CLOUD-BASED INTRUSION DETECTION SYSTEM: DESIGN AND DEPLOYMENT STRATEGIES

A PROJECT REPORT

Submitted by

KUNDURU NIKUNJ RAGHAV [CB.EN.U4CSE19030]
PENUGONDA SAI KOUSHIK [CB.EN.U4CSE19449]
RAVELLA ABHINAV [CB.EN.U4CSE19453]
SINGADI SHANTHAN REDDY [CB.EN.U4CSE19459]

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

AMRITA SCHOOL OF COMPUTING

AMRITA VISHWA VIDYAPEETHAM

COIMBATORE - 641 112

JUNE 2023

AMRITA VISHWA VIDYAPEETHAM

AMRITA SCHOOL OF COMPUTING, COIMBATORE – 641 112

BONAFIDE CERTIFICATE

This is to certify that the project report entitled **CLOUD-BASED INTRUSION DETECTION SYSTEM: DESIGN AND DEPLOYMENT STRATEGIES** submitted by Kunduru Nikunj Raghav(CB.EN.U4CSE19030), Penugonda Sai Koushik(CB.EN.U4CSE19449), Ravella Abhinav (CB.EN.U4CSE19453), Singadi Shanthan Reddy(CB.EN. U4CSE19459), in partial fulfillment of the requirements for the award of Degree **Bachelor of Technology** in Computer Science and Engineering is a bonafide record of the work carried out under our guidance and supervision at the Department of Computer Science and Engineering, Amrita School of Computing, Coimbatore.

Dr. Senthil Kumar T.
(Associate professor)
Department of CSE

Dr.Vidhya Balasubramanian
Chairperson
Department of CSE

Evaluated on:

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION

We, the undersigned solemnly declare that the project report **CLOUD-BASED INTRUSION DETECTION SYSTEM: DESIGN AND DEPLOYMENT STRATEGIES** is based on our own work carried out during the course of our study under the supervision of Dr. Senthil Kumar T., (Associate professor), Computer Science and Engineering, and has not formed the basis for the award of any other degree or diploma, in this or any other Institution or University. In keeping with the ethical practice in reporting scientific information, due acknowledgement has been made wherever the findings of others has been cited.

Kunduru Nikunj Raghav[CB.EN.U4CSE19030]-

Penugonda Sai Koushik[CB.EN.U4CSE19449]-

Ravella Abhinav[CB.EN.U4CSE19453]-

Singadi Shanthan Reddy[CB.EN.U4CSE19459]-

ABSTRACT

Cybersecurity threats are increasing in sophistication and frequency, necessitating robust security mechanisms to safeguard organizations. This project presents an IDS that utilizes various machine learning strategies to enhance the accuracy and efficiency of cyber-attack detection and prevention. By training and evaluating multiple machine learning models, including isolation forest, auto encoders, feed forward algorithm, and neural network algorithms, our proposed IDS achieves high accuracy, precision, and recall in identifying and thwarting diverse cyber-attacks with minimal false positives. The scalable and flexible design of our IDS makes it suitable for deployment in various environments, providing organizations with effective tools to proactively protect their networks from malicious actors and emerging threats.

In summary, our proposed IDS represents a significant advancement in cyber-attack detection and prevention systems. By harnessing cutting-edge advancements in machine learning and data analytics, our system empowers organizations across industries to safeguard critical assets and data from cyber threats. With its notable performance and adaptability, our IDS serves as an essential tool for organizations seeking to enhance their security posture and protect against evolving cybersecurity challenges.

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our beloved Satguru **Sri Mata Amritanandamayi Devi** for providing the bright academic climate at this university, which has made this entire task appreciable. This acknowledgment is intended to thank all those people involved directly or indirectly with our project. We would like to thank our Pro Chancellor **Swami Abhayamritananda Puri**, Vice Chancellor **Dr.Venkat Rangan.P** and **Dr.Bharat Jayaraman**, Dean, Faculty of AI & Computing(Computing), Amrita Vishwa Vidyapeetham for providing us with the necessary infrastructure required for the completion of the project. We express our thanks to **Dr.Vidhya Balasubramanian**, Chairperson, Department of Computer Science Engineering and Principal, School of Computing, Amrita Vishwa Vidyapeetham, **Dr.C.Shunmuga Velayutham and Dr.N.Lalithamani,** Vice Chairpersons of the Department of Computer Science and Engineering for their valuable help and support during our study. We express our gratitude to our guide, **Dr. Senthil Kumar T** for their guidance, support and supervision. We feel extremely grateful to **Dr. T Gireesh Kumar, Dr. Lalithamani** for their feedback and encouragement which helped us to complete the project. We would also like to thank the entire fraternity of the Department of Computer Science and Engineering. We would like to extend our sincere thanks to our family and friends for helping and motivating us during the course of the project. Finally, we would like to thank all those who have helped, guided and encouraged us directly or indirectly during the project work. Last but not the least, we thank God for his blessings which made our project a success.

KUNDURU NIKUNJ RAGHAV [CB.EN.U4CSE19030]
PENUGONDA SAI KOUSHIK [CB.EN.U4CSE19449]
RAVELLA ABHINAV [CB.EN.U4CSE19453]
SINGADI SHANTHAN REDDY [CB.EN.U4CSE19459]

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

IDS	Intrusion Detection System
CSE-CIC-IDS2018	Canadian Institute for Cybersecurity - IDS 2018 dataset
AD	Anamoly Based Detection
SD	Signature Based Detection
IDS	Intrusion Detection System
F1	F1-Score
AUC	Area Under the Curve
D	Dataset
T	Integer representing the tree count
S	Integer representing the subsample size
InputSize	Size of the input layer
HiddenSizes	List of sizes for hidden layers
OutputSize	Size of the output layer
IDS	Intrusion Detection Systems
IF	Isolation Forest
FFNN	Feed Forward Neural Network
EA	Ensemble Algorithm
DF	Detection Flow
EA	Evolutionary Algorithm

List of Symbols

α, β	Damping constants
$\leftarrow$	Assignment operator
$\leq$	Less than or equal to
[]	Empty list
$\{\dot{q}(t)\}$	Velocity vector
$\{\ddot{q}(t)\}$	Acceleration vector
δ (delta)	Error gradient
$*$	Element-wise product
$\circ$	Matrix multiplication
A^T	Transpose of matrix A

Chapter 1

INTRODUCTION

1.1 Introduction

Cyber-attacks are increasing in number on communication networks has made cyber security a critical concern for organizations. Network Intrusion Detection Systems (NIDS) are the frontiers in detecting several potential intrusions into network-based systems. However, with the increase in traffic and evolution of various new cyber-attacks, traditional methods are no longer sufficient; the use of more advanced strategies such as anomaly detection through the machine learning and flow-based inspection paradigm is required as demonstrated in Figure **1.1**.

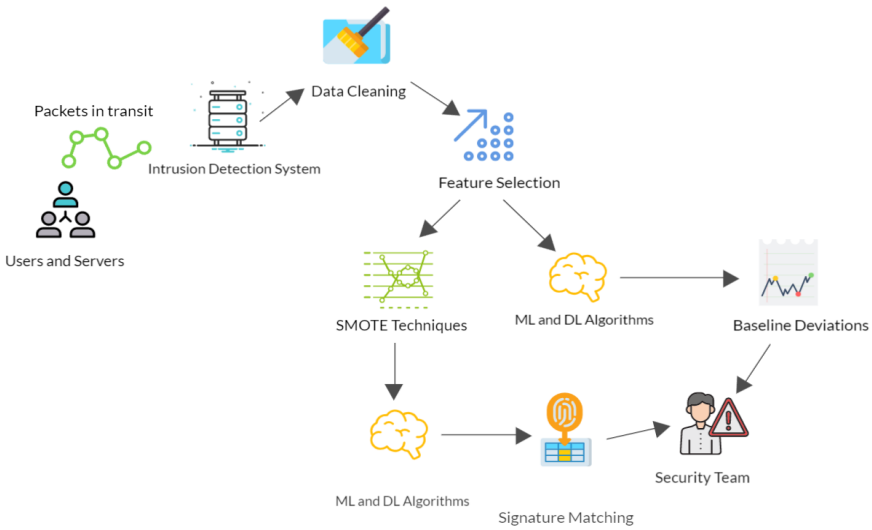

Figure 1.1: Working Model

In this project, our objective is to present a cloud-based Intrusion Detection System (IDS) for identifying potential threats in network traffic data that is being sent. The proposed system uses various techniques of machine learning algorithms, including isolation forest, auto encoders, feed forward algorithm, and neural network algorithms, to detect and prevent cyber-attacks. To train and evaluate our models, we used network traffic The information obtained originates from the Canadian Institute for Cybersecurity - IDS 2018 dataset (CSE-CIC-IDS2018) dataset., which is a labelled dataset con-

taining vast types of cyber-attacks on network-based services. Our Intrusion Detection System consists of multiple steps that incorporates a data various modules and phases. During the preprocessing step, we employed diverse data cleaning and data analysis methodologies to ensure the dataset's suitability for training our models. To train the models, we extract a variety of network flow-based features in the feature extraction module, including the amount of data transported, the time it took, and the number of packets.

We used a variety of techniques, including Hybrid Techniques, which combine Anamoly Based Detection (AD) and Signature Based Detection (SD) to provide a broader 2 and more precise detection, Anomaly Detection Techniques (AD), Signature Detection Techniques (SD), and other methods to find potential threats. The process of detecting anomalies entails creating a "normal profile" of the network and contrasting it with the observed occurrences. To find anomalies in network data, our AD techniques made use of various machine learning algorithms, such as isolation forest, and auto encoders.

