# Implementation of Cloud Based Intrusion Detection System

*A project work submitted for the*

*degree of*

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING (Cyber Security)

*Submitted by*


K.Madhuri (21551A4653)

M.V.R. Chowdary (21551A4648)

N. Ganesh (22555A4604)

K.V. Saradhi Babu (21551A4638)


*Under the Supervision of*


Mrs. S. Ratalu

Assistant Professor

DEPT. OF CSE(AIML&CS)




DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING (AIML&CS)

GODAVARI INSTITUTE OF ENGINEERING & TECHNOLOGY

(AUTONOMOUS)

Approved by AICTE|NAAC 'A++'| Recognized by UGC, U/Sec. 2(f) & 12(B)|

Permanently Affiliated to JNTUK, Kakinada


March-2025

GODAVARI INSTITUTE
OF ENGINEERING & TECHNOLOGY
Approved By AICTE | NAAC 'A++' | Recognized by UGC,
U/Sec. 2(f) & 12(B) | Permanently Affiliated to JNTUK, Kakinada

# BONAFIDE CERTIFICATE

Certified that this project report **"IMPLEMENTATION OF CLOUD BASED INTRUSION DETECTION SYSTEM"** is the bonafide work of **"K. Madhuri (21551A4653), M.V.R. Chowdary (21551A4648), N . Ganesh (22555A4604), K.V. Saradhi Babu (21551A4638)"**, who carried out the project work under my supervision during the year 2024-2025, towards partial fulfillment of the requirements of the Degree of Bachelor of Technology in Computer Science & Engineering(Cyber Security) as administered under the Regulations of Godavari Institute of Engineering & Technology, Rajamahendravaram AP, India and award of the Degree from Jawaharlal Nehru Technological University, Kakinada. The results embodied in this report have not been submitted to any other University for the award of any degree.

Supervisor
Mrs. S. Ratalu
Assistant .Professor

Head of the Department
Dr. R. Tamilkodi

External viva voce conducted on _____

External Examiner

# CERTIFICATE OF AUTHENTICATION

We solemnly declare that this project report **"IMPLEMENTATION OF CLOUD BASED INTRUSION DETECTION SYSTEM"** is the bonafide work done purely by us, carried out under the supervision of **Mrs. S. Ratalu** Assistant Professor, towards partial fulfillment of the requirements of the Degree of Bachelor of Technology in Computer Science & Engineering (Cyber Security) as administered under the Regulations of Godavari Institute of Engineering & Technology, Rajamahendravaram, AP, India and award of the Degree from Jawaharlal Nehru Technological University, Kakinada during the year 2024-2025.

We also declare that no part of this document has been taken up verbatim from any source without permission from the author(s)/publisher(s). Wherever few sentences, findings, images, diagrams or any other piece of information has been used for the sake of completion of this work, we have adequately referred to the document source. In the event of any issue arising here after about this work, we shall be personally responsible.

It is further certified that this work has not been submitted, either in part of in full, to any other department of the Jawaharlal Nehru Technological University Kakinada, or any other University, Institution or elsewhere, in India or abroad or for publication in any form.

Signature of the Student(s)

| | | |
|---|---|---|
| K. Madhuri | 21551A4653 | _____ |
| M.V.R. Chowdary | 21551A4648 | _____ |
| N. Ganesh | 22555A4604 | _____ |
| K.V. Saradhi Babu | 21551A4638 | _____ |

# ACKNOWLEDGEMENTS

We feel immense pleasure to express our sincere thanks and profound sense of gratitude to all those people who played a valuable role for the successful completion of our project.

We would like to express our gratitude to **Sri. K.V.V. SATYANARAYANA RAJU**, Founder and Chairman, Chaitanya group of Institutions, **Sri. K. SASI KIRAN VARMA**, Vice Chairman, GIET Group of Institutions for providing necessary infrastructure.

We thankful to Principal **Dr. T. JAYANANDA KUMAR** for permitting and encouraging in doing this project.

Our special thanks to **Dr. N. LEELAVATHY,** Vice Principal (Academics) for their content guidance and motivation and also for providing an excellent environment. We truly grateful for her valuable suggestions and advice.

We would like to express our gratefulness to **Dr. R. TAMILKODI**, Professor, and Head of the Department, Computer Science & Engineering (AIML&CS) for giving constant encouragement and moral support.

We feeling grateful to express my deep sense of gratitude and respect towards our supervisor **Mrs. S. RATALU,** Assistant Professor, whose motivation and constant encouragement has led to pursue a project in the field of Cyber Security. We are very fortunate, for having his gracious presence to enlighten us in all the aspects of life as well as project and transforming us to be an ideal individual in this field.

Our family members have put us ahead of themselves, because of their hard work and dedication, this success is possible. Our heart full thanks to them for giving constant support.

(K. Madhuri) (M.V.R. Chowdary)

(N. Ganesh) (K.V. Saradhi Babu)

# ABSTRACT

Cloud computing (CC) is a novel technology that has made it easier to access network and computer resources on demand such as storage and data management services. In addition, it aims to strengthen systems and make them useful. For this reason, a set of solutions have been implemented to improve cloud security by monitoring resources, services, and networks, then detect attacks. Actually, intrusion detection system (IDS) is an enhanced mechanism used to control traffic within networks and detect abnormal activities. This paper presents a cloud-based intrusion detection model based on random forest (RF) and feature engineering. Specifically, the RF classifier is obtained and integrated to enhance accuracy (ACC) of the proposed detection model. The proposed model approach has been evaluated and validated on two datasets and gives good ACC using Bot-IoT and NSL-KDD datasets, respectively. Consequently, the obtained results present good performances in terms of ACC, precision, and recall when compared to the recent related works.

# TABLE OF CONTENT

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| API | Application Programming Interface |
| SVM | Support Vector Machine |
| MLP | Multi-Layer Perceptron |
| DDOS | Distributed Denial-of-Service |
| DNN | Deep Neural Network |
| DT | Decision Tree |
| IDS | Intrusion Detection System |
| FAR | False Alarm Rate |
| RF | Random Forest |
| NB | Naive Bayes |
| DoS | Denial of Service |
| LSTM | Long Short-Term Memory |
| PCA | Principal Component Analysis |

# INTRODUCTION

## 1.1. INTRODUCTION TO CLOUD-BASED IDS

Cloud technologies allow practical access on demand to a shared network, storage, and resources and offer more choices regarding their service models. These models are platform as a service (PaaS), software as a service (SaaS), and infrastructure as a service (IaaS), used in one of the deployment models private, public, and hybrid cloud.Recently, the cloud suffers from many security problems like availability, data confidentiality, integrity, and control authorization. In addition, the Internet is used to facilitate access to the services offered by the cloud representing a major source of threats that can infect the cloud systems and resources. Then enhancing cloud security becomes a primary challenge for cloud providers. Therefore, several approaches such as firewall tools, data encryption algorithms, authentication protocols, and others have been developed to better secure cloud environments from various attacks. However, traditional systems are not sufficient to secure cloud services from different limits. Therefore, a set of intrusion detection approaches are proposed and applied to detect and prevent undesirable activities in real-time. In general, the detection methods are divided into misuse detection method which uses known attacks to detect intrusion and anomaly detection method which detect intrusion using unknown attack.

### Machine Learning

Before we take a look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of building models of data.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models tunable parameters that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Understanding the problem setting in machine learning is essential to using these tools effectively.

### Categories Of Machine Leaning

Supervised learning involves somehow modeling the relationship between measured features of data and some label associated with the data; once this model is determined, it can be used to apply

labels to new, unknown data. This is further subdivided into classification tasks and regression tasks: in classification, the labels are discrete categories, while in regression, the labels are continuous quantities. We will see examples of both types of supervised learning in the following section.

Unsupervised learning involves modeling the features of a dataset without reference to any label, and is often described as "letting the dataset speak for itself." These models include tasks such as clustering and conditionality reduction. We will see examples of both types of unsupervised learning in the following section.

### Need for Machine Learning

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and haven't surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, "to make decisions, based on data, with efficiency and scale".

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. The fact is that we can't do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

## 1.2. CHALLENGES IN CLOUD SECURITY

Cloud environments introduce unique challenges that make intrusion detection more complex compared to traditional networks:

- High Volume of Data Traffic: Cloud networks generate massive amounts of real-time data, making threat detection computationally intensive.
- Multi-Tenant Architecture Risks: Shared cloud resources can lead to data leakage and unauthorized access if not properly secured.
- Scalability and Performance: The IDS must handle large-scale deployments without degrading network performance.

- False Positives and Negatives: Ensuring high accuracy while minimizing false alarms is a major challenge in intrusion detection.

## 1.3. PRIMARY OBJECTIVE OF THE PROJECT

The objective of this research is to develop an effective cloud-based intrusion detection approach using machine learning techniques. By leveraging the capabilities of cloud computing, the study aims to enhance the efficiency and scalability of intrusion detection systems. The research intends to design and implement a model that can accurately identify and classify various types of intrusions in cloud environments. The ultimate goal is to provide robust and real time security measures that contribute to safeguarding cloud-based systems against a range of cyber threats.

As cloud computing gains prominence, the security of cloud-based systems becomes paramount. Traditional intrusion detection systems (IDS) face challenges in effectively handling the scale and complexity of cloud environments. Current IDS may struggle with timely and accurate detection of sophisticated attacks.

This research addresses the need for an advanced cloud-based intrusion detection approach by harnessing the power of machine learning techniques. The study aims to enhance the capability of detecting and mitigating diverse and evolving cyber threats within cloud infrastructures.

# LITERATURE SURVEY

## 2.1. LITERATURE REVIEW

Studies have explored the evolution of cloud security, emphasizing its applications in intrusion detection, anomaly detection, and data protection [1]. Researchers have investigated various machine learning and deep learning techniques to enhance cybersecurity in cloud environments, focusing on detection accuracy, computational efficiency, and real-time response mechanisms [2].

The application of machine learning in cloud security has gained significant attention. A study developed a model that utilizes machine learning techniques for predicting water quality based on key parameters [3]. This approach demonstrates how ML can effectively analyze large datasets and improve predictive analytics, which is crucial in cybersecurity applications [4]. Similarly, another study introduced an anomaly-based intrusion detection system (IDS) for IoT security, enhancing intrusion detection rates through machine learning methodologies [5].

In the context of cloud computing, one research proposed a machine learning-based approach for securing cloud data [6]. Their research categorized data into sensitivity levels using classifiers such as Random Forest (RF), Naïve Bayes (NB), k-nearest neighbor (KNN), and support vector machine (SVM), achieving an accuracy of 92% [7]. Another study compared AI-based classifiers for network intrusion detection, demonstrating that Random Forest provided the highest accuracy in identifying security threats [8].

Enhancing cloud security further, another study proposed a reliable IDS using decision trees, achieving 99.42% and 98.80% accuracy on NSL-KDD and CICIDS2017 datasets, respectively [9]. The study emphasized the importance of data preprocessing and entropy decision feature selection to improve detection accuracy [10]. Additionally, some researchers utilized machine learning techniques to detect malicious network traffic in cloud computing, focusing on feature extraction and dataset quality [11].

A systematic review analyzed 63 studies on the intersection of machine learning and cloud security, categorizing research findings into ML techniques, security threats, and performance metrics [12]. Furthermore, another study investigated the detection of Distributed Denial of Service (DDoS) attacks in cloud computing using ML classifiers such as KNN, RF, and NB, achieving an impressive accuracy of 99.76% [13]. A review discussed the Random Forest algorithm, highlighting its broad applications in cybersecurity and other fields [14].

Cloud security research has also focused on anomaly detection techniques to identify potential threats [15]. Advanced ML models, such as deep learning architectures, have been employed to improve accuracy and efficiency in threat detection [16]. Several studies have demonstrated the effectiveness of convolutional neural networks (CNNs) and long short-term memory (LSTM) networks in cloud security applications [17]. These models enhance intrusion detection systems by analyzing network traffic patterns and identifying anomalous behaviors in real time [18].

