National College of Ireland

Deep Learning Based Multi Agent Intrusion Detection Approach for Cloud Security

MSc Research Project
MSc Cloud Computing

Aishwarya Ashok Pokharkar
Student ID: X23141808

School of Computing
National College of Ireland

Supervisor: Prof Vikas Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Aishwarya Ashok Pokharkar
Student ID:	23141808
Programme:	MSc Cloud Computing **Year:** 2024
Module:	MSc Research Project
Supervisor:	Prof Vikas Sahni
Submission Due Date:	12/08/2024
Project Title:	Deep Learning Based Multi Agent Intrusion detection Approach for Cloud Security

Word Count: 5719 **Page Count: 21**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Aishwarya Ashok Pokharkar

Date: 12/08/2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Deep Learning Based Multi Agent Intrusion Detection Approach for Cloud Security

Aishwarya Ashok Pokharkar
23141808

Abstract

Intrusion Detection Systems (IDS) are crucial in today's cloud environment due to the increased complexity and scale of network infrastructures, which are often distributed across multiple locations and accessed globally. Effective IDS are vital for identifying and mitigating network attacks and threats. Deep learning excels at processing complex, high-dimensional, and large-scale traffic data, offering superior detection capability. Current IDS solutions face challenges such as low detection accuracy, sub-optimal structures, and high false positive rates. To address these issues, this paper presents a sophisticated hierarchical intrusion detection model that compared multiple deep learning models and introduces a novel attention mechanism, aiming to enhance detection accuracy and reliability. The major objective achieved by this research is to reduce false positive rates using an attention-based mechanism and to increase the accuracy of detection. After evaluating the comparative analysis between different deep learning models, it has been found that Bi-LSTM with attention mechanism has the highest accuracy score over the test data.

Keywords- Cloud Security, Network Security, Attention mechanism, Deep Learning, Feature selection.

1 Introduction

Cloud computing has become a preferred solution for the rising data influx due to its ability to offer scalable, on-demand resources and better accessibility compared to local storage systems, which suffer from high costs and limited capacity. It provides a range of services, including software, storage, infrastructure, and hardware, over the internet, ensuring reliability, cost-effectiveness, and flexibility. Additionally, cloud computing enhances data security and privacy, making it a trusted choice for organizations and individuals alike. Cloud computing has an open architecture which makes it more vulnerable to privacy and security breaches which is a major cybersecurity challenge. These Breaches can put users data privacy and confidentiality in risk, possibly resulting in misuse of user data. For maintaining trust and protecting sensitive information addressing these issues has become a critical priority as cloud adoption is emerging (Butt *et al*., 2023).

1.1 Motivation

The motivation for this research is rooted in the urgent need to enhance IDS, as traditional IDS methods are insufficient in effectively protecting against sophisticated and evolving cyber threats. Utilizing the potential of deep learning can enhance the accuracy and reliability of intrusion detection which in turn ensures the security and integrity of cloud-based systems. The attention mechanism is a significant boost to deep learning models that allows the network to focus on the most appropriate parts of the input data. This capability improves the model's output quality and transparency, making it especially valuable for enhancing the detection capabilities of IDS.

In the context of IDS, integrating an attention mechanism allows the system to focus on critical features of network traffic data which shows potential threats. This targeted focus enhances detection accuracy by allowing the model to concentrate on significant anomalies and patterns that might otherwise go unnoticed. When used in Integration with deep learning models, attention mechanisms can significantly boost the effectiveness of IDS, leading to more precise identification of both known (signature-based) and unknown (anomaly-based) threats. This conjunction addresses the challenges of high false positive rates and low detection accuracy prevalent in traditional IDS methods, this approach ultimately delivers a stronger and more reliable security solution for cloud computing environments (Li *et al*., 2020).

1.2. Research Question

The research problem is related to optimizing network security, specifically through the development of an advanced intrusion detection system customized for cloud environments which reduces false positives without compromising detection accuracy. The general context within this field of study is developing, implementing, and improving IDS to detect and mitigate cyber threats dynamically and in real time. The above research problem motivated the following research question:

How effectively does an attention mechanism-based intrusion detection approach, utilizing deep learning models, achieve the highest accuracy in identifying breaches in a distributed cloud network environment**?**

1.3. Research Objective

The above research question addresses following research objective:

- Compare Different Deep Learning Models and Implement IDS which enhances the detection accuracy and reduce false positive rates in IDS, thereby improving the reliability and effectiveness of network security in cloud computing environments.

1.4. Document Structure

This research paper has split-up into several sections. The related work of previous research on cloud security threats, types of Intrusion detection system, Machine learning and Deep learning-based IDS provided in Section 2. The Methodology and framework are explained in the section 3. Detail description of tools, Algorithms have been mentioned in the section Design specification along with diagrams. Section 5 shows the detail Implementation of IDS and followed with section 6 discusses about the evaluation and results of the developed project.

2 Related Work

This section reviews various research papers and studies conducted by multiple researchers. It is further divided into the following sub-sections: Security Threats and Challenges in Cloud Computing, IDS in Cloud Framework, Machine Learning-based IDS, and Deep Learning-based IDS.

2.1 Security Threats and Challenges in the Cloud Computing

(Maheswari B. U., 2023) identified twelve security challenges in cloud computing, including vulnerabilities like dysfunctional login information and breached authentication tokens. Essential security requirements include authentication, integrity, confidentiality, availability, and audits. Challenges are categorized into communication, computation, and SLA levels. Mishra *et al*. (2021) surveyed security threats, detailing network-based, virtual machine-based, storage-based, and application-based attacks. The analysis covered issues like port scanning, botnets, cross-VM side-channel attacks, data scavenging, and malware injection.

(Ahmed I. &., 2020) conducted a thorough review of security issues in cloud computing, highlighting key factors such as operating systems, networks, load balancing, virtualization, and memory management. The research also tackled issues related to outsourcing, multi-latency, SLAs, heterogeneity, server downtime, backup, and data redundancy. A comparative analysis was conducted to highlight the impact of these threats on real-world applications. Tabrizchi and Kuchaki Rafsanjani (2020) explored similar security challenges, underscoring the significance of confidentiality, integrity, availability, authentication, and non-repudiation. The study reviewed several research papers, stressing the necessity for durable systems of authentication, authorization, identification, and access control. It proposed both technological and methodological strategies to strengthen cloud security and outlined potential areas for future research.