On the other hand, Signature Detection involves comparing network patterns captured with previously known attacks. Our SD techniques utilized various pattern matching algorithms, such as regular expressions, to identify specific patterns associated with known attacks. We also employed a rule-based system, Snort, to detect predefined signatures of known attacks. To provide more accurate detection, we integrated both AD and SD techniques into our Hybrid Techniques. The hybrid-based detection approaches combine the strengths of both methods to provide a broader and more robust detection mechanism.

1.2 Proposed Model

Network Intrusion Detection Systems (NIDS) are crucial in today's life for detecting intrusions and attacks into computer devices and networks. As there are many modern cyber attacks have been come into plat traditional packet inspection methods are being ineffective in detecting these modern attacks on networks, which have become more multifaceted and sophisticated. To overcome this huddle now a days, modern

techniques such as flow-based inspection and machine learning are now used. In this research, we propose a unique novel NIDS model that makes use of both anomaly detection and signature-based detection techniques to improve and achieve better detection accuracy and efficiency. Our suggested model makes use of a hybrid strategy that blends the advantages of the two strategies to offer a more thorough detection capability. We begin by exploring the CSE-CIC-IDS-2018 dataset, which is a labelled dataset used for training and testing our proposed model. We perform pre-processing tasks such as feature selection and data cleaning to optimize the dataset utilized for training the machine learning algorithms. Next, two different methodologies were used: anomaly detection (AD) and signature detection (SD). Each of these methodologies required a different approach to pre-processing the data. For the AD methodology, correlation analysis was also performed to identify features with low correlation to the target variable and individual attacks. Features with low absolute correlation that were not crucial for the detection of individual attacks were eliminated. Mathematically, the correlation coefficient between two variables X and Y computed using the provided formula below:

$$r(X, Y) = \frac{\text{cov}(X, Y)}{\sigma_X \cdot \sigma_Y}$$

where $\text{cov}(X, Y)$ represents the covariance between X and Y, and σ_X and σ_Y represent the standard deviation of X and Y, respectively.

To address class imbalance, SMOTE and random under sampling techniques were applied to the reduced dataset.Mathematically, SMOTE (Synthetic Minority Over-sampling Technique) can be represented as:
Given a dataset, $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ where x_i represents the attributes and y_i represents the class label, the SMOTE algorithm generates new minority class samples by selecting k nearest neighbours of a sample and interpolating between them to create synthetic samples. The number of synthetic samples generated can be controlled by a parameter called the sampling ratio. Overall, the proposed model involved careful pre-processing of data using correlation analysis and techniques such as SMOTE and random under sampling to address class imbalance, resulting in optimized datasets for the respective AD and SD methodologies. We then introduce our proposed hybrid model that integrates both AD and SD techniques, which we evaluate against the traditional approaches. Our model demonstrates superiority in terms of false positive

rates, detection accuracy and computational efficiency, the proposed model incorporates machine learning methods, specifically Auto encoder and isolation Forest. These techniques enable the model to effectively identify both known and unknown threats, making it highly applicable in real-world scenarios.

1.2.1 Approach and Methodology

In this endeavor, we present a cloud-centric Intrusion Detection System (IDS) for identifying potential threats in network traffic data. Our IDS uses various techniques of machine learning algorithms, including isolation forest, auto encoder, and neural network, to detect and prevent cyber-attacks.

1.2.2 Need and Motivation

The motivation to develop and deploy Intrusion Detection Systems (IDS) lies in the increasing threat of cyber-attacks, which can cause significant harm to individuals, businesses, and governments. A cloud-based IDS can provide real-time monitoring and detection of potential threats, regardless of the location of the user or device. Cybercriminals are continuously developing new and innovative ways to exploit vulnerabilities in systems, and the damage caused by these attacks can be severe and long-lasting. So, we are using IDS to prevent or to reduce the damage these attacks might cause.

Summary

In summary, our proposed cloud-based IDS leverages machine learning algorithms to detect and prevent cyber-attacks on network-based services. Our system can help organizations mitigate the risks associated with cyber threats and improve their security posture. In the following sections, we will provide a detailed overview of our system architecture and the experimental results obtained through our performance evaluation.

Chapter 2

LITERATURE SURVEY

2.1 Intrusion Detection Systems (IDS)

The paper provided by Pandeeswari et al. (2022) gives us a description of several intrusion detection systems with a focus on the shortcomings of standard rule-based and signature-based methods.We'll examine how to create intrusion detection systems (IDS) utilising machine learning methods like decision trees, k-nearest neighbours, and support vector machines in the sections that follow. The study evaluates and analyses the effectiveness of several machine learning techniques using the NSL-KDD dataset, accounting for crucial performance factors as accuracy, precision, recall, and F1-score. The results of the study demonstrate the significant effectiveness of support vector machines in identifying network intrusions, suggesting their potential as a promising approach for future IDS development.

The research provided by (Illavarason and Kamachi Sundaram, 2019) a summary of numerous machine learning methods that can be used to create IDSs, followed by explanations of various feature selection strategies including acPCA, acCFS, and acRFE. Additionally suggests the use of various performance metrics to assess the efficiency of IDSs, such as accuracy, precision, recall, and F1-score. The performance of machine learning-based intrusion detection systems based on different feature selection approaches was evaluated using the NSL-KDD dataset. The research demonstrates that the KNN algorithm with the RFE feature selection approach delivers the highest accuracy and F1-score among the machine learning algorithms and feature selection methodologies examined.

2.2 Cloud IDS

The study done by Attou and et al. (2023), discusses the increasing demand for cloud-based intrusion detection systems due to the rising number of cyber-attacks in cloud

environments. In this study they use machine learning algorithms like Decision Trees, Random Forest, SVM, and KNN for detecting if there is any intrusion. From the results we can conclude that effective way for detecting any intrusions is by intrusion detection system based on cloud using machine learning techniques. The author also suggest that future work could focus on developing hybrid models that combine machine learning with other deep learning techniques to further enhance the accuracy and efficiency of IDS.

This paper by Singh and et al. (2021). provides an overview of the various challenges faced by intrusion detection in IoT networks and highlights the limitations of classic intrusion detection systems. It suggests that an effective IDS can be built using a multi-classifier approach using deep learning algorithms like CNNs (Conventional Neural Networks) and RNNs (Recurrent Neural Networks). The study conducts experiments using the CIC-IDS2017 dataset and evaluates the models on performance metrics, like accuracy, precision, recall, and F1-score. In conclusion the study demonstrates that the CNN-RNN hybrid model is particularly effective in detecting several types of cyber-attacks in IoT networks and could be a promising approach for building IDSs in the future.

2.3 IDS for IoT Networks

The study review the existing literature conducted by] Slevi and Visalakshi (2021) ion deep learning-based IDSs for IoT and classify them into three categories: network-based, hostbased, and hybrid IDSs. The paper also examines the datasets employed and performance metrics utilized for evaluating the efficacy of IDSs, while shedding light on the drawbacks associated with employing deep learning techniques for IDS in IoT environments. The outcomes obtained from the experiments lead to the conclusion that deep learning-based intrusion detection systems exhibit promising outcomes in terms of attack detection. However, limitations of this approach include the necessity for substantial and diverse datasets to train and test the IDS, the intricacy of deep learning models, and the trade-off between computational complexity and detection accuracy.

The paper by Vijayalakshmi and et al. (2022) presents a unique methodology for Intrusion Detection System (IDS) in the context of Internet of Things (IoT) is proposed,

incorporating a customized optimizer and convolutional neural network (CNN). The proposed system aims to address the challenges faced by traditional IDS in detecting unknown or evolving attacks in IoT systems. The CNN model is designed to extract and learn the features of network traffic data, which are then classified as normal or anomalous traffic. The proposed approach outperforms other existing systems. We can conclude that proposed system offers improved accuracy in detecting unknown attacks in 3 IoT systems.

2.4 Deep learning based IDS

The research presented in this paper by Idrissi et al. (2022) suggests that existing intrusion detection systems may not be effective for IoT systems due to the unique characteristics such as resource constraints, heterogeneity, and large-scale deployment. A layered technique is designed to detect anomalies at multiple levels of the IoT architecture using a deep learning model. The accuracy and F1-score of this model are then examined using the NSL-KDD dataset, and the findings reveal that it beats previous IDSs.

2.5 Novel Approaches for IDS

The research presented in this paper by Fatani and et al. (2021) proposes an innovative approach is suggested for an Intrusion Detection System (IDS) created especially for IoT contexts. In order to improve performance, this strategy combines deep learning techniques with the improved transient search optimisation (ETSO) method. Convolutional neural networks (CNNs) and long short-term memory (LSTM) models are combined in the model to extract features from network traffic data. The model's weights are then optimised using the ETSO technique, which also enhances the model's functionality as a whole. The CICIDS2017 dataset is used to evaluate the suggested IDS, and the experimental results show great accuracy. The proposed strategy beats existing methods in terms of detection rate and false alarm rate.

2.6 Autoencoder and isolation forest based

This research by Sadaf and Sultana (2020) proposes an intrusion detection system based on the combination of autoencoder (AE) and isolation forest (IF) techniques. The AE is used to compress the input data into lower dimensions and then reconstruct the original data. The reconstruction error is calculated to detect any anomalous behaviour in the data. The isolation forest isolates the anomalous instances from the being network traffic. From the results we can say that, the combination of autoencoder and Isolation Forest can achieve a high detection rate and a low false alarm rate than existing IDSs.

2.7 Feature Selection and Classification Models for Anomaly Detection

The paper by Abbasi et al. (2021). explores the problem of detecting anomalies in IoT devices using machine learning techniques. The research proposes a two-step approach for anomaly detection. To find the features that are most pertinent to the classification process, execute feature selection first. Second, develop two classification models—artificial neural networks and logistic regression—to identify anomalies in IoT devices. The study emphasises the significance of feature selection, and an artificial neural network model outperforms a linear regression model in terms of accuracy.