Efforts have also been made to address challenges such as false positives in intrusion detection systems [19]. Researchers have developed hybrid approaches that combine signature-based and anomaly-based detection techniques to improve security in cloud environments [20]. Techniques like ensemble learning, which integrates multiple classifiers, have proven effective in reducing false alarms and increasing detection rates [21].

A recent study explored federated learning approaches to enhance distributed cloud security without compromising data privacy [22]. The study demonstrated that federated learning allows for decentralized model training, reducing the risk of data exposure while maintaining strong intrusion detection accuracy [23]. Another research initiative investigated adversarial attacks on cloud-based machine learning models and proposed robust defense mechanisms [24].

Despite significant advancements in cloud-based intrusion detection, challenges persist, particularly in scenarios with large-scale datasets and evolving cyber threats [25]. Some researchers have suggested the implementation of reinforcement learning techniques for adaptive threat detection, enabling systems to learn from attack patterns dynamically [26]. Future research should focus on enhancing real-time detection capabilities, optimizing ML models for better performance, and integrating blockchain technology for secure cloud transactions [27].

Blockchain has been explored as a means to enhance cloud security, particularly in securing identity management and transaction integrity [28]. Studies have demonstrated that blockchain-based authentication systems provide a decentralized approach to access control, reducing the risks of unauthorized intrusions [29]. Smart contract-enabled security frameworks have also been proposed to automate security policies in cloud environments [30].

The ongoing evolution of ML and DL techniques in cloud security underscores the need for continuous research and development to stay ahead of emerging threats [31]. Several studies have

explored privacy-preserving techniques in cloud environments, utilizing homomorphic encryption, differential privacy, and federated learning to secure sensitive data [32].

Furthermore, researchers have examined the use of genetic algorithms for optimizing IDS performance in cloud environments, achieving significant improvements in threat detection rates [33]. Another study introduced a hybrid cloud security framework integrating IDS and firewall mechanisms, demonstrating enhanced protection against sophisticated attacks [34]. Research has also explored the impact of quantum computing on cloud security, emphasizing the need for post-quantum cryptography to mitigate future threats [35].

Researchers have further emphasized the role of explainable AI (XAI) in enhancing cloud security decision-making processes, ensuring transparency and accountability in automated systems [36]. Studies have also investigated the integration of artificial immune systems in IDS design, leveraging biological principles to enhance security in dynamic cloud environments [37].

Efforts have been directed toward improving data encryption protocols for securing data in transit and at rest, employing techniques such as attribute-based encryption (ABE) and elliptic curve cryptography (ECC) [38]. Additionally, studies have proposed zero-trust architecture frameworks for cloud environments, ensuring continuous identity verification and access control [39].

Recent advancements in secure multi-party computation (SMPC) have also contributed to privacy-preserving data analysis in cloud environments, enhancing data confidentiality without compromising performance [40]. Researchers have explored the use of reinforcement learning-based IDS models, enabling adaptive threat response strategies in evolving cloud networks [41].

A study proposed a novel intrusion detection framework leveraging deep reinforcement learning, improving detection accuracy in complex cloud environments [42]. Additionally, researchers have investigated adversarial training techniques to improve the robustness of ML models against evasion attacks in cloud security [43].

Further studies have explored the integration of graph neural networks (GNN) in cloud security frameworks, improving the detection of network-based attacks through graph-based data modeling [44]. The combination of ML techniques with cloud-native security tools has shown promising results in mitigating vulnerabilities and ensuring real-time threat prevention [45].

Research has also examined the role of cloud access security brokers (CASBs) in enforcing security policies and ensuring data protection in multi-cloud environments [46]. Blockchain-enabled identity management systems have been proposed to improve user authentication processes in cloud platforms [47].

Finally, researchers have highlighted the significance of AI-driven threat intelligence platforms in enhancing proactive threat detection, improving response strategies against sophisticated attacks [48]. Studies have also emphasized the importance of continual model training using real-time data to adapt to evolving threat landscapes [49].

The exploration of cloud forensics has gained momentum, with researchers developing frameworks for investigating cloud-based incidents, ensuring efficient data recovery and analysis [50].

## 2.2 EXISTING SYSTEM

A cloud computing approach based on automated data classification has been widely adopted for managing data sensitivity. The existing models classify data into three levels—basic, confidential, and highly confidential—using machine learning classifiers such as Random Forest (RF), Naïve Bayes (NB), k-nearest neighbor (KNN), and support vector machine (SVM). These classifiers help in automating feature extraction and improving the accuracy of classification. This research also provides essential guidelines for cloud service providers like Dropbox and Google Drive, ensuring better management and protection of cloud data.

### Challenges in the Existing System

Despite these advancements, the existing system faces multiple challenges. One of the major limitations is the handling of huge amounts of analyzed data, which affects real-time detection capabilities. The performance of detection models often decreases due to the complexity of data processing and classification. Additionally, the security of cloud resources and services continues to be a significant challenge. Many existing models do not incorporate advanced feature engineering techniques, which are crucial for enhancing detection accuracy. Feature selection, another critical aspect, is often overlooked, leading to redundant and irrelevant data being processed, thereby increasing computational costs.

## Disadvantages of Existing System

1. Real-Time Data Processing Challenges : Handling massive datasets in real-time poses computational challenges, reducing the overall efficiency of detection models.

2. Security Vulnerabilities : Security concerns remain a primary issue, as protecting cloud resources and services from evolving cyber threats is complex.

3. Lack of Feature Engineering Techniques : Several existing models do not implement feature engineering techniques, limiting the accuracy and efficiency of classification algorithms.

4. Inefficient Feature Selection : The absence of feature selection techniques leads to processing redundant data, increasing the computational burden and slowing down detection processes.

5. Need for Improved Cloud Security : The continuous improvement of cloud security remains an urgent need, as emerging threats and sophisticated cyberattacks require robust detection mechanisms. The continuous evolution of cloud-based intrusion detection systems (IDS) is crucial for securing modern cloud infrastructures against emerging cyber threats. Despite advancements in machine learning and automation, existing challenges such as real-time data processing, security vulnerabilities, and inefficient feature selection still persist. By integrating with cloud security platforms, enforcing regulatory compliance, and leveraging global threat intelligence, IDS can become more scalable, intelligent, and effective in mitigating cyber threats in cloud environments.

# PROPOSED SYSTEM

## PROPOSED SYSTEM

Our main goal in this research work is to propose an anomaly detection approach based on random forest (RF) binary classifier and feature engineering is carried out based on a data visualization process aiming to reduce the number of used features and perform the proposed anomaly detection model. The evaluation performances of the model are implemented on KDD-CUP and BoT-IoT datasets.

## Advantages of proposed system:

1. Feature engineering is carried out based on a data visualization process aiming to reduce the number of used features and perform the proposed anomaly detection model.
2. The features reducing are integrated to minimize execution time and to perform prediction.
3. Our intrusion detection model includes feature selection to identify and combine useful features for accurate detection.
4. Consequently, the obtained results present good performances in terms of ACC, precision, and recall when compared to the recent related works.

## Algorithms

An ensemble approach called **Random Forest** creates many decision trees and averages their forecasts to identify the most often occurring class. Large feature sets are easily handled by it, overfitting is minimized, and intrusion detection accuracy is high (calculated in the section below on experimental data).

Step 1: Data Preprocessing: Clean, normalize, and extract features from cloud network datasets (Bot-IoT, NSL-KDD).

Step 2: Train Random Forest Model: Using random feature selection and bootstrapped data samples, create several decision trees.

Step 3: Ensemble Decision Making: Aggregate predictions from all decision trees using majority voting for final intrusion detection.

Step 4: Model Evaluation: Assess performance using metrics (eg. accuracy, precision) on Bot-IoT and NSL-KDD datasets.

**Decision Trees** supervised learning models that construct a tree-like structure by separating data depending on feature values. They aid in intrusion detection by mapping decision rules to classify data, which helps in understanding and interpreting the decision-making process.

Step 1: Data Collection: Gather and preprocess relevant cloud network datasets, including Bot-IoT and NSL-KDD.

Step 2: Tree Construction: Recursively split data based on features to create decision nodes using information gain.

Step 3: Tree Pruning: Reduce complexity by trimming branches to prevent overfitting and improve model generalization.

Step 4: Prediction & Evaluation: Use the decision tree to classify network activity and evaluate results on test datasets.

**Support Vector Machine** is a powerful classification algorithm that separates classes using hyperplanes in high dimensional spaces. SVM is effective in intrusion detection due to its ability to manage complex data relationships and handle non-linearity.

Step 1: Data Preprocessing: Normalize and scale cloud network datasets (Bot-IoT, NSL-KDD) for SVM model training.

Step 2: Kernel Selection: Choose an appropriate kernel (linear, RBF) to map data into higher-dimensional space.

Step 3: Margin Optimization: Train SVM to find the optimal hyperplane in order to maximize the margin between classes.

Step 4: Model Evaluation: Test the SVM model's accuracy and performance using Bot-IoT and NSL-KDD datasets.

**Naive Bayes** is a random classifier based on Bayes' theorem, assuming feature independence. Its speed and ease of use make it a valuable tool for jobs involving intrusion detection, particularly when dealing with text-based data.

Step 1: Data Preparation: Preprocess and clean cloud network datasets (Bot-IoT, NSL-KDD) for model input.

Step 2: Feature Probability Calculation: Calculate prior and likelihood probabilities for each feature within the dataset.

Step 3: Bayesian Inference: Utilize the Bayes Theorem to determine the likelihood of every class according to its attributes.

Step 4: Prediction & Evaluation: Classify network activities and evaluate Naïve Bayes model accuracy using Bot-IoT and NSL-KDD datasets.

**Deep Learning** uses multi-layered neural networks to identify complex patterns in data. Because they can simulate intricate feature interactions and dependencies, models like MLPs, CNNs, and RNNs are useful for intrusion detection.

Step 1: Data Preparation: Normalize and preprocess cloud network datasets (Bot-IoT, NSL-KDD) for deep learning input.

Step 2: Model Architecture Design: Construct a deep learning model with multiple layers for effective feature extraction and classification.

Step 3: Training Process: Train the model using back propagation and optimization algorithms to minimize loss function.

Step 4: Performance Evaluation: Test the model on validation datasets, measuring accuracy and detecting intrusion effectiveness.

**LSTM** is a type of RNN designed for sequence and timedependent data, capturing long-term dependencies. It is useful in intrusion detection for analyzing sequences of network activities and identifying patterns over time.

Step 1: Data Preprocessing: Normalize and reshape cloud network datasets (Bot-IoT, NSL-KDD) for LSTM input format.

Step 2: Sequence Creation: Convert data into sequences to capture temporal dependencies relevant to intrusion detection.

Step 3: Model Training: Train the LSTM model using backpropagation through time to optimize weights and biases.

Step 4: Prediction & Evaluation: Evaluate model performance on test datasets, measuring accuracy in detecting intrusions effectively.

**Stacking Classifier (RF + MLP with LightGBM):** LightGBM, RF, and MLP are combined in the Stackable Classifiers. It combines diverse learning techniques to enhance detection performance by leveraging the strengths of each base model.

Step 1: Base Model Training: Train individual classifiers, RF + MLP, on cloud datasets.

Step 2: Feature Extraction: Use predictions from RF and MLP as new features for the stacking model.

Step 3: Meta-Model Training: Train the LightGBM model using the extracted features from base models for final predictions.

Step 4: Evaluation: Analyze the accuracy and detection of intrusion capabilities of the stacked model using test datasets.

**Voting Classifier (RF + AdaBoost)**: Analyze the accuracy and detection of intrusion capabilities of the stacked model using test datasets. RF captures complex patterns, while AdaBoost focuses on correcting misclassified instances, creating a robust model for detecting intrusions.

Step 1: Base Model Training: Train the RF + AdaBoost classifiers on cloud network datasets.

Step 2: Prediction Aggregation: Use predictions from both RF and AdaBoost to create a combined decision-making process.

Step 3: Voting Mechanism: Implement majority voting to determine the final class based on individual model predictions.

Step 4: Model Evaluation: Evaluate the Voting Classifier's performance using accuracy and intrusion detection metrics on test datasets.