In a similar vein, Bhasker and Murali (2020) delved into security threats within cloud computing, focusing particularly on Distributed Denial of Service (DDoS) attacks, which compromise security, privacy, and system availability. They analysed various strategies and mechanisms to mitigate DDoS attacks, reviewed other potential threats, and highlighted preventive techniques discussed in other studies, stressing the importance of practical application. Narayanan and Naresh (2022) examined security challenges in cloud data storage, pointing out concerns like data privacy, integrity, recoverability, vulnerabilities, and backup. Their study also considered risks posed by malicious insiders, external attackers,

SLAs, and legal complications. They proposed solutions such as See Cloud, FADE, TimePre, and resident data security for data storage, along with SPICE, HASBE, a decentralized role-based framework, and Specs SLA for identity management and access control. In addition to that, Narayanan and Naresh (2022) assessed the effectiveness of encryption methods and multi-factor authentication in bolstering data security. Their research evaluated these strategies, recommending the most effective ones and emphasizing the critical need for continuous monitoring and regular security audits to sustain strong cloud security.

2.2 IDS in cloud computing

Given the rise in security threats and attacks, IDS have become difficulties for threat mitigation. Tariq et al. (2021) explored the challenges and opportunities of IDS within cloud computing, emphasizing various intrusion types across public, private, community, and hybrid clouds, with a particular focus on network-level attacks like IP spoofing, DNS poisoning, and DDoS attacks. They observed that such intrusions compromise confidentiality, integrity, and availability by evading security defences, and although IDS can address these issues, traditional systems struggle with scalability and self-adaptation. Similarly, Gangwani et al. (2023) provided a systematic analysis of IDS and prevention systems, detailing functional layers such as reaction, alarm processing, detection, and monitoring, alongside infrastructure layers including network-based, host-based, and application-based detection. They also discussed the structural configurations of IDS modules, which can be independent, distributed, hierarchical, or mobile, and offered a comprehensive examination of various IDS components.

Belal and Sundaram (2022) carried out an in-depth review of IIDS techniques in cloud computing, concentrating on various forms of attacks such as VM attacks, user-to-robot attacks, insider threats, DoS attacks, port scanning, and backdoor path exploits. Their research provided a thorough understanding of the security challenges in cloud environments and offered recommendations for future studies. In 2024, Qasem et al. introduced a security framework aimed at detecting network intrusions in virtual machines (VMs) within over-the-air infrastructures. The study addresed essential VM capabilities like managing large network packets, swiftly identifying threats, and minimizing transmission costs. Their research also contrasted signature-based with anomaly-based intrusion detection methods, conducting experiments to validate their innovative approach to recognizing emerging attack patterns.

2.3 Machine Learning-based IDS

In their 2022 paper, Abadi et al. introduced an innovative machine learning method for intrusion detection within cloud environments, with a particular focus on feature selection to boost both accuracy and overall performance. The study assessed four different algorithms—neural network, CART, ID3 decision tree, and random forest—using precision, recall, and accuracy as key performance indicators. Among these, the neural network algorithm demonstrated exceptional performance in the final evaluation. In contrast, Mughaid et al. (2023) developed a hybrid IDS utilizing machine learning techniques tailored for an over-the-air framework. Their research aimed to establish a strong detection system for both signature-based and anomaly-based attacks. The model combined K-means clustering with SVM classification algorithms, implemented using the UNSW-NB15 dataset. The findings revealed that K-means outperformed SVM, but the integration of both algorithms allowed the model to capitalize on their individual strengths.

Hasan *et al*. (2020) proposed a robust approach for IDS in cloud computing using machine learning. The paper identified eight types of attacks: DDoS, probing, U2R, R2L, zero-day attacks, distributed attacks, vulnerability reports, and covert channel attacks. It categorized IDS models into three types: detection method, monitoring method, and behaviour pattern. The study reviewed various research studies and compared the proposed model with conventional systems like anomaly detection, intrusion detection and prevention, fuzzy-based hybrid systems, and context-aware anomaly detection. The models were evaluated using metrics such as the PRF score.

In a paper by Singh and Ranga (2021), an ensemble learning approach for intrusion detection in cloud computing was studied. The researchers evaluated four ensemble learning algorithms: AdaBoost Classifier, Bagged Tree Classifier, Subspace Discriminant, and RUS Boosted Tree Classifier. Using the CICIDS 2017 dataset, which was pre-processed through normalization and one-hot encoding, the algorithms were implemented and assessed using an ensemble voting scheme to determine optimal performance. The evaluation metrics included PRF score and accuracy. Despite some drawbacks, the implementation showed promise. Similarly, Mishra *et al.* (2021) explored optimal algorithms for IDS in cloud systems, aiming to find efficient and fast detection methods. The study utilized machine learning algorithms and parallelization, specifically Naïve Bayes, Neural Network, and Decision Tree, with the KDD99 dataset containing various attack types such as DoS, Probe, U2R, and R2L. The evaluation revealed that the Decision Tree algorithm outperformed the others, though there was room for improvement in practical applications.

2.4 Deep Learning-based IDS

Sethi *et al*. (2020) proposed a deep reinforcement learning-based IDS for cloud computing systems to address shortcomings in conventional models, such as reduced accuracy, slower detection, and longer interpretation times. To overcome these challenges, they implemented deep reinforcement learning using the UNSW-NB15 dataset. The study employed five classifiers: Random Forest, Adaptive Boost, Gaussian Naïve Bayes, KNN, and Quadratic Discriminant Analysis. In another study, Fontaine *et al*. (2020) proposed a log-based intrusion detection system for cloud web applications using computational intelligence. This research focused on two algorithms, neural networks and decision trees, finding that neural networks outperformed other approaches upon evaluation.

Lansky *et al*. (2021) conducted a systematic review of deep learning (DL) methods for IDS in cloud and distributed computing, analysing and discussing several relevant papers. Bafna (2022) also examined cloud-based real-time network IDS using DL algorithms, implementing the NSL-KDD dataset and evaluating SVM, Random Forest, Logistic Regression, and Naïve Bayes algorithms, achieving an accuracy score of 83%. Meanwhile, Patil and Joshi (2023) proposed a DL method for a proactive multi-cloud cooperative IDS. The Deep Neural Network (DNN) model was built using a Denoising Autoencoder architecture. The implementation utilized several real-time datasets processed with GPU-enabled TensorFlow, resulting in a model accuracy of 95%.

2.5 Summary of Literature Review

The following section provides a summary of the literature review, highlighting key findings and insights from various studies relevant to the development of advanced IDS for cloud environments.

