**Real-Time Network Monitoring and Anomaly Detection**

**A PROJECT REPORT**

***Submitted by***

**MEHUL BHATT RA2111030020062**

**SHAZIN NAVAS RA2111030020074**

**YUVA BHARGAV RA2111030020112**

**Under the guidance of**

**Mrs. S.Lakshmi ,M.E,(Ph.D).,**

**Assistant Professor,**

**Department of Cybersecurity and Gaming Technology**

***in partial fulfillment for the award of the degree of***

**BACHELOR OF TECHNOLOGY**

***in***

**COMPUTER SCIENCE AND ENGINEERING**

**with specialization in**

**CYBER SECURITY**

**of**

**FACULTY OF ENGINEERING AND TECHNOLOGY**

**SCHOOL OF COMPUTER SCIENCE ENGINEERING**



**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY RAMAPURAM, CHENNAI -600089**

May 2025

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(Deemed to be University U/S 3 of UGC Act, 1956)**

**BONAFIDE CERTIFICATE**

Certified that this project report titled **“REAL-TIME NETWORK MORNITORING AND ANAOMALY DETECTION”** is the Bonafide work of **MEHUL BHATT [REG NO: RA2111030020062],** **SHAZIN NAVAS [REG NO: RA2111030020074], YUVA BHARGAV [REG NO: RA2111030020112]** who carried out the project work under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an occasion on this or any other candidate.

|  |  |
| --- | --- |
| SIGNATURE  **Mrs.S. Lakshmi M.E.,(Ph.D).,**  **Assistant Professor,**  **Department of CS & GT,**  **School of Computer Science Engineering**  S R M Institute of Science and Technology,  Chennai-89. | SIGNATURE  **Dr. Shiny Duela J, M.E., Ph.D.,**  **Associate Professor and HOD**  **Department of CS & GT**  **School of Computer Science Engineering**  S R M Institute of Science and Technology,  Chennai-89. |

Submitted for the project viva-voce held on\_\_\_\_\_\_\_\_\_\_\_at S R M Institute of Science and Technology, Chennai -600089.

**INTERNAL EXAMINER EXTERNAL EXAMINER**

ii

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY RAMAPURAM, CHENNAI - 89**

**DECLARATION**

We hereby declare that the entire work contained in this project report titled “**REAL TIME NETWORK MORNITORING NAD ANAOMALY DETECTION**” has been carried out by **MEHUL BHATT [REG NO: RA2111030020062], SHAZIN NAVAS [REG NO: RA2111030020074], YUVA BHARGAV [REG NO: RA2111030020112]**, at SRM Institute of Science and Technology, Ramapuram, Chennai- 600089, under the guidance of **Mrs.S.Lakshmi, M.E., (Ph.D)., Assistant Professor,** School of Computer Science and Engineering.

**Place: Chennai**

**Date: MEHUL BHATT**

**SHAZIN NAVAS**

**YUVA BHARGAV**

iii

 **Own Work Declaration**

Department of Computer Science and Engineering

**SRM Institute of Science and Technology**

**Own Work Declaration form**

**Degree/ Course :** B. Tech Computer Science and Engineering with Specialization in cyberscurity

**Student Name :** MEHUL BHATT, SHAZIN NAVAS, YUVA BHARGAV

**Registration Number :** RA2111030020062, RA2111030020074, RA2111030020112

**Title of Work :** REAL TIME NETWORK MORNITORING AND ANAOMALY DETECTION

We hereby certify that this assessment complies with the University’s Rules and Regulations relating to Academic misconduct and plagiarism, as listed in the University Website, Regulations, and the Education Committee guidelines.

We confirm that all the work contained in this assessment is our own except where indicated, and that We have met the following conditions:

* Clearly references / listed all sources as appropriate
* Referenced and put in inverted commas all quoted text (from books, web, etc)
* Given the sources of all pictures, data etc. that are not our own
* Not made any use of the report(s) or essay(s) of any other student(s) either past or present
* Acknowledged in appropriate places any help that we have received from others (e.g., fellow students, technicians, statisticians, external sources)
* Compiled with any other plagiarism criteria specified in the Course handbook / University website

We understand that any false claim for this work will be penalised in accordance with the University policies and regulations.

|  |
| --- |
| **DECLARATION:** |
| We are aware of and understand the University’s policy on Academic misconduct and plagiarism and we certify that this assessment is our own work, except where indicated by referring, and that we have followed the good academic practices noted above. |
| RA2111030020062 RA2111030020074 RA2111030020112 |

iv

**ACKNOWLEDGEMENT**

We place on record our deep sense of gratitude to our honourable Chairman **Dr. R. SHIVAKUMAR, MBBS., MD.,** for providing us with the requisite infrastructure throughout the course.

We take the opportunity to extend our hearty and sincere thanks to our **Dean, Dr. M. SAKTHI GANESH., Ph.D.,** for maneuvering us into accomplishing the project.

We take the privilege to extend our hearty and sincere gratitude to the Professor and Chairperson, **Dr. K. RAJA, Ph.D.,** for his suggestions, support and encouragement towards the completion of the project with perfection.

We thank our honorable Head of the department **Dr.SHINY DUELA J., Ph.D., Associate Professor and HOD – CS & GT** for her constant motivation and unwavering support.

We express our hearty and sincere thanks to our guide **Mrs.S.LAKSHMI, AP, Department of CSE** for her encouragement, motivation and constant guidance throughout this project work.

Our thanks to the teaching and non-teaching staff of the Department of Computer Science and Engineering of SRM Institute of Science and Technology, Chennai, for providing necessary resources for our project

v

**ABSTRACT**

Anomaly-based detection systems identify potential threats by monitoring network behavior and detecting deviations from expected patterns. Although many models have been proposed, there is a gap in research when it comes to evaluating these models across a range of publicly available datasets. As cyber threats evolve quickly, it is essential to consistently update and benchmark intrusion detection datasets. Traditional methods for multi-class intrusion detection, such as deep neural networks, often fail to recognize spatial relationships and long-term dependencies in traffic data. This project introduces a new deep learning approach that addresses these challenges and aims to build a dependable system for detecting cyberattacks. Our framework includes three core strategies, the first being an autoencoder integrated with various optimization techniques. Experimental results demonstrate the effectiveness of this hybrid model in identifying contemporary threats. The CICIDS2017 dataset was used to test the model, which successfully classified multiple types of attacks. The model showed strong performance, particularly with the Adamax optimizer, in terms of accuracy, detection rate, and minimizing false alarms. Comparative analysis confirms that our solution outperforms other machine learning and deep learning models in these key areas.

vi

# TABLE OF CONTENTS

## Page. No

**ABSTRACT vi**

**LIST OF FIGURES**

[**LIST OF ACRONYMS AND ABBREVIATIONS xi**](#_gjdgxs)

1. **INTRODUCTION 1**

[1.1 Problem Statement](#_30j0zll) 2

[1.2 Aim of the Project](#_1fob9te) 3

[1.3 Project Domain](#_3znysh7) 3

[1.4 Scope of the Project](#_2et92p0) 8

[1.5 Methodology](#_tyjcwt) 10

1.6 Organization of the Report 10

1. **LITERATURE REVIEW 13**
2. **PROJECT DESCRIPTION 15**

[3.1 Existing System](#_3dy6vkm) 15

[3.2 Proposed System](#_1t3h5sf) 16

3.2.1 Advantages 17

[3.3 Feasibility Study](#_4d34og8) 17

* + 1. Economic Feasibility 17

vii

* + 1. Technical Feasibility 17
    2. Social Feasibility 18

3.4 System Specification 18

3.4.1 Hardware Specification 18

3.4.2 Software Specification 18

## PROPOSED WORK 20

* 1. General Architecture 20
  2. Design Phase 20
     1. Data Flow Diagram 22
     2. UML Diagram 23
     3. Use Case Diagram 24
     4. Sequence Diagram 25
  3. Module Description 25
     1. Module 1: Exploratory Data Analysis 25
     2. Module 2: Feature Selection 26
     3. Module 3: Iot BoT Prediction 26
     4. Step 2: Processing of Data 27
     5. Step 3: Split the Data 27
     6. Dataset Sample 27
     7. Step 4: Building the Model 28
     8. Step 5: Compiling and Training the Model 28

## IMPLEMENTATION AND TESTING 29

* 1. Input and Output 29
     1. Predicted Output 29

viii

5.2 Testing 29

5.2.1 Types of Testing 30

5.2.2 Unit testing 30

5.2.3 Integration testing 31

5.2.4 Functional testing 32

5.2.5 Test Result 32

## RESULTS AND DISCUSSIONS 33

* 1. Efficiency of the Proposed System 33
  2. Comparison of Existing and Proposed System 33

## CONCLUSION AND FUTURE ENHANCEMENTS 34

* 1. Conclusion 34
  2. Future Enhancements 34
  3. Results 34

## SOURCE CODE & POSTER PRESENTATION 36

* 1. Sample Code 36

## REFERENCES 37

## Sample screenshots

## Proof of Publication/Patent filed/ Conference Certificate

ix

**LIST OF FIGURES**

## Architecture Diagram 20

* 1. **Data Flow Diagram** 22

## Uml Diagram 23

* 1. **Use Case Diagram** 24

## Sequence Diagram 25

## Predicted Output of the Project 29

* 1. **Unit testing of the code** 30
  2. **Result of unit testing** 31
  3. **Testing anomaly detection integration** 31
  4. **Output of anomaly detection** 31
  5. **Training and split data** 32

## 7.1 Result of anomaly detection 34

7.2 **Final result of anomaly detection** 35

x

# LIST OF ACRONYMS AND ABBREVIATIONS

AI ARTIFICIAL INTELLIGENCE

API APPLICATION PROGRAMMING INTERFACE

PCA PRINCIPAL COMPONENT ANALYSIS

CSV COMMA SEPERATED VALUES

EDA EXPLORATORY DATA ANALYSIS

IOT INTERNET OF THINGS

xii

# CHAPTER 1

# INTRODUCTION

Modern life is deeply intertwined with digital systems, ranging from personal devices to large-scale enterprise and government networks. This widespread reliance on interconnected services has significantly increased vulnerability to cyber threats. As the use of internet-connected systems continues to grow, so does the risk of network-based attacks. Weaknesses in hardware, software, or configurations can be exploited, resulting in serious breaches with far-reaching consequences.

Addressing these challenges requires more than traditional tools. While firewalls and signature-based intrusion detection systems (IDS) offer some defence, they often fall short due to their static nature and inability to detect complex, evolving threats like spoofed addresses or large-scale denial-of-service attacks. These conventional systems are not well-equipped to analyse high volumes of diverse and dynamic traffic.

To overcome these limitations, advanced solutions must be employed. Real-time intrusion detection frameworks that leverage automation and intelligent algorithms can help organizations monitor, detect, and respond to malicious behaviour more effectively. By integrating technologies such as machine learning and real-time traffic analysis, it becomes possible to spot unusual patterns and alert security teams before damage is done. These intelligent systems represent a proactive shift in cybersecurity, enabling early threat detection and better overall resilience against increasingly sophisticated attacks.

