ECTI Transactions on Computer and Information Technology
Journal homepage: https://ph01.tci-thaijo.org/index.php/ecticit/
Published by the ECTI Association, Thailand, ISSN: 2286-9131

Optimizing Cloud-Integrated Computer Networks: Strategies for Enhanced Performance and Security

Teeb Hussein Hadi[1]

ABSTRACT

This study proposes a unified, Python-based simulation framework to enhance the performance and security of cloud-integrated computer networks. The framework concurrently addresses two critical aspects, load balancing and intrusion detection, within a single reproducible environment. Using the CICIDS2017 dataset, a Random Forest classifier was used to detect a wide range of network attacks with high accuracy. To simulate realistic traffic behavior, synthetic data were generated for performance metrics such as latency, throughput, and packet loss. Load balancing is evaluated using round-robin and random assignment strategies across virtual servers, illustrating the trade-offs between uniformity and randomness in the request distribution. The experimental results demonstrated a classification accuracy of 99.79%, with precision and recall metrics supporting the robustness of the selected model. Feature importance analysis highlights the key indicators of anomalous traffic, and confusion matrices and precision-recall curves validate the detection performance. Additionally, the simulated network KPIs provide a scalable approximation of the Quality of Service under varying load scenarios. The proposed research is the only one that suggests an integrated and data-driven approach to fill the gap documented in previous studies, where network security and resource optimization are frequently studied independently. The proposed framework provides a convenient basis for additional scholarly and industrial investigations in the field of secure cloud networking.

Article information:

Keywords: Cloud Networks, Intrusion Detection, Network Security, Performance Optimization, Random Forest

Article history:
Received: June 22, 2025
Revised: July 31, 2025
Accepted: August 21, 2025
Published: September 13, 2025 (Online)

DOI: 10.37936/ecti-cit.2025194.262550

1. INTRODUCTION

Cloud-integrated computer networks serve as the digital backbone of today's technology ecosystems, providing a scalable, adaptable, and cost-effective solution that enables organizations to leverage remotely hosted computing resources from geographically distributed data centers [1]. With the increased use of cloud as their operational infrastructure, organizations' reliance on virtualized computing has increased performance constraints and network security concerns for cloud infrastructure [2]. Cloud environments are affected by high-bandwidth-consuming workloads and heterogeneous applications combined with growing user demands. Network performance is usually affected by increases in latency, service unavailability, and decreases in throughput. In addition, the degree and intensity of cyber threats, such as DDoS attacks and new exploit variants, as well as the growing threat of information theft, have increased the demand for dynamic and defensive intrusion detection programs [3]. These issues typically require an integrative security solution that aims not only to improve the responsiveness of a system but also to harden it against malicious behavior.

To meet this challenge, researchers are developing simulation-based approaches that replicate real-world cloud environments, allowing them to define environmental control over several important system variables. A Python simulation framework has been reported in the literature that has gained much attention for its modular, adaptable, and inexpensive ability to simulate cloud workloads and model security threats [4]. This paper presents a simulation framework designed to simulate and benchmark the performance of load-balancing algorithms, such as Round

[1]The author is with the IT Department, Technical College of Management, Middle Technical University, Iraq, Email: eng.teebhussien@mtu.edu.iq

Robin and Random Assignment, while implementing supervised machine learning models, especially Random Forest, for multi-class intrusion detection. The simulation environment uses a combination of real datasets (CICIDS2017) and synthetic performance metrics to provide an all-encompassing testbed that supports researchers in addressing the trade-offs and interdependencies between cloud system performance and cloud security [5].

1.1 Background

Cloud computing has transformed the delivery of IT services and offers solutions such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) that allow organizations to access on-demand computing and storage resources [6]. Virtualization decouples applications from physical hardware, and software-defined networking (SDN) allows for programmable and responsive control over data flows [7]. This means that organizations can now deploy mission-critical applications (e.g., artificial intelligence (AI), real-time analytics, IoT systems, and e-commerce systems) over highly scalable and resilient resources [8]. Although the same flexibility and complexity that are a part of cloud computing help with the deployment of various applications, they also add the potential for problems in terms of bottlenecks in the system or a greater potential for cyberattacks.