Section	Description	Key areas of Contribution
Security Threats and Challenges in Cloud Computing	These papers identify major security challenges in cloud computing, such as dysfunctional login information and breached authentication tokens.	Emphasizes the need for essential security measures including authentication, integrity, confidentiality, availability, and audits.
IDS in Cloud Computing	Discusses the challenges and opportunities for IDS in cloud environments, focusing on various types of intrusions and their impact.	Highlights the limitations of traditional IDS systems and the need for scalable, adaptive solutions.
Machine Learning Approach to IDS in Cloud	Proposes a novel machine learning approach for IDS, evaluating algorithms like neural networks and decision trees.	Focuses on feature selection and performance metrics such as precision and accuracy.
Deep Learning Methods for Intrusion Detection in Cloud	Reviews the use of deep learning methods for IDS, analyzing various techniques and their applicability to cloud security.	Discusses the potential of deep learning in improving IDS accuracy and adaptability in cloud environments.

Table 1. Summary of Literature review

3 Research Methodology

Cloud computing has revolutionized data management by offering scalable and on-demand resources. This advantage reduces the need for clients to worry about hardware and software maintenance and upgrades. However, vulnerabilities in cloud computing architectures pose significant security concerns. This section discusses Cloud-based IDS and their key components. A typical cloud-based module relies on the virtualization of resources, where a server is responsible for meeting client needs. Malicious attacks can disrupt server services and compromise user data. Therefore, a Network-based Intrusion Detection System (NIDS) is crucial for detecting and managing anomalies.

This section discusses Cloud-based IDS and their key components. A normal cloud-based module relies on the virtualization of resources, where a server should provide for client needs. Malicious attacks can interrupt server services and compromise user data. Therefore, a Network-based IDS is important for identifying and managing anomalies. This framework is developed to defend against multiple system penetrations in virtual architectures, including Denial of Service (DoS) attacks, Remote to Local (R2L) attacks, Probing, and User to Root (U2R) attacks. The framework leverages deep learning methodologies to provide precise detection, minimizing false positives under various network attacks by utilizing an attention mechanism. In a cloud network, routers and firewalls are the most important component. Firewalls prevent unauthorized access, and if an attacker bypasses the firewall, the NIDS identifies and eliminates the threat. The cloud-based IDS architecture processes large data packets, reducing packet loss and accurately detecting attacks. Upon detecting malicious activity, the IDS determines the attack type and takes appropriate action, generating

comprehensive reports for the cloud service provider to prevent future incidents (Ouiazzane, Addou and Barramou, 2021).

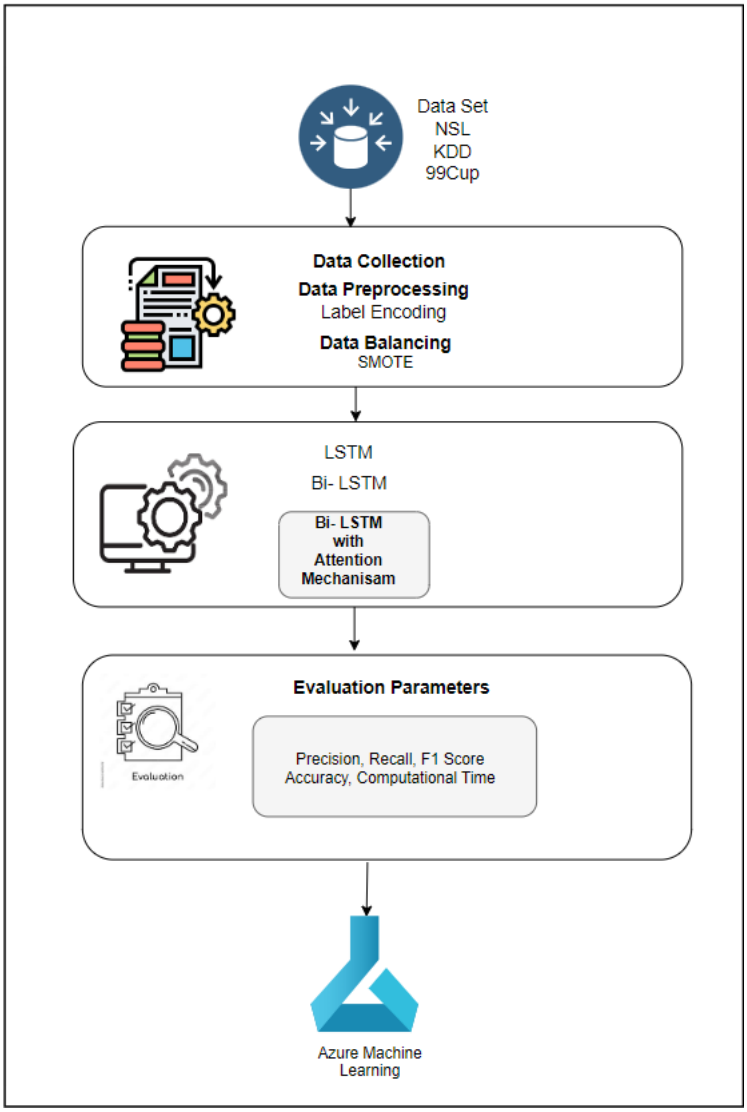

Fig 1: Architectural flow

3.1. Feature Identification

The feature identification process was crucial in determining the predictive power of various attributes related to network traffic. The dataset used was the KDD Cup 1999 [1]dataset, which is widely used for evaluating intrusion detection systems. The following steps were taken:

- **Data Collection**: The dataset comprises 41 features, with each record classified into either normal traffic or one of several types of attacks, including dos, probe, r2l, and u2r.

[1] https://www.kaggle.com/datasets/galaxyh/kdd-cup-1999-data

- **Feature Engineering**: The original features were augmented by creating an additional feature, AttackType, which maps specific attack labels to broader categories, facilitating a more streamlined analysis. The mapping was performed using a dictionary to associate each unique attack with a category.

3.2. Material and Techniques

The following tools and platforms were utilized to process and analyse the data:

Material Used: This project has used Azure AI machine learning Lab for model development and execution. Azure Notebooks were used for data collection and pre-processing, while Azure's compute resources efficiently handled intensive processing tasks, enabling a scalable approach to intrusion detection.

Techniques: The project developed a robust machine learning model for network intrusion detection using the KDD Cup 1999 dataset and advanced techniques. Comprehensive data pre-processing included data cleaning, label encoding, normalization, and data splitting into training, validation, and test sets. To address class imbalance, SMOTE was utilized, ensuring a balanced and effective dataset for detecting both common and rare network intrusions.