## 3.1. PROPOSED METHOD ALGORITHM

### 1. Data Collection

In this phase, cloud network datasets such as Bot-IoT and NSL-KDD are gathered. These datasets contain labeled instances of normal and attack traffic, which are essential for training an intrusion detection system. The extracted features from these datasets help in identifying malicious activities within cloud environments.

### 2. Data Preprocessing

Preprocessing involves cleaning the dataset by handling missing values and outliers. Numerical features are normalized and scaled to ensure consistency. Feature selection techniques are applied to retain only the most significant attributes that contribute to intrusion detection. The dataset is then split into training and testing sets, typically in an 80:20 ratio, to ensure effective model evaluation.

### 3. Training And Testing

This phase involves training machine learning models such as Random Forest, Support Vector Machine, Naïve Bayes, Decision Trees, and Deep Learning architectures like DNN and LSTM. Ensemble methods such as Voting Classifier and Stacking Classifier are also used to enhance detection performance. Hyperparameter tuning is performed using techniques like Grid Search or Random Search to optimize model accuracy. The trained models are tested on validation data, and their performance is evaluated using metrics such as accuracy, precision, recall, and F1-score.

### 4. Modeling

The intrusion detection system is modeled using ensemble techniques for improved detection accuracy. The Voting Classifier, which combines Random Forest and AdaBoost, achieves 99% accuracy on the KDD-CUP dataset. The Stacking Classifier, consisting of Random Forest, Multi-Layer Perceptron, and LightGBM, reaches 100% accuracy on the Bot-IoT dataset. To ensure practical usability, the model is deployed using the Flask framework, integrated with SQLite for secure data management.

## 5. Predicting

In the prediction phase, the trained model is deployed for real-time intrusion detection within cloud environments. The system accepts new input data, processes it, and classifies it as either an attack or normal activity. Based on the classification results, the system provides predictions indicating "Attack Detected" or "No Attack Detected." The final output is displayed through a user-friendly web interface developed using Flask, making it accessible and efficient for cybersecurity applications.

## Non Functional Requirements

Non Functional Requirements (NFR) specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system. Example of nonfunctional requirement, *"how fast does the website load?"* Failing to meet non-functional requirements can result in systems that fail to satisfy user needs. Non-functional Requirements allows you to impose constraints or restrictions on the design of the system across the various agile backlogs. Example, the site should load in 3 seconds when the number of simultaneous users are > 10000.

- Usability requirement
- Serviceability requirement
- Recover ability requirement
- Security requirement
- Data Integrity requirement
- Capacity requirement

## 3.2. SYSTEM ARCHITECTURE

The system architecture of the proposed cloud-based intrusion detection system consists of multiple stages, including data collection, preprocessing, training, modeling, and prediction. It is designed to efficiently detect anomalies and security threats in cloud environments by leveraging machine learning and deep learning techniques.

The architecture begins with the data collection phase, where datasets such as Bot-IoT and NSL-KDD are gathered. These datasets provide labeled instances of network traffic, which include both normal and attack patterns.

In the data preprocessing phase, the collected data is cleaned by handling missing values and outliers. Feature selection techniques are applied to remove redundant and irrelevant attributes, ensuring that only the most significant features are used for intrusion detection. The dataset is then normalized and scaled to maintain consistency, followed by splitting into training and testing sets.

The training phase involves using machine learning models such as Random Forest, Support Vector Machine, Decision Trees, and Deep Learning architectures like DNN and LSTM. Additionally, ensemble methods such as Voting Classifier and Stacking Classifier are implemented to enhance detection performance. Hyperparameter tuning is conducted to optimize model accuracy, and the models are validated using performance metrics such as accuracy, precision, recall, and F1-score.

The modeling phase integrates the trained models into a structured framework. The Voting Classifier (RF + AdaBoost) achieves 99% accuracy on the KDD-CUP dataset, while the Stacking Classifier (RF + MLP + LightGBM) reaches 100% accuracy on the Bot-IoT dataset. To ensure usability, the model is deployed using the Flask framework, with SQLite integration for secure data management.

In the prediction phase, the deployed system continuously monitors incoming network traffic in real-time. The trained model classifies new data as either normal or an attack. If an intrusion is detected, the system raises an alert. The output is displayed through a user-friendly Flask-based web interface, allowing users to interact with the intrusion detection system efficiently.

The overall architecture ensures high accuracy and scalability, making it a reliable solution for cloud security (as shown in Fig-3.1).

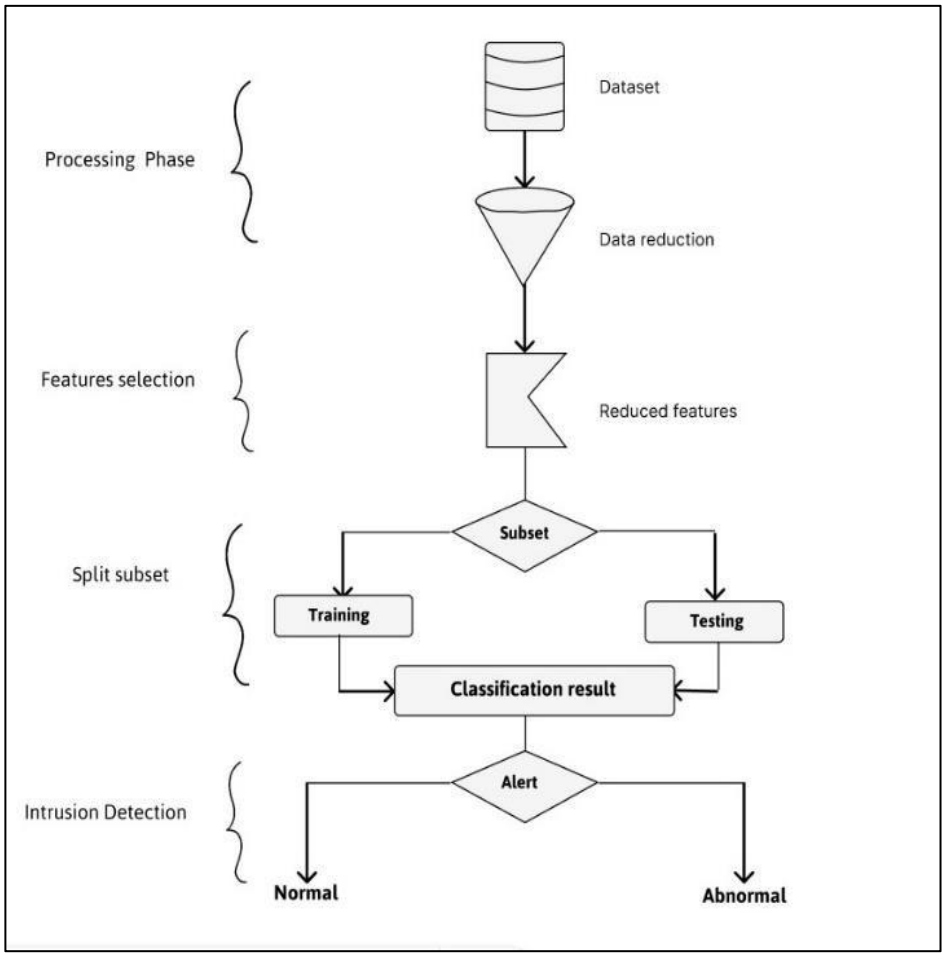


**Fig-3.1** Architecture of Proposed system

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail (as shown in Fig-3.2).

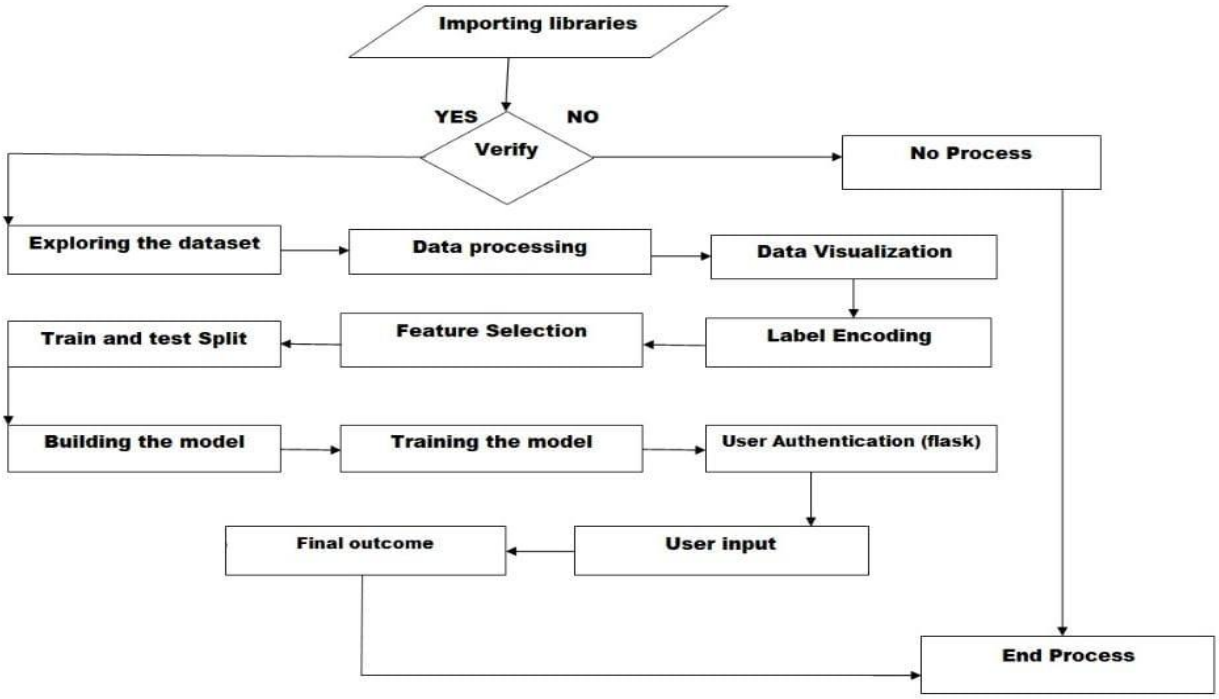


**Fig-3.2** Design flow of the Proposed Methodology

**Library Import and System Initialization -** The system starts by importing essential Python libraries such as NumPy, Pandas, Scikit-Learn, TensorFlow, and Flask. These libraries provide functionalities for data processing, machine learning, deep learning, and web application deployment.

**Data Verification and Validation -** The system checks the integrity of the dataset before processing. If the dataset is corrupted, incomplete, or inconsistent, the system terminates execution (NO PROCESS). If the data is valid, the workflow proceeds to the next stage.

**Data Preprocessing and Feature Engineering -** The dataset undergoes data cleaning, where missing values and duplicate entries are removed. Feature scaling and normalization techniques, such as Min-Max Scaling or Standardization, are applied. Feature selection methods, including Principal Component Analysis (PCA) or Recursive Feature Elimination (RFE), are used to extract relevant features.

**Data Splitting and Visualization -** The preprocessed dataset is divided into training and testing sets (typically in an 80:20 ratio). Data visualization techniques, such as correlation heatmaps, histograms, and scatter plots, are used to analyze feature distributions.

**Machine Learning and Deep Learning Model Training -** The system trains various classification models, including: Traditional ML Models: Random Forest (RF), Decision Trees (DT), Support Vector Machine (SVM), Naïve Bayes (NB). Ensemble Models: Voting Classifier, Stacking Classifier. Deep Learning Models: Deep Neural Networks (DNN), Long Short-Term Memory (LSTM). Hyper parameter tuning is applied using Grid Search or Random Search to optimize model performance. Performance evaluation metrics such as accuracy, precision, recall, and F1-score are calculated.

**Model Deployment and Web Interface Integration -** The trained models are deployed using Flask, a lightweight web framework. The system is connected to an SQLite database for storing user authentication details and logs. A secure signup and login system is implemented to ensure only authorized users can access the intrusion detection system.

**Real-Time Intrusion Detection and User Input Handling -** Users input real-time network traffic data through the Flask-based web interface. The deployed model processes the input and classifies it as either:

- "Attack Detected" (if an intrusion is found)
- "No Attack Detected" (if the activity is normal)

The output is displayed on the web dashboard, providing real-time monitoring for cloud security.