## PROBLEM STATEMENT

In the modern digital era, organizations, governments, and individuals all depend heavily on interconnected networks and services. Security, reliability, and performance of computer networks are now critical to day-to-day operations. The exponentially increasing growth of network infrastructures such as cloud services, Internet of Things (IoT) devices, and distributed systems has significantly increased volume and complexity ofnetwork traffic. The growth makes it a serious challenge to ensure consistent performance and detect potential threats in a timely manner. The traditional network

monitoring tools are primarily reactive in nature. Most of them are static rule and threshold or look-back analysis based, lacking the speed and agility to detect and respond to anomalies in real-time. The response delays could result in catastrophic consequences such as undetected intrusions, extended service outage, loss of data, financial and reputational loss. Moreover, the threats from today's cyber space are sophisticated, dynamic, and stealthy, often evading traditional signature-based or rule-based detection. Moreover, separating normal traffic behavior from truly suspicious or anomalous behavior is not an easy task. Networks inherently are dynamic due to changing user activity, application updates, load balancing, and other operating changes. Thus, the traditional approaches tend to generate a high false positive rate or miss low-signal indicators of compromise. Increased adoption of encrypted traffic, mobile devices, and decentralized network architectures further complicate visibility and analysis. Keeping these challenges in mind, there is an urgent need for intelligent real-time network monitoring and anomaly detection system that can analyze high-speed traffic, detect anomalies on the fly, and provide early warning signs before things snowball. Such a system needs to use advanced techniques, including machine learning, deep learning, statistical modeling, and data visualization, to identify known patterns of attacks and learn to identify new ones. The objective of this project is to design and implement a real-time, smart, and scalable solution to network activity that can effectively monitor network traffic, identify real-time anomalies, and eliminate false positives. Such a system needs to be able to support managing heterogeneous network infrastructures, learning from dynamic changing patterns of data, and providing actionable intelligence for the network administrators as well as security experts.  
By addressing this problem, the project intends to contribute towards the development of proactive network security solutions that encourage organizational resilience, offer system availability, and protect sensitive data from constantly evolving cyber attacks.

* 1. **AIM OF THE PROJECT**

**To develop a real-time network monitoring system** capable of continuously analyzing live traffic data without significant delays or performance degradation.

 **To implement an intelligent anomaly detection mechanism** using machine learning or statistical techniques that can identify unusual or malicious patterns in network behavior.

 **To detect both known and unknown threats**, including network intrusions, DDoS attacks, unauthorized access, and abnormal traffic spikes.

 **To reduce the number of false positives and false negatives** by improving detection accuracy through adaptive learning models.

 **To build a scalable solution** that can handle large volumes of network data in diverse environments (e.g., corporate networks, cloud systems, IoT infrastructures).

 **To provide real-time alerts and notifications** to administrators for quick response and mitigation of potential threats.

 **To create a user-friendly interface or dashboard** for visualizing network activity, detected anomalies, and system health metrics.

 **To enhance the overall network security posture** by enabling proactive threat detection and faster incident response.

 **To support decision-making for network administrators** through actionable insights derived from traffic analysis and anomaly reports.

 **To contribute to the advancement of modern cybersecurity practices** by demonstrating the application of real-time analytics and AI in network defence.

## PROJECT DOMAIN

Cyber security is keeping information and information systems secure from unauthorized disclosure, use, disruption, modification, or destruction. Information security, computer security and information assurance are synonymous. The three fields are connected but share a common goal of safeguarding the confidentiality, integrity and availability of information with some differences between them. These differences are mostly in the approach to the subject, the methodology used, and the focus areas. Cyber security deals with the assurance of the confidentiality, integrity and availability of information in whatever form the information takes: electronic, print, or others. Cyber security covers the tools and processes that businesses use to secure information. This includes policy settings that prevent unauthorized users from accessing business or personal data. InfoSec is a dynamic and evolving field that covers a wide range of areas from network and infrastructure security to testing and auditing. Cyber security ensures that sensitive information is safeguarded against unauthorized activity, including inspection, alteration, recording, and any disruption or destruction. The aim is to guard the safety and privacy of critical information like customer account information, financial information or intellectual property. The CIA triad consists of three basic principles â€" confidentiality, integrity, and availability (CIA). Together, these principles form the foundation that information security policies follow.

1. Confidentiality

Information should be kept for only authorized individuals. Sanctions against unauthorized disclosure of information are in place to ensure confidentiality. The main purpose of the rule of confidentiality is to maintain private information confidential and protect it from visibility or reach by anyone else except its owners or individuals who require it to attain their organizational goals.

2. Integrity

Data must always be consistent, accurate, and reliable. Data integrity is an aspect that encompasses protection against unauthorized modifications (additions, deletions, revisions, etc.) to data. The integrity principle asserts that data is accurate, consistent, and not inappropriately altered, by accident or design.

3. Availability

Data should be made available to the correct parties even in the event of failures (with little or no impact). Availability is protecting a system's ability to make software systems and data entirely available when a user needs it or at a specified time. Availability attempts to make the technology environment, applications, and data as available when necessary for an organizational process or customers of an organization.

1. Application security

Application security controls are used to protect applications and application programming interfaces (APIs). You can use these to avoid vulnerabilities, identify and resolve bugs or other application weaknesses. Application and API weaknesses, when not secured, can provide a point of entry into your systems overall, violating your information. Much application security depends on application assurance-specific tools to protect, scan and test applications. These can help you learn about weaknesses and other applications and surrounding circumstances. Once you find them, you can patch these weaknesses before applications are released or weaknesses are exploited. Application security applies to applications you're using and you develop because both need to be protected.

2. Infrastructure security

Security for infrastructure guards infrastructure elements, such as networks, servers, client desktops, mobile devices, and data centers. Increased interconnectivity between these, and other infrastructure elements, exposes information to risk unless adequate measures are in place. That risk is that interconnectivity spreads vulnerabilities throughout your systems. If any component of your infrastructure crashes or is

compromised, the dependent components also get impacted. Because of this, one key objective of infrastructure security is to reduce dependencies and compartmentalize components while continuing to enable intercommunications.

3. Cloud Security

Cloud security provides similar protections as application and infrastructure security but is applied in cloud or cloud connected components and data. Cloud security adds additional protections and tools to deal with the vulnerabilities that are created by Internet-facing services and shared environments, such as public clouds. Cloud security also has a tendency to include a consideration of centralizing security management and tooling. Centralization provides security teams visibility of information and information threats on distributed resources. Another aspect of cloud security is a relationship with your cloud provider or third-party services. When you have resources and applications hosted in the cloud, you usually cannot control your environments completely because the infrastructure is typically managed for you. This means that cloud security practices must consider limited control and impose controls to limit accessibility and vulnerabilities due to contractors or sellers.

4. Cryptography

Cryptography uses a method called encryption to secure information by concealing its contents. Information that has been encrypted can be read only by those who have the correct encryption key. Those who do not have the key cannot read information that has been encrypted. Security personnel can use encryption to establish confidentiality and integrity to information in all phases of its life cycle, including transmission and storage. The moment that the user decrypts the information, however, it can be stolen, subjected to

modification. To encrypt data, security staff use tools like encryption algorithms or technologies like blockchain. Encryption algorithms, like the advanced encryption standard (AES), are more common as there is greater support for such tools and less usage overhead.

5. Encryption

Encryption techniques encrypt information such that it will only be intelligible to individuals who possess confidential keys. Encryption is very successful in avoiding data loss or damage in case of technology loss or theft or the probability that violators breach organizational systems.

Encryption algorithms are specifically created to eliminate any correlation between plaintext and the corresponding ciphertext. A good encryption algorithm will have a ciphertext that is indistinguishable from a random number. The only way to know what plaintext a given ciphertext is included in is to use the proper key to decrypt it. Being able to perform math on encrypted data means that there has to be a correlation between plaintexts and ciphertexts. It has to be possible to add or multiply two ciphertexts together and have the result be the same as doing the same operation on the two plaintexts and encrypting it. homomorphic encryption is a type of encryption that allows any data to be stored encrypted while things being processed and manipulated. It allows you or a third party (such as a cloud provider) to operate functions on encrypted data without requiring to know the values of the data. A homomorphic cryptosystem is like other forms of public encryption in that it uses a public key to encrypt data and only provides access to its unencrypted data to the owner of the corresponding private key (although there are also symmetric key homomorphic encryption examples as well). But what sets it apart from other forms of encryption is that it uses an algebraic system to allow you or other individuals to perform a large number of computations (or operations) on the encrypted data.

## SCOPE OF THE PROJECT

The scope of the project is the design, development, and testing of an intelligent real-time network monitoring system. The project will focus on live network traffic monitoring to identify abnormal patterns that may indicate security threats, loss of performance, or system misconfiguration. The system will be able to support dynamic and large-scale network environments like corporate networks, cloud networks, and data centers. It will employ machine learning or statistical models to distinguish normal variations of the network from potentially malicious activity, enhance detection accuracy, and minimize false positives. The system will also support real-time alerting and have a straightforward interface for easy utilization by network administrators to act fast on identified anomalies. The project will also explore scalability and efficiency, so the solution is still effective even during peak-traffic hours. It will not, however, tackle advanced intrusion prevention or mitigation methods, nor enhance traditional firewalls or security appliances. It will, instead, be an add-on monitoring device to enhance existing network defense mechanisms through early anomaly detection and continuous traffic analysis.

## METHODOLOGY

The methodology adopted in this project follows a systematic approach to design, develop, and evaluate an intelligent system capable of monitoring network traffic in real time and detecting anomalies using machine learning. The process is divided into several key stages

**1. Problem Definition and Requirement Analysis**

Begin by identifying the core problems in existing network monitoring systems (e.g., delay in detection, high false positives). Analyze user requirements, such as the need for real-time alerts, visualization tools, or integration with existing systems.

**2. Data Collection**

Clean and preprocess the collected data:

* Remove irrelevant or redundant features
* Normalize/standardize values
* Handle missing or corrupted data
* Extract useful features (e.g., IP addresses, packet size, protocol type, time intervals)

.

**3. Feature Engineering**

Select or derive meaningful features that help differentiate normal from abnormal traffic patterns. Common features include:

* Traffic volume over time
* Number of connections per IP
* Packet size variance
* Protocol distribution

**4. System Architecture Design**

Design the structure of the real-time monitoring system. Components may include:

* Packet sniffer or traffic capture module
* Data preprocessing engine
* ML-based anomaly detection module
* Real-time alert system
* Dashboard for visualization

**5. Real-Time Integration**

Integrate all components into a system capable of analysing live network traffic. Use tools like:

* **Scapy**, **Tshark**, or **Socket programming** for packet capture
* Python for backend logic
* Web frameworks (Flask/Django) for UI/dashboard Ensure the model processes data with minimal

delay for real time performance.

## ORGANIZATION OF THE REPORT

This document is structured to provide an easy and brief summary of the entire process of creating a real-time network monitoring and anomaly detection system. Each chapter is structured to follow on from the previous one, guiding the reader through the background, research, system design, implementation, and testing of the project. The document is structured in the following chapters:

Chapter-1: Introduction. The chapter presents a general introduction to the project, including the background and objectives of the study. The chapter defines the problem being addressed, states the purpose and objectives of the project, and discusses the project scope with limits and boundaries of the work. The chapter also explains the need for real-time anomaly detection in modern network security and concludes with a summary of the report structure

Chapter-2 Literature Survey.

This chapter discusses the existing research work in network monitoring, intrusion detection, and anomaly detection techniques. It discusses traditional techniques and modern techniques with a focus on machine learning and artificial intelligence-based systems. Surveying identifies the advantages and limitations of existing techniques and gives theoretical foundation to the design decisions taken here in this project. It also identifies the research work gap to be fulfilled by this project.