3.3. Machine Learning Algorithms

The project utilized two deep learning-based models to enhance the network IDS. A Bidirectional Long Short-Term Memory (LSTM) network was implemented for its ability to capture long-term dependencies in sequential data, improving the model's understanding of complex patterns. An attention mechanism was integrated to focus on the most relevant parts of the input data, enhancing the model's precision and reducing false positives. Additionally, Convolutional Neural Networks (CNNs) were employed to capture spatial relationships within the data. The models were built and trained using the Keras and TensorFlow frameworks, leveraging their powerful capabilities for deep learning applications. This combination of advanced deep learning techniques provided robust and precise detection of various network attacks.

4 Design Specification

The system design includes several key components, each essential for detecting and responding to threats, and the following explanation will detail how each part works and contributes to the method's overall effectiveness.

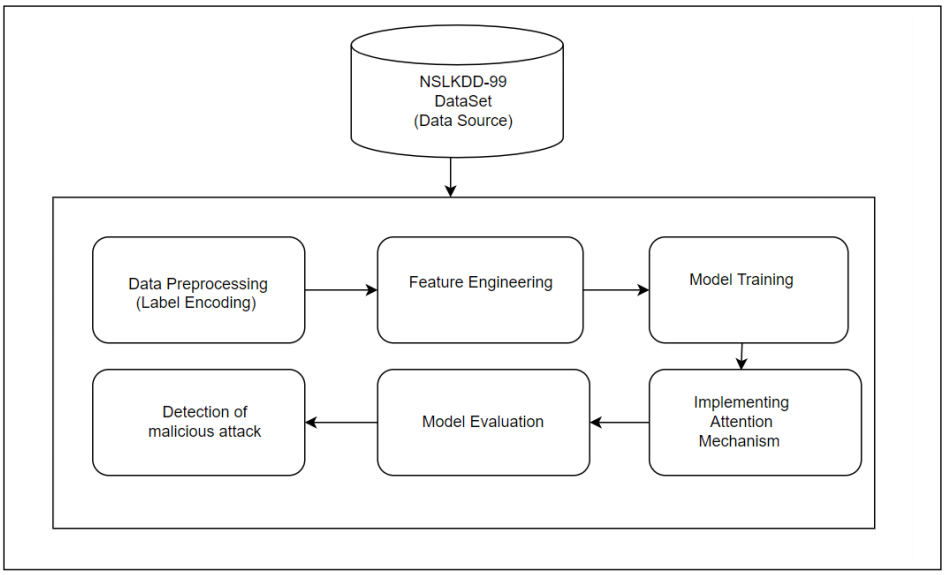

Fig 2: Intrusion detection model training using deep learning algorithms

4.1. Data Collection

The initial phase of this project involves gathering relevant data essential for training and evaluating the machine learning model. The dataset utilized is the KDD Cup 1999 dataset, a well-established benchmark in the domain of network intrusion detection systems. This dataset includes a variety of network traffic records, each labeled to indicate whether it represents normal traffic or a specific type of intrusion. The data collection process ensures the dataset's integrity and authenticity by sourcing it from a verified repository. The data format encompasses multiple features that describe different aspects of the network traffic, such as duration, protocol type, service, flag, among others, along with a label denoting the type of traffic, whether normal or indicative of a specific attack.

4.2. Data Pre-processing

During this project, to prepare the dataset for optimal model training a data processing approach is used. This process included several key steps, starting with Data Cleaning. Missing values were carefully addressed, either through imputation or exclusion, depending on their nature and impact. To prevent bias and overfitting, duplicates were identified and removed ensuring that the model learned from a diverse set of unique examples. Additionally, any inconsistencies within the data, such as conflicting entries, were corrected to maintain the dataset's integrity and consistency.

Following the data cleaning, to convert categorical features, such as protocol type and service, into numerical values label Encoding was used. For the machine learning algorithms to process and analyse these features effectively this conversion was essential. The next step involved Normalization, where features were standardized to a common scale, typically between 0 and 1. This standardization ensured that each feature had an equal influence on the model's training process, aiding in faster convergence and more reliable predictions. Finally,

Data Splitting was performed, dividing the dataset into training, validation, and test sets. This splitting was crucial for tuning hyperparameters, refining the model, and providing an unbiased evaluation of the model's performance.

4.3. Data Balancing

SMOTE (Synthetic Minority Over-sampling Technique) is used to address class imbalance by generating synthetic samples for underrepresented classes, particularly in the context of network intrusion detection where certain attack types may be rare. Implemented via the imblearn library, SMOTE selects minority class samples and creates synthetic data points along the line segments between these samples and their nearest neighbors. This process enhances the training dataset by balancing the representation of all classes, thereby improving the models ability to detect less frequent attacks and ensuring a more strong and unbiased performance.

4.4. Model Architecture

- **Bidirectional LSTM (Bi-LSTM) Layer**: The base model employs a Bidirectional LSTM layer to capture temporal dependencies from both past and future states in the input sequences. This layer is configured with 64 units and is set to return sequences, which is essential for the subsequent attention mechanism.

- **Attention Mechanism**: A custom attention layer is integrated into the model to focus on the most appropriate parts of the input sequence. This mechanism calculates attention scores for each time step, normalizes these scores using a softmax function, and then computes a context vector by taking a weighted sum of the input features. This context vector highlights the important features that contribute most efficiently to the models decision-making process.

  In the provided neural network, the ReLU (Rectified Linear Unit) function is applied in the dense layers after the Bidirectional LSTM and attention mechanism. The ReLU function introduces non-linearity which is mandatory for the network to learn complex patterns, by setting any negative input to zero and passing positive inputs unchanged. The formula for ReLU is:

  $$\mathrm{ReLU}(x) = \max(0, x)$$

  This function is computationally efficient and helps prevent issues like the vanishing gradient problem, making it suitable for deep learning applications. ReLU enables the model to efficiently capture and learn from diverse and intricate data features.

  The **Softmax** function is employed in the models final output layer to produce a probability distribution over multiple class. This function transforms the raw scores

(logits) into probabilities, which sum to one, providing a probabilistic interpretation of the model's outputs. The formula for Softmax is:

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

Here, zi represents the score for each class i. Softmax is crucial in multi-class classification tasks, as it allows the model to assign probabilities to each class, identifying the most likely class for each input. This combination of ReLU and Softmax functions facilitates effective learning and accurate classification in the neural network.