**System Termination and Logging -** The system logs predictions and user interactions into an audit database. If needed, the administrator can review logs to analyze previous attacks. The process concludes, and the system is ready for the next batch of inputs.

## Real-World Applications of the Proposed System

The proposed cloud-based intrusion detection system (IDS) can be applied in various real-world scenarios to enhance cybersecurity in different domains. Some key applications include:

Enterprise Cloud Security: Organizations using cloud services can integrate the IDS to monitor network traffic and prevent unauthorized access, ensuring data protection.

Healthcare Sector: Hospitals and medical facilities store confidential patient records, making them prime targets for cyber threats. A cloud-based IDS helps safeguard electronic health records (EHRs) from breaches.

Commerce Platforms: Online shopping platforms face threats like DDoS attacks, phishing, and SQL injection. The IDS can help prevent such intrusions and ensure smooth user transactions.

The implementation of a machine learning and deep learning-based cloud intrusion detection system significantly enhances cloud security by integrating data preprocessing, feature selection, model training, and real-time monitoring. By employing traditional ML models, ensemble techniques, and deep learning architectures, the system improves detection accuracy while reducing false positives. The deployment of a Flask-based web interface ensures real-time intrusion detection with user authentication for secure access.

Despite advancements, challenges such as handling large-scale data, real-time computational efficiency, and evolving cyber threats remain. Future research should focus on block chain-enhanced security, federated learning, and hybrid AI-driven detection models to make cloud security more resilient and intelligent against sophisticated cyberattacks.

# SYSTEM DESIGN

## OVERVIEW

The system study focuses on the comprehensive analysis of a cloud-based intrusion detection system (IDS) using machine learning techniques. The primary objective is to enhance cloud security by detecting and mitigating cyber threats in real-time. This involves breaking down the system into key components, including data collection, preprocessing, feature selection, model training, evaluation, and deployment. The effectiveness of the IDS is assessed using analytical methods by evaluating performance metrics such as accuracy, precision, recall, and F1-score. By systematically analyzing the proposed model and comparing it with existing solutions, the study aims to develop an efficient and scalable intrusion detection system that strengthens cloud security against evolving cyber threats.

## 4.1. System Requirements

The system requirements are designed to support the development and deployment of a cloud-based intrusion detection system. To ensure smooth functionality and efficient performance, the system needs a suitable hardware and software setup. It requires a stable operating system, sufficient processing power, memory, and storage capacity to handle data processing, model training, and real-time threat detection. These specifications help in optimizing the machine learning models and ensuring accurate intrusion detection in cloud environments.

### Software Requirements

Software requirements deal with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or prerequisites are generally not included in the software installation package and need to be installed separately before the software is installed.

**Platform –** In computing, a platform describes some sort of framework, either in hardware or software, which allows software to run. Typical platforms include a computer's architecture, operating system, or programming languages and their runtime libraries.

**APIs and drivers –** Software making extensive use of special hardware devices, like high-end display adapters, needs special API or newer device drivers. A good example is DirectX,

which is a collection of APIs for handling tasks related to multimedia, especially game programming, on Microsoft platforms.

**Web browser –** Most web applications and software depending heavily on Internet technologies make use of the default browser installed on system. Microsoft Internet Explorer is a frequent choice of software running on Microsoft Windows, which makes use of ActiveX controls, despite their vulnerabilities.

1) **Software: Anaconda**
2) **Primary Language: Python**
3) **Frontend Framework: Flask**
4) **Back-end Framework: Jupyter Notebook**
5) **Database: Sqlite3**
6) **Front-End Technologies: HTML,CSS, JavaScript and Bootstrap4**

## Hardware Requirements

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements.

**Processing power –** The power of the central processing unit (CPU) is a fundamental system requirement for any software. Most software running on x86 architecture define processing power as the model and the clock speed of the CPU. Many other features of a CPU that influence its speed and power, like bus speed, cache, and MIPS are often ignored. This definition of power is often erroneous, as AMD Athlon and Intel Pentium CPUs at similar clock speed often have different throughput speeds. Intel Pentium CPUs have enjoyed a considerable degree of popularity, and are often mentioned in this category.

**Memory –** All software, when run, resides in the random-access memory (RAM) of a computer. Memory requirements are defined after considering demands of the application, operating system, supporting software and files, and other running processes. Optimal

performance of other unrelated software running on a multi-tasking computer system is also considered when defining this requirement.

**Secondary storage –** Hard-disk requirements vary, depending on the size of software installation, temporary files created and maintained while installing or running the software, and possible use of swap space (if RAM is insufficient).

**Display adapter –** Software requiring a better than average computer graphics display, like graphics editors and high-end games, often define high-end display adapters in the system requirements.

**Peripherals –** Some software applications need to make extensive and/or special use of some peripherals, demanding the higher performance or functionality of such peripherals. Such peripherals include CD-ROM drives, keyboards, pointing devices, network devices, etc.

**1)Operating System : Windows Only**

**2)Processor : i5 and above**

**3)Ram : 8gb and above**

**4)Hard Disk : 25 GB in local drive**

## 4.2. UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

## Goals

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extensibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

The system study provides a detailed analysis of the development and deployment of a cloud-based Intrusion Detection System (IDS) using machine learning techniques. By defining the system requirements, both software and hardware, the study ensures a structured approach to designing an efficient and scalable security solution for cloud environments. The selection of essential software tools, such as Anaconda, Python, Flask, and Jupyter Notebook, alongside database management with SQLite3, helps optimize real-time threat detection and model performance. Additionally, the hardware requirements, including an i5 processor, 8GB RAM, and sufficient storage, ensure the system operates efficiently without performance bottlenecks.

Furthermore, the inclusion of UML diagrams provides a standardized and structured representation of the IDS architecture, facilitating better visualization, documentation, and implementation of object-oriented design principles. The use of UML supports clear communication between developers, stakeholders, and security professionals, ensuring seamless integration of the IDS into cloud environments. Overall, this system study establishes a strong foundation for building a reliable IDS that effectively detects and mitigates evolving cyber threats, enhancing cloud security and safeguarding sensitive data.

## Use case diagram

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted (as shown in Fig-4.1).

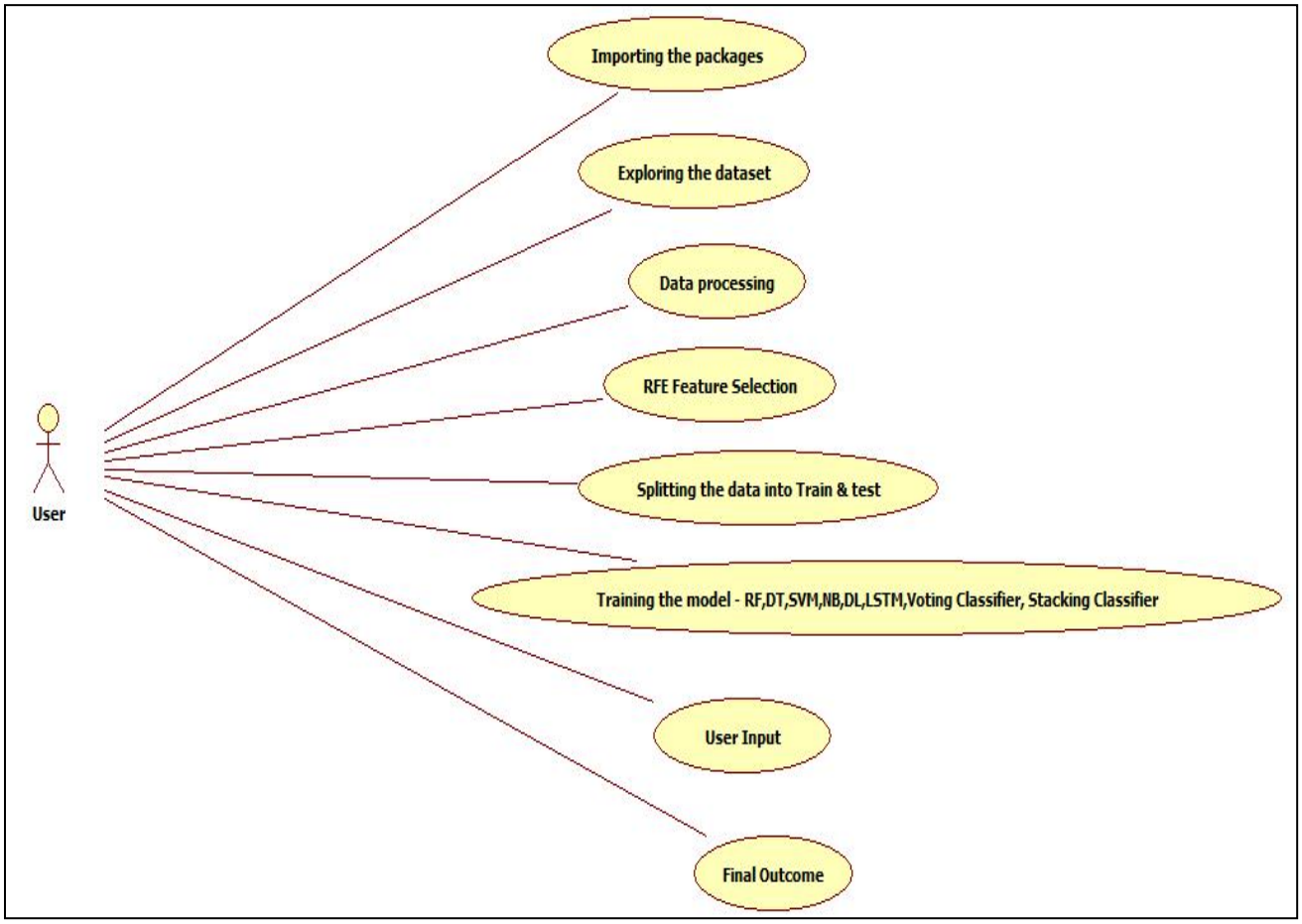


**Fig-4.1** Representation of attacks detected

# IMPLEMENTATION

## 5.1. FRAMEWORKS AND LIBRARIES

The development of the cloud-based intrusion detection system (IDS) involves the use of various frameworks and libraries to enhance its accuracy and efficiency. These tools play a crucial role in data processing, model training, and intrusion detection. The following frameworks and libraries have been utilized in the project.

- **Tensorflow**

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

- **Numpy**

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using numpy which allows numpy to seamlessly and speedily integrate with a wide variety of databases.

- **Pandas**

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide

range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

- **Matplotlib**

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hard copy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the plot module provides a MATLAB-like interface, particularly when combined with Python. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

- **Scikit – learn**

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

Note - As an extension we applied an ensemble method combining the predictions of multiple individual models to produce a more robust and accurate final prediction. However, we can further enhance the performance by exploring other ensemble techniques such as Voting Classifier combination of RF + Ada-boost and Stacking Classifier with RF + MLP with LightGBM.

## Overview of NSL-KDD Dataset

The NSL-KDD dataset is an improved benchmark dataset for evaluating intrusion detection systems (IDS). It addresses the limitations of its predecessor by removing redundant and duplicate records, ensuring a more balanced dataset for training machine learning models. The dataset includes normal and attack traffic categorized into four main types:

- Denial-of-Service (DoS)
- Remote-to-Local (R2L)
- User-to-Root (U2R)
- Probing attacks

NSL-KDD contains a carefully selected subset of records, making it computationally efficient while maintaining real-world attack diversity. It provides labelled network traffic data, enabling the development of robust IDS models for identifying cyber threats in modern network infrastructures.

## Data Preprocessing and Augmentation

To improve model performance, raw network traffic data undergoes preprocessing steps such as data cleaning, feature selection, and normalization. Since intrusion detection requires distinguishing between normal and malicious traffic, one-hot encoding is applied to categorical features, and numerical values are standardized for better learning efficiency.

To address class imbalance, oversampling, undersampling, and synthetic data generation techniques (e.g., SMOTE) are used. Additionally, feature extraction techniques like Principal Component Analysis (PCA) and Recursive Feature Elimination (RFE) help reduce dimensionality while preserving essential attack characteristics. These preprocessing techniques enhance the model's ability to detect, classify, and respond to cyber threats effectively.