1.2 Motivation

Although cloud systems have many advantages and offer a high degree of flexibility, they also have limitations that result in significant bottlenecks. Network issues such as high jitter, unequal throughput, and server throttling are still observed far too often during peak use times or in a distributed nature [9]. These performance glitches harm the user experience, particularly in latency-sensitive applications such as self-driving cars, telemedicine, and high-frequency trade executions [10]. Beyond performance issues, cloud services and data are also opportunities for attacks as they continue to evolve. Multitenant architectures augment the vulnerability between tenants and shared resources. Hybrid/multicloud deployments worsen poor security governance. Some existing performance tuning methods or rule-based security tools do not always have the right level of flexibility to adapt to the changing threat landscape and workload conditions [11]. A solid planning environment is required to simulate all potential scenarios and deliver intelligent automation with load management and threat detection.

1.3 Problem Statement

Although cloud technology has matured, achieving optimal performance and airtight security in cloud-integrated networks remains challenging [12]. Load-balancing algorithms, such as Round Robin or Least Connections, although fast and straightforward, do not dynamically adapt to real-time workload fluctuations or resource availability [13]. On the security front, conventional intrusion detection systems often rely on predefined rules and are plagued by high false-positive rates, making them unreliable for detecting novel or stealthy attacks [14]. Furthermore, performance and security are often treated as disjoint objectives, with little effort made to co-optimize them within a single unified framework. There is a real need for a solution that integrates the ability to simulate both load balancing decisions and security policies using an adaptive, data-driven, and extensible approach.

1.4 Scope of the Study

This study was designed to create and deploy a Python simulation framework for performance and security analysis of networks that integrate cloud architecture with flow monitoring capabilities. Publicly available datasets (e.g., CICIDS2017) were utilized, in addition to synthetic data generation methods, to model critical network performance indicators that can impact latency, packet loss, and throughput. The framework is not intended to be used for live testing in the cloud or at the production level; instead, it targets research and academic contexts, such as algorithmic benchmarking, machine learning model validation, and visualizing the interaction of traffic routing and anomaly detection. It was designed to provide a controlled environment for examining cross-cutting network behaviors by evaluating different load-balancing strategies with an added machine learning-based intrusion detection module.

1.5 Significance of the Study

ormance and security to be examined in tandem, rather than in isolation, as is often the case. The dual-focus framework allows users to see how changes to load-distribution strategies affect the system efficiency (in terms of intrusion detection system performance/impact) and vice versa, providing a more accurate and reliable representation of what it means to be part of cloud computing. Reproducible workflows and scalable architectures will also make it accessible to as enormous scope of usage as possible by varying study participants and with varying research interests in different institutions. Ultimately, varying levels of visual analytics, integrated classification measures, simulated performance data included, community offering their evidence to contextual decision-making about cloud infrastructure design and policy, and of the autonomous 'self-healing' philosophy of cloud systems that can monitor, adapt and optimize based on aggregating incoming data in real time

- all elements represent research that contributes toward the development of a collective, evidence-based knowledge associated with cloud computing.

2. LITERATURE REVIEW

This section reviews recent developments in cloud-integrated IoT and cyber–physical systems, including decentralized learning and blockchain-SDN architectures for increased security, energy-aware offloading for optimal performance, and multi-timescale orchestration frameworks for improved scalability and resource utilization in dynamic networks.

2.1 Security Enhancement in Cloud-Integrated IoT Networks

Gonçalves *et al.* (2024) suggested a decentralized machine learning architecture to improve the security, privacy, and performance of cloud-based IoT devices [15]. They discussed the limitations of centralized designs for managing IoT data, reducing reliance on centralized servers, evading security vulnerabilities, and facilitating efficient resource utilization. The architecture also protects sensitive data at the edge and can thus be used in privacy-sensitive applications.

Haritha *et al.* (2024) designed a Blockchain-Software Defined Networking paradigm for enhancing cloud-based IoT networks' data protection [16]. They combined the transparency and immutability characteristics of blockchain with SDN's flexibility of SDN to facilitate real-time traffic management and secure data handling. The model is robust in preventing attacks such as data tampering, unauthorized access to data, and DDoS attacks, and enables fine-grained policy enforcement and decentralized trust management.