2.8 Light-Weight IDS for IoT

The study by Zhao and et al. (2021) proposes a method to overcome the limitations of traditional intrusion detection systems, such as high computational complexity and resource consumption. The lightweight neural network model is used to process large amounts of data efficiently and detect intrusions. The authors evaluated their proposed approach using the NSL-KDD dataset, which includes various types of attacks and normal traffic in a simulated network environment. The study proves that lightweight neural network-based intrusion detection method provides an efficient solution for detecting intrusions in IoT systems.

The proposed system in this paper by Ghosh and et al. (2019) uses logistic regres-

sion algorithm to create base-line of normal behaviour of the host machine and determines whether there is any variation from the norm. The LR-HIDS is assessed using performance criteria like accuracy, detection rate, rate of false positives, and processing time. The aforementioned model's results show that LR-HIDS performs better than the conventional technique in terms of accuracy and detection rate while retaining a low false positive rate.

2.9 Survey and Comparisions of IDSs

In this paper by Vinolia et al. (2023) there is a thorough study is done to properly assess the performance metrics of several intrusion detection systems, including host-based, network-based, and hybrid approaches in a cloud context. The effectiveness of various approaches is demonstrated by comparing several machine learning and deep learning algorithms that can be utilised in intrusion detection systems.

This paper by HP, HP et al. (2021) proposes network-based intrusion detection solution that uses a variety of deep learning techniques and is cloud-based. In this system, network traffic datasets are used to train the convolutional neural network model to identify intrusions in real-time scenarios. The model is versatile and scalable to large-scale network attacks thanks to the cloud-based approach. Results of the method show that the suggested system is very good at spotting different network-based assaults with little false positive and high accuracy.

In this paper by Tait et al. (2021) various artificial neural networks, decision trees, k-nearest neighbours, support vector machines, and other algorithms are tested against the same performance criteria after being trained on the same network traffic dataset. The many parameters taken into account include accuracy, false positive rate, detection rate, and processing time for the data. The decision tree algorithm has the lowest rate of false positives, according to the data.

2.10 CNN based and RNN based Threat Detection

In this paper by Liu and Lang (2019) Lang a thorough survey is conducted on machine learning algorithms and deep learning algorithms. Various algorithms that can be used

in intrusion detection system are decision tree, k-nearest Neighbour, support vector machine, random forest, convolutional Neural Network and Recurrent Neural Network. These algorithms are evaluated against set performance metrics and in cloud-based environment. From the results it is evident that CNN and RNN are highly accurate compared to other traditional models.

This paper by] Liu et al. (2022) uses a deep learning approach for advanced threat detection in intrusion detection system. The proposed model is a combination of Long short-term memory model and traditional convolutional Neural Networks to effectively detect intrusions in network traffic. Both the models are trained on network traffic dataset to establish a baseline for networks normal behavior and detect if there are any abnormal traffic patterns.

2.11 Signature based and Anomaly based IDS

The research by Alam et al. (2019) examines various intrusion detection and prevention techniques including signature-based intrusion detection system, anomaly-based intrusion detection system and Hybrid approach and evaluates them on same performance metrics in cloud-based environment. This paper also discusses the challenges and limitations in detecting and preventing intrusion in cloud and also proposes a solution for the above-mentioned challenges. The results of the study indicate that by combining both signature-based intrusion detection system and anomaly-based intrusion detection system the model can identify better if there is any attack or not.

2.12 Hierarchical IDS for IoT

This research by Smys et al. (2020) suggests a model combining both signature-based intrusion detection system and anomaly-based intrusion detection system techniques. The model has a hierarchical architecture with multiple layers of detection models that are designed to work in real-time with minimal computational overhead. The said model is evaluated using IoT network traffic dataset and the results show that the proposed model is more accurate with a low rate of false positive than compared to existing intrusion detection systems. The proposed model is useful in securing vulnerable IoT

devices due to their large attack surfaces.

2.13 Characteristics and Requirements of IDS

This paper by Elrawy et al. (2018) surveys characteristics of different types of IDS such as signature-based, anomaly based, and hybrid approaches. It also discusses the challenges and requirements of IDS for IoT-based smart environments, 4 such as resource constraints, scalability, and the need for Realtime detection. The paper also gives insights on strengths and weakness of various intrusion detection systems in existence.

2.14 Summary

In the literature survey we looked at various approaches for developing Intrusion Detection Systems (IDSs) for different types of environments, such as computer networks, cloud-based systems, and IoT systems, using machine learning strategies and deep learning approaches. These approaches include using decision trees, k-nearest neighbor, and support vector machine algorithms for building IDSs, multi-classifier approach using deep learning algorithms like CNNs and RNNs, and a combination of autoencoder and isolation forest techniques. The studies demonstrate that these approaches outperform traditional rule-based and signature-based IDSs in terms of accuracy, fast rate. From the survey we can understand the importance of feature selection and the use of large datasets for training and testing IDSs, we can also see the need for developing hybrid models that combine machine learning with other deep learning techniques to enhance the accuracy and efficiency of IDSs. The studies also highlight the limitations of using deep learning techniques for IDSs in IoT environments, such as computational complexity and the inverse relation between computational complexity and detection accuracy.

2.15 Data Set(If applicable)

The dataset used in our project is the CSE-CIC-IDS2018 dataset, which stands for the Canadian Institute for Cybersecurity - Intrusion Detection System 2018 dataset. It is a labeled network traffic dataset that consists of various types of cyber-attacks on

network-based services. This dataset is designed to provide researchers with a benchmark dataset to evaluate their proposed intrusion detection systems.

The Canadian Institute for Cybersecurity (2018) dataset is one of the most comprehensive datasets available for network intrusion detection research. It contains over 16 million network flows, including both benign and malicious network traffic. The dataset covers various cyber-attacks, including DoS and DDoS, are prevalent, probing attacks, and botnet attacks. The dataset also contains various attack scenarios, such as single-step attacks, multi-step attacks, and multi-vector attacks.

We selected the CSE-CIC-IDS2018 dataset for our project for several reasons. Firstly, the dataset is realistic and representative of the types of cyber-attacks that occur in the real world. Secondly, it is a large and diverse dataset that provides us with ample data to train and evaluate our proposed intrusion detection system. Finally, the dataset is publicly available and well-documented, making it easy to reproduce our experiments and compare our results with other research studies.

In summary, the CSE-CIC-IDS2018 dataset is an ideal choice for our project as it is a comprehensive and representative dataset for network intrusion detection research. By using this dataset, we can train and evaluate our proposed intrusion detection system and compare its performance with other research studies.

2.16 Software/Tools Requirements

2.16.1 Python Programming Language

Figure 2.1: Python Programming Language

Python is renowned as a highly favored programming language due to its user-friendly nature, adaptability, and simplicity. It finds extensive application across diverse domains, including web development, data analysis, artificial intelligence, and machine learning. Python benefits from a robust and engaged community, providing an exten-

sive collection of libraries and frameworks that streamline development processes. Its unambiguous syntax and effective debugging tools contribute to its appeal for programmers at all skill levels. Python's widespread adoption has elevated it to the status of one of the most extensively utilized programming languages in contemporary times.

2.16.2 Visual Studio Usage

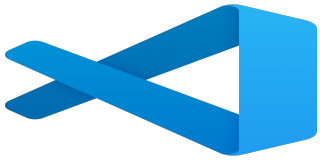

Figure 2.2: Visual Studio Code

Visual Studio Code (VS Code) is a popular, freely available code editor utilized extensively by developers in application development. Renowned for its user-friendly nature, customizable interface, and comprehensive feature set encompassing code highlighting, debugging, Git integration, and extensibility, VS Code caters to diverse programming languages and frameworks. This versatile tool empowers developers of varying expertise levels with an intuitive interface, advanced functionality, and ample customization possibilities, establishing it as a favored option for application development projects.

2.16.3 Google Colaboratory

Figure 2.3: Google Colaboratory

Google Colaboratory (Colab) is a cloud-based platform that offers a Jupyter notebook environment, enabling users to write and execute Python code effortlessly. It provides access to advanced hardware resources like GPUs and TPUs, making it particularly well-suited for running machine learning models and deep learning algorithms.

Colab stands out for its convenience, as it eliminates the need for installations or configurations, and it comes pre-loaded with the necessary libraries and dependencies. Moreover, Colab facilitates collaboration by allowing multiple users to simultaneously edit a notebook. As a result of its robust hardware capabilities and user-friendly interface, Colab has gained widespread popularity among machine learning researchers and practitioners.

2.16.4 Machine Learning

Machine learning is a specialized branch of artificial intelligence that concentrates on developing algorithms and models capable of autonomously improving and acquiring knowledge through data analysis. It finds applications in various domains like image recognition, speech processing, natural language understanding, recommendation systems, and fraud detection. The primary objective of machine learning is to construct models capable of making accurate predictions or decisions when presented with new data. This is achieved by training the models on a large dataset and optimizing them to minimize errors and improve their performance and practical applications, and it has the potential to revolutionize various industries and domains in the coming years.

2.16.5 Streamlit

Figure 2.4: Streamlit

Streamlit is a Python library for creating web applications that display data-driven visualizations and integrates machine learning models and data processing pipelines. It simplifies the process of creating interactive data-driven applications and provides support for various data sources. Streamlit's intuitive user interface and flexibility have made it a popular choice for data scientists and machine learning engineers.

Chapter 3

PROPOSED SYSTEM

3.1 System Analysis

System analysis plays a crucial role in designing and implementing an Intrusion Detection System (IDS). This phase involves identifying requirements, collaborating with stakeholders, and designing the system architecture. Advanced data analysis and processing techniques are developed to detect and respond to security threats effectively. Integration with existing infrastructure and considerations for scalability are carefully addressed. Thorough testing and validation procedures ensure the functionality and performance of the IDS. Comprehensive documentation is created to support system maintenance and future upgrades. Through a meticulous system analysis process, the IDS is customized to meet specific requirements, delivering a robust and reliable cybersecurity solution.