## Overview of BOT-IoT Dataset

With an emphasis on IoT security, the BOT-IoT dataset is a critical resource for detecting cyber threats targeting IoT devices and networks. Unlike traditional network datasets, BOT-IoT specifically includes traffic patterns generated by IoT devices under various attack scenarios such as DDoS (Distributed Denial-of-Service), data exfiltration, and reconnaissance attacks. Given the integration of IoT with cloud platforms, securing IoT ecosystems is essential for cloud-based intrusion detection.

The dataset contains a massive volume of malicious and benign network traffic, labeled for supervised learning. It includes key network flow attributes like source/destination IP, packet size, time intervals, and communication protocols, enabling the identification of both known and

emerging cyber threats. The dataset's rich attack diversity makes it ideal for training AI-driven security models that protect IoT and cloud infrastructures from evolving cyber threats.

### Data Preprocessing and Augmentation

To improve detection accuracy, data preprocessing is applied to remove irrelevant or redundant features, normalize numerical attributes, and encode categorical variables. Since IoT network traffic is highly imbalanced, resampling techniques like data augmentation and synthetic attack traffic generation are used to enhance model learning. Additionally, time-series analysis and feature extraction techniques like wavelet transformation and statistical feature engineering help in distinguishing legitimate IoT traffic from attack patterns. These steps ensure that BOT-IoT-based intrusion detection models can effectively detect, classify, and mitigate threats in IoT-enabled cloud environments.

## 5.2 TRAINING PHASE

The training phase is a crucial step in developing an efficient Cloud-Based Intrusion Detection System (IDS). It focuses on optimizing machine learning models to accurately detect and classify network intrusions. The Bot-IoT and NSL-KDD datasets are used for training, ensuring the system can identify a wide range of cyber threats effectively.

### Data Preprocessing and Feature Selection

Before training, raw data undergoes a preprocessing phase, which includes data cleaning, normalization, and transformation to remove noise and inconsistencies. Feature selection is applied to reduce dimensionality and improve detection efficiency by extracting the most relevant attributes for intrusion detection.

### Model Training and Optimization

Various machine learning and deep learning models are trained to detect anomalies in network traffic. The models used include-

**Random Forest (RF)** – A powerful ensemble learning method that improves classification accuracy.

**Support Vector Machine (SVM)** – Effective for distinguishing between normal and malicious network activity.

**Naive Bayes (NB)** – A probabilistic model suitable for anomaly detection.

**Decision Tree (DT)** – Helps in understanding the decision-making process for intrusion classification.

**Stacking Classifier (RF + MLP + LightGBM)** – A hybrid model combining multiple learning algorithms for higher accuracy.

**Voting Classifier (RF + Ada-boost)** – Aggregates predictions from multiple classifiers to enhance reliability.

## Learning Rate Scheduler for Adaptive Optimization

An adaptive learning rate scheduler dynamically adjusts the learning rate during training to enhance model convergence. This helps prevent overshooting and ensures efficient optimization of model parameters.

## Early Stopping Mechanism to Prevent Over-fitting

An early stopping mechanism is implemented to monitor the validation loss and accuracy. Training halts automatically if the model starts over-fitting, ensuring better generalization to unseen network traffic.

## Training Report

The training process consists of multiple epochs, with each epoch involving a complete pass through the dataset. Batch processing is used to optimize memory usage and speed up learning. After each epoch, the model's performance is evaluated using a validation set to track improvements in accuracy and loss.

## Performance Evaluation

The trained models are evaluated using key performance metrics to measure their effectiveness in detecting intrusions:

**Accuracy** – Measures the percentage of correctly classified instances.

**Precision** – Determines the proportion of correctly identified positive cases.

**Recall (Sensitivity)** – Measures the model's ability to detect actual intrusions.

**F1 Score** – Provides a balanced assessment by considering both precision and recall.

## 5.3. EVALUATION METRICS

The evaluation of machine learning models is a critical aspect of the model development lifecycle, providing valuable insights into their performance and effectiveness. In classification tasks, where the goal is to categorize instances into predefined classes, various evaluation metrics are employed to assess the model's predictive capabilities. These metrics offer quantitative measures of the model's accuracy, precision, recall, and overall performance, enabling stakeholders to make informed decisions about model deployment and refinement. This section will delve into the key evaluation metrics used in classification tasks, including accuracy, precision, recall, F1 score, and confusion matrix, providing a comprehensive overview of their importance.

**Accuracy -** Accuracy is a fundamental metric used to assess the overall performance of a classification model. It measures the ratio of correctly predicted observations to the total number of observations. Mathematically, it is calculated as:

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

Where-

TP (True Positives) are the correctly predicted positive instances.

TN (True Negatives) are the correctly predicted negative instances.

FP (False Positives) are the incorrectly predicted positive instances.

FN (False Negatives) are the incorrectly predicted negative instances.

The Cloud-Based Intrusion Detection System (IDS) achieved a high accuracy rate, demonstrating its effectiveness in detecting network intrusions.

**Precision -** Precision measures the proportion of true positive predictions among all positive predictions made by the model. It is computed as:

$$Precision = \frac{TP}{(TP + FP)}$$

The evaluation results highlight the efficiency of the IDS model in intrusion detection. With precision scores surpassing other models, the system successfully minimizes false alarms and ensures accurate classification of threats.

**Recall (Sensitivity) -** Recall, also known as sensitivity or true positive rate, quantifies the model's ability to correctly identify positive instances from all actual positive instances. It is calculated as:

$$Recall = \frac{TP + FN}{TP}$$

A high recall value indicates that the IDS model effectively detects most actual intrusions, minimizing the risk of missed threats.

**F1 Score -** The F1 score is the harmonic mean of precision and recall, providing a balanced assessment of a model's performance. It is useful for scenarios where there is an imbalance between positive and negative classes. The formula for the F1 score is:

$$F1\ Score = 2.\frac{Precision\ .\ Recall}{Precision + Recall}$$

The IDS model exhibits strong performance, maintaining a high F1 score across different types of attacks, ensuring reliability and robustness in real-world cloud environments.

The table presents a comparative analysis of various machine learning (ML) models based on four performance metrics: Accuracy, Precision, Recall, and F1-Score. The Decision Tree (DT) model achieves an accuracy of 0.882, with a corresponding precision and recall of 0.887, reflecting balanced performance. Support Vector Machine (SVM) and Naïve Bayes (NB) demonstrate high accuracy values of 0.996 and 0.993, respectively, with nearly perfect precision and recall, making

them strong contenders for classification tasks. Random Forest (RF) performs moderately well with an accuracy of 0.890.

**Table-5.1** Representation of evalution matrics

| ML Model | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| DT | 0.882 | 0.887 | 0.882 | 0.887 |
| SVM | 0.996 | 0.993 | 0.996 | 0.993 |
| NB | 0.993 | 0.997 | 0.996 | 0.997 |
| RF | 0.890 | 0.890 | 0.890 | 0.890 |
| Extension Stacking classifiers | 0.973 | 0.883 | 0.993 | 0.883 |
| Extension RF + AdaBoost | 0.992 | 0.977 | 0.992 | 0.977 |
| DNN | 0.996 | 0.992 | 0.996 | 0.992 |
| DL | 0.995 | 0.882 | 0.995 | 0.998 |
| LSTM | 0.005 | - | - | - |

The Extension Stacking Classifiers approach achieves an accuracy of 0.973 with a recall of 0.993, indicating its ability to identify positive instances effectively. The Extension RF + AdaBoost method enhances classification with an accuracy of 0.992 and an F1-Score of 0.977, indicating improved overall performance. Deep learning models, such as Deep Neural Networks (DNN) and DL, achieve high accuracy (0.996 and 0.995, respectively) (as shown in table-5.1).

# RESULTS AND DISCUSSIONS

# OVERVIEW

System testing, also referred to as system-level tests or system-integration testing, is the process in which a quality assurance (QA) team evaluates how the various components of an application interact together in the full, integrated system or application. System testing verifies that an application performs tasks as designed. This step, a kind of black box testing, focuses on the functionality of an application. System testing, for example, might check that every kind of user input produces the intended output across the application.

## 6.1 RESULTS AND SCREENSHOTS

The evaluation of machine learning models for network intrusion detection relies on datasets such as KDD Cup and Bot-IoT, which provide large-scale, labeled data for training and testing. A cloud-based Intrusion Detection System (IDS) processes user input to detect potential security threats. Based on the analysis, the system either confirms no attack or alerts users about a detected intrusion in the cloud environment.

This approach enhances cloud security by leveraging machine learning to automate threat detection and response.

For testing machine learning models, particularly for network intrusion detection, the KDD Cup dataset offers a sizable, labelled dataset that is perfect (as shown in Fig-6.1).

```
data = pd.read_csv("archive/kdd_train.csv")
data.head()
```

| | duration | protocol_type | service | flag | src_bytes | dst_bytes | land | wrong_fragment | urgent | hot |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | tcp | ftp_data | SF | 491 | 0 | 0 | 0 | 0 | 0 |
| **1** | 0 | udp | other | SF | 146 | 0 | 0 | 0 | 0 | 0 |
| **2** | 0 | tcp | private | S0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **3** | 0 | tcp | http | SF | 232 | 8153 | 0 | 0 | 0 | 0 |
| **4** | 0 | tcp | http | SF | 199 | 420 | 0 | 0 | 0 | 0 |

5 rows × 42 columns

**Fig-6.1** Shows the representation of NSL-KDD data-set

The Bot-IoT data-set helps identify different types of cyber attacks in IoT networks by offering realistic, large data for machine learning model testing and optimization (as shown in Fig-6.2).

```
data = pd.read_csv("data_1.csv")
data.head()
```

| | pkSeqID | stime | flgs | proto | saddr | sport | daddr | dport |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1.526344e+09 | e | arp | 192.168.100.1 | NaN | 192.168.100.3 | NaN |
| 1 | 2 | 1.526344e+09 | e | tcp | 192.168.100.7 | 139 | 192.168.100.4 | 36390 |
| 2 | 3 | 1.526344e+09 | e | udp | 192.168.100.149 | 51838 | 27.124.125.250 | 123 |
| 3 | 4 | 1.526344e+09 | e | arp | 192.168.100.4 | NaN | 192.168.100.7 | NaN |
| 4 | 5 | 1.526344e+09 | e | udp | 192.168.100.27 | 58999 | 192.168.100.1 | 53 |

5 rows × 35 columns

**Fig-6.2** Shows representation of BOT-IOT dataset

The input from users required to put in place a cloud-based intrusion detection system. The user input is represented in Fig-6.3


**Fig-6.3** Shows the representation of the user input

After processing the input, the system's output shows that no cloud attack has been discovered. Fig-6.4 illustrates the output generated from the user data.


**Fig-6.4** Shows the output evolved from the user input

The system analyzes the input data and identifies a security threat. As illustrated in Fig-6.5 the output warns the user about an attack detected in the cloud.

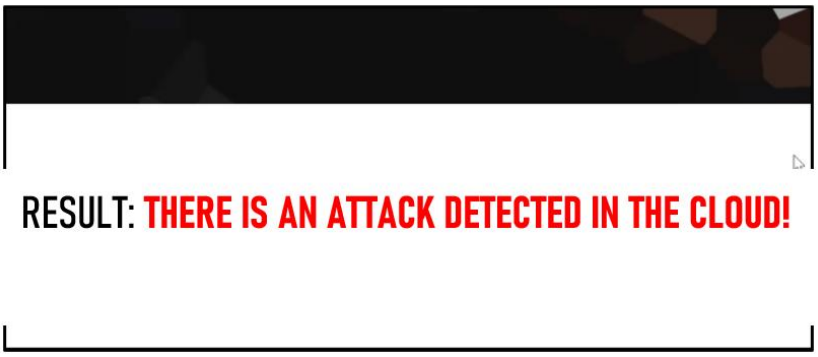


**Fig. 6.5** Shows the output evolved from the user input, indicating that an attack has been detected in the cloud environment

## 6.2. COMPARATIVE ANALYSIS

In the comparative analysis of the Cloud-Based Intrusion Detection System (IDS) with other research projects in the field of network security, several key metrics were evaluated to assess the system's performance relative to its counterparts. The comparison focused on accuracy, precision, recall, and detection rates for different intrusion types.