Chapter-3 System Analysis and Methodology. Functional and non-functional system requirements are established. The methodology used is discussed in detail, i.e., the order of operations performed for data collection, preprocessing, feature extraction, model selection, training, and evaluation. This chapter explains why machine learning algorithms and tools used for anomaly detection have been chosen. It also provides the proposed workflow of the entire system, starting from real-time data input to alert generation.

Chapter-4 System Design and Implementation,The design of the real-time monitoring and detection system. It provides the architecture, components, and technologies employed in the system development. The functional role of each module—traffic capturing, preprocessing, anomaly detection, alert generation, and visualization—are explained. The implementation details, i.e., programming languages, libraries, and frameworks employed (e.g., Python, Scikit-learn, Flask), are also explained in this chapter. detection performance, Low latency and computational overhead suitable for resource-constrained environments, High detection rates for unknown and stealthy anomalies

Chapter-5 Testing and Evaluation, talks about testing the performance of the system developed. It explains the testing process followed to ensure the functionality and efficiency of the system. Some performance metrics—accuracy, precision, recall, F1-score, detection rate, and false positive rate—are used to evaluate the anomaly detection model. Test cases in real-time, system response, and traffic load scalability are also considered. Results are presented in tabular as well as graphical form for better understanding.

Chapter-6 Conclusion and Future Work The final chapter gives an overview of the entire work, the significant findings and contributions of the project, etc. It gives an overview of the benefits of using a real-time network monitoring and anomaly detection system and how it can resolve current issues in cybersecurity. The limitations of the current system are discussed, and possible areas of future improvements are proposed, for instance, integration with deep learning models, use of real-time threat intelligence feeds, or hosting the system on cloud-based systems.

Chapter-7 Reference, the chapter consists of a comprehensive list of all the books, research papers, internet sources, data sets, and tools cited or consulted in the course of the project. Chapter-8 Appendices, the appendices hold supplementary materials that are relevant to the key issue of the report, including sample code, user interface snapshots, model training results, network logs, and descriptions of the datasets. These materials hold extra detail about the technical implementation and serve to authenticate the project outcomes.

# CHAPTER 2

# LITERATURE REVIEW

This chapter presents a digest of the body of current research and academic literature relevant to the project's topic. Through a digest, a summary of key findings, technical advancements, and method development is provided below. Critical reviews and comparisons of several methods and models also serve as a good starting point for comprehending the state-of-the-art at the time and potentially any gaps or scope for future research and innovation. This chapter is required to set the project in the broader context by referring to existing knowledge and lessons of past research.

Bhuyan et al [1] Using highlights the central role of packet inspection in the context of anomaly-based intrusion detection systems (IDS). Their study highlights the importance of deep packet inspection (DPI) in gathering detailed information regarding network traffic behavior, particularly for the detection of subtle and novel anomalies. The authors offer a comprehensive study of signature-based and anomaly-based detection techniques. Signature-based detection relies on known attack patterns and can effectively detect known threats but is less effective against novel or evolving attacks. Anomaly-based detection monitors departures from known baseline behaviors and is more appropriate for unknown or zero-day attack detection but suffers from high false positive rates. Bhuyan et al. suggest that a combination of these detection techniques—using hybrid detection models—can greatly enhance the overall security posture of network systems. Their study categorizes numerous anomaly detection techniques based on traffic features, detection techniques, and performance measures and offers a comparative study to aid in the design of effective IDS frameworks. The study also highlights the constraints of high-speed network environments such as handling large volumes of data and sustaining real-time detection capabilities, and suggests the use of advanced machine learning models and adaptive models to address these challenges and improve the accuracy and responsiveness of detection in real-world environments.

Biondi & Fournier. [2] provide a detailed overview of Scapy, an interactive Python-based packet inspection and manipulation tool specifically designed for network packet inspection and manipulation. Their paper describes Scapy's programmability and flexibility, which set it apart from other packet analysis systems. Unlike standard network analyzers, which can only handle fixed protocols and operations, Scapy enables users to construct, send, sniff, dissect, and manipulate packets on various layers of the OSI model with fine-grained control. This makes it particularly effective in both offensive security testing and defensive network monitoring applications. The authors demonstrate how Scapy facilitates scriptable and customizable traffic analysis, making it a useful tool for researchers, penetration testers, and network engineers. With a simple Python-based syntax, users can specify custom packet structures, conduct deep packet inspection, and examine real-time traffic flows with accuracy. Biondi and Fournier also mention the extensibility of the tool—users can just add support for new protocols or extend existing ones, enabling Scapy to be effective in fast-evolving network environments. In real-time network monitoring and anomaly detection application, Scapy offers a building block functionality for capturing and analyzing live traffic. It can be integrated into more extensive security systems to facilitate advanced capabilities such as packet-based anomaly detection, traffic pattern modeling, and network attack simulation for IDS system testing. The authors recommend Scapy be employed in experimental network configurations due to its control and flexibility, making it an essential tool for constructing and testing custom monitoring frameworks that require fine-grained examination of network activity.

Chandola et al. [3] present an extensive and authoritative overview of methods for anomaly detection, emphasizing their algorithmic classification, theoretical basis, and applications—particularly network security. Their paper serves as a reference to observe how anomalies, patterns in data not expected to occur, can be aptly detected using various computational techniques. The overview classifies methods for anomaly detection as statistical, proximity-based, clustering-based, and classification-based models, and discusses at great length each of these methods' suitability, advantages, and shortcomings in various applications. A major part of their paper discusses how clustering techniques have been used to detect anomalies, particularly network attack detection. Clustering, being an unsupervised learning approach, can detect inherent clusters within data without knowledge of labels in advance. The aspect makes it particularly suitable to be used within network environments, where labelled data about attacks isn't available or is outdated. Chandola et al. emphasize again that methods of clustering—such as k-means, DBSCAN, and hierarchical clustering—can aptly distinguish normal and anomalous traffic by identifying outliers or poorly contained data points within defined classes.

Kabiri & Ghorbani. [4] provide an extensive analysis of incident response mechanisms and their critical role in mitigating the impact of cyberattacks on network systems. Their analysis delves into the systematic, step-by-step mechanisms that must be embraced by organizations to effectively counter and recover from security incidents. These incident response mechanisms are crucial in providing an immediate and coordinated response to the detected threats, minimizing the damage potential of attacks, and restoring the system to normal operating state as quickly as possible. The authors highlight the importance of combining incident response mechanisms with network detection systems to develop an extensive and proactive security model. Through the combination of real-time monitoring and detection capabilities with pre-defined response mechanisms, organizations are able to not only identify threats in real time but also initiate automated or manual responses to contain, minimize, and eliminate the attack effects. The combined system reduces response time and increases the overall efficiency of the security operations.

Javaid et al. [5] outline the use of deep neural networks (DNNs) for intrusion detection in network traffic, the promise of more advanced models such as machine learning to enhance IDS performance. Their work demonstrates how DNNs, an artificial neural network with multiple layers of processing elements, can be used to learn to recognize complex patterns and relationships in network traffic data that are overlooked by conventional approaches. By exploiting the hierarchical nature of DNNs, the authors demonstrate how such models can automatically learn significant features from raw traffic data, with minimal manual feature engineering, and are thus well-suited to cope with the high dimensionality and variability of contemporary network traffic.

Gao et al. [6] make a useful contribution to network security by addressing the new challenge of threat detection in encrypted HTTPS traffic using machine learning models. Contemporary web communication, in which HTTPS plays a growing role in protecting user privacy and confidentiality of data, sees traditional intrusion detection systems—especially those based on deep packet inspection—obstructed by encryption, which masks the packet payload. Gao et al. recognize this new challenge and introduce a framework that enables effective threat detection without undermining the integrity of the encrypted data, thereby protecting user privacy.

Their contribution is the use of machine learning algorithms to the traffic metadata and behavioral feature analysis, rather than decrypting data. These features include packet size, timing, direction, session length, and traffic statistics, all of which can be observed even when the data is encrypted. By mining and analyzing such side-channel information, the authors demonstrate that malicious activity is still detectable with high confidence even when the payload is not readily available.

Redmon et al. [7] revolutionized the field of real-time image processing and computer vision with the advent of YOLO (You Only Look Once), a real-time object detection system that was both accurate and fast. Their new approach revolutionized the traditional object detection approaches by casting the task as a single regression problem where it predicts class probabilities and bounding boxes directly from full images in a single pass. This design allowed YOLO to surpass the then current state-of-the-art detectors that used multi-stage pipelines or region proposal networks by a considerable margin.

**CHAPTER 3**

**PROJECT DESCRIPTION**

## EXISTING SYSTEM

In-vehicle controller area network (CAN) is susceptible to a wide range of cyberattacks due to its broadcast-based communication. An attacker can launch forged messages to a vehicle's CAN via wireless communication, the infotainment system, or the onboard diagnostic port. An effective intrusion detection system is therefore necessary to distinguish between legitimate CAN messages and forged messages. In the current research, the current system authors designed a hybrid quantum-classical CAN intrusion detection system based on a classical neural network (NN) and a quantum restricted Boltzmann machine (RBM). The classical NN handles feature extraction of images from a vehicle CAN bus data. On the other hand, the quantum RBM handles CAN image reconstruction for classification-based intrusion detection. The contribution of this current research is applying the generative ability of an RBM to reconstruct the pixels of a CAN image, part of which is responsible for labelling. Then, that portion of the reconstructed image is used for classifying the image as an attack image or a normal image. In order to evaluate the performance of the proposed hybrid quantum-classical CAN intrusion detection system, the current system authors used a real-world dataset of CAN fuzzy attacks to create three individual attack datasets, where each dataset represents a unique set of features related to the vehicle. We compared our hybrid framework's performance against a similar but classical-only framework. Our experiments proved that the hybrid framework outperforms the classical-only framework in CAN intrusion detection. For the three datasets considered in the current research, the best models in the hybrid framework achieved 97.5%, 97%, and 98.3% intrusion detection accuracies and 94.7%, 93.9%, and 97.2% recalls, respectively. Contrarily, the best-performing models in the classical-only configuration achieved 92.5%, 95%, and 93.3% accuracies of intrusion detection and 84.2%, 89.8%, and 88.9% recalls, respectively. Quantum computing can provide an ironclad defense against a variety of cyberattacks in a transportation M. S. Salek et al.: Novel Hybrid Quantum-Classical Framework for an In-Vehicle CAN Intrusion Detection cyber-physical systems environment. However, because of the current state of quantum computers, the best way to utilize them is to use hybrid quantum-classical methods. In this paper, the authors of this present system proposed a hybrid quantum-classical CAN intrusion detection framework based on a classical NN and a quantum RBM. In the framework, data preprocessing is done in a classical computer in order to create CAN images with embedded labelling pixels from CAN messages. A quantum RBM is used in the framework in order to reconstruct each CAN image and its labelling pixels, which is then used for an image classification-based CAN intrusion detection. We tested our hybrid quantum-classical CAN intrusion detection frame- work on three different real-world fuzzy attack datasets and compared the CAN intrusion detection performance of the hybrid framework with a similar but classical-only frame- work. From the experiments with the datasets, the lowest accuracy and recall of the hybrid framework were 97% and 93.9%, respectively, while for the similar but classical-only framework, the lowest CAN intrusion detection accuracy and recall were 92.5% and 84.2%, respectively. The innovation of this research is in using the gener- ative capability of a generative NN (i.e., RBM) in reconstructing the labelling pixels hidden inside a CAN image, which could lead to a correct image classification-based CAN intrusion detection. It is indicated that while the hybrid quantum-classical CAN intrusion detection framework employs a quantum computer for training the RBM models, once the RBM models have been trained to satisfactory intrusion detection performance, the quantum computer is no longer involved. The trained models can then be deployed to an in-vehicle computing unit where the whole process of CAN intrusion detection will be executed. This will ease the end- to-end latency in a CAN IDS, which then could enable the implementation of a real-time in-vehicle intrusion detection application. This research employed QA-based training to train the RBM models for the hybrid quantum-classical CAN IDS. Future research should implement gate-based RBM models for CAN IDS and compare them with the QA-based RBM models. Any views, findings, conclusions, or recommendations expressed in this material are those of the author(s) and are not necessarily those of C2M2, and the U.S. Government assumes no responsibility for the contents or use thereof.