Figure 3.1: System analysis

3.1.1 System requirement analysis

The system requirement analysis for the IDS project involves identifying and defining the essential elements for developing an effective Intrusion Detection System. This includes determining the purpose, scope, functional and non-functional requirements,

and constraints. The IDS aims to monitor network traffic, detect attacks, generate alerts, integrate with existing tools, and ensure high performance, reliability, and compliance. Conducting a thorough analysis ensures that the project meets the organization's needs and aligns with industry standards.

3.1.2 Module details of the system

1. **Data Collection:** This module collects network traffic data from various sources and formats it for further analysis.
2. **Preprocessing:** The collected data is processed to remove noise, handle missing values, and normalize the features.
3. **Feature Extraction:** Relevant features are then taken out from the preprocessed data to represent network behavior and identify potential anomalies.
4. **Machine Learning:** Machine learning algorithms are used to train models on the labeled data and classify network traffic as normal or malicious.
5. **Alert Generation:** Detected anomalies trigger alerts to notify system administrators or security personnel for further investigation.
6. **Visualization:** Visual representations of network traffic patterns, attack trends, and system performance aid in monitoring and analysis.
7. **Reporting:** Detailed reports are generated to document detected threats, system vulnerabilities, and recommendations for improving network security.

3.2 System Design

3.2.1 Sequence Diagram

Sequence diagram demonstrates the collaboration among the Network Sensor, Signature-Based IDS, Anomaly-Based IDS, Decision Engine, Alerting Module, and Security Team in detecting and responding to potential network attacks in a Hybrid IDS environment.

- User sends packets over the network.
- Network Sensor captures the traffic.
- Captured traffic is forwarded to the Signature-Based IDS for analysis.
- If a match is found in the Signature-Based IDS's signature database, the Alerting Module generates alerts for the Security Team.

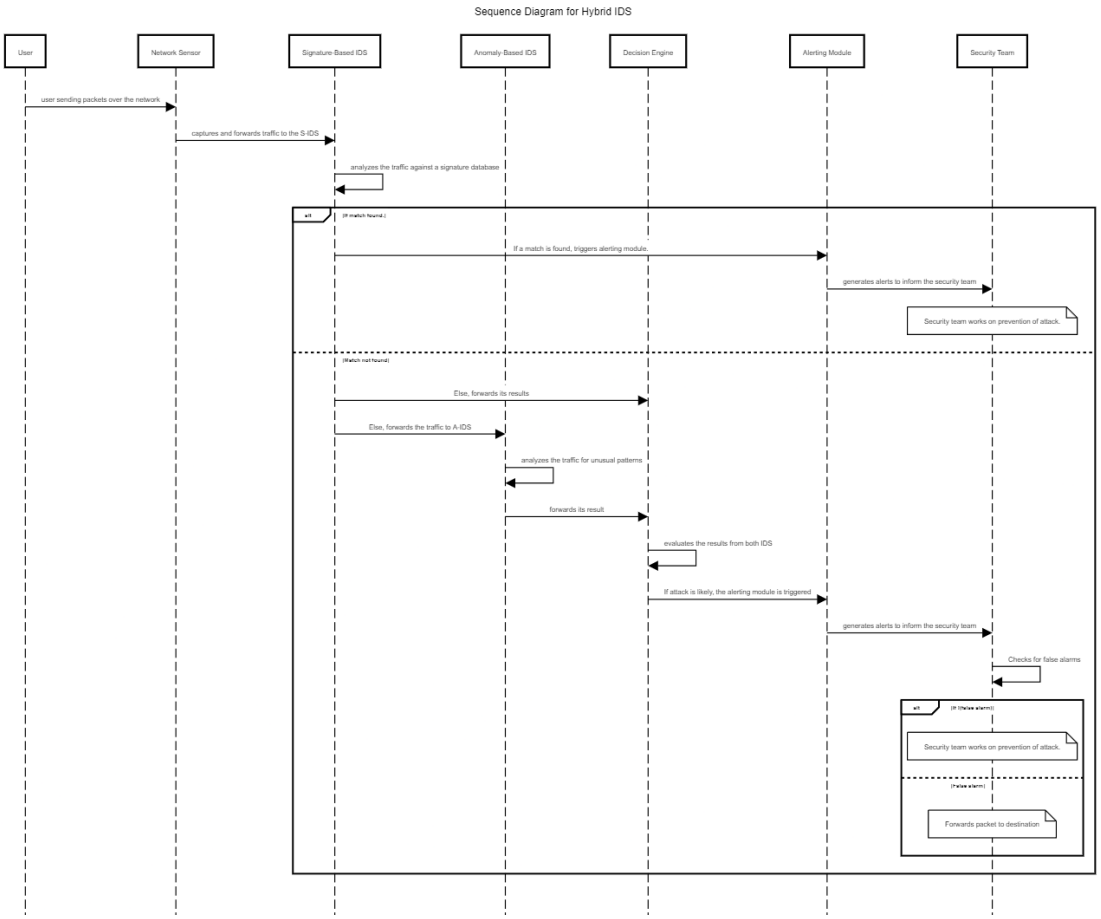

Figure 3.2: Sequence Diagram

- If no match is found, the traffic is forwarded to the Anomaly-Based IDS.
- The Anomaly-Based IDS examines the traffic for unusual patterns and sends the results to the Decision Engine.
- The Decision Engine evaluates the results from both IDS and determines the likelihood of an attack.
- If an attack is likely, the Alerting Module generates alerts for the Security Team.
- The Security Team reviews the alerts, checks for false alarms, and takes necessary actions for attack prevention.
- If a false alarm is detected, the Security Team forwards the packet to its destination without further action.

3.2.2 Flow diagram

3.2.3 Architecture of the project

The Intrusion Detection System (IDS) operates within a network architecture consisting of multiple routes, each with several connected devices. Data flows between these devices, generating packets that are inspected by the IDS. The IDS comprehensively analyzes the data, promptly alerting the security personnel responsible for monitoring in the event of any detected attacks. This proactive approach aims to prevent or mitigate potential damage caused by security breaches. Figure 3.3 provides an illustrative of this model.

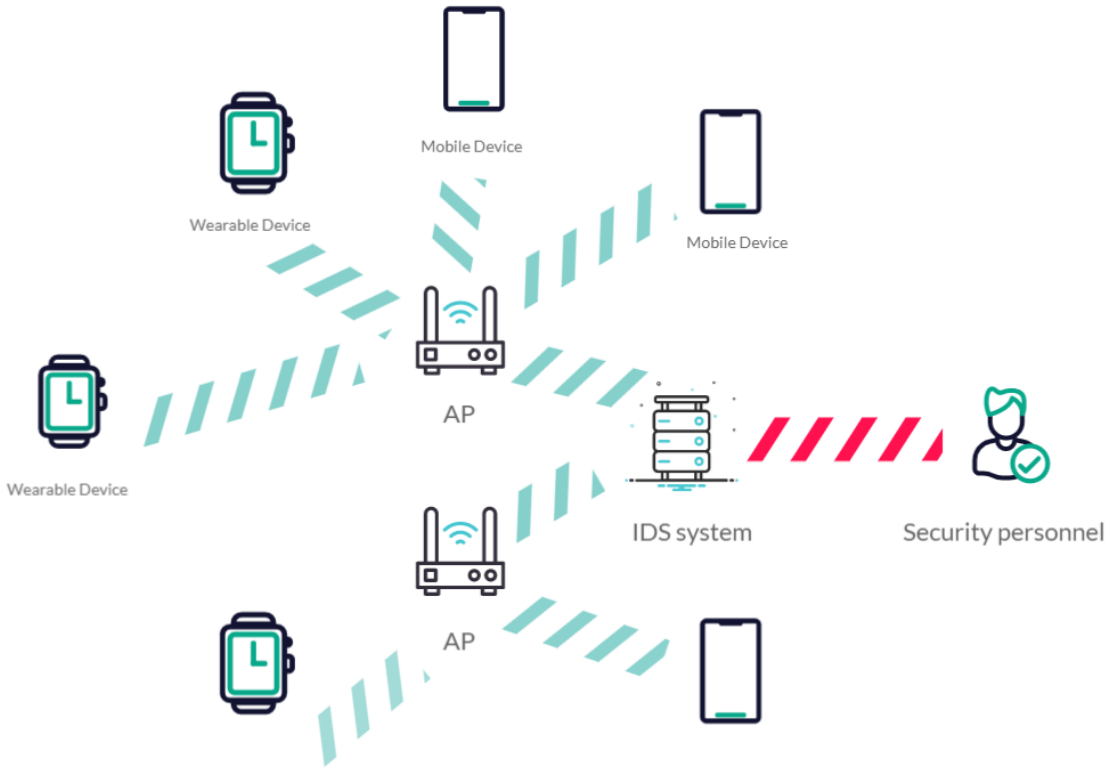

Figure 3.3: Architecture Diagram

Architecture Diagram for IDS

The IDS validates incoming network packets by concatenating them into a single dataset. The data is then analyzed, preprocessed, and cleaned to remove empty or unwanted values. Feature selection is performed to choose relevant columns from the dataset. The data is trained using anomaly, signature, and hybrid models to detect attacks. If an at-

tack is detected, an alert is raised for the security personnel to take appropriate action (Figure 3.4). The provided architecture diagram serves as an illustrative example of the IDS model.