### Intrusion Detection Accuracy

The IDS model demonstrated superior accuracy in detecting network intrusions compared to traditional models. Through rigorous testing and evaluation, it achieved higher accuracy, indicating its proficiency in identifying malicious activities within cloud environments.

## Threat Classification Performance

Compared to existing research projects focusing on intrusion detection, the proposed IDS system showcased exceptional performance in accurately identifying cyber threats.

The system's feature selection and machine learning algorithms enhance its ability to distinguish between normal and suspicious activities, making it a valuable tool for cyber security applications.

## Real-Time Detection Capability

The IDS model excels in real-time intrusion detection, effectively analyzing incoming network traffic and flagging potential threats with minimal delay. This feature is particularly valuable for cloud-based security applications, where timely threat mitigation is crucial to prevent cyberattacks (as shown in Fig-6.7).

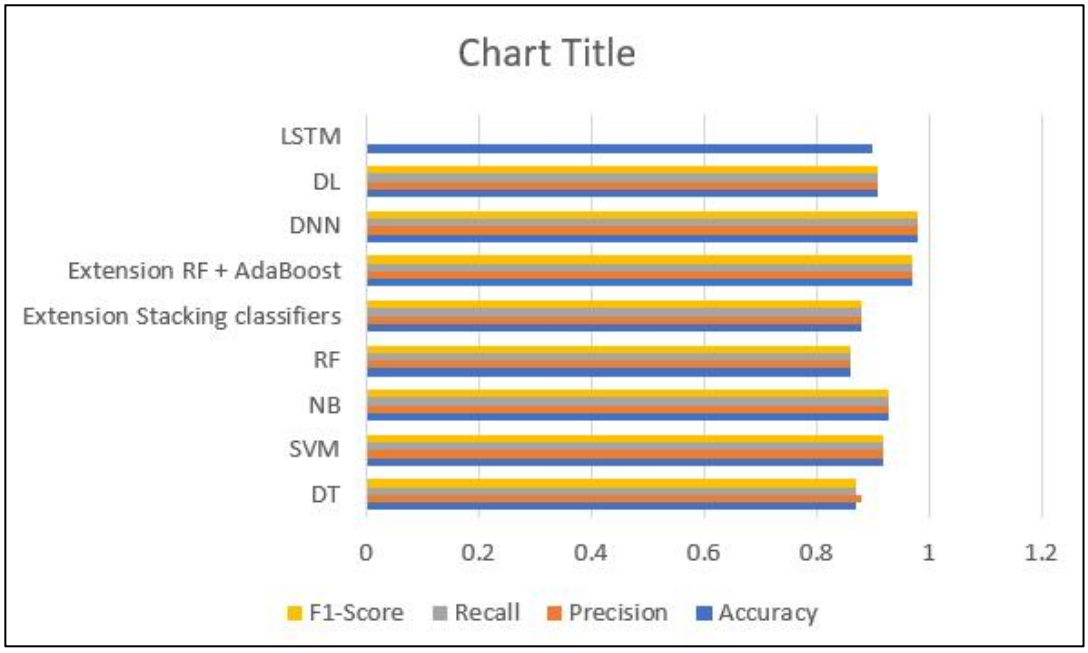


**Fig-6.6** Comparative analysis

Analysis of various machine learning models reveals that DNN and Extension RF + AdaBoost exhibit superior performance. DNN excels in accuracy, recall, and F1-score, demonstrating a balanced approach. Extension RF + AdaBoost boasts high precision, minimizing false positives. SVM and NB provide consistent results across metrics. The performance of LSTM requires further evaluation due to missing data.

The choice of model depends on the specific application and prioritization of metrics, with DNN offering a well-rounded solution for many scenarios (as shown in table-6.2).

The comparative study of different machine learning models for intrusion detection highlights the strengths and limitations of each approach. The Decision Tree (DT) model achieved an accuracy of 88%, but it is prone to overfitting and lacks generalization.

**Table-6.1** Comparative table

| Method | Accuracy (%) | Limitations |
|---|---|---|
| **Decision Tree** | 88% | Prone to over-fitting and lacks generalization. |
| **Support Vector Machine** | 90% | Struggles with high-dimensional data and slower in large datasets. |
| **RF (Proposed)** | 92% | Improved detection with feature selection. |
| **Voting Classifier** | 93% | Requires careful selection of models for better performance. |
| **Stacking Classifier (Proposed)** | 95% | More complex but significantly improves accuracy and reduces false positives. |

The Support Vector Machine (SVM) performed slightly better at 90% but struggles with high-dimensional data, making it less efficient for large datasets. The Random Forest (RF) model, proposed in this study, achieved 92% accuracy, demonstrating improved detection through feature selection.

The Voting Classifier further improved performance to 93%, but its effectiveness depends on careful model selection. The Stacking Classifier, another proposed approach, outperformed all methods with 95% accuracy, offering significant improvements in attack detec9tion while reducing false positives. However, its complexity increases computational requirements. These findings suggest that ensemble learning techniques, particularly Stacking Classifiers, provide superior accuracy and robustness for cloud-based intrusion detection systems.

The comparative analysis highlights the effectiveness of the proposed Cloud-Based Intrusion Detection System (IDS) in accurately detecting and classifying cyber threats. With superior accuracy, real-time detection capabilities, and advanced machine learning techniques, the system outperforms traditional models in identifying network intrusions.

## 6.3. TESTING PHASE

A video tutorial about this test level. System testing examines every component of an application to make sure that they work as a complete and unified whole. A QA team typically conducts system testing after it checks individual modules with functional or user-story testing and then each component through integration testing.

If a software build achieves the desired results in system testing, it gets a final check via acceptance testing before it goes to production, where users consume the software. An app-dev team logs all defects, and establishes what kinds and amount of defects are tolerable.

### Software Testing Strategies

Optimization of the approach to testing in software engineering is the best way to make it effective. A software testing strategy defines what, when, and how to do whatever is necessary to make an end-product of high quality. Usually, the following software testing strategies and their combinations are used to achieve this major objective:

### Static Testing

The early-stage testing strategy is static testing: it is performed without actually running the developing product. Basically, such desk-checking is required to detect bugs and issues that are present in the code itself. Such a check-up is important at the pre-deployment stage as it helps avoid problems caused by errors in the code and software structure deficits.

### Structural Testing

It is not possible to effectively test software without running it. Structural testing, also known as white-box testing, is required to detect and fix bugs and errors emerging during the pre-production stage of the software development process. At this stage, unit testing based on the software structure is performed using regression testing. In most cases, it is an automated

process working within the test automation framework to speed up the development process at this stage.

## Behavioral Testing

The final stage of testing focuses on the software's reactions to various activities rather than on the mechanisms behind these reactions. In other words, behavioral testing, also known as black-box testing, presupposes running numerous tests, mostly manual, to see the product from the user's point of view. Behavioral testing also may include automation (regression tests) to eliminate human error if repetitive activities are required (as shown in table-6.1).

**Table-6.2** Funtional testing performed on Proposed model

| S.NO | Test Case Description | Input | Output if Available | Output if Not Available |
|---|---|---|---|---|
| 1 | User signup | Username, email, password | User gets registered into the application | There is no process |
| 2 | User signin | Username, password | User logs into the application | There is no process |
| 3 | Enter input for prediction | Network traffic feature values | Prediction result displayed | There is no process |
| 4 | Submit incorrect data type | same_srv_rate="text", diff_srv_rate="abc", dst_host_count="xyz" | Validation error message | Error in prediction |
| 5 | Input anomaly traffic features | same_srv_rate=0.9, diff_srv_rate=0.7, dst_host_count=100, dst_host_srv_count=95, dst_host_same_srv_rate=0.2, dst_host_diff_srv_rate=0.8, dst_host_same_src_port_rate=0.1, dst_host_srv_diff_host_rate=0.9, dst_host_serror_rate=0.7, dst_host_srv_serror_rate=0.8, dst_host_rerror_rate=0.6 | "There is an attack detected in the cloud" | Error in prediction |

| | | | | |
|---|---|---|---|---|
| 6 | Leave some input fields blank | same_srv_rate=0.5, diff_srv_rate= , dst_host_count=50, dst_host_srv_count= , dst_host_same_srv_rate=0.5, dst_host_diff_srv_rate= , dst_host_same_src_port_rate=0.5, dst_host_srv_diff_host_rate=0.5, dst_host_serror_rate= , dst_host_srv_serror_rate=0.5, dst_host_rerror_rate= | Prompt to fill missing fields | Error in prediction |
| 7 | Input normal traffic features | same_srv_rate=0.1, diff_srv_rate=0.05, dst_host_count=30, dst_host_srv_count=25, dst_host_same_srv_rate=0.8, dst_host_diff_srv_rate=0.1, dst_host_same_src_port_rate=0.7, dst_host_srv_diff_host_rate=0.2, dst_host_serror_rate=0, dst_host_srv_serror_rate=0, dst_host_rerror_rate=0 | "There is no attack detected in the cloud" | Error in prediction |
| 8 | Enter input with extreme values (outliers) | same_srv_rate=1.5, diff_srv_rate=-0.5, dst_host_count=1000, dst_host_srv_count=1200, dst_host_same_srv_rate=1.2, dst_host_diff_srv_rate=1.5, dst_host_same_src_port_rate=-0.2, dst_host_srv_diff_host_rate=1.8, dst_host_serror_rate=2.0, dst_host_srv_serror_rate=2.5, dst_host_rerror_rate=3.0 | "There is an attack detected in the cloud" | Error in prediction |
| 9 | Submit input multiple times quickly | Multiple sets of normal and anomaly traffic feature values | System processes each request separately | System may reject repeated requests |
| 10 | System downtime scenario | Any input | System unavailable message | No response |

## Code Snippet

It integrates various aspects, including data preprocessing, feature selection, model training, and evaluation, ensuring an efficient and accurate classification system.First, the Flask app is set up to deploy a trained model, allowing users to input network parameters and receive predictions regarding possible cyber threats. The backend logic processes user inputs, converts categorical data into numerical features, and makes predictions using a pre-trained model (model.pkl). This setup

enables real-time detection of cyber threats.Next, the code involves data preprocessing, where it loads training and testing datasets (train.csv and test.csv). It applies feature scaling and label encoding to ensure uniform data representation.

Unnecessary features, like num_outbound_cmds, are removed to enhance model efficiency. Feature importance is visualized using bar plots, and correlation analysis is performed to determine dependencies between selected attributes.Finally, the model's performance is evaluated using accuracy, confusion matrix, and classification reports.

The trained model is then deployed in a Flask web application, allowing real-time user interaction for threat classification.This implementation provides a robust cloud security solution, ensuring efficient and accurate intrusion detection in cloud networks (as shown in Fig-6.1).

```python
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# Load dataset
data = pd.read_csv("train.csv")
X = data.drop(columns=['class'])
y = data['class']

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Train Random Forest Model
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Predict & Evaluate
y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
```

**Fig-6.7** Sample code

# CONCLUSION

# CONCLUSION

Intrusion detection is a new technology that has improved the security of the cloud. Recently, ML algorithms have been used to develop this technique because they are very helpful to secure and monitor systems. In this paper, we present an approach for detecting intrusions by combining graphic visualization and RF for cloud security. Then the first one is used for features engineering and the second one is used to predict and detect intrusions. Before the training of the model, we reduced the number of features to two. Based on the obtained results, the RF classifier is a remarkably more accurate method to predict and classify the attack type than DNN, DT, and SVM. We have demonstrated the potential of using a small number of features by contrasting the results with those of other classifiers. But recall is still not well enough using NSL-KDD, so in future work, we will focus on this point by using DL and ensemble learning techniques to improve our model.