**PROPOSED SYSTEM**

We address existing issues by proposing a novel model that utilizes three different deep learning models. The main contributions of this paper can be listed as follows. We propose a single Autoencoder based on the combination of multi-scale convolutional neural network and long short-term memory for network traffic anomaly detection. Our proposed model works in an unsupervised manner by effectively removing the need for manual labelling of the data. The part of our model is capable of extracting inherited spatial patterns from network traffic. The LSTM-AE part of our model extracts the temporal patterns from network traffic in addition to the spatial patterns extracted by the AE. In order to further improve the classification accuracy, we utilize a two-staged detection approach using isolation forest to effectively remove the false positives and false negatives produced by the threshold-based approach utilized in the Autoencoder. We utilized the Local Outlier Factor in data preprocessing to remove the normal packets overlapping with the density position of abnormal packets. Then, a cleaned (reduced) set of normal packets is utilized to train another local outlier model to detect abnormal packets.

We focused primarily on autoencoder (AE) to build efficient IDS because they are easy to implement and low computational expense. Several research works have attempted to build variants of AE with improved discriminative intrusion detection. Regarding the performance enhancement of the deep learning model, we optimize the neural network with one of the most popular algorithms, which is gradient descent. There are three categories of gradient descent algorithms: batch gradient descent, stochastic gradient descent, and mini-batch gradient descent. We need to select the gradient descent depending on the amount of data we want to calculate the gradient of the objective function. We use sigmoid as an input activation function and SoftMax as an output activation function to classify the attacks in our model.

* + 1. **ADVANTAGES**

• Lower learning rate will prevent overfitting and provide improved stability of training.  
• Minimize the amount of trainable parameters and simplify the model training.  
• Could lead to higher accuracy and shorter training time.  
• The meaning of parameters is preserved and the model stabilizes.  
• Successfully overcomes the unclassifiable region and enhances the computational complexity.

## FEASIBILITY STUDY

A Feasibility analysis was conducted to validate the efficacy and feasibility of the hybrid anomaly detection system in real-time industrial IoT in a real-world setting. The study highlights:  
• Technical Feasibility  
• Operational Feasibility  
• Economic Feasibility

**TECHNICAL FEASIBILITY**

This This phase determines if the technology, tools, and infrastructure available can be utilized to build the desired system. Packet sniffing, traffic inspection, and machine learning algorithms are required by the real-time monitoring system in order to properly detect anomalies.

• Tools and Technologies: Tools and technologies such as Scapy, Wireshark, and Python libraries such as Scikit-learn, TensorFlow, and Kera's can be used easily to support real-time traffic analysis as well as machine learning-based detection.

• System Requirements: The system is installable on generic hardware and open-source platforms, reducing reliance on costly specialist equipment.

• Scalability and Integration: The subject system can be integrated and scaled as part of existing network security infrastructure, such as firewalls and intrusion prevention systems, for future adaptability.

* + 1. **OPERATIONAL FEASIBILITY**

This determines if the system in question will function optimally in the real environment and if the administrators and users will adopt and use it.

• Ease of Use: The solution will include a graphical dashboard for real-time alerting and traffic visualization, which will be intuitive and easy to use for network administrators.

• Real-Time Capability: It can process and manage real-time streams of information without introducing significant latency, thereby being suitable for dynamic network situations.

• Security and Reliability: Combined alerting capabilities and anomaly logging will enable fast response and traceability in the event of suspicious activity.

* + 1. **ECONOMIC FEASIBILITY**

Because the proposed hybrid anomaly detection system is highly compatible with user expectation, occupational safety needs, and societal demands, it is socially feasible. This factor determines the extent to which the advantages of project implementation outweigh the costs.

• Development Costs: The project is mainly based on open-source software and available infrastructure, consequently little initial investment.

• Maintenance Costs: Model maintenance and updates of traffic patterns are feasible with low resource demands.

• Benefits: Organization will be able to avoid the expense of pricey data breaches, downtime, and potential legal costs with early awareness of cyber attacks. ROI measured in terms of damage avoided is likely to be very high.

## SYSTEM SPECIFICATION

This section explores the hardware and software requirements, along with tools and technologies used for the development of the system.

* + 1. **HARDWARE SPECIFICATION**
* Processor: Intel 10th gen or higher with i5 or higher
* Stable internet connection
* At least 200GB HDD/SSD or more
* Minimum 16 GB; Recommend 32 or above
  + 1. **SOFTWARE SPECIFICATION**
* Python For AI/ML/DL ProgrammingAnaconda
* Jupyter Notebook IDE (Integrated Development Environment) for Development.TensorFlow
* PyTorch or TensorFlow for Deep Learning Coding
* Sklearn for Machine Learning/Feature Extraction/Evaluation Metrics Coding
* Numpy for implementing Linear Algebra
* Plotly for Data Visualization (For Graphs)
* Matplotlib for Data Visualization (For Graphs).
* Seaborn for Data Visualization (For Graphs).
* Pandas for dealing with Tabular Data.

**CHAPTER 4**

**PROPOSED WORK**

## GENERAL ARCHITECTURE

****

Figure 4.1: **Architecture Diagram**

## DESIGN PHASE

The design phase highlights a strong emphasis on developing a plan of the system architecture and functionality in a way that each component of it is structured systematically and can deliver the intended results.

The current research emphasizes real-time detection and prevention of botnet attacks in an industrial IoT system.

The framework consists of multiple modules connected with each other such as hijacked IoT devices (bots), command-and-control (C&C) server (represented by the center malicious computer), and victim devices like smart doorbells and cameras. Bots are manipulated by the command and control server, share information through the internet using HTTP protocols, and attempt to infect compromised IoT devices on the network.

The model shows a common botnet attack route, where the attack is launched from compromised machines operated by an offender. These machines constitute a network that talks through a centralized server and tries to infect or modify other target machines. Information diffusion emphasizes the following considerations in the application of anomaly detection by hybrid machine learning approaches, especially in the traffic observation and analysis layer.

The design stage defines the role of every component and data flow, and therefore the anomaly detection system is able to effectively identify unusual activity and respond to possible threats before affecting operational systems. The stage also evaluates system scalability, integration with the existing IoT infrastructure, and real-time processing.

* + 1. **DATA FLOW DIAGRAM**

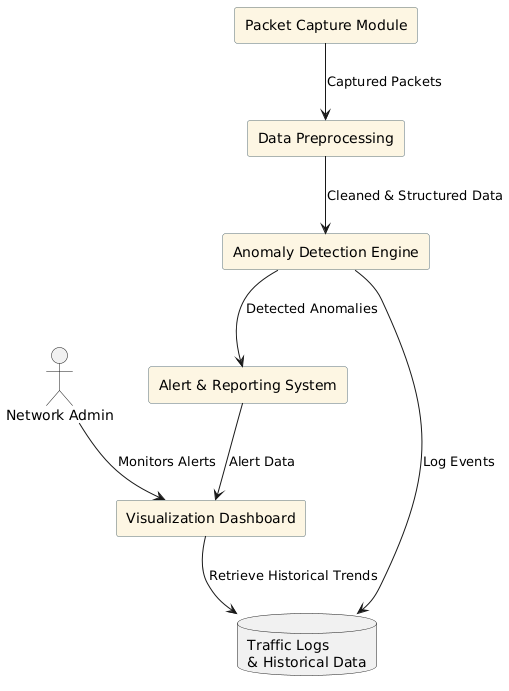


Figure 4.2: **Data Flow Diagram**

Figure 4.2 The Real-Time Network Monitoring and Anomaly Detection System's primary features are described in the Data Flow Diagram. The Packet Capture Module records network traffic first, and the Data Preprocessing unit processes and organizes it. The Anomaly Detection Engine examines this cleaned data to find odd trends. The Alert and Reporting System receives detected anomalies and notifies the Network Administrator through the Visualization Dashboard. For future use and trend analysis, all traffic logs and events are kept in one central database. In order to facilitate proactive network security, this system guarantees real-time monitoring, precise detection, and efficient alerting.system administrator for further action.

* + 1. **UML DIAGRAM**

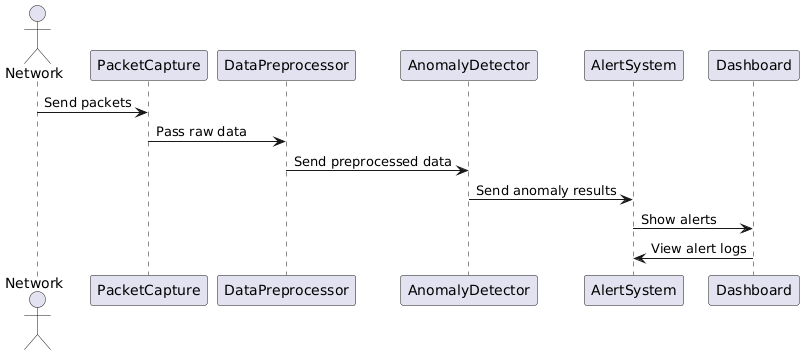


Figure 4.3: **UML Diagram**

The Real-Time Network Monitoring and Anomaly Detection System's data flow is depicted in Figure 4.3, a UML diagram sequence diagram. The Packet Capture module receives packets from the network and sends the raw data to the Data Preprocessor. The Anomaly Detector then examines the pre-processed data to find any unusual activity. The Alert System receives the results if an anomaly is found, creates alerts, and shows them to the administrator on the Dashboard. The dashboard also has access to previous alert logs for additional monitoring and analysis.

* + 1. **USE CASE DIAGRAM**

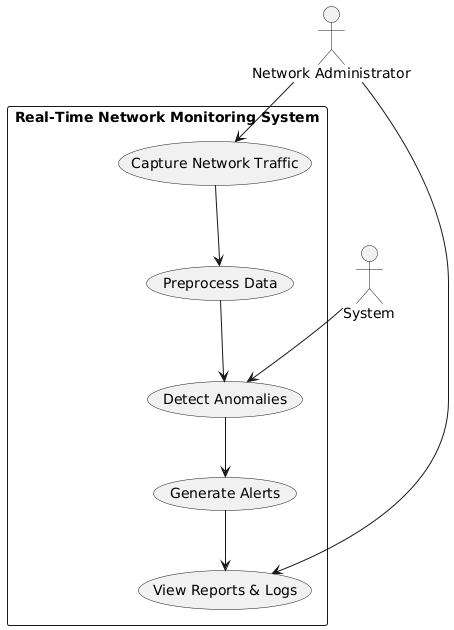


Figure 4.4: **Use Case Diagram**

The figure 4.4 diagram The main interactions in the Real-Time Network Monitoring and Anomaly Detection System are depicted in this use case diagram. The network administrator starts traffic capture and uses logs and reports to keep an eye on system activity. Capturing network traffic, preprocessing the data, identifying anomalies, and producing alerts are some of the interrelated use cases that the system itself uses to process data. A streamlined detection and response process is ensured by the logical flow of each function into the next. The figure illustrates the real-time collaboration between human and system actors to maintain network security.