- **Dense and Output Layers**: Following the attention mechanism, the model includes dense layers for further feature processing. Specifically, there is a dense layer with 32 units and ReLU activation, followed by another dense layer with 18 units. The final output layer uses a softmax activation function to classify the input data into one of several classes. Below is the pseudo code for the Attention mechanism:

```
Class AttentionLayer(Layer):

    Method __init__():
        Initialize layer

    Method build(input_shape):
        W = random weight matrix of shape (input_shape[-1], 1)
        b = zero bias vector of shape (input_shape[1], 1)

    Method call(inputs):
        score = tanh(inputs * W + b)
        attention_weights = softmax(score)
        context_vector = sum(attention_weights * inputs)

        return context_vector

    Method compute_output_shape(input_shape):
        return (input_shape[0], input_shape[-1])
```

Fig 3: pseudo code for attention mechanism

A significant enhancement to the model was the integration of a custom attention mechanism. This mechanism allows the model to focus on the most relevant parts of the input sequences which in turn improves its ability to differentiate between normal and attack traffic. The

attention mechanism was implemented as a custom Keras layer, which calculated attention scores for each time step in the sequence, normalized these scores, and then computed a weighted sum of the input features to form a context vector. This context vector represented the most significant information extracted from the sequence, guiding the classification decision.

4.5. Training and Validation Process

The model training process involves several key components to ensure effective learning and performance optimization. The categorical cross entropy loss function is utilized, appropriate for handling multi-class classification tasks by measuring the difference between the predicted and actual class distributions. To optimize the learning process, the Adam optimizer is chosen for its efficiency and adaptive learning rate capabilities, facilitating faster convergence during training. Additionally, Dropout regularization with a rate of 0.5 is applied to mitigate overfitting by randomly setting a fraction of input units to zero during each training step, thereby preventing the model from becoming overly reliant on specific neurons.
The model is trained on the dataset (X_train1, y_train) using a batch size of 64 over 10 epochs, with validation conducted on a separate set (X_test1, y_test) to track performance and adjust training strategies accordingly. Although not explicitly stated, employing early stopping is recommended as a potential measure to halt the training process when the validation performance ceases to improve, thereby preventing overfitting and ensuring the model's robustness and generalizability.

4.6. Model Evaluation

The model's performance is comprehensively evaluated using a range of metrics, including accuracy, precision, recall, F1-score, and the confusion matrix. These metrics collectively provide a detailed understanding of the model's classification capabilities, particularly its effectiveness in distinguishing between normal and attack categories. To further support this evaluation, the training and validation accuracy and loss are plotted, offering a visual representation of the learning progress and helping to identify potential issues such as overfitting or underfitting. Additionally, the confusion matrix and classification report offer in-depth insights into the model's strengths and weaknesses across different classes, highlighting areas for potential improvement.

5 Implementation

5.1. Data Collection and Transformation

The initial phase involves gathering and transforming data necessary for training and evaluating the machine learning models. The KDD Cup 1999 dataset is loaded from a

verified repository into Azure Jupyter Notebook on Azure AI ML Studio. Data cleaning processes are applied to handle missing values and remove duplicates, ensuring data integrity.

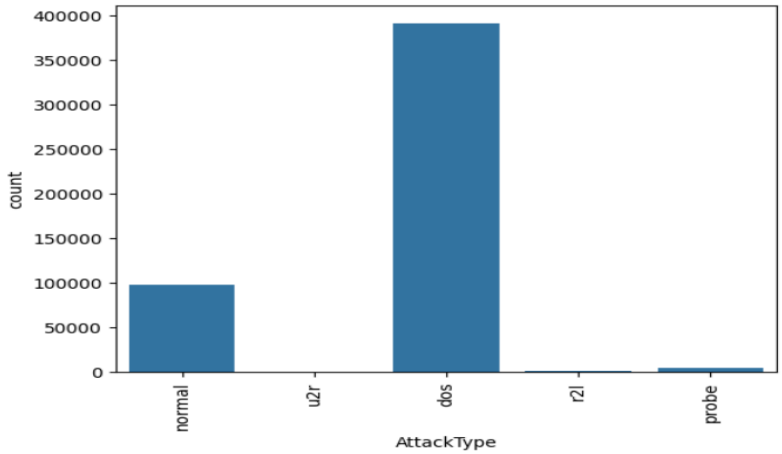

Fig 4: Distribution of attack types before

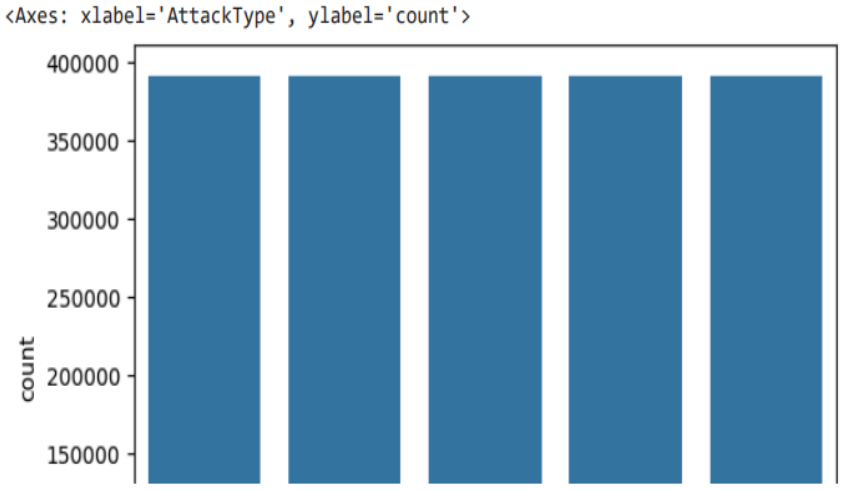

Fig 5: Distribution of attack types after SMOTE sampling

Categorical features such as protocol type and service are encoded into numerical values using label encoding, and the features are normalized to a common scale. The dataset is then split into training, validation, and test sets. Additionally, SMOTE is used to address class imbalance by generating synthetic samples for underrepresented classes, enhancing the model's ability to detect rare attacks.

5.2. Training Deep Learning Models

Deep learning models are developed and trained using Azure Jupyter Notebook and compute instance of Standard_DS12_v2 (4 cores, 28 GB RAM, 56 GB disk) in Azure AI ML Studio. The primary model is a Bidirectional Long Short-Term Memory (LSTM) network, chosen for its ability to capture long-term dependencies and temporal patterns in sequential data. The

model is defined using ‘tf.keras.layers.Bidirectional’ and LSTM layers, with ‘return_sequences’ set to true. The training process involves compiling the model with the Adam optimizer and categorical cross-entropy loss function, followed by training on the pre-processed dataset with a specified number of epochs and batch size. The model's performance is monitored using accuracy and loss metrics. However, specifically training time for models have not been considered rather overall computational time have taken into account.