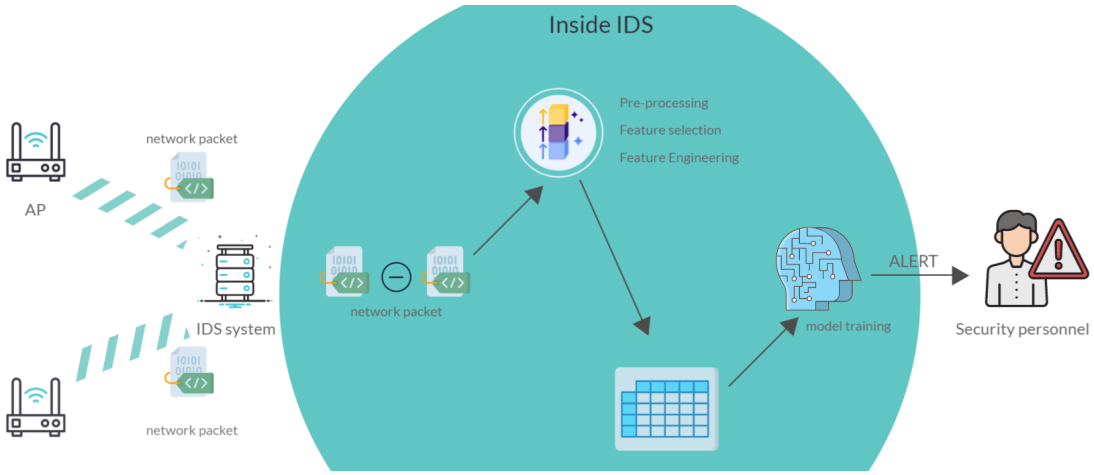

Figure 3.4: Architecture Diagram for IDS

Block Diagram showing process of Data collection

The CSE-CIC-IDS2018 dataset is a collection of real-world and synthetic network traffic captured from a simulated organization network environment (see Figure 3.5). It comprises approximately 16,000 labeled network flows, categorized as normal or malicious traffic. The malicious traffic encompasses various attack types, including botnet, DDoS, and port scanning. The network topology consists of interconnected subnets with hosts running on various operating systems such as Windows XP, Windows 10, Linux, Ubuntu and etc. The dataset also includes servers, gateway devices, and an attack subnet with Linux and Ubuntu operating systems. Overall, the CSE-CIC-IDS2018 dataset provides a comprehensive representation of network traffic for research and analysis purposes.

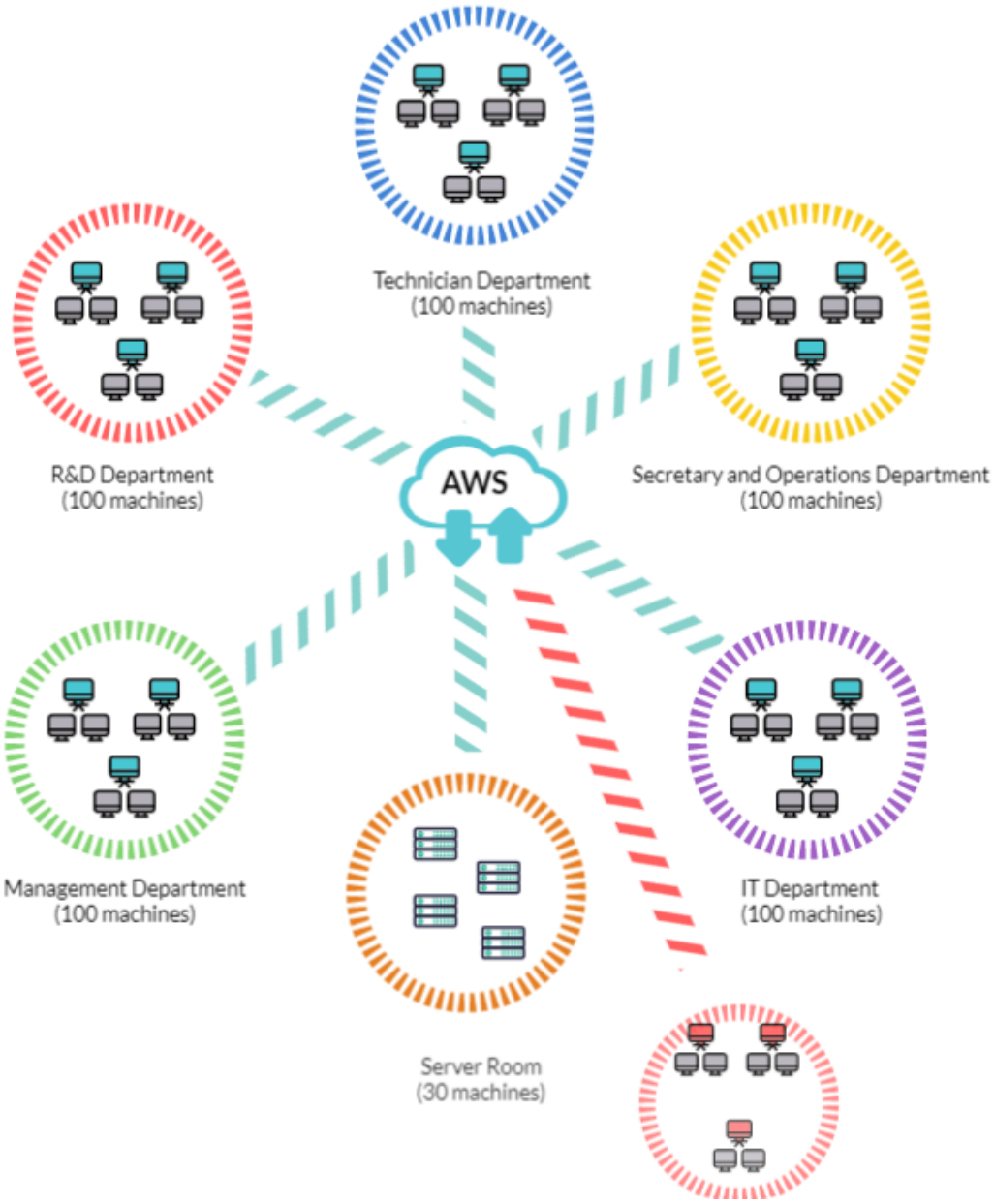

Figure 3.5: Block Diagram showing process of Data collection

Chapter 4

IMPLEMENTATION AND TESTING

Our project involved several key components, including machine learning, user interface development, and webpage creation. We successfully implemented these components to build an effective intrusion detection system.

4.1 Machine Learning phase

In the machine learning phase, we utilized state-of-the-art algorithms and techniques to train the intrusion detection system. We processed the CSE-CIC IDS 2018 dataset, which consisted of a comprehensive collection of network traffic data. This dataset comprised millions of rows and a large number of columns, capturing various attributes and characteristics of network flows. By applying advanced machine learning models, we achieved high accuracy and robust performance in detecting and classifying different types of attacks.

4.1.1 Anomaly Based Detection

Our approach, illustrated in Figure 4.1, encompasses multiple steps, including data collection, pre-processing, model building, anomaly detection, and alert generation. Firstly, we gathered data from various sources, utilizing the CSE-CIC-IDS2018 dataset, which encompasses network traffic, system logs, and application logs. The collected data was pre-processed to eliminate noise, irrelevant information, and any superfluous data.

Next, we employed the Isolation Forest algorithm as our machine learning technique to construct a model that captures the normal behavior of the IDS system. The algorithm was trained on a labeled dataset containing instances of both normal and known malicious activities, enabling the model to learn distinctive patterns and characteristics

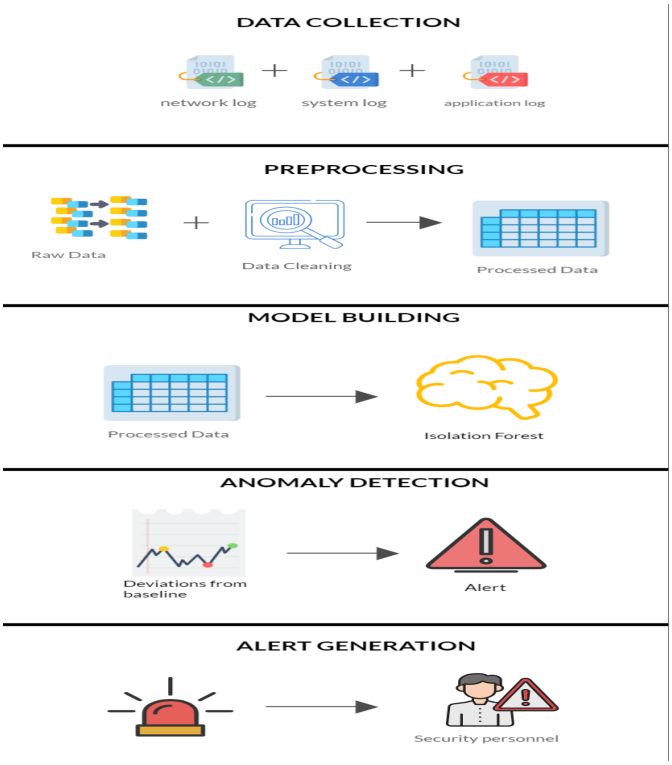

Figure 4.1: Working of Anomaly Based Detection

of different activities. At this stage, features with low absolute correlation, which were not crucial for the detection of specific attacks, were eliminated.

Once the model representing the normal behavior of the system was built, the IDS utilized it to detect any deviations from the expected behavior in real-time. Incoming data was compared against the established baseline of normal behavior, and if the data significantly deviated from the learned normal pattern, it was flagged as anomalous or malicious.

Whenever an anomaly was detected, the IDS generated an alert to notify the system administrator or security team, enabling them to take appropriate actions to prevent potential security incidents. While our approach proved effective in detecting anomalies, we encountered challenges such as the accurate labeling of training data, potential false negatives, and false positives, which could increase the workload for security analysts.