* + 1. **SEQUENCE DIAGRAM**

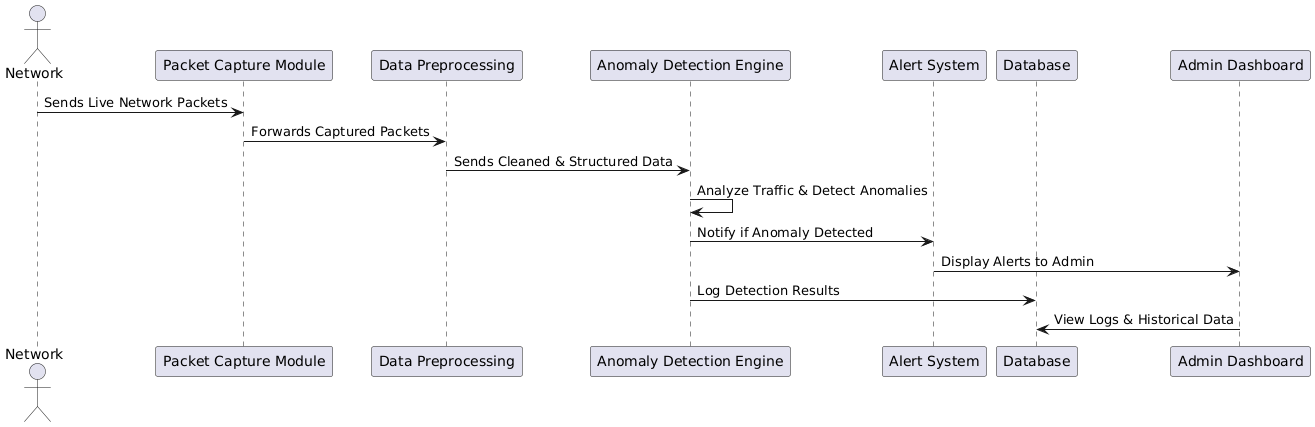


Figure 4.5: **Sequence Diagram**

The Real-Time Network Monitoring and Anomaly Detection System's step-by-step interaction is depicted in this sequence diagram. The Packet Capture Module receives live network packets from the network and forwards them to the Data Preprocessing unit for structuring and cleaning. The Anomaly Detection Engine then examines the refined data to find any odd trends. The Alert System sends real-time alerts to the Admin Dashboard upon detecting an anomaly. Effective network monitoring and decision-making are supported by the simultaneous logging of detection results into the database, which the dashboard can access to view historical trends and logs.

**MODULE DESCRIPTION**

The Essential components for effective network security are included in the Real-Time Network Monitoring and Anomaly Detection System. The Data Preprocessing Module cleans and arranges the data after the Packet Capture Module gathers real-time traffic. The data is examined by the Anomaly Detection Engine to look for odd trends that might point to security risks. The Alert System flags detected anomalies and notifies administrators via an intuitive Admin Dashboard.

* + 1. **MODULE1: DATASET PREPROCESSING `**

Since most machine learning algorithms are parameterized, it is impossible to predict how they will behave based just on the data that has been processed. Furthermore, the effectiveness of AIDS models is significantly impacted by random factors. After that, the parameters' behavior needs to be adjusted to obtain a satisfactory assessment. Finding and fixing mistakes, inconsistencies, and discrepancies in data before using it for analysis or modeling is known as data cleaning. It is a crucial stage in the preprocessing of data, particularly for applications involving deep learning.

* + 1. **MODULE 2: DATA TRANSFORMATION**

For ML-based NIDS, protecting a network of heterogeneous connected devices is a difficult undertaking. Data transformation or normalization is essential to ML-based NIDS optimization and facilitation. Real network traffic or benchmarked datasets are skewed and not normally distributed. Because normalization tends to increase the general structure and relation among features, machine learning algorithms tend to perform better when the data is normalized. Finding the best data transformation or normalization for the data or dataset is a challenging task, though. To determine the best normalization for the three datasets, the statistical method suggested in the paper is applied in this investigation. In terms of implementation and computational requirements, the suggested statistical method is straightforward and effective. For a feature selection based on correlation, normalizing the dataset makes the correlation between the features more noticeable and appropriate.

* + 1. **MODULE 3: AUTOENCODER & DECODER**

An artificial neural network type called an autoencoder (AE) is made to learn a condensed and effective representation of the input data, which can subsequently be utilized to reconstruct the original input. An autoencoder's main goal is to decrease the input data's dimensionality while keeping the key characteristics that reveal its underlying structure. Autoencoders are useful for tasks like data compression, denoising, and anomaly detection because they train the network to map the input data into a lower-dimensional space, eliminating noise and redundant information. An encoder, a bottleneck layer, and a decoder make up an autoencoder's architecture.

In order to capture the most significant features and gradually reduce the dimensionality of the input data, the encoder must compress it into a lower-dimensional representation. The compressed version of the data is stored in the bottleneck layer, sometimes referred to as the latent space or latent vector. This layer is crucial because it determines the model's ability to capture and represent the input in a compact form while keeping only the most crucial information and eliminating less crucial details. In contrast, the decoder uses this compressed latent vector to try to regenerate the original input as precisely as possible from the learned representation. An advanced variant known as a stacked autoencoder (SAE) uses a deep neural network with multiple hidden layers in both the encoder and decoder.

More intricate feature extraction is made possible by this deep learning technique, which also helps the model learn hierarchical representations and identify higher-level, more abstract patterns in the data. A stacked autoencoder is especially useful for tasks like image processing, speech recognition, and anomaly detection because each layer learns a more complex representation of the data. The output of one encoder layer is used as the input for the next in a stacked architecture. This helps to create increasingly complex abstractions, which eventually improve the original input's reconstruction and boost performance across a range of applications.

* + 1. **STEP 2: PROCESSING OF DATA**

To guarantee that your data is in a format that can be used to train a model, data preprocessing is essential. The data can be processed as follows:

• Packet Capture: Use tools for packet sniffing to record live network traffic.

• Data preprocessing: organize and clean the recorded data, then extract important characteristics (such as protocol behaviors and traffic patterns).

• Anomaly Detection: Use statistical models or machine learning to examine the preprocessed data for odd trends.

• Normalization/Standardization: To prevent features with disparate scales from skewing the model's performance, normalize numerical values. Z-score standardization or Min-Max scaling can be used to accomplish this.

* + 1. **STEP:3 SPLIT THE DATA**

Following preprocessing, the data must be divided into training and test sets. Typically, there might be a 20% test and 80% training ratio.

* + 1. **DATASETS SAMPLE**

**Event Type:** Type of activity (e.g., login, error).

**Error Code:** The error code (if any).

**CPU Usage (%):** Percentage of CPU usage at the time of the event.

**Memory Usage (%):** Percentage of memory usage.

**Label:** Whether the event is normal or anomalous.

* + 1. **STEP 4: BUILDING THE MODEL**
* **Packet Capture**: Capture live network traffic using packet-sniffing tools.
* **Data Preprocessing**: Clean, structure, and extract key features from raw data.
* **Feature Engineering**: Generate meaningful features for anomaly detection.
* **Anomaly Detection**: Use machine learning or statistical methods to identify unusual patterns.
* **Alert Generation**: Trigger real-time alerts when anomalies are detected.
* **Visualization**: Display alerts and network health on a dashboard.
* **Database Logging**: Log traffic, anomalies, and alerts for future analysis.
* **Response and Mitigation**: Take actions to block threats or mitigate risks.
* **Continuous Improvement**: Update models using historical data to enhance detection accuracy.
  + 1. **STEP 5: COMPILING AND TRAINING THE MODEL**
* **Compilation**: **Data Compilation**: Capture and preprocess network traffic data, extracting key features for analysis.
* **Model Training**: Train machine learning models (e.g., Decision Trees, Autoencoders) on the data to learn normal patterns and detect anomalies.
* **Continuous Learning**: Continuously update the model with real-time data and feedback to improve detection accuracy.

This process enables effective and adaptive anomaly detection in real-time network monitoring.

# CHAPTER 5

# IMPLEMENTATION AND TESTING

## INPUT AND OUTPUT

**Network traffic data: obtained from real datasets or simulated configurations, such as Mininets, in IoT environments**

* + 1. **PREDICTED Output**

A screenshot of a computer

AI-generated content may be incorrect.

**Figure 5.1** below shows the synthesized output

Figure 5.1**:** **Predicted Output of The Project**

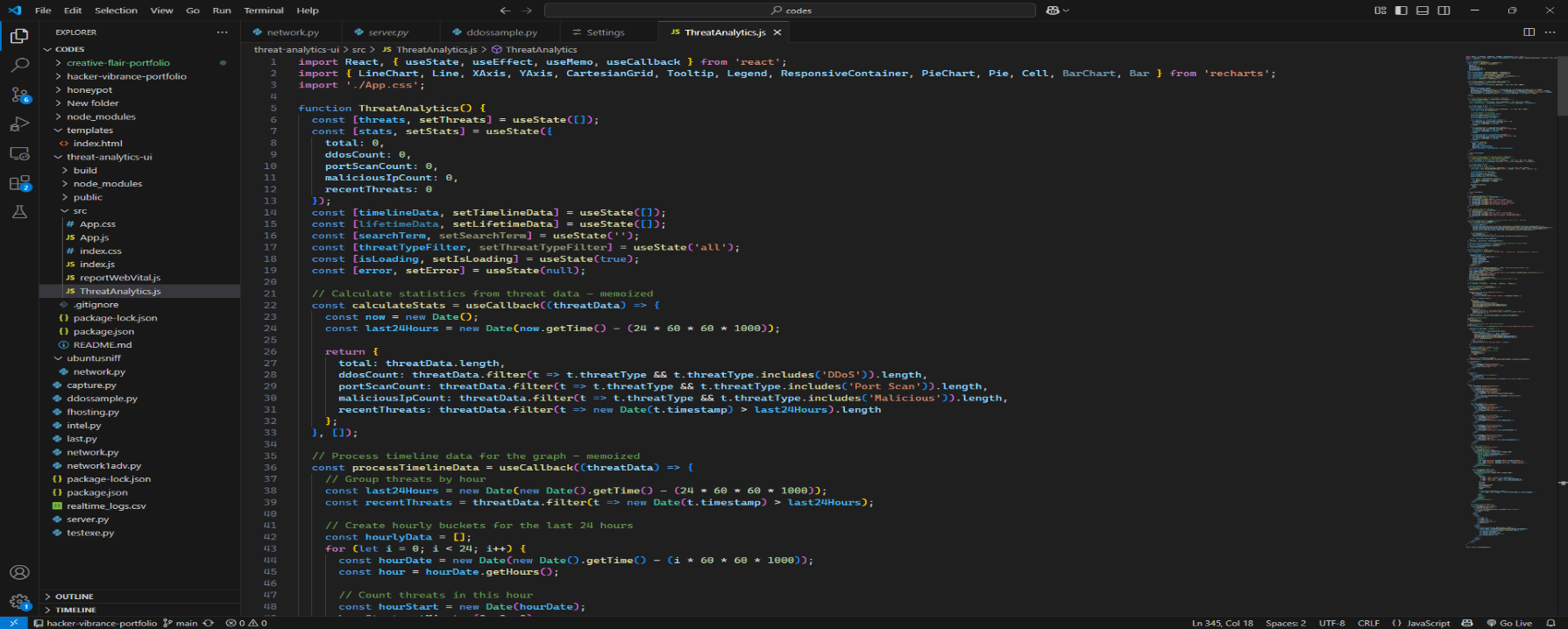
## TESTING

The Real-Time Network Monitoring and Anomaly Detection system's accuracy, dependability, and efficiency in real-world scenarios are guaranteed by the testing phase. It entails confirming that the system can identify irregularities in network traffic while reducing false positives and false negatives. Functional testing is the first step in the process, during which each module—packet capture, preprocessing, anomaly detection, alert generation, and reporting—is examined separately to make sure it operates as intended. Integration testing is then done to make sure that every module functions as a whole in a real-time setting.