Kumar *et al.* (2024) highlighted the need for improved network security for cloud-connected IoT devices by enumerating common vulnerabilities and proposing architectural improvements [17]. They underlined the need for multilayer security and lightweight encryption, and the inclusion of anomaly detection systems and real-time monitoring to improve threat mitigation functionality. Furthermore, several machine learning-based intrusion detection systems have been developed using the CICIDS2017 dataset. Ustebay *et al.* [21] used the Random Forest and trained the recursive features and removed the feature elimination and deep learning classification to increase the rate of detection. Stiawan *et al.* [22] exploited the use of feature importance via information gain, seeking to assess the performance of a model on CICIDS2017. Despite their strong classification performance, these studies neither investigate the behavior of the system under different network loads nor consider unified frameworks covering both intrusion detection and performance simulation.

2.2 Performance Optimization in Cyber–Physical and Cloud-Based Systems

Ma *et al.* (2024) investigated the performance, reliability, and energy usage of cloud-integrated cyber-physical systems (CPS) with shared servers and parallel services [18]. They found latency, computation load balance, and system effectiveness issues. They improved the processing speed and energy efficiency but found potential reliability trade-offs under heavy workloads.

Materwala *et al.* (2022) designed an Energy-SLA-aware genetic algorithm for vehicular network computation offloading in edge-cloud systems [19]. The algorithm promotes task assignment based on energy usage and service-level agreement requirements, reducing power consumption without compromising the quality of service. Therefore, it is suitable for real-time vehicular applications requiring high responsiveness and optimum utilization of resources.

2.3 Orchestration and Resource Management in Next-Generation Cloud Networks

Pagliuca *et al.* (2024) proposed a double timescale orchestration framework for elastic management in next-generation cloud-integrated networks [20]. It applies short- and long-term orchestration mechanisms to manage network functions dynamically, thereby enhancing scalability and flexibility. The research demonstrated that it improves resource allocation efficiency and user experience in dynamic cloud network environments.

2.4 Research Gap

Despite extensive advancements in cloud security and performance management, existing studies tend to focus on either security (e.g., intrusion detection) or performance (e.g., load balancing) in isolation. A unified, reproducible framework that simultaneously addresses both aspects within a simulated environment using real and synthetic data is lacking. This study fills this gap by integrating machine learning-based intrusion detection with performance optimization strategies, providing a holistic approach that has been largely overlooked in previous studies. In addition, versions of CICIDS2017 have been well utilized in previously conducted machine learning studies [21][22], yet they do not combine performance testing or provide simulation of network-level behavior faced when using load balancing, which this research seeks to cover.

3. RESEARCH OBJECTIVES AND QUESTIONS

This study aims to design a Python-supported simulation environment that solves performance enhancement and security issues in cloud-combined networks. The individual objectives are as follows:

- To simulate and evaluate network performance metrics, including latency, throughput, jitter, and packet loss, under various load conditions using a Python-based framework.
- To assess and compare the efficiency of traditional and advanced load-balancing algorithms for optimizing traffic distribution across cloud-integrated network servers.
- To simulate and analyze network security by integrating intrusion detection datasets and machine learning models for the classification and mitigation of cyber threats.
- To develop a modular and extensible simulation environment capable of generating synthetic traffic data and visualizing performance and security results using advanced plotting tools.

Based on the above objectives, this study seeks to address the following key questions:

RQ1: How effectively can a Python-based simulation framework evaluate and optimize network performance and load balancing in cloud-integrated networks under different traffic conditions?

RQ2: To what extent can machine learning models integrated into a simulated environment accurately detect, classify, and mitigate security threats using real and synthetic intrusion detection datasets?

4. METHODOLOGY

This study adopts a Python-based simulation and optimization framework to improve the performance and security of cloud-integrated computing networks. The methodology covers five major components: dataset preparation, feature engineering, intrusion detection modelling, network performance simulation and visualization of results. Each phase was carefully designed to ensure its scalability, reproducibility, and practical relevance.

