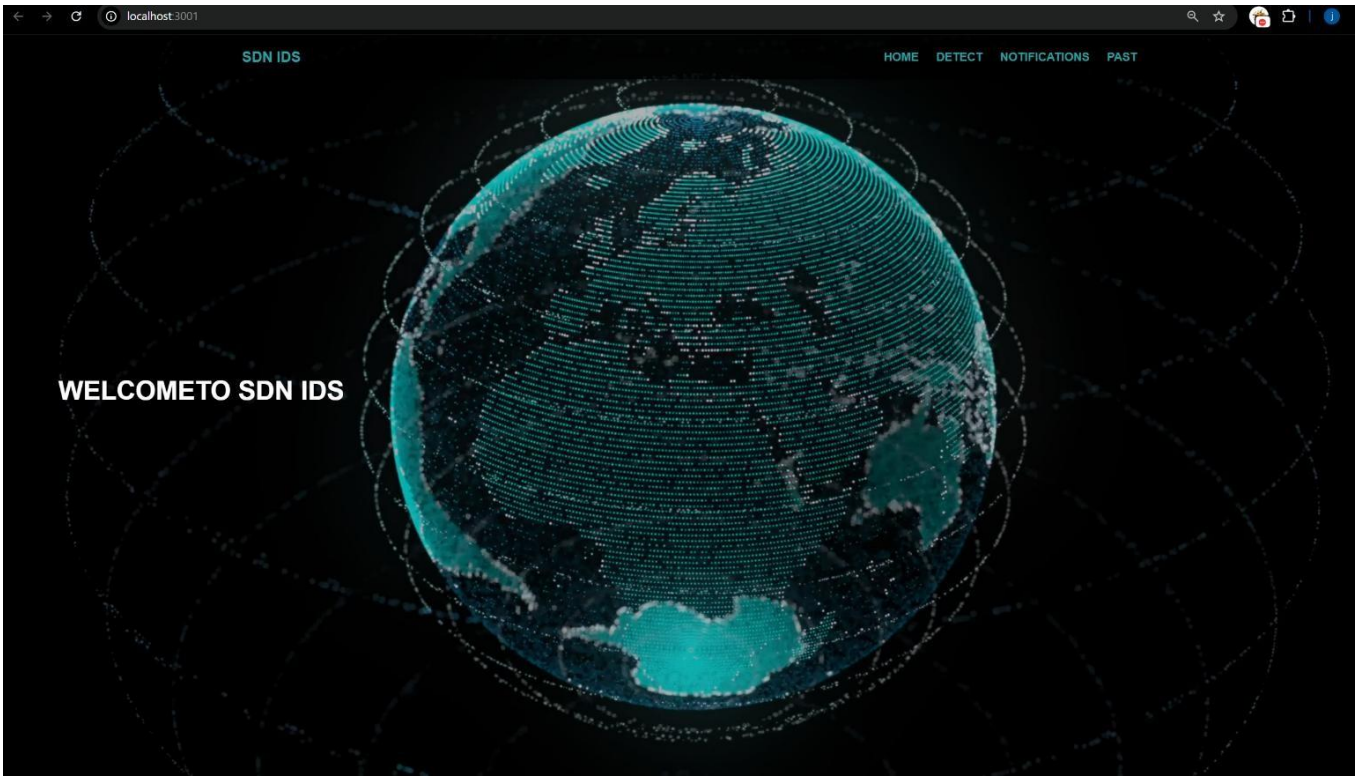
Paper Submission - Open	05th May 2025 Submit Now
Paper Submission - Close	20th August 2025
Acceptance Notification	20th October 2025
Camera Ready & Registration - Open	20th October 2025
Camera Ready & Registration - Close	10th November 2025
Conference	09th and 10th December 2025

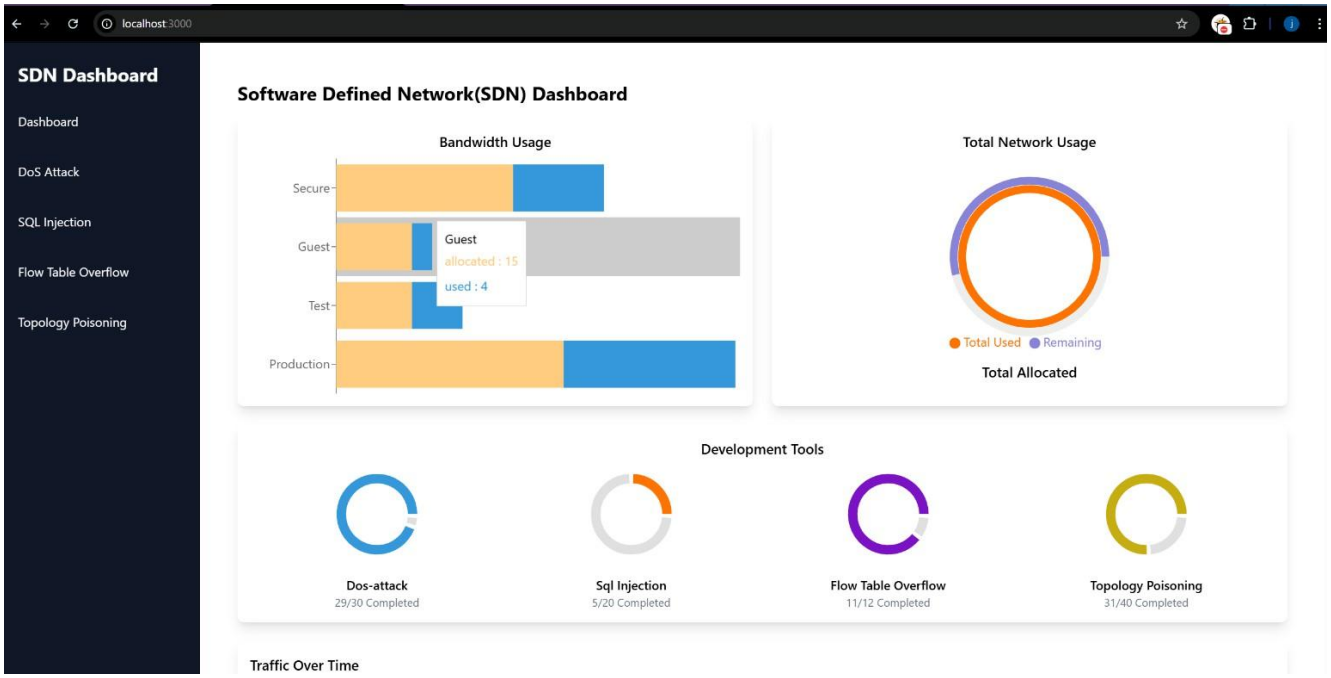
► Completed Tasks and Conversation Highlights

- 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 U
topology-att...
favicon.ico U
globals.css U
layout.js U
page.js U
.gitignore U
eslint.config.mjs U
jsconfig.json U
next.config.mjs U
package-lock.js... U
package.json U
postcss.config.... U
README.md U
uploads
OUTLINE
TIMELINE
...\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 LEARNING