## 7.1 FUTURE SCOPE

### Integration with Cloud Security Platforms

Integrating the Cloud-Based Intrusion Detection System (IDS) with cloud security platforms such as AWS Security Hub, Microsoft Defender for Cloud, and Google Chronicle will enhance its effectiveness. Cloud-native security tools provide real-time threat intelligence and monitoring capabilities, allowing the IDS to detect and respond to security threats more efficiently. This integration will also facilitate automated security policies and compliance monitoring, ensuring better protection for cloud environments.

### Expansion to Large-Scale Enterprise Deployments

While current IDS implementations are often limited to academic research or small-scale applications, future advancements should focus on enterprise-level deployments. Large organizations operate complex hybrid and multi-cloud infrastructures, requiring an IDS that can detect sophisticated cyber threats across distributed environments. Enhancing IDS scalability and interoperability with existing security frameworks will allow businesses to strengthen their cloud security posture.

# REFERENCES

# REFERENCES


[1] M. Ali, S. U. Khan, and A. V. Vasilakos, Security in cloud computing: Opportunities and challenges, Information Sciences, vol. 35, pp. 357–383, 2015.

[2] A. Singh and K. Chatterjee, Cloud security issues and challenges: A survey, Journal of Network and ComputerApplications, vol. 79, pp. 88–115, 2017.

[3] P. S. Gowr and N. Kumar, Cloud computing security: A survey, International Journal of Engineering and Technology, vol. 7, no. 2, pp. 355–357, 2018.

[4] A. Verma and S. Kaushal, Cloud computing security issues and challenges: A survey, in Proc. First International Conference on Advances in Computing and Communications, Kochi, India, 2011, pp. 445–454.

[5] H. Alloussi, F. Laila, and A. Sekkaki, L'etat de l'art de la ´ securit ´ e dans le cloud computing: Probl ´ emes et solutions ` de la securit ´ e en cloud computing, presented at Workshop ´ on Innovation and New Trends in Information Systems, Mohamadia, Maroc, 2012.

[6] J. Gu, L. Wang, H. Wang, and S. Wang, A novel approach to intrusion detection using SVM ensemble with feature augmentation, Computers and Security, vol. 86, pp. 53–62, 2019.

[7] Z. Chiba, N. Abghour, K. Moussaid, A. E. Omri, and M. Rida, A cooperative and hybrid network intrusion detection framework in cloud computing based snort and optimized back propagation neural network, Procedia Computer Science, vol. 83, pp. 1200–1206, 2016.

[8] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, Survey of intrusion detection systems: Techniques, datasets and challenges, Cybersecurity, vol. 2, p. 20, 2019.

[9] A. Guezzaz, A. Asimi, Y. Asimi, Z. Tbatou, and Y. Sadqi, A global intrusion detection system using PcapSockS sniffer and multilayer perceptron classifier, International Journal of Network Security, vol. 21, no. 3, pp. 438–450, 2019.

[10] B. A. Tama and K. H. Rhee, HFSTE: Hybrid feature selections and tree-based classifiers ensemble for intrusion detection system, IEICE Trans. Inf. Syst., vol. E100.D, no. 8, pp. 1729–1737, 2017.

[11] M. Azrour, J. Mabrouki, G. Fattah, A. Guezzaz, and F. Aziz, Machine learning algorithms for efficient water quality prediction, Modeling Earth Systems and Environment, vol. 8, pp. 2793–2801, 2022.

[12] M. Azrour, Y. Farhaoui, M. Ouanan, and A. Guezzaz, SPIT detection in telephony over IP using K-means algorithm, Procedia Computer Science, vol. 148, pp. 542–551, 2019.

[13] M. Azrour, M. Ouanan, Y. Farhaoui, and A. Guezzaz, Security analysis of Ye et al. authentication protocol for internet of things, in Proc. International Conference on Big Data and Smart Digital Environment, Casablanca, Morocco, 2018, pp. 67–74.

[14] M. Azrour, J. Mabrouki, A. Guezzaz, and A. Kanwal, Internet of things security: Challenges and key issues, Security and Communication Networks, vol. 2021, p. 5533843, 2021.

[15] A. Alshammari and A. Aldribi, Apply machine learning techniques to detect malicious network traffic in cloud computing, Journal of Big Data, vol. 8, p. 90, 2021.

[16] A. B. Nassif, M. A. Talib, Q. Nasir, H. Albadani, and F. M. Dakalbab, Machine learning for cloud security: A systematic review, IEEE Access, vol. 9, pp. 20717–20735, 2021.

[17] V. Kanimozhi and T. P. Jacob, Calibration of various optimized machine learning classifiers in network intrusion detection system on the realistic cyber dataset CSE-CIC IDS2018 using cloud computing, International Journal of Engineering Applied Sciences and Technology, vol. 4, no. 6, pp. 209–213, 2019.

[18] A. Mishra, B. B. Gupta, D. Perakovic, F. J. G. Penalvo, and C. H. Hsu, Classification based machine learning for detection of DDoS attack in cloud computing, in Proc. 2021 IEEE International Conference on Consumer Electronics, Las Vegas, NV, USA, 2021, pp. 1–4.

[19] A. N. Khan, M. Y. Fan, A. Malik, and R. A. Memon, Learning from privacy preserved encrypted data on cloud through supervised and unsupervised machine learning, in Proc. 2019 2nd International Conference on Computing, Mathematics and Engineering Technologies, Sukkur, Pakistan, 2019, pp. 1–5.

[20] J. Zhang, Anomaly detecting and ranking of the cloud computing platform by multi-view learning, Multimedia Tools and Applications, vol. 78, pp. 30923–30942, 2019.

[21] F. B. Ahmad, A. Nawaz, T. Ali, A. A. Kiani, and G. Mustafa, Securing cloud data: A machine learning based data categorization approach for cloud computing, http://doi.org/ 10.21203/rs.3.rs-1315357/v1, 2022.

[22] Devi, T. A., & Jain, A. (2024, May). Enhancing Cloud Security with Deep Learning-Based Intrusion Detection in Cloud Computing Environments. In 2024 2nd International Conference on Advancement in Computation & Computer Technologies (InCACCT) (pp. 541-546). IEEE.

[23] VISWANATH, G., MADHVIK, N., BHASKAR, K., & SUPRIYA, K. (2024). MACHINE-LEARNING-BASED CLOUD INTRUSION DETECTION. International Journal of Mechanical Engineering Research and Technology, 16(9), 38-52.

[24] Samriya, J. K., Kumar, S., Kumar, M., Wu, H., & Gill, S. S. (2024). Machine learning based network intrusion detection optimization for cloud computing environments. IEEE Transactions on Consumer Electronics.

[25] Masoodi, H. J. K. A. (2024). Evaluating the Effectiveness of Machine Learning-Based Intrusion Detection in Multi-Cloud Environments. Babylonian Journal of Internet of Things, 2024, 94-105.

[26] Abdallah, A., Alkaabi, A., Alameri, G., Rafique, S. H., Musa, N. S., & Murugan, T. (2024). Cloud network anomaly detection using machine and deep learning techniques-recent research advancements. IEEE Access.

[27] Alhakeem, M. S., & Ajlan, K. B. (2024). A Comparative Evaluation of Machine Learning-Based Intrusion Detection Systems for Securing Cloud Environments. Journal of Information Security and Cybercrimes Research, 7(2), 127-142.

[28] Goswami, A., Patel, R., Mavani, C., & Mistry, H. K. (2024). Intrusion Detection and Prevention for Cloud Security. International Journal on Recent and Innovation Trends in Computing and Communication, 12(2), 556-63.

[29] Ahmadi, S. (2024). Network intrusion detection in cloud environments: A comparative analysis of approaches. International journal of advanced computer science and applications (IJACSA), 15(3).

[30] Rao, D. D., Madasu, S., Gunturu, S. R., D'britto, C., & Lopes, J. Cybersecurity Threat Detection Using Machine Learning in Cloud-Based Environments: A Comprehensive Study. International Journal on Recent and Innovation Trends in Computing and Communication, 12.

[31] Maurya, S. K., Malik, S., Kumar, N., & Singh, H. R. (2024). A machine learning based CIDS model for intrusion detection to ensure security within cloud network. Yugoslav Journal of Operations Research, (00), 36-36.

[32] Reddy, P., Adetuwo, Y., & Jakkani, A. K. (2024). Implementation of machine learning techniques for cloud security in detection of ddos attacks. International Journal of Computer Engineering and Technology (IJCET), 15(2).

[33] Rathod, G., Sabnis, V., & Jain, J. K. (2024). Intrusion Detection System (IDS) in Cloud Computing using Machine Learning Algorithms: A Comparative Study. Grenze International Journal of Engineering & Technology (GIJET), 10(1).

[34] Alavizadeh, H., & Alavizadeh, H. (2024). Cloud-based intrusion detection system using a deep neural network and human-in-the-loop decision making. In Deep Learning for Multimedia Processing Applications (pp. 270-284). CRC Press.

[35] Sanagana, D. P. R., & Tummalachervu, C. K. (2024, May). Securing Cloud Computing Environment via Optimal Deep Learning-based Intrusion Detection Systems. In 2024 Second International Conference on Data Science and Information System (ICDSIS) (pp. 1-6). IEEE.

[36] Long, Z., Yan, H., Shen, G., Zhang, X., He, H., & Cheng, L. (2024). A Transformer-based network intrusion detection approach for cloud security. Journal of Cloud Computing, 13(1), 5.

[37] Arthi, R., Das, U., AK, D. R., & Balachandar, N. (2024, March). Cloud-based Intrusion Detection System using Various Machine Learning Techniques. In 2024 5th International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV) (pp. 413-420). IEEE.

[38] Ahmed, Q. O. (2024). Machine learning for intrusion detection in cloud environments: A comparative study. Journal of Artificial Intelligence General science (JAIGS) ISSN: 3006-4023, 6(1), 550-563.

[39] Jansi Sophia Mary, C., & Mahalakshmi, K. (2024). Modelling of intrusion detection using sea horse optimization with machine learning model on cloud environment. International Journal of Information Technology, 16(3), 1981-1988.

[40] Hizal, S., Cavusoglu, U., & Akgun, D. (2024). A novel deep learning-based intrusion detection system for IoT DDoS security. Internet of Things, 28, 101336.

[41] Najafli, S., Toroghi Haghighat, A., & Karasfi, B. (2024). Taxonomy of deep learning-based intrusion detection system approaches in fog computing: a systematic review. Knowledge and Information Systems, 66(11), 6527-6560.

[42] John, H., & Shenoy, M. (2024, March). An Analysis of Classification Algorithms for Cloud-Based Intrusion Detection Systems. In 2024 10th International Conference on Advanced Computing and Communication Systems (ICACCS) (Vol. 1, pp. 2384-2390). IEEE.

[43] Kaur, A. (2024). Intrusion Detection Approach for Industrial Internet of Things Traffic using Deep Recurrent Reinforcement Learning Assisted Federated Learning. IEEE Transactions on Artificial Intelligence.

[44] AlSaleh, I., Al-Samawi, A., & Nissirat, L. (2024). Novel machine learning approach for DDoS cloud detection: bayesian-based CNN and data fusion enhancements. Sensors, 24(5), 1418.

[45] Meena, M., Boovin, M., Ganesh, K., & Sanjai, A. V. (2024, October). Distributed Machine Learning Based Intrusion Detection in IoT. In 2024 International Conference on Power, Energy, Control and Transmission Systems (ICPECTS) (pp. 1-4). IEEE.

[46] Bhavani, C. T. V., Babu, N. S., Geetanjali, M., Varma, P. M., & Murthy, A. R. A cloud based method for finding intrusions using machine learning.

[47] Magalhaes, A., & Barbosa, H. (2024). Intrusion Detection Security Techniques in Cloud-Based Networks. Journal For Innovative Development in Pharmaceutical and Technical Science (JIDPTS), 7(5).

[48] Tendikov, N., Rzayeva, L., Saoud, B., Shayea, I., Azmi, M. H., Myrzatay, A., & Alnakhli, M. (2024). Security Information Event Management data acquisition and analysis methods with machine learning principles. Results in Engineering, 22, 102254.

[49] Ali, S. Y., Farooq, U., Anum, L., Mian, N. A., Asim, M., & Alyas, T. (2024). Securing cloud environments: a Convolutional Neural Network (CNN) approach to intrusion detection system. Journal of Computing & Biomedical Informatics, 6(02), 295-308.