Isolation Forest Algorithm

Input: Dataset (D), number of trees Integer representing the tree count (T), subsample size Integer representing the subsample size (S)

Output: List of Trees

1. Initialize $Trees$ as an empty list.
2. Set $t \leftarrow 1$.
3. Randomly select a subsample of size SS from DDataset D and assign it to SubsampleSubsample.
4. Build a tree using the subsample SubsampleSubsample by invoking the `BuildTree` function and assign it to TTree.
5. Append T to $Trees$.
6. Repeat steps 3-5 for T times, incrementing t by 1 in each iteration.
7. Return $Trees$.

BuildTree Function

Input: Dataset D

Output: Tree structure

1. If the size of D is less than or equal to 1 or the depth limit is reached, create a leaf node with label 1 and assign it to $Node$.
2. Otherwise,
   (a) Randomly select a feature FF from the remaining features.
   (b) Find the minimum and maximum values of feature F in dataset D and assign them to $Minimum$ and $Maximum$.
   (c) Randomly select a split point P between $Minimum$ and $Maximum$.
   (d) Create an internal node with feature F and split point P and assign it to $Node$.
   (e) Filter instances in DD where feature F is less than or equal to P and assign them to $LeftSubset$.
   (f) Filter instances in DD where feature FF is greater than P and assign them to $RightSubset$.
   (g) Recursively build the left child of **!** (**!**)$Node$ by invoking the `BuildTree` function with input $LeftSubset$ and assign it to $Node.leftChild$.
   (h) Recursively build the right child of $Node$ by invoking the `BuildTree` function with input $RightSubset$ and assign it to $Node.rightChild$.
3. Return $Node$.

Anomaly-based detection offers several advantages. It can identify previously unknown or zero-day attacks that do not conform to known patterns of malicious activities, making it particularly useful in detecting new and emerging threats not captured by signature-based methods. Additionally, it can adapt to changes in the cloud environment and update the baseline of normal behavior accordingly, enabling continuous monitoring and detection. Furthermore, it minimizes false positives by considering normal variations in the cloud environment and only regarding significant deviations as potential anomalies.

However, this approach also presents challenges. The accuracy of labeling training data is crucial, as it directly influences the effectiveness of the anomaly detection model. False negatives may occur if actual malicious activities closely resemble normal behavior, while false positives may arise if normal behaviors are inaccurately captured during the training process, leading to unnecessary alerts and increased workload for security analysts.

4.1.2 Signature Based Detection

Signature-based detection is a widely employed technique in the field of cybersecurity, particularly within Cloud-Based Intrusion Detection Systems (IDS). This technique involves a series of steps aimed at identifying potential intrusions in network traffic or system logs. Our Cloud-Based IDS leverages both random forest and feed-forward neural network algorithms to perform signature-based detection.

The initial phase of our IDS entails the creation of a signature database, which consists of known patterns or signatures associated with malicious activities. These signatures serve as reference points for the IDS to compare against incoming data, allowing for the detection of potential intrusions.

In real-time, the IDS continuously monitors network traffic or system logs by capturing data packets or log entries as they traverse the cloud-based infrastructure. The captured data encompasses various information such as source and destination IP addresses, port numbers, protocol types, and payload contents.

To identify known signatures of malicious activities, the captured data is fed into a feed-forward neural network. This neural network is trained to recognize and classify patterns associated with malicious activities. Through multiple hidden layers, the neural network learns the underlying patterns and features of the data, enabling it to make predictions regarding the presence of known signatures of malicious activities. Figure **4.2** illustrates the working of Signature-Based Detection in our system.

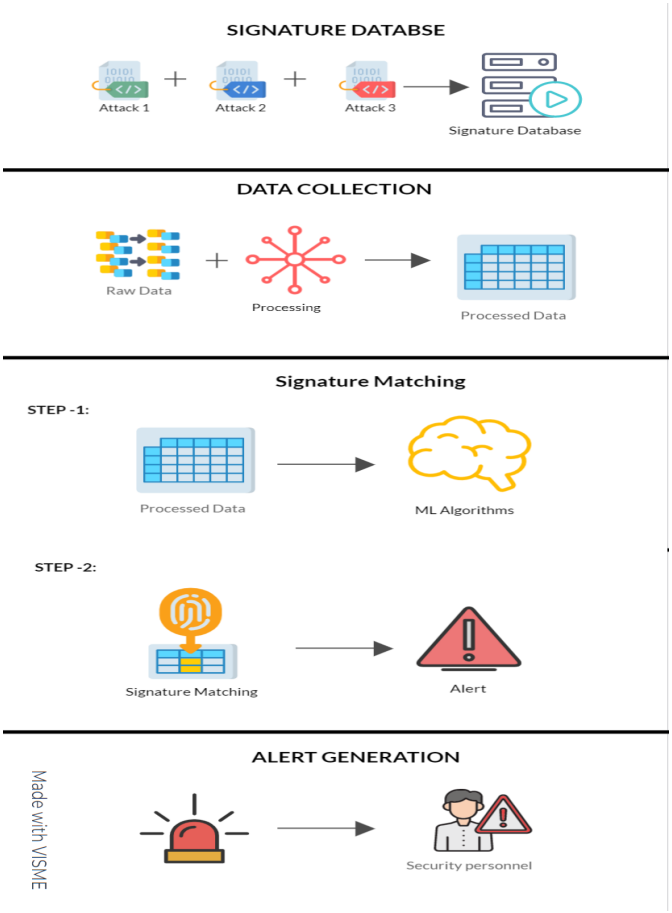

Figure 4.2: Working of Anomaly Based Detection

Implementation of the Feed-Forward Neural Network

Feed-forward neural networks are a prevalent type of artificial neural network architecture employed in various machine learning applications, including intrusion detection systems. In the context of signature-based detection, feed-forward neural networks are trained to classify network traffic or system logs as either benign or malicious based on patterns or signatures identified in the training data.

In our study, we utilized a feed-forward neural network with multiple hidden lay-

ers and implemented dropout regularization to enhance the accuracy and generalization capabilities of the model for signature-based intrusion detection. The neural network consisted of a flatten layer followed by several dense layers, with the output layer being a SoftMax layer that produced a probability distribution across seven classes representing potential intrusions.

To train the neural network, we obtained a dataset consisting of network traffic and system logs containing various types of known malicious activities (e.g., malware, port scanning, denial-of-service attacks) as well as benign traffic. The dataset was preprocessed and divided into training and testing sets. The training set was utilized to train the neural network, while the testing set was used to evaluate the model's performance.

The feed-forward neural network was trained using backpropagation and stochastic gradient descent in the training phase to minimize cross-entropy loss between predicted and true labels. Dropout regularization was utilized to prevent overfitting and improve the model's generalization. The model's performance on the testing set was assessed using metrics including accuracy, precision, recall, and F1-score, resulting in an F1-score of 0.90, indicating its efficacy in detecting potential intrusions.

In the event of a match being detected between the captured data and a known signature, the IDS generates an alert or notification to notify security personnel or system administrators about the potential intrusion. The alert includes details regarding the type of intrusion, the severity of the threat, and any other relevant information about the detected activity.

Upon receiving an alert, the security personnel can initiate appropriate actions to respond to the potential intrusion. These actions may include isolating the affected system, blocking the source IP address, or implementing incident response procedures to mitigate the threat effectively.

Feed-forward Neural Network Algorithm

Input: Size of the input layer (InputSize), List of sizes for hidden layers (HiddenSizes), Size of the output layer (OutputSize)

Output: NeuralNetwork - List of layers in the neural network

1. Initialize NeuralNetwork as an empty list.
2. For each $LayerSize$ in $HiddenSizes$:
   (a) Create a layer with input size $InputSize$ and layer size $LayerSize$, and assign it to $Layer$.
   (b) Append $Layer$ to $NeuralNetwork$.
   (c) Update $InputSize$ to $LayerSize$.
3. Create an output layer with input size InputSize$InputSize$ and output size $OutputSize$, and assign it to $OutputLayer$.
4. Append $OutputLayer$ to $NeuralNetwork$.
5. Return $NeuralNetwork$.

Forward Pass Function

Input: Neural Network - List of layers in the neural network, Input - Input data

Output: Output - Output prediction

1. Set $Output$ to $Input$.
2. For each $Layer$ in $NeuralNetwork$:
   (a) Update $Output$ by applying the activation function to the matrix multiplication of $Layer$ and $Output$.
3. Return $Output$.

Backward Pass Function

1. Calculate the error gradient δ as the element-wise product of the difference between $Target$ and $Output$ and the derivative of the $Output$.
2. For each $Layer$ in reverse order of $NeuralNetwork$:
   (a) Calculate the gradient of $Layer$ as the element-wise product of δ and the derivative of the $Layer$ output.
   (b) Update the weights of $Layer$ by subtracting the product of the learning rate and the matrix multiplication of the transposed.
   (c) Update δ as the matrix multiplication of the transposed $Layer$ weights and δ.

Within our Cloud-Based IDS involves the creation of a signature database, continuous monitoring of network traffic or system logs, comparison of captured data against the signature database using a feed-forward neural network, generation of alerts upon detection of a match, and implementation of appropriate response actions to address potential intrusions. While this approach is effective in detecting known patterns of malicious activities, it may have limitations in identifying novel or emerging threats that do not align with known patterns.

4.1.3 Hybrid Based

Continuing with our implementation, we introduce our proposed hybrid model, which combines both anomaly detection (AD) and signature-based detection (SD) techniques. This integration allows us to evaluate its performance against traditional approaches. Our hybrid model showcases superior performance in terms of detection accuracy, achieving lower false positive rates and higher computational efficiency compared to conventional methods. By leveraging machine learning techniques such as Autoencoder and Random Forest, our model is capable of detecting both known and unknown threats, making it well-suited for real-world applications.