Sriskandarajah J.P

IT21261978

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

Department of Information Technology

Sri Lanka Institute of Information Technology Sri Lanka

April 2025

i

DECLARATION

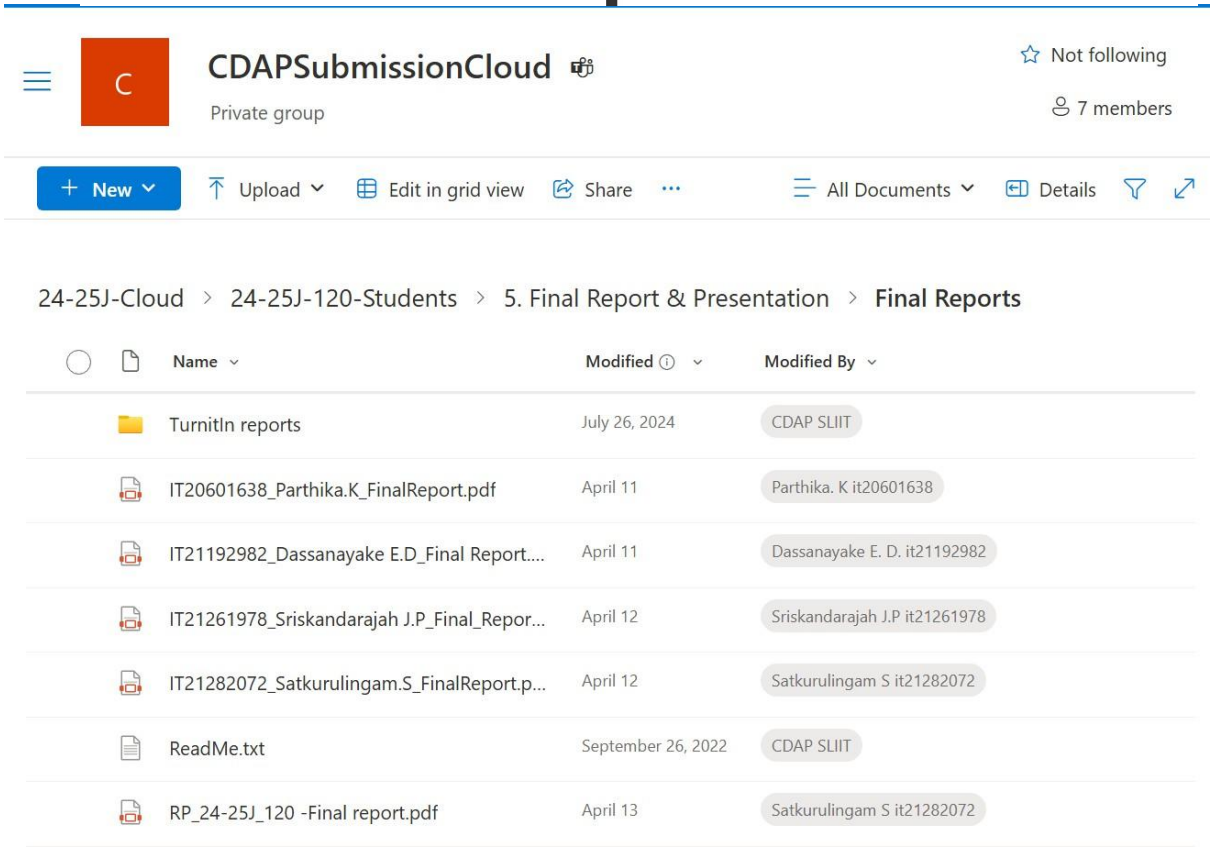
I declare that this is my own work and this dissertation does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously 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 in 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

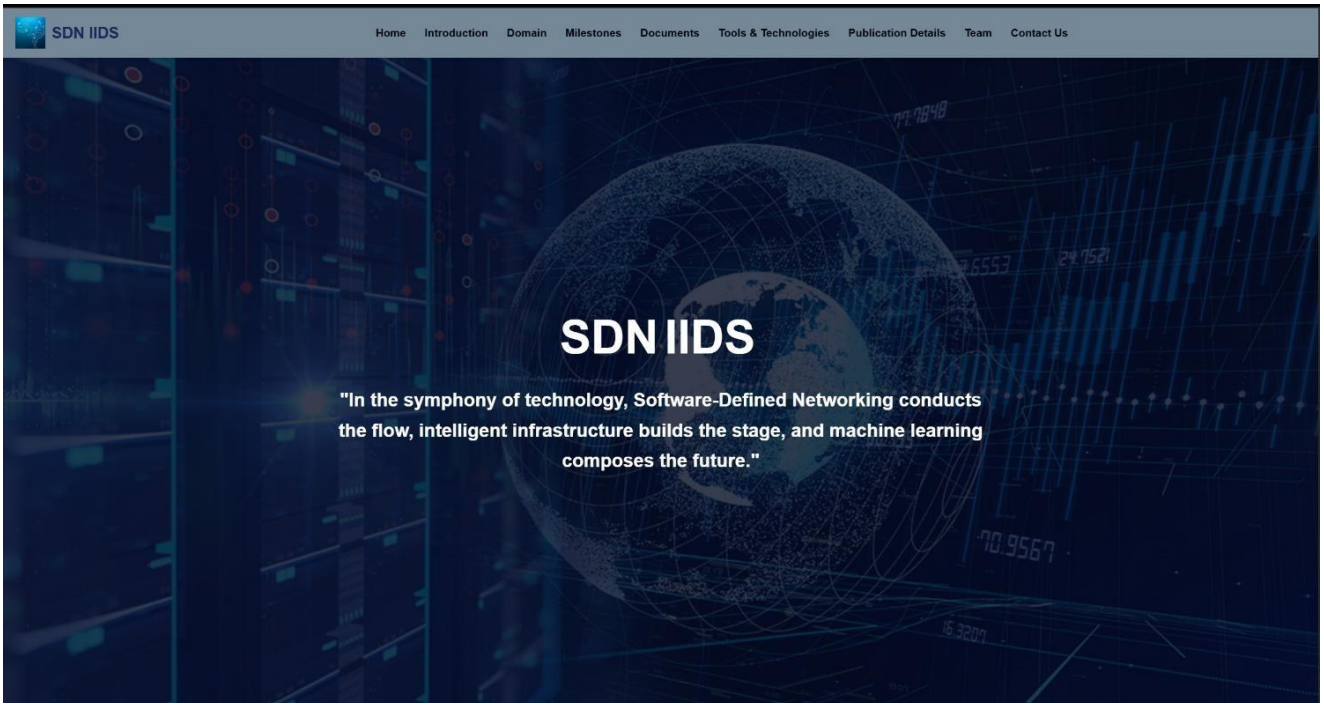
Signature of the supervisor: Date:

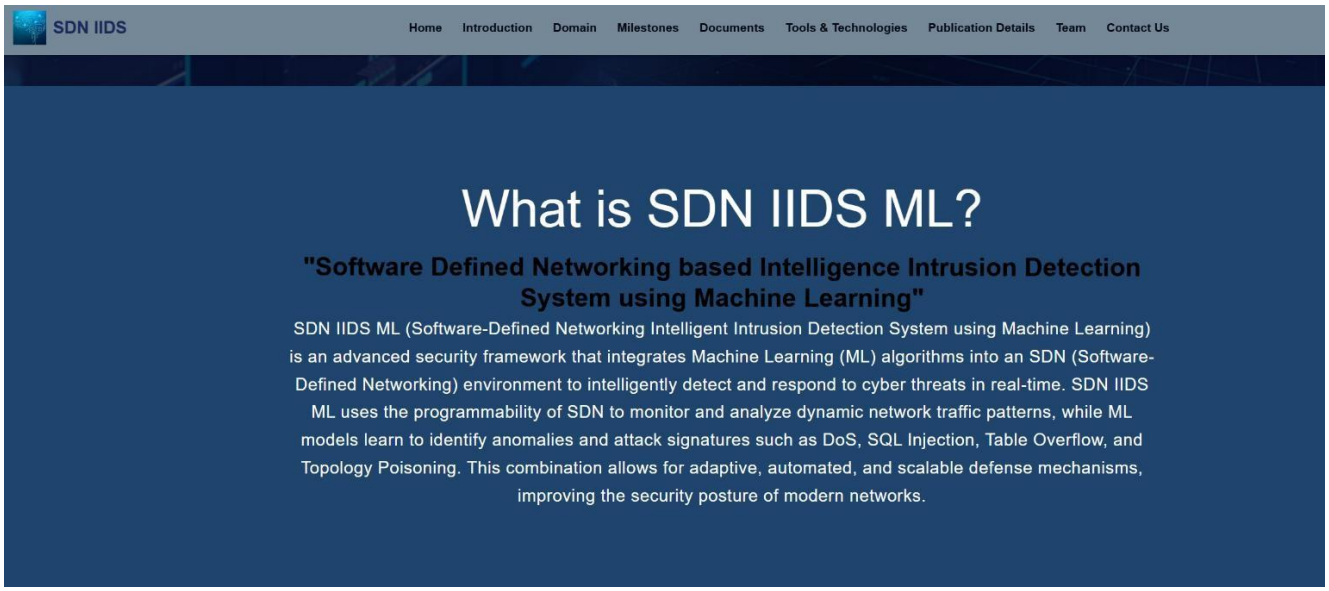
i



► 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

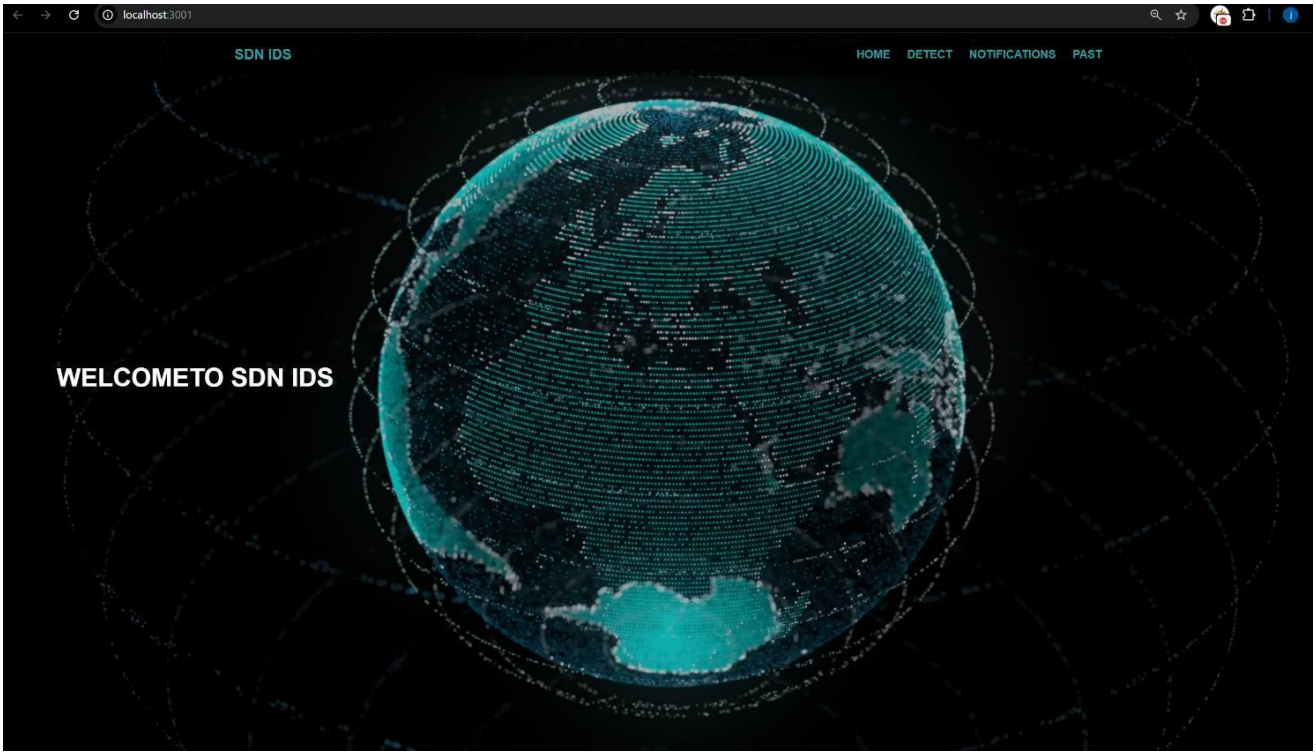
Sriskandarajah J.P
Member

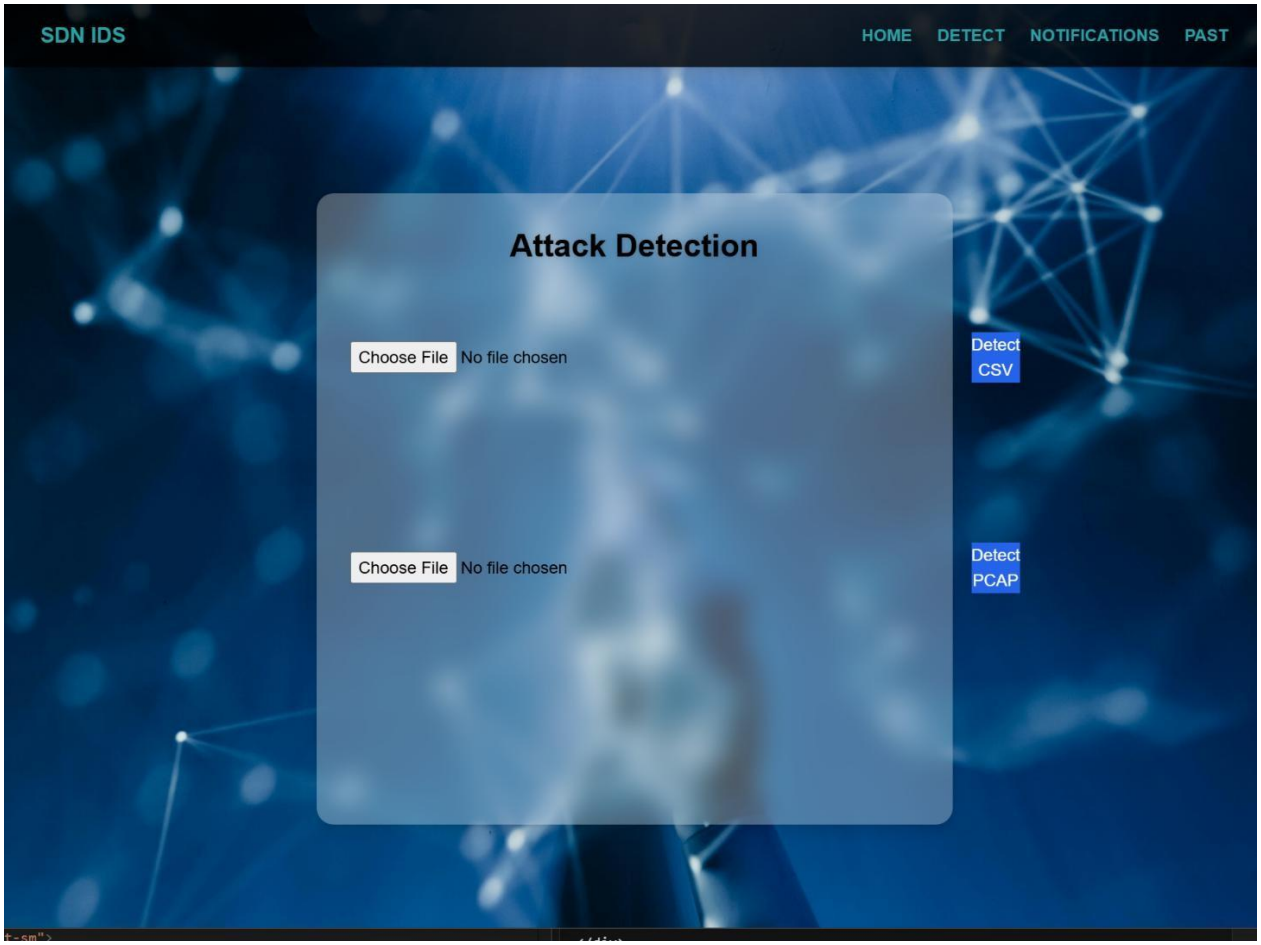
Dassanayake E.D
Member

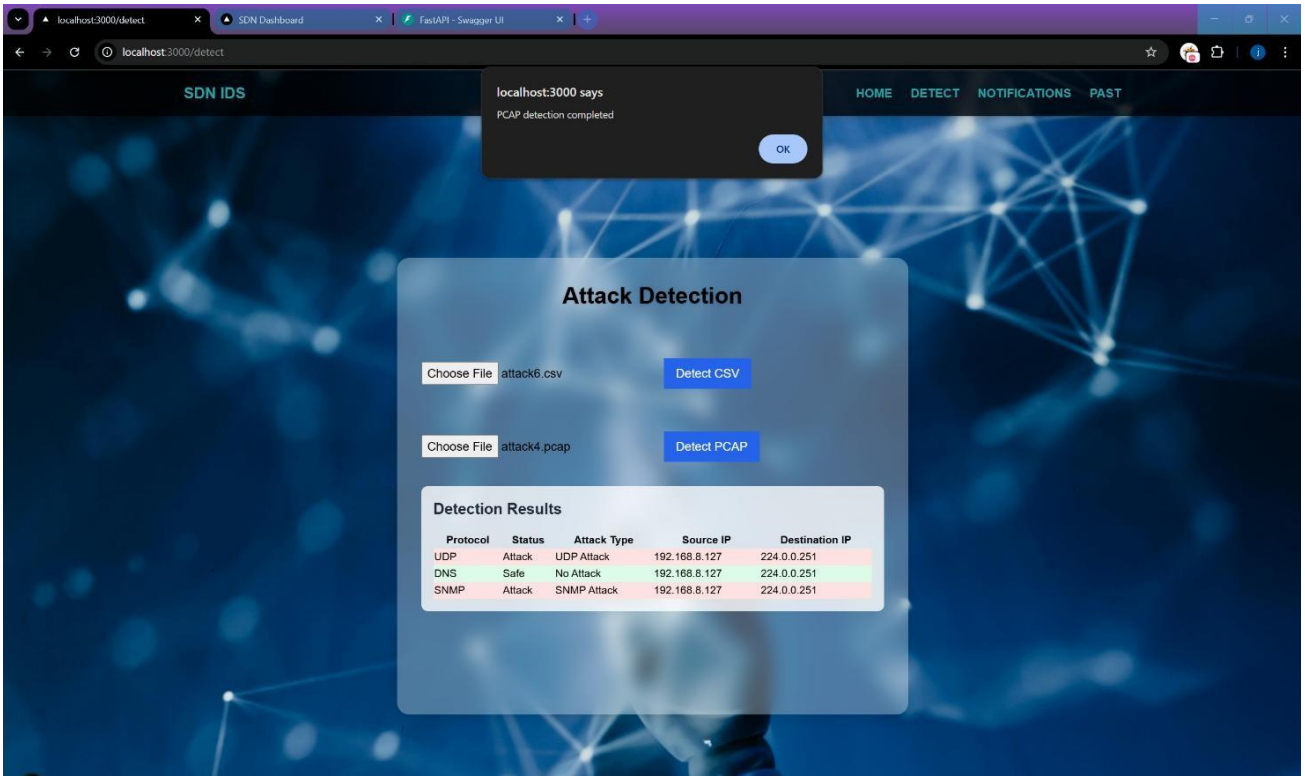
🛠 Tools and Technologies Used

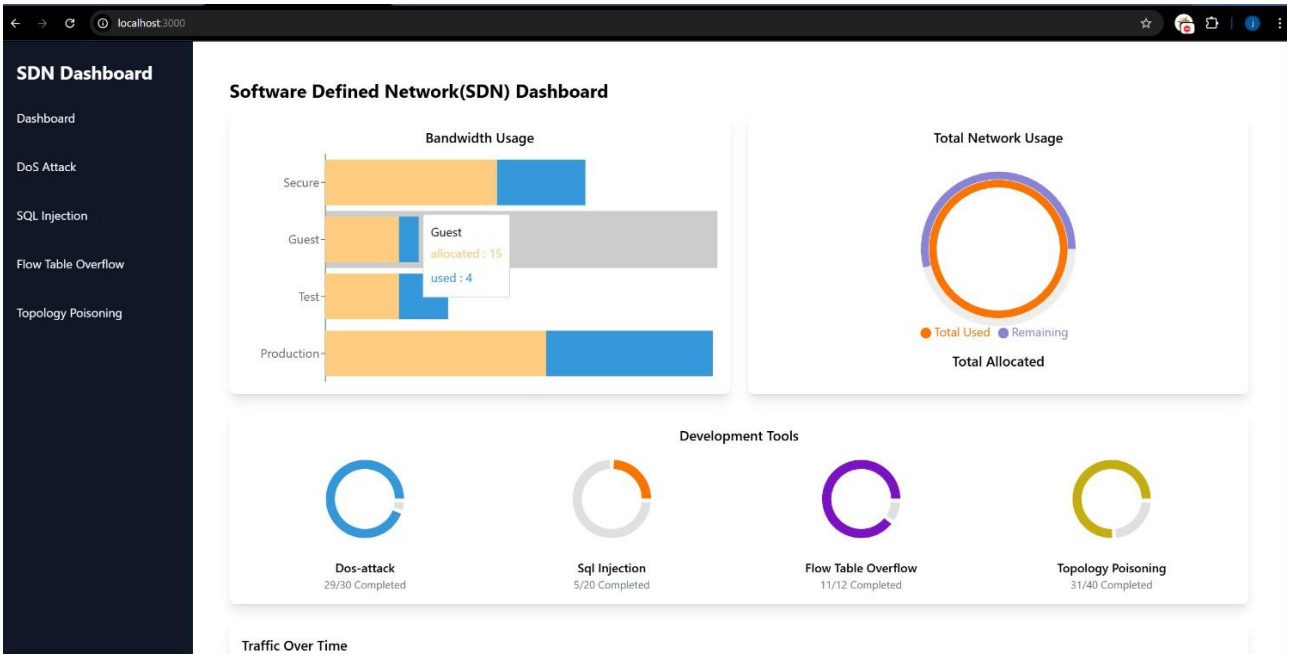
► **Completed Tasks and Conversation Highlights**

- Final Research Project Product





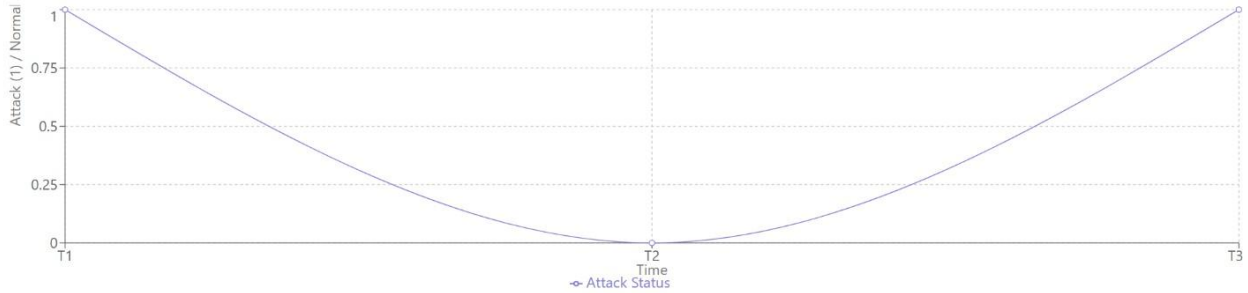

localhost:3000/detect
SDN Dashboard
FastAPI - Swagger UI
SDN IDS
HOME
DETECT
NOTIFICATIONS
PAST
localhost:3000 says
PCAP detection completed
OK
Attack Detection
Choose File
attack6.csv
Detect CSV
Choose File
attack4.pcap
Detect PCAP
Detection Results
Protocol
Status
Attack Type
Source IP
Destination IP
UDP
Attack
UDP Attack
192.168.8.127
224.0.0.251
DNS
Safe
No Attack
192.168.8.127
224.0.0.251
SNMP
Attack
SNMP Attack
192.168.8.127
224.0.0.251