* + 1. **TYPES OF TESTING**

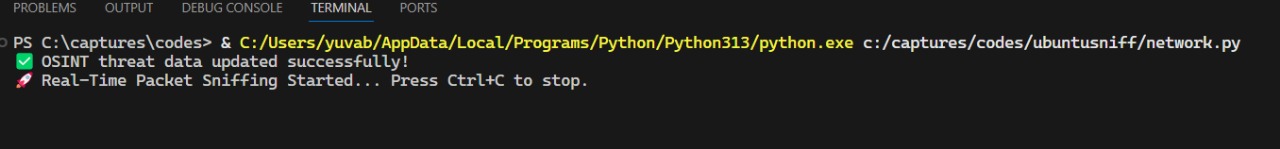
* + 1. **UNIT TESTING**

Unit testing is a beneficiable software testing method where the units of source code is tested to check the efficiency and correctness of the program. **Figure 5.2** below contains the code for the correctness and efficiency

****  **INPUT:**

**Figure 5.2**: Unit Testing of the code

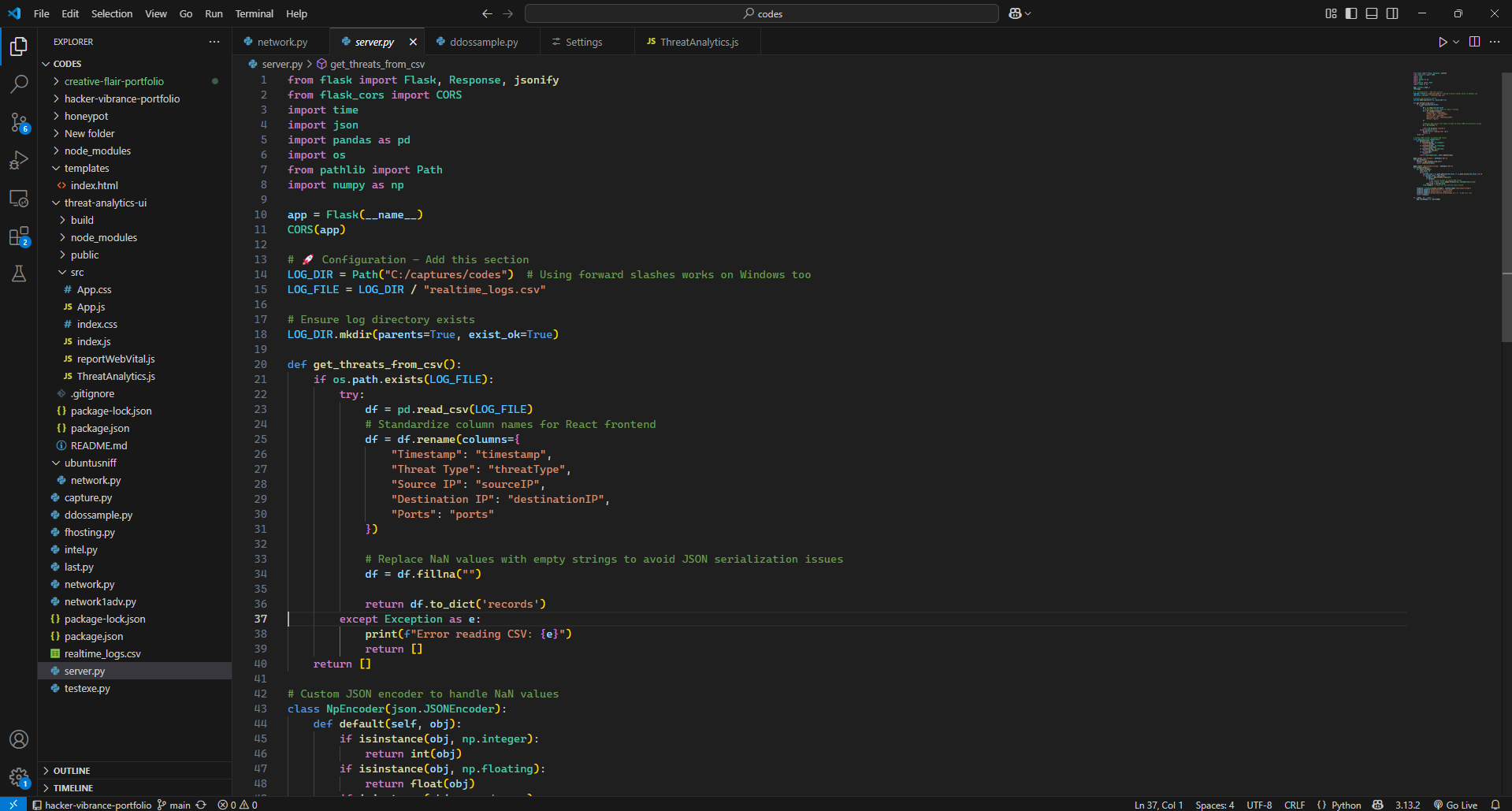
 **TEST RESULT**



**Figure 5.3**: Result of Unit Testing

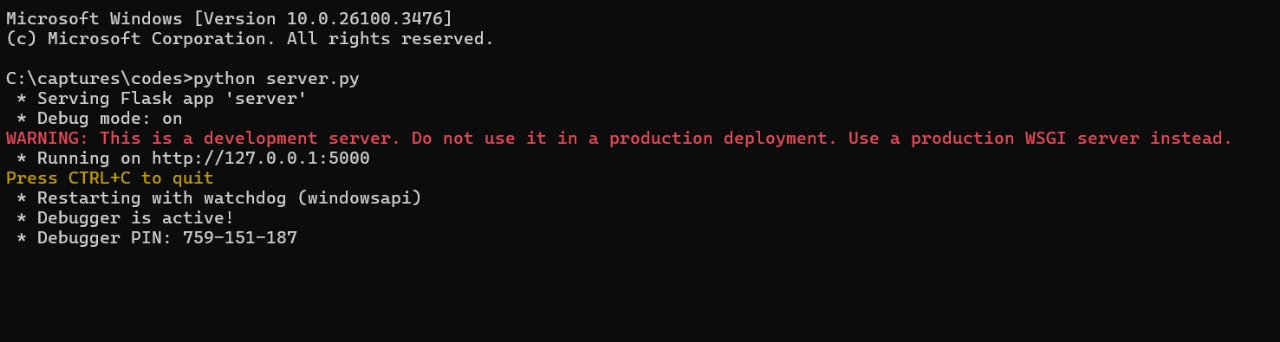
* + 1. **INTEGRATION TESTING**

**INPUT:**



**Figure 5.4:** Testing Anomaly detection Integration

**TEST RESULT**



**Figure 5.5**: Output of anomaly detection

* + 1. **FUNCTIONAL TESTING**

**INPUT**

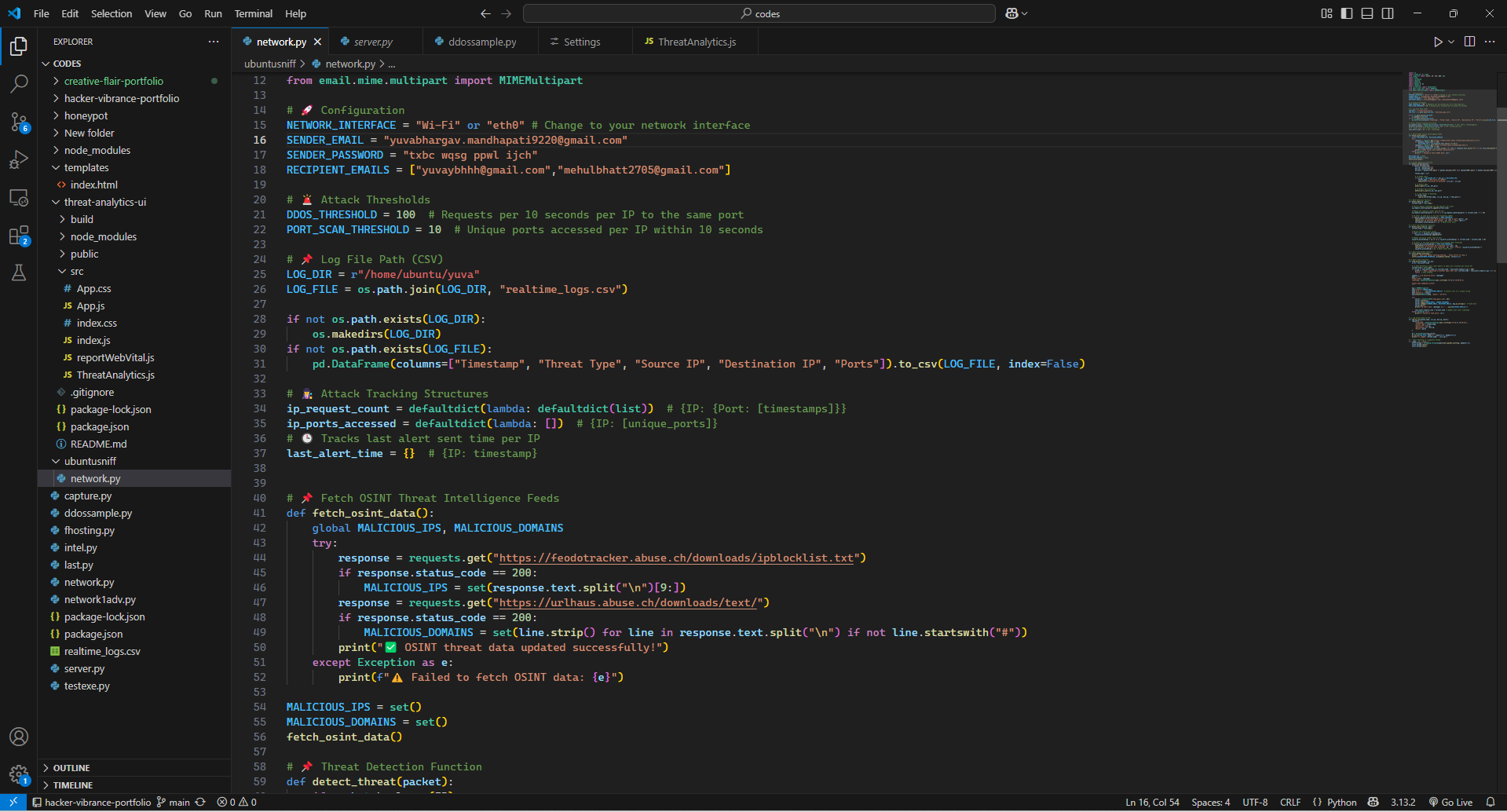
****

Figure 5.6: **Training And Testing Split**

**TEST RESULT**

• Verified CSV output creation

• Checked for anomaly columns and anomaly counts

• Complete pipeline tested with clean input and expected anomalies

# CHAPTER 6

# RESULTS AND DISCUSSIONS

## EFFICIENCY OF THE PROPOSED SYSTEM

The ability of the suggested Real-Time Network Monitoring and Anomaly Detection system to provide quick, precise, and scalable threat detection in dynamic network environments demonstrates its effectiveness. The system's real-time processing capability, which permits ongoing monitoring of live network traffic without causing appreciable latency, is one of its main advantages. While the data preprocessing module efficiently filters, cleans, and organizes incoming traffic for quick analysis, the use of optimized packet capturing tools guarantees little packet loss. Only pertinent data is sent to the detection engine thanks to this optimized data flow, which also lowers overhead.