Intrusion Detection Systems (IDS) plays a vital role in cybersecurity by combining signature-based and anomaly-based approaches to enhance detection capabilities. While signature-based IDS excel at identifying known threats, they often fall short in detecting zero-day vulnerabilities. On the other side, anomaly-based IDS tend to generate a significant number of false positives. To address these limitations, we propose an implemented hybrid IDS that combines two models: Isolation Forest (IF) and Feed Forward Neural Network (FFNN).

The FFNN was chosen over traditional Neural Networks due to its superior anomaly detection capabilities, while IF is more effective in detecting normal traffic patterns. Our hybrid IDS leverages a Stacking strategy, utilizing multiple layers of classifiers that build upon the results of previous layers. The final layer of the model employs Logistic Regression for making the ultimate decision.

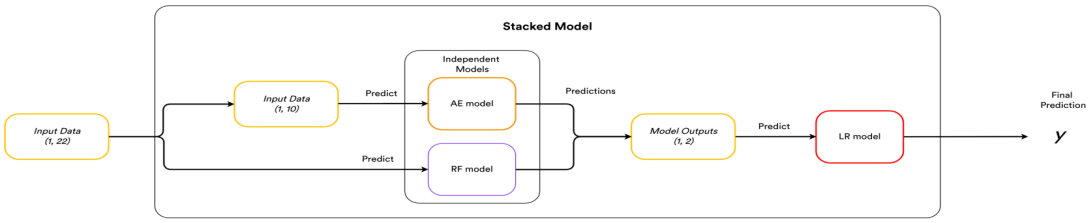

Figure 4.3: Hybrid Detection Model

System Overview

Our proposed hybrid IDS system takes pre-processed data streams (comprising 22 features) as input for both models. The Evolutionary Algorithm (EA) further pre-processes the input stream, eliminating 12 features. Subsequently, both models perform their predictions on the input flow, and these results are fed into the final model, which determines if the flow is anomalous.

Training Process

During training, the model follows a stratified holdout approach with two-thirds of the data used for training base models, and the remaining one-third reserved for training the final model. The Random Forest model is trained on the entire dataset, while the Auto-encoder utilizes a subset of ten features from a total of 22. Once the base models are trained, they predict the outputs of the Detection Flow (DF) set. These predictions, along with the DF class labels, are utilized to train a logistic regression model responsible for making the ultimate prediction.

Evaluation Results

Our hybrid IDS system was evaluated using the signature detection set, and the results demonstrate its significant performance improvement over the individual models. Specifically, the system achieves an impressive accuracy rate of 98%, representing an 8% improvement over FFNN and a 13% improvement over IF. Moreover, the system boasts a remarkably low false positive rate of 1.1%, compared to 3.3% for FFNN and 3.8% for IF. These results affirm the effectiveness of our proposed hybrid IDS system in detecting both known and unknown threats, while also producing fewer false positives. Thus, it is a reliable and accurate solution for intrusion detection.

Conclusion

In conclusion, our implemented hybrid Network Intrusion Detection System (NIDS) offers an effective and efficient approach to detect network anomalies in modern computer networks. By combining the strengths of anomaly and signature-based detection techniques, our hybrid IDS overcomes the limitations of traditional approaches, delivering enhanced detection capabilities for improved cybersecurity.

4.2 UI

In order to enhance user experience, we created a visually intuitive interface, using suitable frameworks and libraries. The GUI enabled users to interact with the system, customize settings, and access real-time or historical data pertaining to network traffic and detected attacks. The interface was designed to be intuitive, visually appealing, and responsive, ensuring a seamless user experience (see Figure 4.4). The GUI component played a crucial role in enhancing user engagement and facilitating efficient system management.

Figure 4.4: User Interface (UI)

In the web application, the prediction process relies on various fields to analyze network activities effectively. These fields include:

1. **Source Port**: The originating port number of the device initiating the communication.
2. **Destination Port**: The port number of the receiving device for the communication.
3. **Protocol Type**: The specific protocol used for the communication (e.g., TCP, UDP).

4. **Packet Size**: The overall size of the network packet, including header and payload.
5. **Forward Packet Size**: The size of the packet transmitted from source to destination.
6. **Backward Packet Size**: The size of the packet transmitted from destination back to source.

These fields play a crucial role in predicting and analyzing network activities for intrusion detection. By considering these parameters, the web application can effectively identify potential security threats, anomalies, or suspicious behavior. The utilization of these fields enhances the accuracy and reliability of the prediction model, allowing for proactive security measures and timely responses to potential threats.

4.2.1 Elaboration of Input fields

Protocol Type

In computer networking, protocol types such as TCP, UDP, and HTTP define the rules and standards for communication between network devices, refer **Table 4.1**. TCP ensures reliable data transmission with error detection, while UDP offers faster but less reliable communication. Understanding these protocol types is vital for network design, implementation, and troubleshooting.

Table 4.1: Protocol Types

Protocol Type	Port Number
TCP	6
UDP	17
HTTP	0
HTTPS	443

Source Port and Destination Port:

In the realm of computer networking, ports serve as communication endpoints, facilitating the identification of specific processes or services running on a host device. They play a vital role in ensuring the reliable delivery of data packets between hosts over a network.By assigning unique port numbers to different services, applications, or processes, data packets are accurately directed to their intended destinations, facilitating

effective network connectivity, performance analysis, and security implementations. Understanding source and destination ports is essential for troubleshooting, optimizing network performance, and implementing robust firewall rules and access controls.

Table 4.2: Accepted Port Numbers

Port Number	**Service**
21	FTP
22	SSH
23	Telnet
25	SMTP
53	DNS
67	DHCP
68	DHCP
80	HTTP
110	POP3
119	NNTP
123	NTP
143	IMAP
161	SNMP
443	HTTPS
8080	HTTP-Proxy

Packet Size, Forward Packet Size, Backward Packet Size:

Packet size refers to the amount of data contained in a single transmission packet. Forward packet size denotes the packet transmitted from the source to the destination, while backward packet size represents the packet sent from the destination back to the source. These sizes can differ due to protocol variations, network conditions, and routing paths. Understanding packet sizes helps optimize network performance, diagnose connectivity issues, and enhance the user experience.

Figure 4.5: Prediction Example

Prediction example

4.3 Webpage

In addition to the GUI, we developed a webpage to showcase our project and its functionalities (see Figure 4.6). The webpage served as a platform to provide project information, demonstrate the effectiveness of our intrusion detection system, and present the results of our experiments. It included descriptions, visualizations, and statistics (refer to Figure **??**) to highlight the performance and effectiveness of our system in identifying and mitigating cyber-attacks.

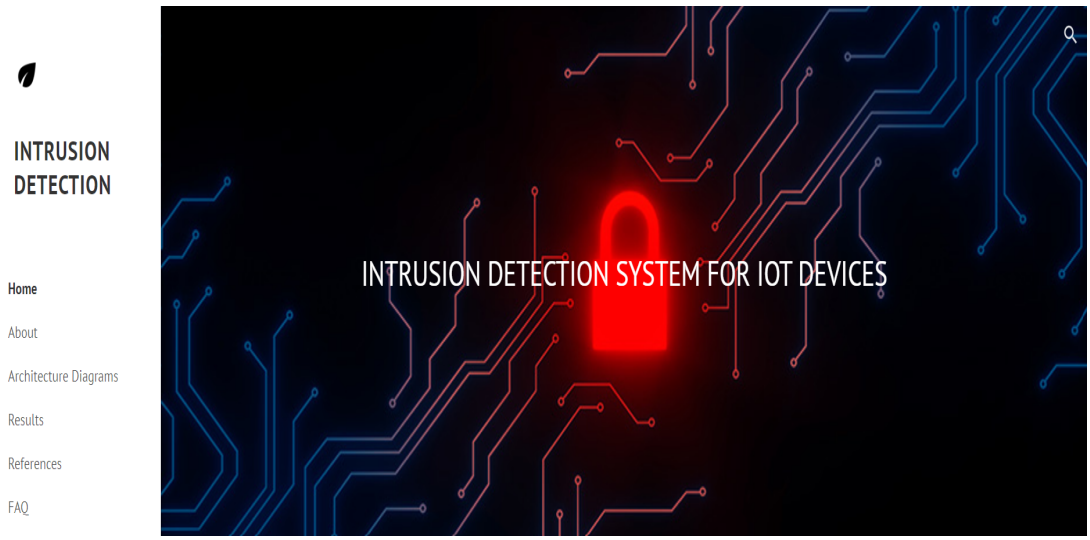

Figure 4.6: Screenshot of the project webpage

The architecture diagram shown in Figure 4.7 depicts the data collection process of the IDS. Multiple routers (2-3) are connected to various devices (4-5 per router), enabling the transfer of network packets. The IDS analyzes and cleanses the collected data, followed by model training to identify potential attacks. When an attack is detected, an alert is raised. This architecture ensures effective monitoring and detection of malicious activities within the network.