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

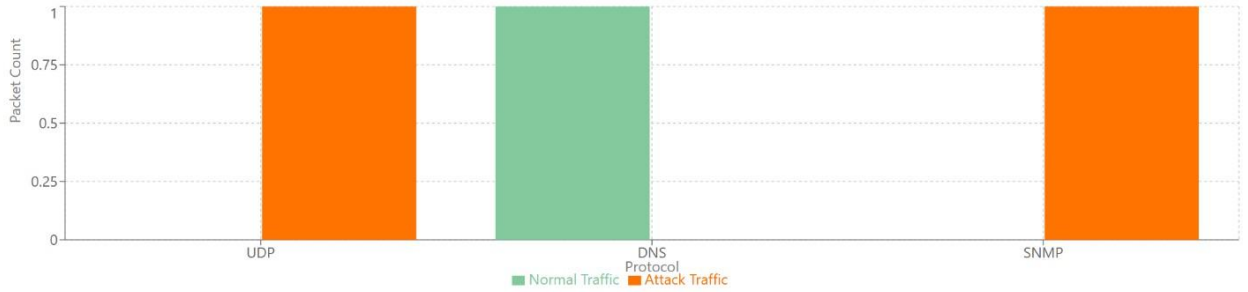
Upload PCAP to Detect Attacks

Choose File attack4.pcap

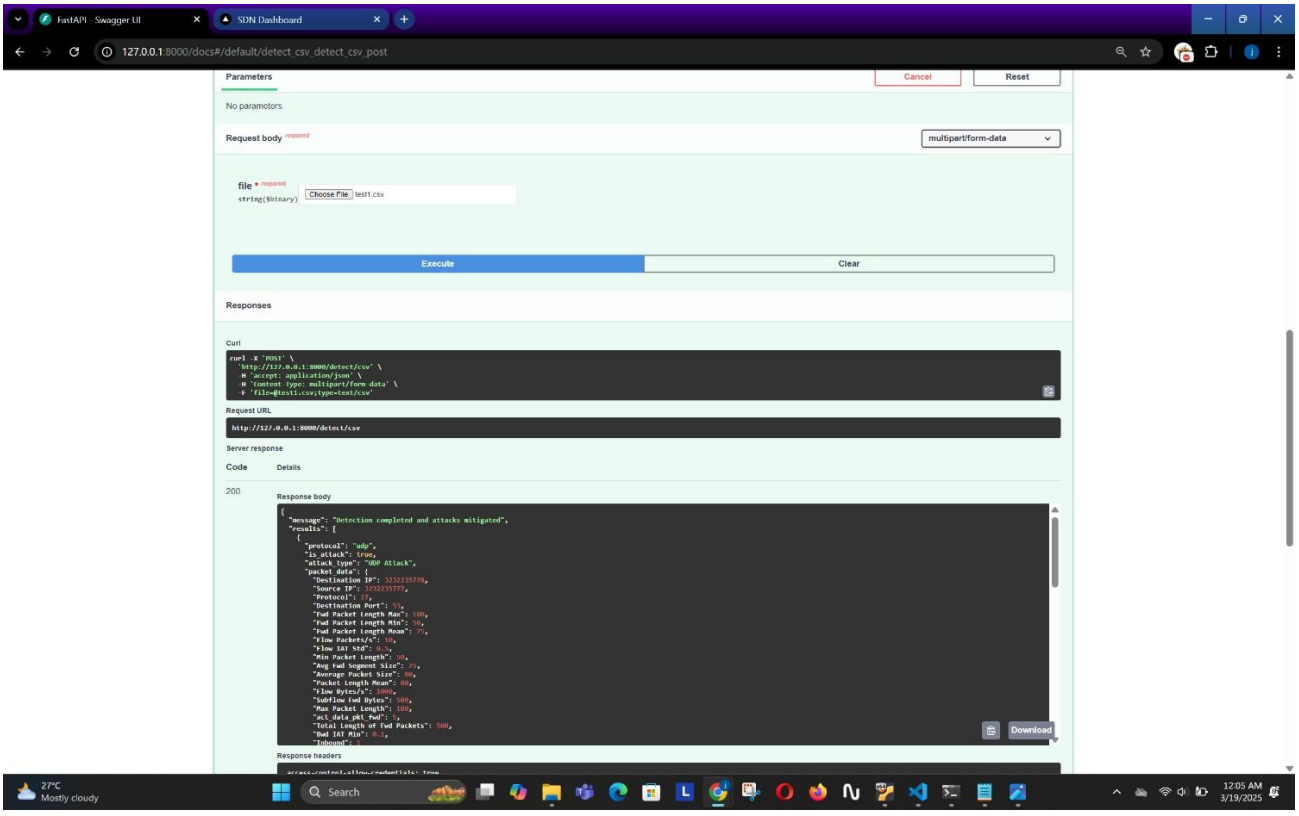
Packets Over Time

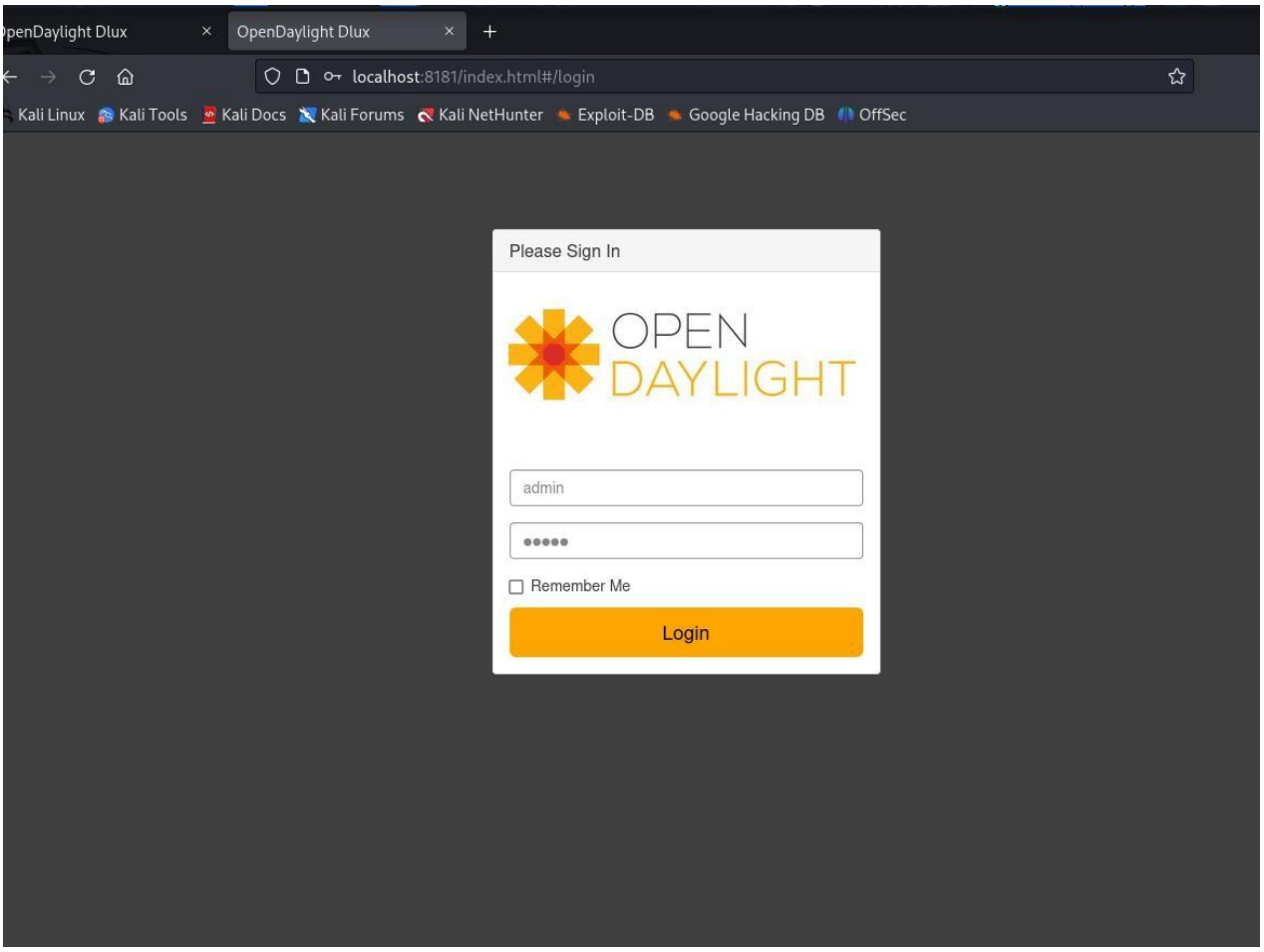


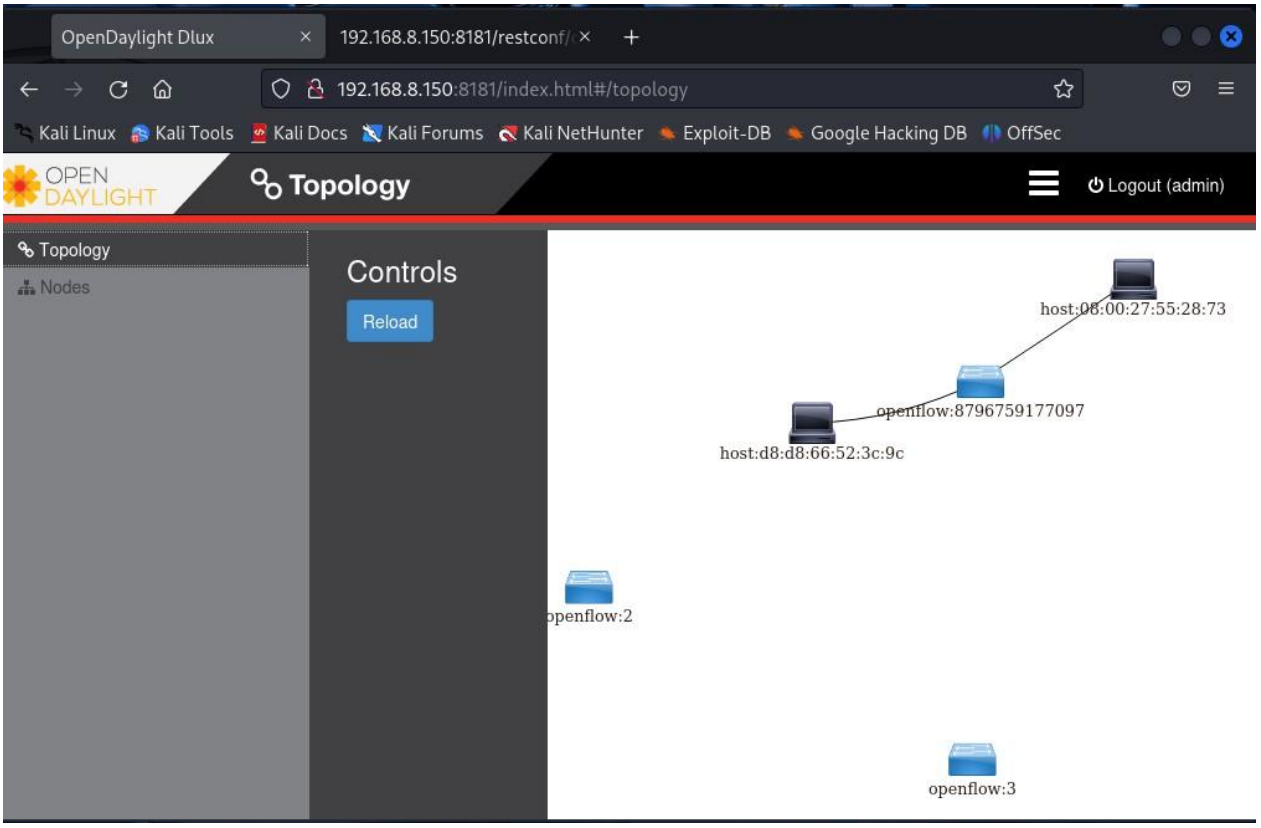
Normal vs Attack Traffic by Protocol

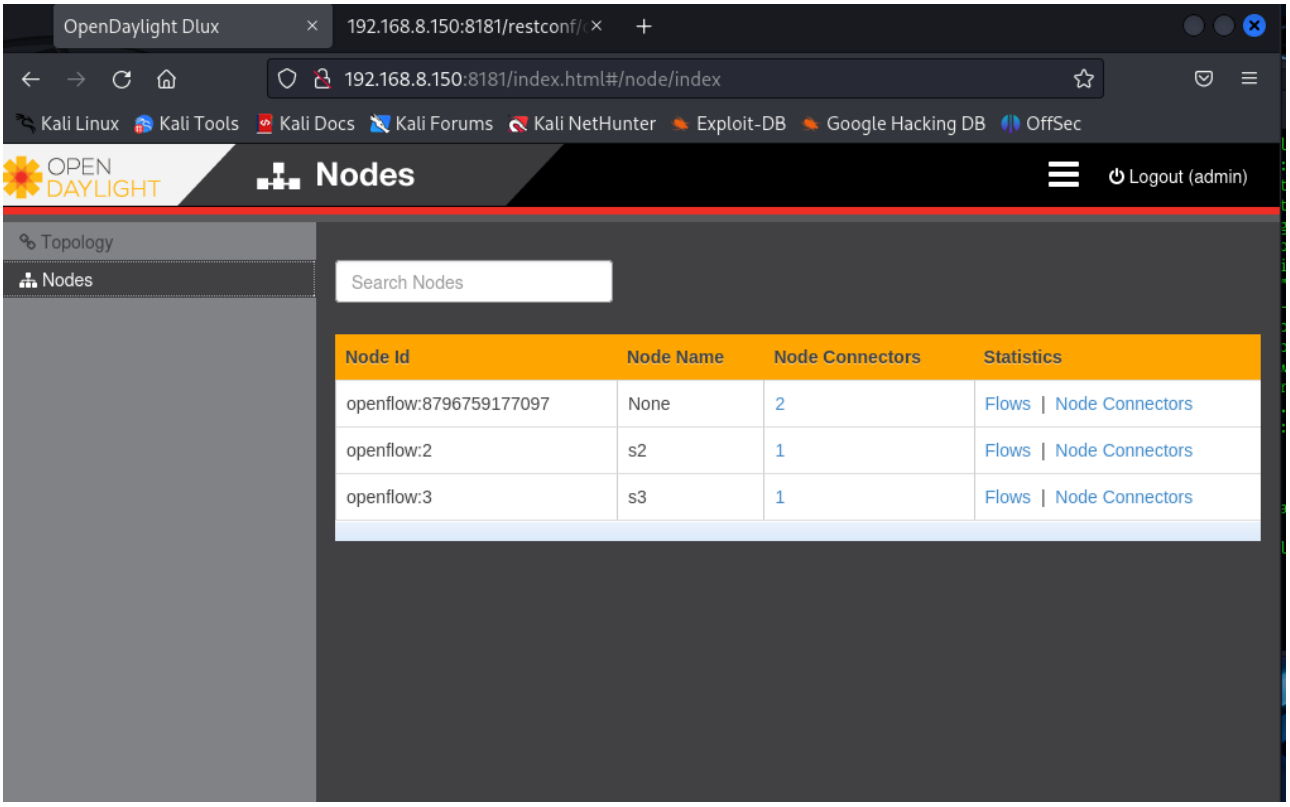


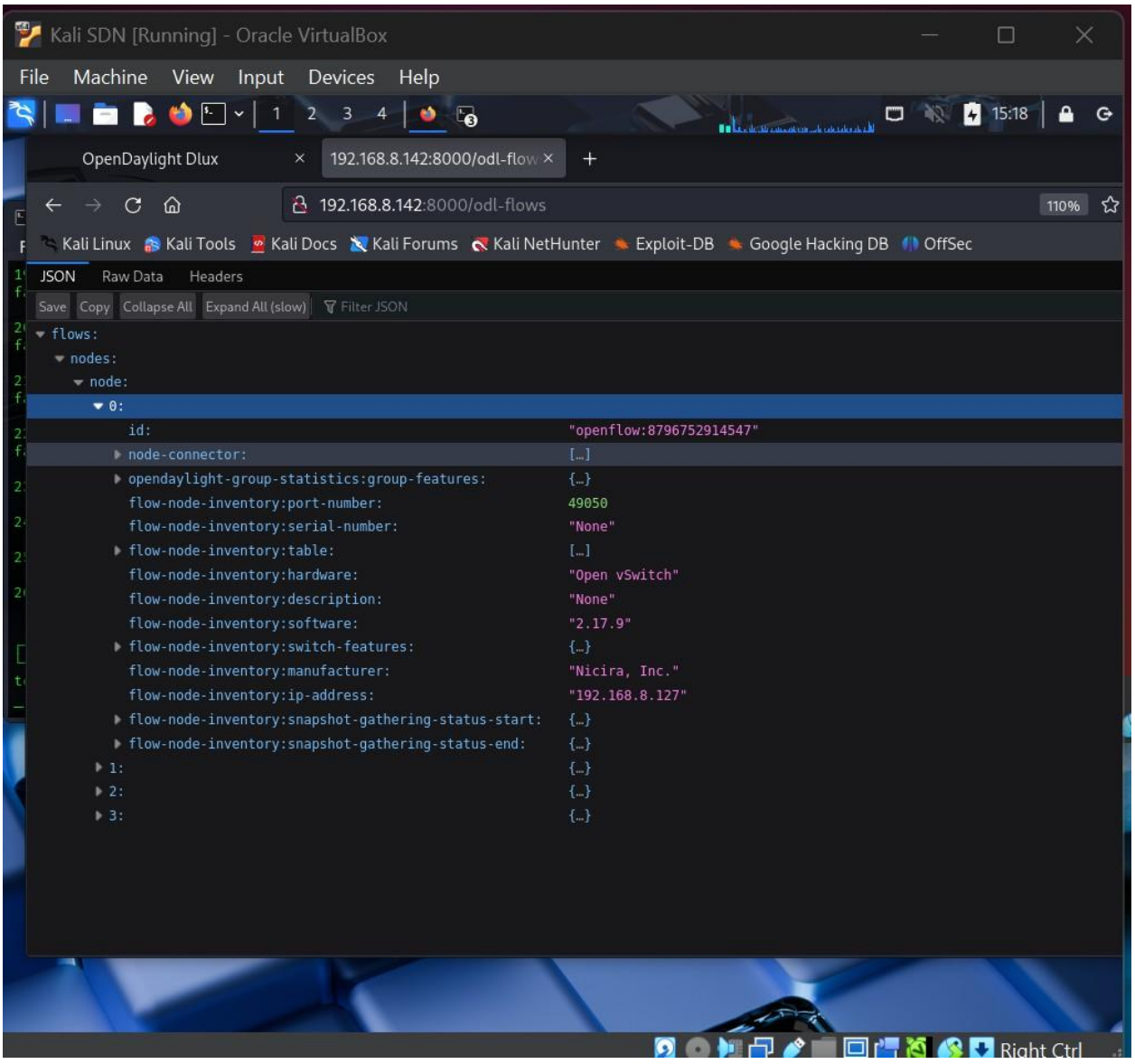