At the anomaly detection module, which is driven by deep learning methods like autoencoders or machine learning algorithms like decision trees and clustering models, is the brains behind the system. These models are taught to identify typical network behavior patterns and highlight any deviations that might indicate possible security risks. The high detection accuracy and low false-positive rate of these models demonstrate their effectiveness, which makes the system intelligent and dependable. In order to facilitate prompt intervention and shorter incident response times, real-time alert generation makes sure that administrators are informed of suspicious activity right away.

## COMPARISON OF EXISTING AND PROPOSED SYSTEM

By providing increased efficiency, accuracy, and adaptability, the suggested Real-Time Network Monitoring and Anomaly Detection system overcomes the drawbacks of current systems. Conventional systems frequently rely significantly on detection techniques that are based on rules or signatures and are limited to identifying known threats. These systems frequently produce a large number of false positives and have trouble identifying new or developing cyberattacks. Furthermore, a large number of current solutions work in batch mode, which analyzes traffic data after it has been gathered. This causes a delay in threat detection and response.

On the other hand, the suggested system includes real-time packet processing and capture, which makes it possible to identify and react to questionable activity right away. It uses machine learning and anomaly detection techniques to detect deviations from typical traffic behavior, which enables it to detect zero-day or previously unseen attacks. When compared to conventional methods, this approach dramatically lowers the false-positive rate. The modular and scalable architecture of the suggested system is another significant enhancement. The suggested solution is made to scale well, allowing for continuous monitoring without compromising performance, whereas existing systems might become unreliable or inefficient when managing large-scale or high-speed networks. By giving network administrators access to visual analytics and real-time alerts, the incorporation of an intuitive dashboard also improves system usability.   
Additionally, the suggested system facilitates ongoing education. In contrast to static rule-based models, it can be retrained using administrator input and updated traffic patterns, gradually enhancing its detection capabilities. This guarantees that the system will continue to adapt to changing cyberthreats.  
In conclusion, the suggested system provides a real-time, intelligent, and scalable approach to network security, capable of effectively and efficiently detecting both known and unknown threats, whereas existing systems are constrained by their static nature, dependence on known signatures, and delayed analysis.

# CHAPTER 7

**CONCLUSION AND FUTURE ENHANCEMENTS**

## CONCLUSION

To sum up, the Real-Time Network Monitoring and Anomaly Detection project effectively illustrates a clever and effective strategy for protecting contemporary networks. The system efficiently and quickly detects known and unknown threats by combining real-time data capture, sophisticated preprocessing, and machine learning-based anomaly detection. This suggested model, in contrast to conventional rule-based systems, greatly lowers false positives and increases detection accuracy by adjusting to changing network behaviors and developing over time through continuous learning. Administrators can react swiftly to possible threats thanks to the system's improved usability brought about by the installation of a centralized alerting and visualization dashboard. Furthermore, the system can manage high-traffic environments and adjust to future network growth and complexity thanks to its modular and scalable design.

Overall, this project emphasizes how crucial intelligent, real-time monitoring solutions are to preserving network security by providing a proactive defence against ever-more-sophisticated cyberattacks. The developed system provides a solid basis for future improvements, including support for encrypted traffic analysis, automated response mechanisms, and deeper AI integration.

## FUTURE ENHANCEMENTS

By adding automated response mechanisms to immediately block or isolate threats, the Real-Time Network Monitoring and Anomaly Detection system can be further enhanced. In secure settings, increasing support for encrypted traffic analysis with privacy-preserving methods can improve detection precision. Anomaly prediction could be enhanced by incorporating sophisticated deep learning models, such as transformers or LSTM. The system's scalability and coverage can also be increased through cloud integration and IoT device monitoring. Last but not least, by incorporating a self-learning feedback loop, the model will be able to continuously adjust to changing threats with little assistance from humans.

**7.3 RESULTS**

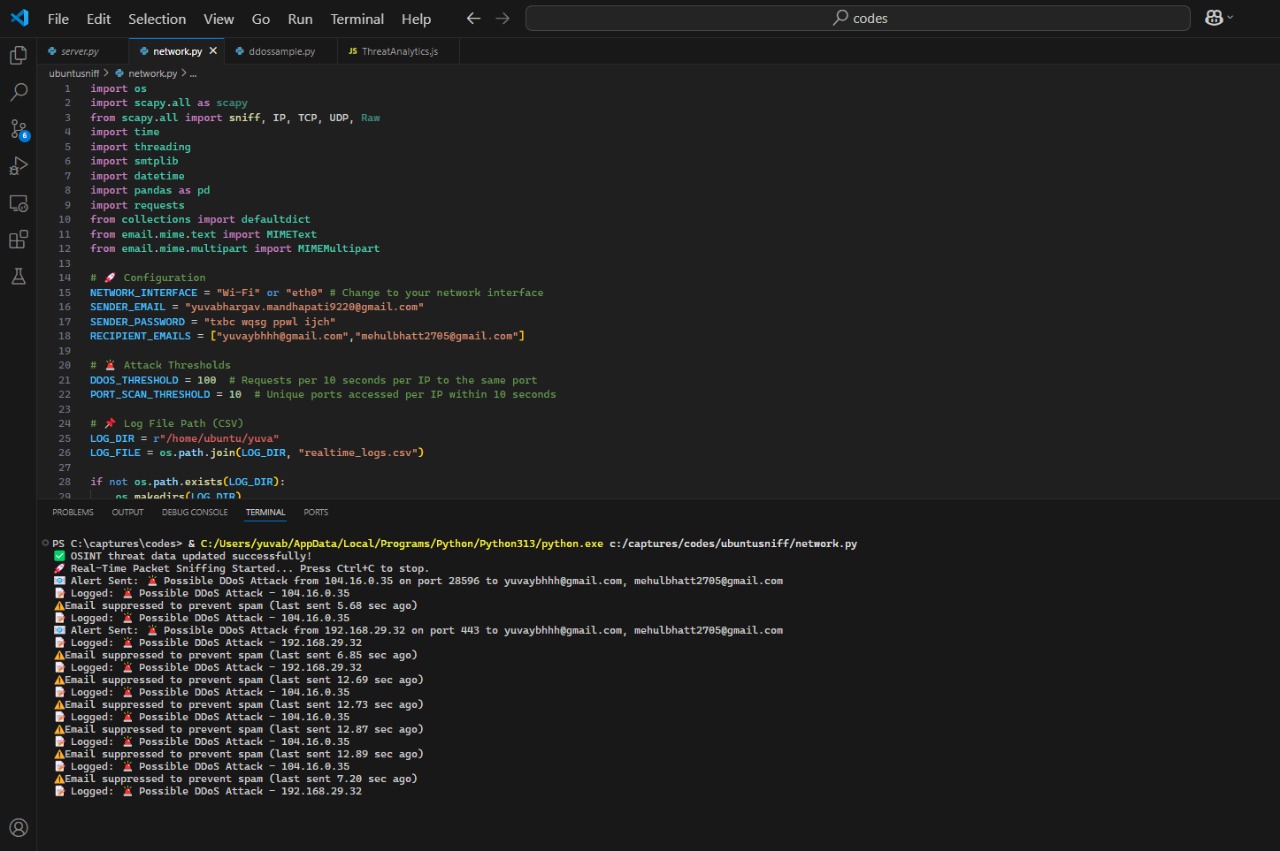


Figure 7.1: **Result of anomaly detection**

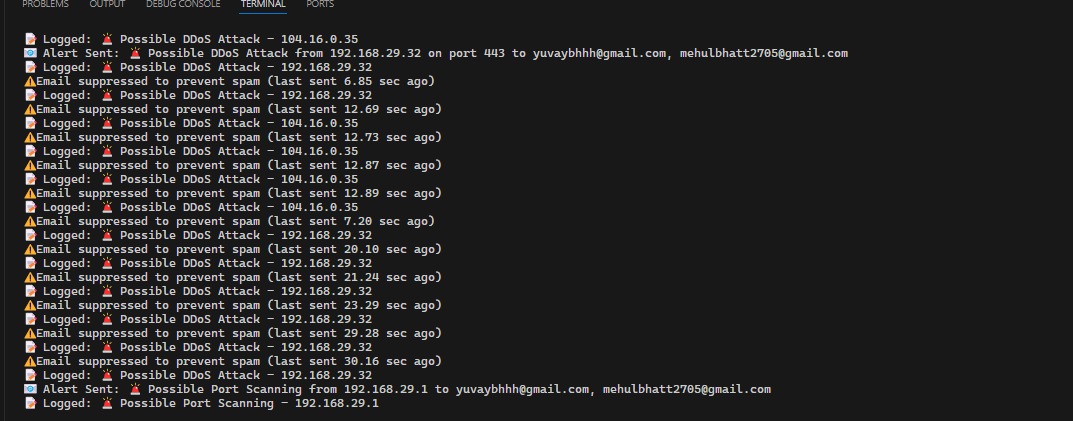
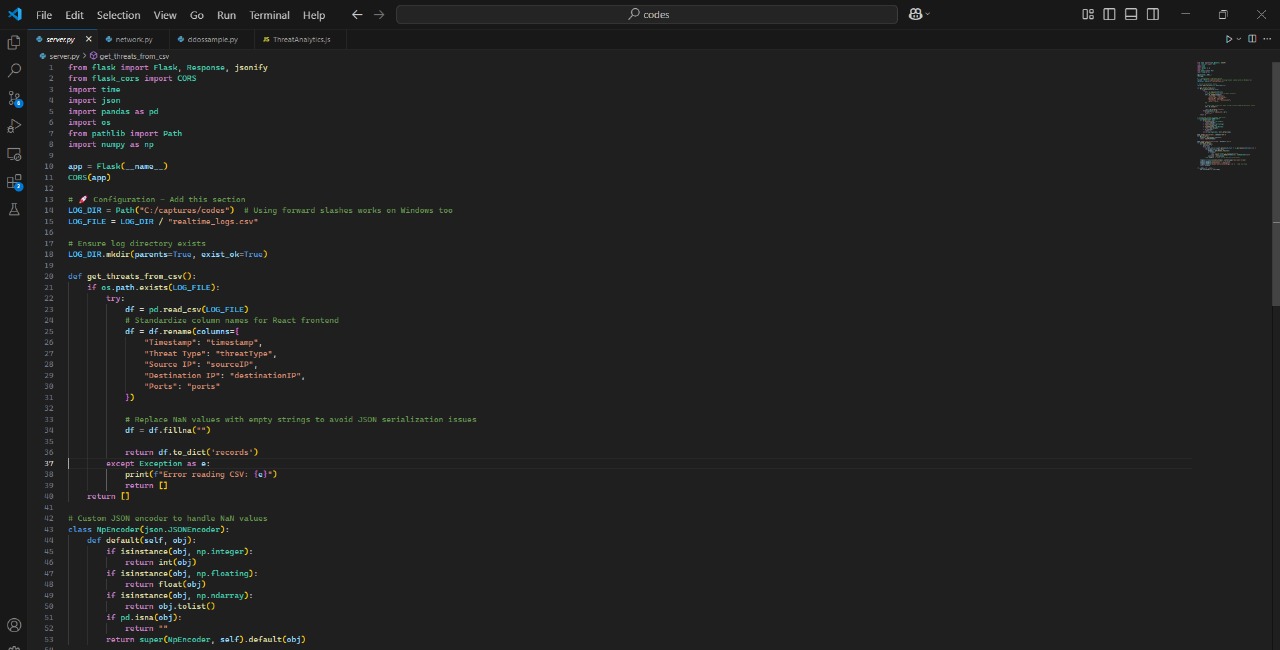