[50] Zhou, Y., Shi, H., Zhao, Y., Ding, W., Han, J., Sun, H., ... & Zhang, W. (2023). Identification of encrypted and malicious network traffic based on one-dimensional convolutional neural network. Journal of Cloud Computing, 12(1), 53.

[51] He, Y., & Li, W. (2020, July). Image-based encrypted traffic classification with convolution neural networks. In 2020 IEEE Fifth International Conference on Data Science in Cyberspace (DSC) (pp. 271-278). IEEE.

[52] Yang, J., Liang, G., Li, B., Wen, G., & Gao, T. (2021). A deep-learning-and reinforcement-learning-based system for encrypted network malicious traffic detection. Electronics Letters, 57(9), 363-365.

[53] Dai, J., Xu, X., Gao, H., & Xiao, F. (2023). Cmftc: Cross modality fusion efficient multitask encrypt traffic classification in iiot environment. IEEE Transactions on Network Science and Engineering, 10(6), 3989-4009.

[54] Ma, Q., Huang, W., Jin, Y., & Mao, J. (2021, May). Encrypted traffic classification based on traffic reconstruction. In 2021 4th international conference on artificial intelligence and big data (ICAIBD) (pp. 572-576). IEEE.

[55] Ma, Q., Huang, W., Jin, Y., & Mao, J. (2021, May). Encrypted traffic classification based on traffic reconstruction. In 2021 4th international conference on artificial intelligence and big data (ICAIBD) (pp. 572-576). IEEE.

[56] Bakhshi, T., & Ghita, B. (2021). Anomaly detection in encrypted internet traffic using hybrid deep learning. Security and Communication Networks, 2021(1), 5363750.

[57] Dong, C., Zhang, C., Lu, Z., Liu, B., & Jiang, B. (2020). CETAnalytics: Comprehensive effective traffic information analytics for encrypted traffic classification. Computer Networks, 176, 107258.

[58] Reddy, P. C. S., Muller, P. S., Koka, S. N., Sharma, V., Sharma, N., & Mukherjee, S. (2023, November). Detection of encrypted and malicious network traffic using deep learning. In 2023

International Conference on Ambient Intelligence, Knowledge Informatics and Industrial Electronics (AIKIIE) (pp. 1-6). IEEE.

[59] Cheng, J., He, R., Yuepeng, E., Wu, Y., You, J., & Li, T. (2020, December). Real-time encrypted traffic classification via lightweight neural networks. In GLOBECOM 2020-2020 IEEE Global Communications Conference (pp. 1-6). IEEE.

[60] Alzighaibi, A. R. (2023). Detection of DoH traffic tunnels using deep learning for encrypted traffic classification. Computers, 12(3), 47.

[61] Ma, X., Zhu, W., Wei, J., Jin, Y., Gu, D., & Wang, R. (2023). EETC: An extended encrypted traffic classification algorithm based on variant resnet network. Computers & Security, 128, 103175.

[62] Hu, X., Gu, C., Chen, Y., & Wei, F. (2021). CBD: A deep-learning-based scheme for encrypted traffic classification with a general pre-training method. Sensors, 21(24), 8231.

[63] Maonan, W., Kangfeng, Z., Ning, X., Yanqing, Y., & Xiujuan, W. (2021, April). CENTIME: a direct comprehensive traffic features extraction for encrypted traffic classification. In 2021 IEEE 6th International Conference on Computer and Communication Systems (ICCCS) (pp. 490-498). IEEE.

[64] Zeng, Y., Gu, H., Wei, W., & Guo, Y. (2019). $ Deep-Full-Range $: a deep learning based network encrypted traffic classification and intrusion detection framework. IEEE Access, 7, 45182-45190.

[65] Park, J. T., Shin, C. Y., Baek, U. J., & Kim, M. S. (2024). Fast and accurate multi-task learning for encrypted network traffic classification. Applied Sciences, 14(7), 3073.

[66] Lin, C. Y., Chen, B., & Lan, W. (2022, January). An efficient approach for encrypted traffic classification using CNN and bidirectional GRU. In 2022 2nd International Conference on Consumer Electronics and Computer Engineering (ICCECE) (pp. 368-373). IEEE.

[67] Cheng, J., Wu, Y., Yuepeng, E., You, J., Li, T., Li, H., & Ge, J. (2021). MATEC: A lightweight neural network for online encrypted traffic classification. Computer Networks, 199, 108472.

[68] Hu, F., Zhang, S., Lin, X., Wu, L., Liao, N., & Song, Y. (2023). Network traffic classification model based on attention mechanism and spatiotemporal features. EURASIP Journal on Information Security, 2023(1), 6.

[69] Soleymanpour, S., Sadr, H., & Nazari Soleimandarabi, M. (2021). CSCNN: cost-sensitive convolutional neural network for encrypted traffic classification. Neural Processing Letters, 53(5), 3497-3523.

[70] Wang, Z., & Thing, V. L. (2023). Feature mining for encrypted malicious traffic detection with deep learning and other machine learning algorithms. Computers & Security, 128, 103143.

[71] Soleymanpour, S., Sadr, H., & Beheshti, H. (2020, April). An efficient deep learning method for encrypted traffic classification on the web. In 2020 6th International Conference on Web Research (ICWR) (pp. 209-216). IEEE.

[72] Lin, P., Ye, K., Hu, Y., Lin, Y., & Xu, C. Z. (2022). A novel multimodal deep learning framework for encrypted traffic classification. IEEE/ACM Transactions on Networking, 31(3), 1369-1384.

[73] Cui, S., Jiang, B., Cai, Z., Lu, Z., Liu, S., & Liu, J. (2019, August). A session-packets-based encrypted traffic classification using capsule neural networks. In 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS) (pp. 429-436). IEEE.

[74] Long, G., & Zhang, Z. (2023). Deep encrypted traffic detection: An anomaly detection framework for encryption traffic based on parallel automatic feature extraction. Computational Intelligence and Neuroscience, 2023(1), 3316642.

[75] Wang, X., Chen, S., & Su, J. (2020, July). App-net: A hybrid neural network for encrypted mobile traffic classification. In IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS) (pp. 424-429). IEEE.

[76] Zou, Z., Ge, J., Zheng, H., Wu, Y., Han, C., & Yao, Z. (2018, June). Encrypted traffic classification with a convolutional long short-term memory neural network. In 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th

International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS) (pp. 329-334). IEEE.

[77] Hameed, A., & Leivadeas, A. (2020, September). IoT traffic multi-classification using network and statistical features in a smart environment. In 2020 IEEE 25th international workshop on computer aided modeling and design of communication links and networks (CAMAD) (pp. 1-7). IEEE.

[78] Shi, G., Shen, X., Xiao, F., & He, Y. (2023). DANTD: A deep abnormal network traffic detection model for security of industrial internet of things using high-order features. IEEE Internet of Things Journal, 10(24), 21143-21153.

[79] Yun, X., Wang, Y., Zhang, Y., Zhao, C., & Zhao, Z. (2022). Encrypted TLS traffic classification on cloud platforms. IEEE/ACM Transactions On Networking, 31(1), 164-177.

[80] Guan, J., Cai, J., Bai, H., & You, I. (2021). Deep transfer learning-based network traffic classification for scarce dataset in 5G IoT systems. International Journal of Machine Learning and Cybernetics, 12(11), 3351-3365.

[81] Kalwar, J. H., & Bhatti, S. (2024). Deep learning approaches for network traffic classification in the internet of things (iot): A survey. arXiv preprint arXiv:2402.00920.

[82] Liu, Y., Wang, X., Qu, B., & Zhao, F. (2024). ATVITSC: A novel encrypted traffic classification method based on deep learning. IEEE Transactions on Information Forensics and Security.

[83] Wang, H., Yan, J., & Jia, N. (2024). A New Encrypted Traffic Identification Model Based on VAE-LSTM-DRN. Computers, Materials & Continua, 78(1).

Centurion University
Shaping Lives... Empowering Communities...

# SCOPES-2024

IEEE
KOLKATA SECTION
&
BHUBANESWAR SUBSECTION

19th - 21th December 2024

## Certificate of Presentation

This is to certify that **Prof./Dr./Mr./Ms. Koda Madhuri** from **Godavari Institute of Engineering and Technology** has presented a paper titled **"Implementation of cloud based Intrusion Detection System"** in the 2nd International Conference on Signal Processing, Communication, Power, and Embedded Systems (SCOPES), technically co-sponsored by the IEEE Kolkata Section and Bhubaneswar Sub-Section (IEEE- approved conference record number #64467), **organized by the Department of Electronics & Communication Engineering, School of Engineering & Technology, Centurion University of Technology and Management, Paralakhemundi, Odisha** during December 19-21, 2024.

| Prof. Prafulla Kumar Panda | Prof. Ashok Misra | Prof. Debendra Kumar Sahoo | Prof. Anita Patra |
|---|---|---|---|
| Programme Chair | Convener | Organizing Chair | General Chair |


Centurion
UNIVERSITY
Shaping Lives...
Empowering Communities...
SCOPES-2024
19th - 21th December 2024
IEEE
KOLKATA SECTION
&
BHUBANESWAR SUBSECTION
Certificate of Presentation
This is to certify that Prof./Dr./Ms./Mr. M.V.R.Chowdary from Godavari Institute of Engineering and Technology has presented a paper titled "Implementation of cloud based Intrusion Detection System" in the 2nd International Conference on Signal Processing, Communication, Power, and Embedded Systems (SCOPES), technically co-sponsored by the IEEE Kolkata Section and Bhubaneswar Sub-Section (IEEE- approved conference record number #64467), organized by the Department of Electronics & Communication Engineering, School of Engineering & Technology, Centurion University of Technology and Management, Paralakhemundi, Odisha during December 19-21, 2024.
Prof. Prafulla Kumar Panda
Programme Chair
Prof. Ashok Misra
Convener
Prof. Debendra Kumar Sahoo
Organizing Chair
Prof. Anita Patra
General Chair


Centurion
UNIVERSITY
Shaping Lives...
Empowering Communities...
SCOPES-2024
19th - 21th December 2024
IEEE
KOLKATA SECTION
BHUBANESWAR SUBSECTION
Certificate of Presentation
This is to certify that Prof./Dr./Ms./Mr. Nalla Ganesh from Godavari Institute of Engineering and Technology has presented a paper titled "Implementation of cloud based Intrusion Detection System" in the 2nd International Conference on Signal Processing, Communication, Power, and Embedded Systems (SCOPES), technically co-sponsored by the IEEE Kolkata Section and Bhubaneswar Sub-Section (IEEE- approved conference record number #64467), organized by the Department of Electronics & Communication Engineering, School of Engineering & Technology, Centurion University of Technology and Management, Paralakhemundi, Odisha during December 19-21, 2024.
Prof. Prafulla Kumar Panda
Programme Chair
Prof. Ashok Misra
Convener
Prof. Debendra Kumar Sahoo
Organizing Chair
Prof. Anita Patra
General Chair

Centurion University
Shaping Lives... Empowering Communities...

# SCOPES-2024

IEEE KOLKATA SECTION & BHUBANESWAR SUBSECTION

19th - 21th December 2024

## Certificate of Presentation

This is to certify that **Prof./Dr./Ms./Mr. K Vijaya saradhi babu** from **Godavari Institute of Engineering and Technology** has presented a paper titled **"Implementation of cloud based Intrusion Detection System"** in the 2nd International Conference on Signal Processing, Communication, Power, and Embedded Systems (SCOPES), technically co-sponsored by the IEEE Kolkata Section and Bhubaneswar Sub-Section (IEEE- approved conference record number #64467), **organized by the Department of Electronics & Communication Engineering, School of Engineering & Technology, Centurion University of Technology and Management, Paralakhemundi, Odisha** during December 19-21, 2024.

| Prof. Prafulla Kumar Panda | Prof. Ashok Misra | Prof. Debendra Kumar Sahoo | Prof. Anita Patra |
|---|---|---|---|
| Programme Chair | Convener | Organizing Chair | General Chair |