Traffic Volume with Attack Highlights




OpenDaylight 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


OpenDaylight Dlux
192.168.8.150:8181/restconf/
192.168.8.150:8181/index.html#/topology
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
OPEN DAYLIGHT
Topology
Logout (admin)
Topology
Nodes
Controls
Reload
host:08:00:27:55:28:73
openflow:8796759177097
host:d8:d8:66:52:3c:9c
openflow:2
openflow:3

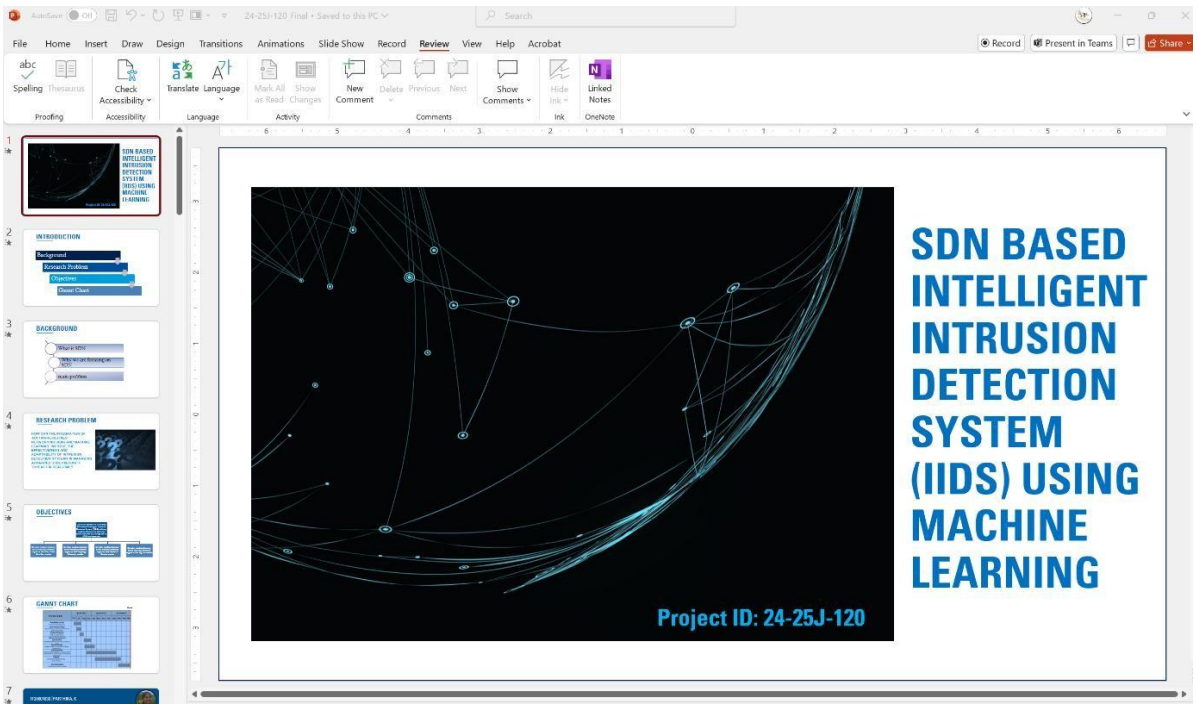

OpenDaylight Dlux
192.168.8.150:8181/restconf/
192.168.8.150:8181/index.html#/node/index
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
OPEN DAYLIGHT
Nodes
Logout (admin)
Topology
Nodes
Search Nodes
Node Id
Node Name
Node Connectors
Statistics
openflow:8796759177097
None
2
Flows | Node Connectors
openflow:2
s2
1
Flows | Node Connectors
openflow:3
s3
1
Flows | Node Connectors