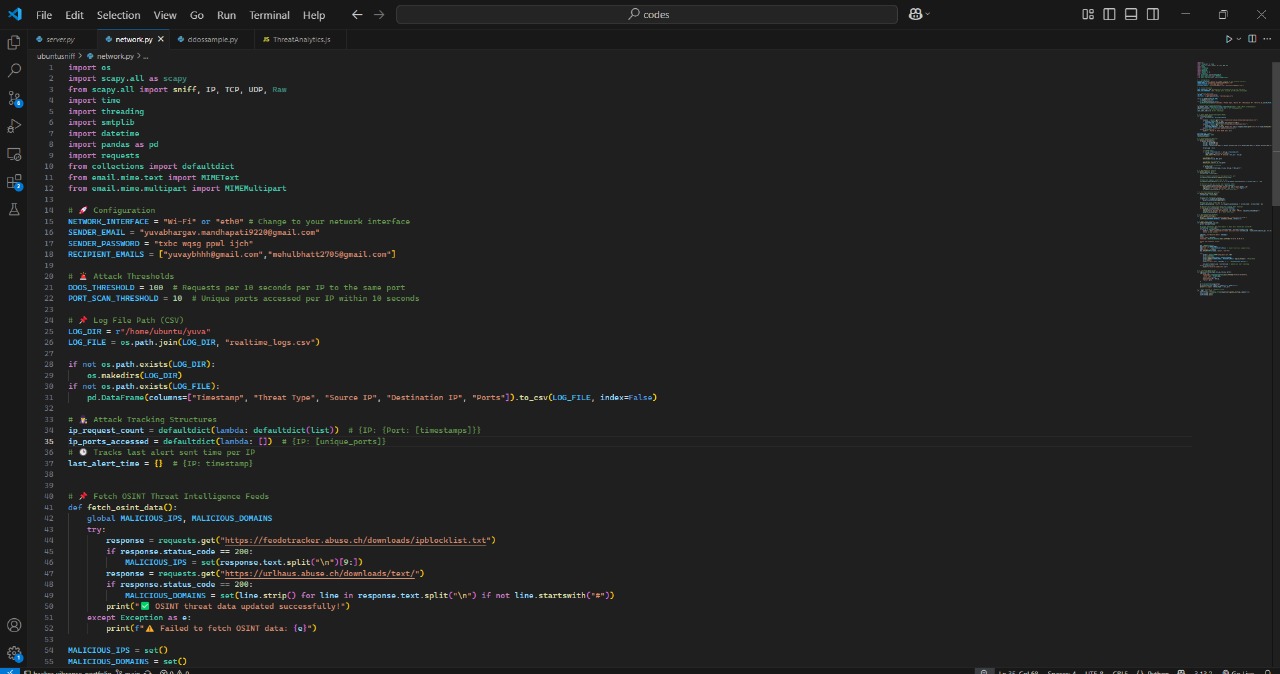
Figure 7.2: **Final Result of anomaly detection**

# CHAPTER 8

**SOURCE CODE & POSTER PRESENTATION**

## 8.1 SAMPLE CODE





# REFERENCES

1. **M Sabbir Salek, Pronab Kumar Biswas, Jacquan Pollard, Jordyn Hales, Zecheng Shen,Vivek Dixit, Mashrur Chowdhury, Sakib Mahmud khan, Yao Wang** A Novel Hybrid Quantum-Classical Framework for an In-Vehicle Controller Area Network Intrusion Detection IEEE Access, 2023
2. **Hongyu Yang, Fengyan Wang** Wireless Network Intrusion Detection Based on Improved Convolutional Neural Network IEEE Access, 2019
3. **Liqun Yang, Jianqiang Li, Liang Yin, Zhonghao Sun, Yufei Zhao, Zhoujun Li** Real-Time Intrusion Detection in Wireless Network: A Deep Learning-Based Intelligent Mechanism IEEE Access, 2020
4. **Josy Elsa Varghese, Balachandra Muniyal** An Efficient IDS Framework for DDoS Attacks in SDN Environment IEEE Access, 2021
5. **Laisen Nie, Yixuan Wu, Xiaojie Wang, Lei Guo, Guoyin Wang, Xinbo Gao,Shengtao L**i Intrusion Detection for Secure Social Internet of Things Based on Collaborative Edge Computing: A Generative Adversarial Network-Based Approach IEEE Transactions on Computational Social Systems, 2021
6. **Xiaokang Zhou, Wei Liang, Weimin Li, Ke Yan, Shohei Shimizu, Kevin I-Kai Wang** Hierarchical Adversarial Attacks Against Graph-Neural-Network-Based IoT Network Intrusion Detection System IEEE Internet of Things Journal, 2021
7. **Paulo Freitas De Araujo-Filho AntÃ´nio J. Pinheiro, Georges Kaddoum, Divanilson R. Campelo, Fabio L. Soares** An Efficient Intrusion Prevention System for CAN: Hindering Cyber-Attacks with a Low-Cost Platform IEEE Access, 2021
8. **C. Young, J. Zambreno, H. Olufowobi and G. Bloom**, Survey of automotive controller area network intrusion detection systems, IEEE Design Test, vol. 36, no. 6, pp. 48-55, Dec. 2019.
9. **W. Lo, H. Alqahtani, K. Thakur, A. Almadhor, S. Chander and G. Kumar**, A hybrid deep learning based intrusion detection system using spatial temporal representation of in-vehicle network traffic, Veh. Commun., vol. 35, Jun. 2022.
10. **L. Ashiku and C. Dagli**, Network intrusion detection system using deep learning, Proc. Computer. Sci., vol. 185,2021.
11. **M. A. Siddiqi, W. Pak and M. A. Siddiqi**, A study on the psychology of social engineering-based cyberattacks and existing countermeasures, Jun. 2022.
12. **F. L. Fessant, A. Papadimitriou, A. C. Viana, C. Sengul and E. Palomar**, A sinkhole resilient protocol for wireless sensor networks: Performance and security analysis, computer. Commun., vol. 35, no. 2, pp. 234-248, Jan. 2012.
13. **R. Dorai**, Performance evaluation of worm hole attack on ad hoc on demand multipath distance vector routing, Wireless Pers. Commun., vol. 66, no. 3, pp. 864-871, 2014.Y. Xu and H. Zhao, Intrusion detection alarm filtering technology based on ant colony clustering algorithm, Proc. 6th Int.Conf. Intel. Syst. Design Eng. Appl., pp. 470-473, May 2016.
14. **S. Agrawal and J. Agrawal**, Survey on anomaly detection using data mining techniques, Procedia compute. Sci., vol. 60, pp. 708-713, Jan. 2015.
15. **Z. Fan, P. Kulkarni, S. Gormus, C. Efthymiou, G. Kalogridis, M. Sooriyabandara, et al.,** Smart grid communications: Overview of research challenges solutions and standardization activities, IEEE Commun. Surveys Tuts., vol. 15, no. 1, pp. 21-38, 1st Quart.2013.

|  |  |  |
| --- | --- | --- |
| SRM INSTITUTE OF SCIENCE AND TECHNOLOGY  (Deemed to be University u / s 3 of UGC Act, 1956) | | |
| Office of Controller of Examinations | | |
| REPORT FOR PLAGIARISM CHECK ON THE DISSERTATION / PROJECT REPORT FOR UG / PG PROGRAMMES  **(To be attached in the dissertation / project report)** | | |
| 1 | Name of the Candidate | MEHUL BHATT  SHAZIN NAVAS  YUVA BHARGAV |
| 2 | Address of Candidate | Chennai -600089  **Mobile Number: +**91 8851773917,  +91 7356845754,  +91 9100814317 |
| 3 | Registration Number | RA2111030020062, RA2111030020074, RA2111030020112 |
| 4 | Date of Birth | 27/05/2004, 03/03/2003, 01/07/2004 |
| 5 | Department | Computer Science and Engineering with specialization in Cybersecurity |
| 6 | Faculty | Engineering and Technology |
| 7 | Title of the Dissertation / Project | REAL TIME NETWORK MONITORING AND ANOMALY DETECTION |
| 8 | Whether the above project / dissertation is done by | Individual or group: Group  (Strike whichever is not applicable)   1. If the project / dissertation is done in group, then how many students together completed the project : 03 2. Name and Reg NO:   MEHUL BHATT [RA2111030020062], SHAZIN NAVAS [RA2111030020074], YUVA BHARGAV[RA2111030020112] |
| 9 | Name and address of the Supervisor / Guide | S.Lakshmi,  Assistant Professor,  Department of CS & GT,  SRMIST, Ramapuram  **Mail ID : lakshmi9@srmist.edu.in**  **Mobile Number :** 9962485494 |
|  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 10 | Name and address of the Co- Supervisor /Guide | **Mail ID:**  **Mobile Number:** | | |
| 11 | Software Used | Python, Anaconda, Jupyter Notebook Tensorflow, Keras, Numpy, Pandas,matplotlib,scikit-learn | | |
| 12 | Date of Verification |  | | |
| 13 | **Plagiarism Details: (to attach the final report from the software)** | | | |
| **Chapter** | **Title of the Report** | **Percentage of similarity index (including self citation)** | **Percentage of similarity index (Excluding self citation)** | **% of plagiarism after excluding Quotes,**  **Bibliography, etc.,** |
| **1** | REAL TIME NETWORK MONITORING AND ANOMALY DETECTION |  |  |  |
| **Appendices** | |  |  |  |
| We declare that the above information has been verified and found true to the best of our knowledge. | | | | |
| **Signature of the Candidate** | | **Name and Signature of the Staff**  **( Who uses the plagiarism check software )** | | |
| **Name and Signature of the Supervisor / Guide** | | **Name and Signature of the Co-Supervisor / Co-Guide** | | |
| **Name and Signature of the HOD** | | | | |

.