5.3. Integration of Attention Mechanism with Deep Learning Models

To enhance the model's performance, an attention mechanism is incorporated with the Bidirectional LSTM. This mechanism is implemented as a custom subclass of ‘tf.keras.layers.Layer’, overriding the call method to compute attention scores that dynamically weigh the importance of different time steps in the input sequence. This allows the model to focus on the most relevant features for intrusion detection. The attention layer is added to the model architecture, following the LSTM layer. Additionally, Convolutional Neural Networks (CNNs) are used to capture spatial relationships within the data, further enhancing the detection capabilities.

5.4. Analysis of the Result

The model's performance is assessed using validation and test datasets within Azure AI ML Studio. To gauge the model's effectiveness, key metrics such as accuracy, precision, recall, and F1-score are computed. These results are then visualized through plots illustrating training and validation accuracy, loss curves, confusion matrices, and detailed classification reports. By incorporating an attention mechanism, the model focuses on the most critical aspects of the input data, thereby minimizing false positives and enhancing overall detection accuracy. The evaluation confirms the systems performance and precision, showcasing its ability to accurately detect and respond to various network threats.

6 Evaluation

6.1 Experiment 1: Evaluation based on computational time

The main objective of this experiment is to analyse the Processing time for every Model. The evaluation of the model's performance in terms of execution time provides insights into its efficiency and computational requirements. The screenshots display the prediction phase for three different models, each processing 12,234 instances.

```
1   pred = model.predict(X_test1)
2   pred = np.argmax(pred,axis=1)
```

✓

```
12234/12234 [==============================] - 32s 3ms/ste
```

```
pred = model.predict(X_test1)
pred = np.argmax(pred,axis=1)
```

```
12234/12234 [==============================] - 37s 3ms/ste
```

```
pred = model.predict(X_test1)
pred = np.argmax(pred,axis=1)
```

```
12234/12234 [==============================] - 64s 5ms/ste
```

Fig 6: Processing Time for LSTM, Bi- LSTM and Attention Mechanism respectively

The evaluation of model execution time shows that the LSTM model is the most time-efficient, completing predictions in 32 seconds with an average of 3 milliseconds per step. The BiLSTM model takes 37 seconds with a similar average step time. The attention mechanism model, while achieving higher accuracy, has the highest computational cost, taking 64 seconds with 5 milliseconds per step. This highlights the trade-offs between accuracy and computational efficiency, crucial for real-time network intrusion detection deployment.

6.2 Experiment 2: Evaluation Based on Accuracy

Accuracy serves as the key metric for assessing the detailed and overall performance of a model. Accuracy is calculated as the ratio of correctly predicted instances to the total instances in the dataset.

Accuracy = 100 X (Number of correct Predictions \ Total Predictions)

For LSTM base model the training accuracy (blue line) steadily increases from about 50% to over 80%, indicating that it has effectively learn from the training data. The validation accuracy (red line) also rises, stabilizing around 85%, which shows good generalization to unseen data. The convergence of these accuracies interprets that the model is accurately identifying network intrusions without overfitting.

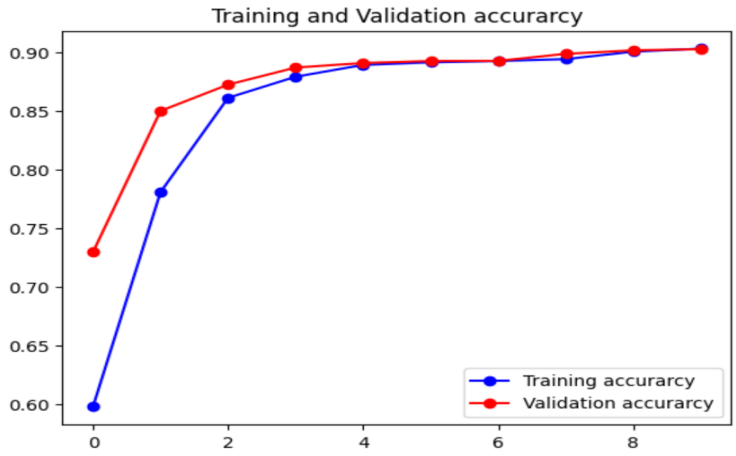

Fig 7: Validation Accuracy for LSTM Model

The BiLSTM model achieves strong performance, with training accuracy increasing from 60% to over 90% and validation accuracy rising from 75% to around 90% over 9 epochs. The close alignment between training and validation accuracy indicates good generalization to unseen data, suggesting minimal overfitting. This demonstrates the BiLSTM model's effectiveness in learning and accurately classifying network traffic for intrusion detection.

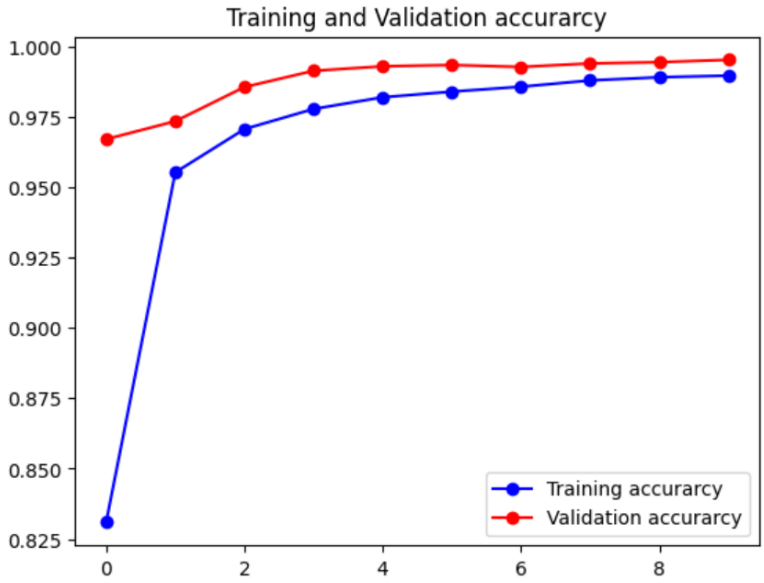

Fig 8: Validation Accuracy for Bi-LSTM Model

For the model with an attention mechanism, the plot demonstrated its performance by showing both training and validation accuracy over 9 epochs. The training accuracy (blue line) rapidly increased from about 82.5% to nearly 99%, indicating that the model effectively learned from the training data. The validation accuracy (red line) followed closely, stabilizing around 98%, which suggested excellent generalization to unseen data. The close alignment between the training and validation accuracy curves highlighted the models minimal overfitting which confirms that the attention mechanism effectively enhanced the models performance for network intrusion detection.