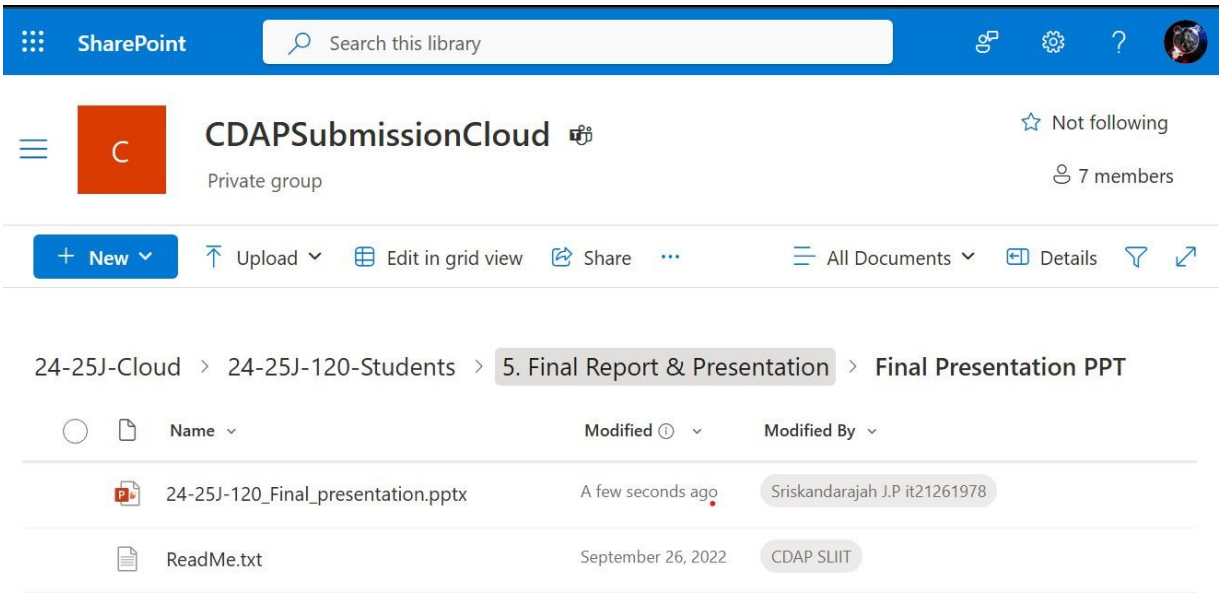

Kali SDN [Running] - Oracle VirtualBox
File
Machine
View
Input
Devices
Help
15:18
OpenDaylight Dlux
192.168.8.142:8000/odl-flow
192.168.8.142:8000/odl-flows
110%
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All (slow)
Filter JSON
flows:
nodes:
node:
0:
id: "openflow:8796752914547"
node-connector: […]
opendaylight-group-statistics:group-features: {…}
flow-node-inventory:port-number: 49050
flow-node-inventory:serial-number: "None"
flow-node-inventory:table: […]
flow-node-inventory:hardware: "Open vSwitch"
flow-node-inventory:description: "None"
flow-node-inventory:software: "2.17.9"
flow-node-inventory:switch-features: {…}
flow-node-inventory:manufacturer: "Nicira, Inc."
flow-node-inventory:ip-address: "192.168.8.127"
flow-node-inventory:snapshot-gathering-status-start: {…}
flow-node-inventory:snapshot-gathering-status-end: {…}
1: {…}
2: {…}
3: {…}
Right Ctrl

```
flow-node-inventory:snapshot-gathering-status-end:  {…}
▼ 2:
    id:                                             "openflow:2"
  ▶ node-connector:                                 […]
  ▶ opendaylight-group-statistics:group-features:   {…}
    flow-node-inventory:port-number:                36286
    flow-node-inventory:serial-number:              "None"
  ▶ flow-node-inventory:table:                      […]
    flow-node-inventory:hardware:                   "Open vSwitch"
    flow-node-inventory:description:                "s2"
    flow-node-inventory:software:                   "3.5.0"
  ▶ flow-node-inventory:switch-features:            {…}
    flow-node-inventory:manufacturer:               "Nicira, Inc."
    flow-node-inventory:ip-address:                 "192.168.8.150"
  ▶ flow-node-inventory:snapshot-gathering-status-start:  {…}
  ▶ flow-node-inventory:snapshot-gathering-status-end:    {…}
▼ 3:
    id:                                             "openflow:3"
  ▶ node-connector:                                 […]
  ▶ opendaylight-group-statistics:group-features:   {…}
    flow-node-inventory:port-number:                36280
    flow-node-inventory:serial-number:              "None"
  ▶ flow-node-inventory:table:                      […]
    flow-node-inventory:hardware:                   "Open vSwitch"
    flow-node-inventory:description:                "s3"
    flow-node-inventory:software:                   "3.5.0"
  ▶ flow-node-inventory:switch-features:            {…}
    flow-node-inventory:manufacturer:               "Nicira, Inc."
    flow-node-inventory:ip-address:                 "192.168.8.150"
  ▶ flow-node-inventory:snapshot-gathering-status-start:  {…}
  ▶ flow-node-inventory:snapshot-gathering-status-end:    {…}
```


► Completed Task and Conversation Highlights

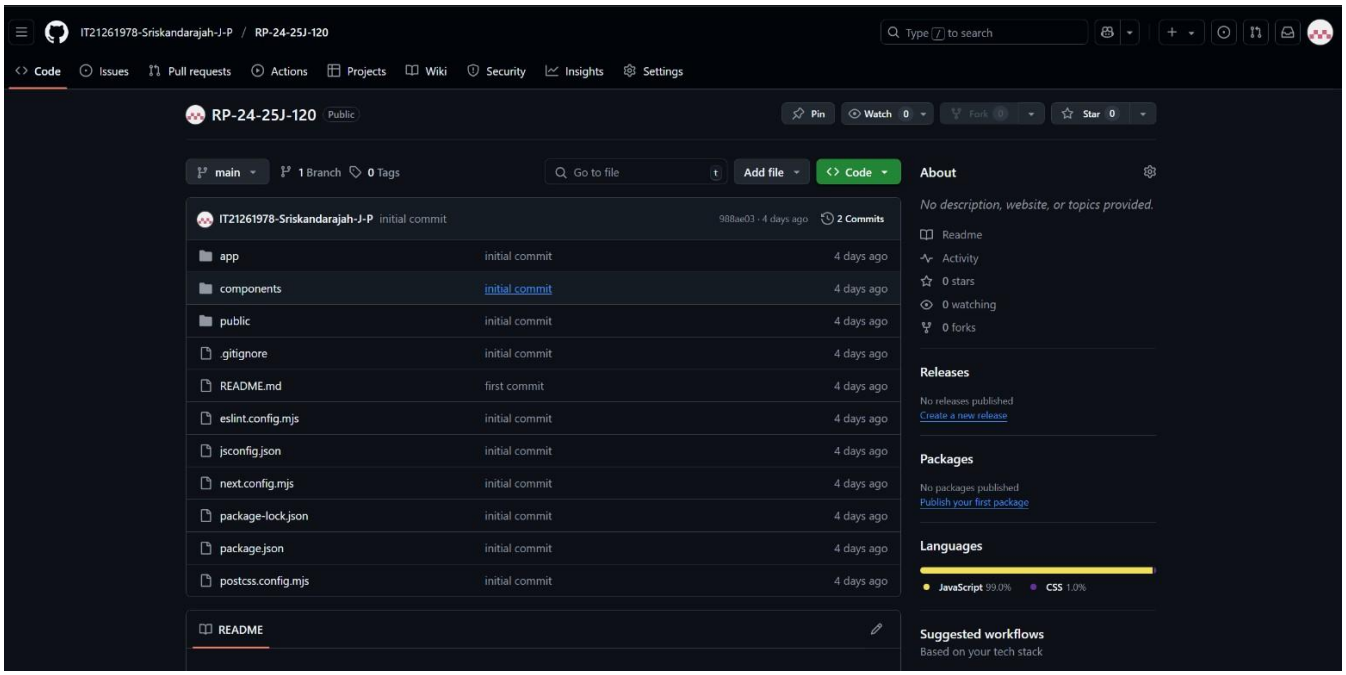
- Prepare for Final Presentation
- Creating the presentation.





► 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.