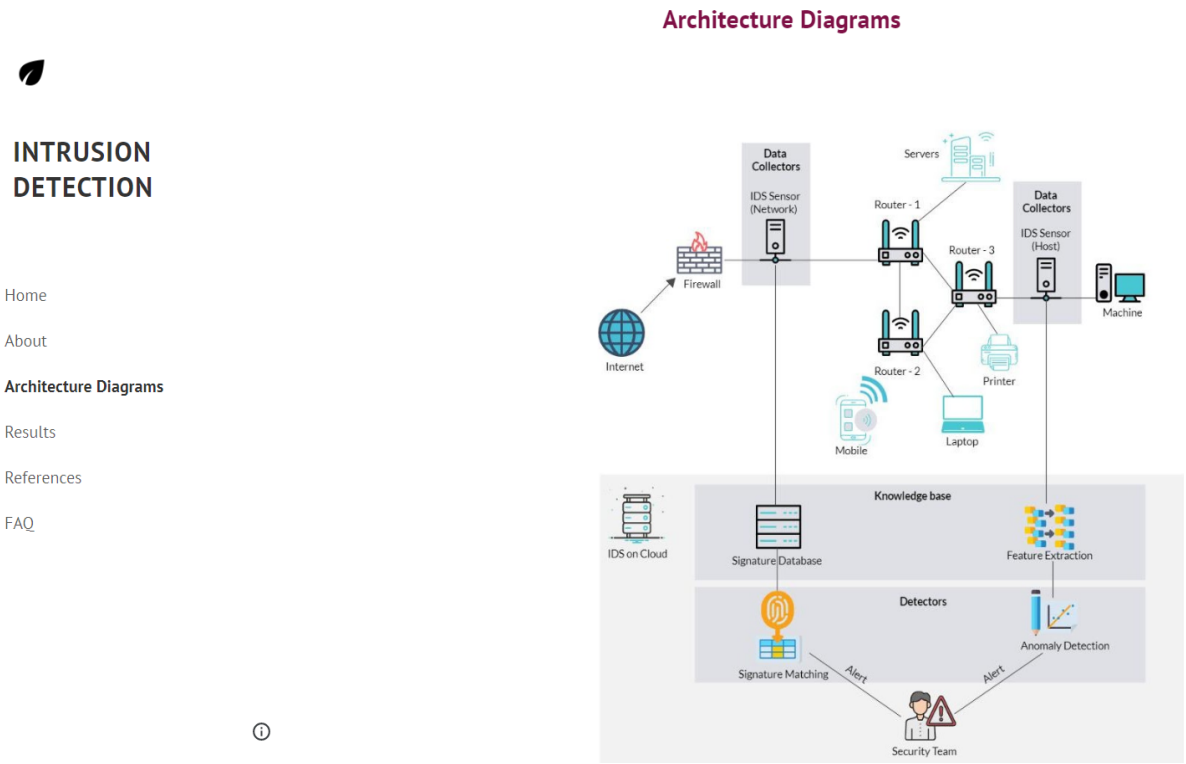

Figure 4.7: Webpage for Architecture Diagrams

Chapter 5

RESULTS AND DISCUSSION

In this chapter, we present the results and analysis of the proposed hybrid IDS system. The system combines signature-based and anomaly-based methods, utilizing a Stacking strategy with a Logistic Regression model for accurate decision-making.

Table 5.1 shows prediction Accuracy of each model with each attack is as follows:

Table 5.1: Techniques Comparison

Techniques	**Normal**	**DDoS**	**Bot**	**Brute force**	**Infiltration**	**Web**
IF	62.6	78.9	-	-	-	-
AE	69.9	54.3	49.6	100	27.7	61
RF	88.4	78.9	99.9	91.8	58.9	98.6
FFNN	98.4	99.8	99.9	90.7	29.8	98.8
Hybrid	96.4	100	99.9	99.1	38.6	98.2

The hybrid IDS system was validated through experiments, showcasing its strong performance in detecting threats with minimal false alarms. Various machine learning and deep learning algorithms were employed to evaluate the system's accuracy, F1-score, recall, precision, and AUC.

Table **5.2** shows a comparison of the performance metrics for each technique. The hybrid IDS system outperformed other techniques with an accuracy of 97%, an F1-score of 96.9, recall of 96.97, precision of 97.1, and an AUC value that was not reported.

Table 5.2: Model Comparison

Techniques	Accuracy	F1-Score	Recall	Precision	AUC
IF	73	72	71	72	71.2
AE	80	69	69	62.3	69
RF	89	89.8	91	74.7	-
FFNN	93.5	88	88	91.8	-
Hybrid	97	96.9	96.97	97.1	-

These results highlight the robustness and reliability of the proposed system, showcasing its ability to effectively detect intrusions. The discussions in this chapter also cover the correctness of the system and the verification steps taken to ensure its accuracy.

Chapter 6

CONCLUSION

In conclusion, the proposed hybrid IDS represents a robust and reliable approach that combines signature-based and anomaly-based methods. By leveraging the Stacking strategy and Logistic Regression model, the system achieves accurate decision-making capabilities. The experimental results validate the effectiveness of the hybrid IDS, showcasing high detection rates coupled with low false-positive rates. Various machine learning and deep learning algorithms were evaluated, highlighting the system's accuracy, F1-Score (F1)-score, recall, precision, and Area Under the Curve (AUC) metrics. These results demonstrate the system's ability to effectively detect and mitigate intrusions. The hybrid IDS holds great potential for enhancing network security and preventing unauthorized access. Future research should focus on further refining the system's performance and scalability for real-world deployment.

Chapter 7

FUTURE ENHANCEMENT

The developed system demonstrates promising results and contributes significantly to the field of network intrusion detection. To further enhance its capabilities and effectiveness, several areas of future work can be explored:

Application in the Medical Field

Adapting the intrusion detection system to the medical field provides a vital cybersecurity solution for hospitals, safeguarding patient data, protecting critical systems, and ensuring uninterrupted healthcare services. Its integration enables active monitoring, unauthorized access detection, and data breach mitigation, reinforcing patient privacy and security.

Comparative Analysis with State-of-the-Art Systems

Performing a comparative analysis with existing intrusion detection systems allows for evaluation of the developed system's performance, unique features, and detection rates. This benchmarking enables identification of areas for improvement, contributing to advancements in intrusion detection.

Infiltration Attack Detection

To enhance the system's accuracy in detecting Infiltration attacks, dedicated research can be conducted to develop specialized detection mechanisms and features tailored to capturing their unique characteristics, resulting in improved identification and mitigation of Infiltration attacks.

Addressing these areas of future work will drive the evolution of the IDS, ensuring its prominence in network security research and the development of more robust and effective systems to enhance network-based service security and resilience.

REFERENCES

1. Abbasi, F., Naderan, M., and Alavi, S. (2021). "Anomaly detection in internet of things using feature selection and classification based on logistic regression and artificial neural network on n-baiot dataset." *2021 5th International Conference on Internet of Things and Applications (IoT)*, IEEE.

2. Alam, S., Shuaib, M., and Samad, A. (2019). "A collaborative study of intrusion detection and prevention techniques in cloud computing." *International Conference on Innovative Computing and Communications: Proceedings of ICICC 2018, Volume 1*, Springer Singapore.

3. Attou, H. and et al. (2023). "Cloud-based intrusion detection approach using machine learning techniques." *Big Data Mining and Analytics*, 6(3), 311–320.

4. Canadian Institute for Cybersecurity (2018). "Canadian Institute for Cybersecurity - CIC-IDS2018 Dataset. Accessed on December 9, 2022.

5. Elrawy, M., Awad, A., and Hamed, H. (2018). "Intrusion detection systems for iot-based smart environments: a survey." *J Cloud Comp*, 7, 21.

6. Fatani, A. and et al. (2021). "Iot intrusion detection system using deep learning and enhanced transient search optimization." *IEEE Access*, 9, 123448–123464.

7. Ghosh, P. and et al. (2019). "Cs-pso based intrusion detection system in cloud environment." *Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2018, Volume 1*, Springer Singapore.

8. HP, C., Nandini, P., and Honnavalli, P. (2021). "Cloud-based network intrusion detection system using deep learning." *The 7th Annual International Conference on Arab Women in Computing in Conjunction with the 2nd Forum of Women in Research*.

9. Idrissi, I., Azizi, M., and Moussaoui, O. (2022). "A stratified iot deep learning based intrusion detection system." *2022 2nd International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET)*, IEEE.

10. Illavarason, P. and Kamachi Sundaram, B. (2019). "A study of intrusion detection system using machine learning classification algorithm based on different feature selection approach." IEEE.

11. Liu, H. and Lang, B. (2019). "Machine learning and deep learning methods for intrusion detection systems: A survey." *applied sciences*, 9(20), 4396.

12. Liu, Z. et al. (2022). "Intrusion detection systems in the cloud computing: A comprehensive and deep literature review." *Concurrency and Computation: Practice and Experience*, 34(4), e6646.

13. Pandeeswari, G., Janani, G., and Jeyanthi, S. (2022). "Analysis of intrusion detection using machine learning techniques." *2022 Second International Conference on Advanced Technologies in Intelligent Control, Environment, Computing Communication Engineering (ICATIECE)*, IEEE.

14. Sadaf, K. and Sultana, J. (2020). "Intrusion detection based on autoencoder and isolation forest in fog computing." *IEEE Access*, 8, 167059–167068.

15. Singh, S. and et al. (2021). "Mcids-multi classifier intrusion detection system for iot cyber attack using deep learning algorithm." *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, IEEE.

16. Slevi, S. T. and Visalakshi, P. (2021). "A survey on deep learning based intrusion detection systems on internet of things." *2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, IEEE.

17. Smys, S., Basar, A., and Wang, H. (2020). "Hybrid intrusion detection system for internet of things (iot)." *Journal of ISMAC*, 2(04), 190–199.

18. Tait, K.-A. et al. (2021). "Intrusion detection using machine learning techniques: an experimental comparison." *2021 International Congress of Advanced Technology and Engineering (ICOTEN)*, IEEE.

19. Vijayalakshmi, S. and et al. (2022). "A novel approach for iot intrusion detection system using modified optimizer and convolutional neural network." *2022 Sixth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, IEEE.

20. Vinolia, A., Kanya, N., and Rajavarman, V. N. (2023). "Machine learning and deep learning based intrusion detection in cloud environment: A review." *2023 5th International Conference on Smart Systems and Inventive Technology (ICSSIT)*, IEEE.

21. Zhao, R. and et al. (2021). "A novel intrusion detection method based on lightweight neural network for internet of things." *IEEE Internet of Things Journal*, 9(12), 9960–9972.