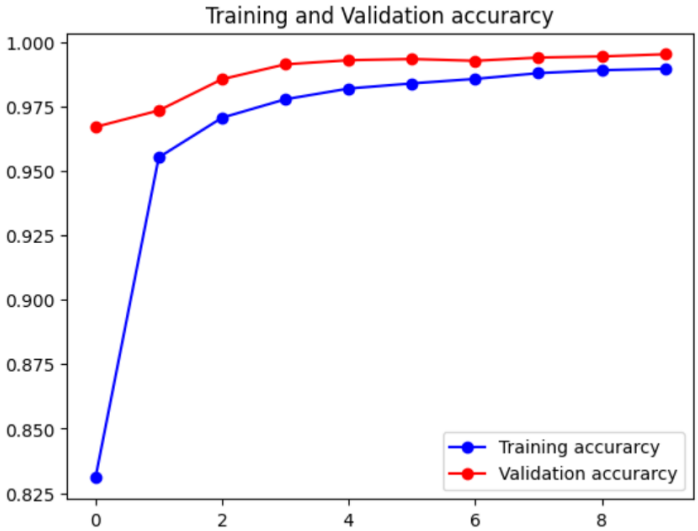

Fig 9: Validation Accuracy for Attention based Model

6.3 Experiment 3: Evaluation Based on Precision, Recall and F1 Score

The LSTM model showed good performance with precision, recall, and F1-score for individual classes ranging from 0.74 to 0.99. The overall accuracy was 0.86. The macro and weighted averages were both 0.87, reflecting a decent but less consistent performance across different classes. The model had lower recall and F1-scores for classes 3 and 4, indicating potential challenges in detecting certain types of network intrusions accurately.

Classification Report :

	precision	recall	f1-score	support
0	0.92	0.98	0.95	78292
1	0.93	0.88	0.90	78291
2	0.99	0.91	0.95	78292
3	0.74	0.78	0.76	78291
4	0.77	0.78	0.77	78292
accuracy			0.86	391458
macro avg	0.87	0.86	0.87	391458
weighted avg	0.87	0.86	0.87	391458

Fig 10: Classification Report for LSTM Model

The BiLSTM model showed strong performance with precision, recall, and F1-score for individual classes ranging from 0.80 to 0.99. The overall accuracy was 0.90. The macro and weighted averages were both 0.91, suggesting that the model performed well across different classes. There was a slight drop in precision and recall for certain classes which indicates that the BiLSTM model had some difficulty in consistently identifying specific types of network intrusions.

Classification Report :

	precision	recall	f1-score	support
0	0.99	0.99	0.99	78292
1	0.91	0.89	0.90	78291
2	0.98	0.94	0.96	78292
3	0.80	0.91	0.85	78291
4	0.85	0.78	0.81	78292
accuracy			0.90	391458
macro avg	0.91	0.90	0.90	391458
weighted avg	0.91	0.90	0.90	391458

Fig 11: Classification Report for Bi-LSTM Model

The classification report for the model with an attention mechanism indicated comparatively better performance across all classes. The precision, recall, and F1-score for each class were almost perfect, with values close to 1.00. The overall accuracy was 1.00, indicating that the model correctly classified all instances in the test set. The macro and weighted averages also

reflected this high performance, highlighting the model's robustness and effectiveness in accurately identifying all types of network traffic and intrusions.

Classification Report :

	precision	recall	f1-score	support
0	1.00	1.00	1.00	78292
1	1.00	0.99	1.00	78291
2	1.00	1.00	1.00	78292
3	1.00	0.98	0.99	78291
4	0.98	1.00	0.99	78292
accuracy			1.00	391458
macro avg	1.00	1.00	1.00	391458
weighted avg	1.00	1.00	1.00	391458

Fig 12: Classification Report for Attention Mechanism Model

6.4 Discussion

The experimental analysis of the developed system was extremely promising, as Experiments conducted provided insights on the efficiency and performance of different models for network intrusion detection, including LSTM, BiLSTM, and models incorporating attention mechanisms. The primary objective was to evaluate the models based on processing time, accuracy, and other critical metrics such as precision, recall, and F1-score. LSTM model demonstrated high time efficiency, that makes it suitable for real-time applications even if it exhibits a lower accuracy and consistency with different classes while BiLSTM model provides a balance of execution time and accuracy with a focus on strong learning and classification capability, but still, it encounters some difficulty in consistently identify some types of intrusions.

Evaluating the precision, recall, and F1-score showed good results from the LSTM model; however, inconsistencies were observed across different classes, especially in class 3 and class 4. BiLSTM demonstrated strong and consistent performance but faced challenges in maintaining higher precision and recall values for certain classes. Expected results were achieved from the attention mechanism model for precision, recall, and F1-score, which were close to 1.0 across all classes, reflecting its robustness and reliability in identifying different types of network intrusions. However, this model requires significant computational resources, highlighting the trade-offs between accuracy and computational efficiency. Reviewing these findings with existing literature on this topic, it was clear that models with attention mechanisms provide significant advantages in terms of accuracy and feature focus, aligning with previous research that emphasizes its effectiveness in complicated classification tasks.

7 Conclusion and Future Work

The research aimed at developing a deep learning-based multi-agent IDS for cloud security that focuses on enhancing detection accuracy and thus reducing false positives. The primary

objective included comparing deep learning models and the integration of an attention mechanism that improves IDS capability. This study answers the research question of whether an attention-based LSTM model achieves high accuracy in detecting network intrusions. The findings of this study reveal that while traditional LSTM models provide good accuracy and efficiency, the attention mechanism, when integrated, provides a significant improvement in the detection of network intrusions. The limitation of this technique is that it requires additional computational resources. This research highlights the potential of deep learning techniques in improving IDS efficiency in cloud environments. This approach addresses the need for good security measures in cloud computing, where traditional approaches fall short.

This shows the improvement in network intrusion detection accuracy by deep learning techniques but also faces limitations due to increased computational resources from integrating the attention mechanism that significantly improves network intrusion detection accuracy and thus reduces false positives, highlighting the constraints related to computational resources and time availability. Given more time, additional improvement techniques for the attention mechanism could have been studied that would reduce computational intensity and thus make the model more efficient.

References

Abadi, E. A. J., Sahu, H., Javadpour, S. M., & Goharimanesh, M. (2022). Interpretable machine learning for developing high-performance organic solar cells. *Materials Today Energy*, 25, 100969. doi: 10.1016/j.mtener.2022.100969

Ahmed, I., & Alam, F. I. (2020). Integrity verification for an optimized cloud architecture. *Telkomnika*, *18*(1), 166-173.

Bafna, E. (2022). Real Time Cloud Based Intrusion Detection, in Rathore, P. S., Dutt, V., Agrawal, R., Sasubilli, S. M. and Swarna, S. R. (eds.) *Deep learning approaches to cloud security*, 207-224. doi: 10.1002/9781119760542.ch13

Belal, M. M., & Sundaram, D. M. (2022). Comprehensive review on intelligent security defences in cloud: Taxonomy, security issues, ML/DL techniques, challenges and future trends. *Journal of King Saud University-Computer and Information Sciences*, 34(10), 9102-9131. doi: 10.1016/j.jksuci.2022.08.035

Bhasker, B., & Murali, S. (2020). A survey on security issues in sensor cloud environment for agriculture irrigation management system. *Journal of Critical Reviews*, 7(4), 796-805. doi: 10.31838/jcr.07.04.148

Butt, U. A., Amin, R., Mehmood, M., Aldabbas, H., Alharbi, M. T., & Albaqami, N. (2023). Cloud security threats and solutions: A survey. *Wireless Personal Communications*, 128(1), 387-413. doi: 10.1007/s11277-022-09960-z

Fontaine, J., Kappler, C., Shahid, A., & Poorter, E. D. (2020). Log-based intrusion detection for cloud web applications using machine learning. in *Advances on P2P, Parallel, Grid, Cloud and Internet Computing: Proceedings of the 14th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC-2019)* 14. Springer International Publishing, pp. 197-210. doi: 10.1007/978-3-030-33509-0_18

Gangwani, D., Sanghvi, H. A., Parmar, V., Patel, R. H., & Pandya, A. S. (2023). A comprehensive review on cloud security using machine learning techniques, in Bhardwaj, T., Upadhyay, H., Sharma, T. K. and Fernandes, S. L. (eds.) *Artificial intelligence in cyber security: Theories and applications*, 1-24. doi: 10.1007/978-3-031-28581-3_1

Hasan, M., Balamurugan, E., Almamun, M., & Sangeetha, K. (2020). An intelligent machine learning and self adaptive resource allocation framework for cloud computing environment. *EAI Endorsed Transactions on Cloud Systems*, 6(18), e2. doi: 10.4108/eai.13-7-2018.165501

Lansky, J., Mohammadi, M., Mohammed, A. H., Karim, S. H. T., Rashidi, S., Rahmani, A. M., & Hosseinzadeh, M. (2021). Scientific workflow scheduling in mobile edge computing based on a discrete butterfly optimization algorithm. Available at: https://www.researchsquare.com/article/rs-208986/v1 [Accessed 8 August 2024].

Li, R., Su, J., Duan, C., & Zheng, S. (2020). Linear attention mechanism: An efficient attention for semantic segmentation. *arXiv preprint arXiv:2007.14902*. doi: 10.48550/arXiv.2007.14902

Maheswari, B. U., Imambi, S. S., Hasan, D., Meenakshi, S., Pratheep, V. G., & Boopathi, S. (2023). Internet of things and machine learning-integrated smart robotics. In *Global Perspectives on Robotics and Autonomous Systems: Development and Applications* (pp. 240-258). IGI Global.

Mishra, P., Aggarwal, P., Vidyarthi, A., Singh, P., Khan, B., Alhelou, H. H., & Siano, P. (2021). VMShield: Memory introspection-based malware detection to secure cloud-based services against stealthy attacks. *IEEE Transactions on Industrial Informatics*, 17(10), 6754-6764. doi: 10.1109/TII.2020.3048791

Mughaid, A., Alqahtani, A., AlZu'bi, S., Obaidat, I., Alqura'n, R., AlJamal, M., & AL-Marayah, R. (2023, May). Utilizing machine learning algorithms for effectively detection IoT DDoS attacks. In *International Conference on Advances in Computing Research*. Cham: Springer Nature Switzerland, pp. 617-629. doi: 10.1007/978-3-031-33743-7_49

Narayanan, K. L., & Naresh, R. (2022). A effective encryption and different integrity schemes to improve the performance of cloud services, in *2022 International Conference for Advancement in Technology (ICONAT)*. Goa, India, 21-222 January 2022, 1-5. doi: 10.1109/ICONAT53423.2022.9725904

Ouiazzane, S., Addou, M., & Barramou, F. (2021). A multiagent and machine learning based hybrid NIDS for known and unknown cyber-attacks. *International Journal of Advanced Computer Science and Applications*, 12(8), 375-382.

Patil, N. B. and Joshi, S. (2023) Extensive analysis of intrusion detection system using deep learning', in *Intelligent Systems and Applications*, pp.191-205. doi: 10.1007/978-981-19-6581-4_16

Qasem, M. A., Thabit, F., Can, O., Naji, E., Alkhzaimi, H. A., Patil, P. R., & Thorat, S. B. (2024). Cryptography algorithms for improving the security of cloud-based internet of things. *Security and Privacy*, 7(4), e378. doi: 10.1002/spy2.378

Sethi, K., Sai Rupesh, E., Kumar, R., Bera, P., & Venu Madhav, Y. (2020). A context-aware robust intrusion detection system: a reinforcement learning-based approach. *International Journal of Information Security*, 19, 657-678. doi: 10.1007/s10207-019-00482-7

Singh, P., & Ranga, V. (2021). Attack and intrusion detection in cloud computing using an ensemble learning approach. *International Journal of Information Technology*, 13(2), 565-571. doi: 10.1007/s41870-020-00583-w

Tabrizchi, H., & Kuchaki Rafsanjani, M. (2020). A survey on security challenges in cloud computing: issues, threats, and solutions. *Journal of Supercomputing*, 76(12), 9493-9532. doi: 10.1007/s11227-020-03213-1

Tariq, M. U., Babar, M., Jan, M. A., Khattak, A. S., Alshehri, M. D., & Yahya, A. (2021). Security requirement management for cloud-assisted and internet of things enabled smart city. *Computers, Materials and Continua*, 67(1), 625-639