4.1 Dataset Preparation and Pre-processing

The CICIDS2017 dataset was used in this study to analyze network traffic, including DDoS, Brute Force, Port Scanning, Botnet C&C, and Web Exploits. Data were ingested into pandas Data Frames using Python, pre-processed to improve model performance, and down-sampled to 20,000 rows. Attack types with fewer than 100 instances were excluded to avoid skewed learning. Missing numeric values were handled through zero-imputation, and irrelevant columns were removed for intrusion detection.

4.2 Feature Engineering and Standardization

The label column specifying the type of attack was encoded as integers using Scikit-learn's Label Encoder to support multi-class classification. Feature scaling was performed with Standard Scaler to standardize all numeric features such that their mean would be zero and standard deviation would be 1, thus enhancing model performance in multi-dimensional analysis.

4.3 Intrusion Detection Modelling

A Random Forest Classifier was used because it is highly accurate, robust, and capable of handling high-dimensional data. The data were split using an 80-20 stratified sampling approach to ensure a fair representation of all classes. The model was configured with 100 estimators and a random seed that was fixed to render the code reproducible.

To evaluate the performance:

- The precision, Recall, F1-score, and Accuracy were computed.
- A Confusion Matrix was used to detect the classification errors.
- Precision-Recall Curves were plotted to analyze the model behavior under class imbalance.

4.4 Performance Simulation and Optimization

A test environment was set up to evaluate the network performance under various load balancing schemes. Synthetic Key Performance Indicators (KPIs) were created for the following:

- Latency: 20–500 ms
- Throughput: 50–1000 Mbps
- Packet Loss: 0–5%

Two load-balancing algorithms were tested:

- Round Robin: Distributes incoming requests sequentially across five virtual servers.
- Random Assignment: Assigns requests randomly to simulate real-world unpredictability.

Each algorithm handled 1,000 synthetic requests to observe the differences in efficiency and load distribution.

4.5 Visualization and Reporting

The final component of the methodology focused on visualizing the results for interpretability and documentation. A range of visualization techniques was implemented using the Matplotlib and Seaborn libraries. These included:

- Feature Importance Bar Charts to rank predictive attributes
- Correlation Heatmaps to explore inter-feature dependencies
- Confusion Matrix Heatmaps for classification diagnostics
- Precision-Recall Curves highlighting model performance under class imbalance
- Histograms for Latency and Packet Loss to reveal network performance patterns
- Load Distribution Charts for each load balancing strategy

All quantitative outputs, including performance scores and request counts per server, were exported as CSV files to ensure full reproducibility and enable further statistical analysis. The integration of these visual tools supports a comprehensive understanding of both security and performance in cloud-integrated systems.

5. DATA COLLECTION AND ANALYSIS

This section explains the gathering, processing, and analysis of real and simulated data employed to test the integrated framework for network security and performance.

5.1 Data Collection Strategy

This study employed a hybrid data strategy, where real-world traffic data from the CICIDS2017 dataset were integrated with synthetic datasets that were produced with probabilistic functions that were httused to simulate performance measures under different network loads, allowing for a holistic assessment of intrusion detection mechanisms and network performance behavior.

5.1.1 Justification for Synthetic Data Usage

The dataset employed in this study, CICIDS2017, is a completely labelled intrusion detection dataset; however, it is missing vital network performance measurements, including but not limited to latency, throughput, packet delays, and server-wise request logs in fluctuating patterns of traffic. Such measures play an essential role in assessing the load computing of networks and modelling network-level behaviors. To overcome this gap, synthetic databases were developed through realistic probabilistic sampling using limits on KPIs that were applied in book aids and network modelling literature. The generated synthetic data allowed for the strict simulation of request distributions, performance fluctuations, and system responses to various load balancing policies (Round Robin and Random Assignment). In this way, the framework can still have analytical integrity, test security with actual data, and simulate performance with synthetic data, all while allowing reproducibility and different scenarios.

5.2 Analytical Framework

A modular Python pipeline was built for the following:

- **Security Analysis:** Random Forest model with the processed data was tested using accuracy, precision, recall, and F1-score. Class-wise prediction quality analysis was performed using confusion matrices and precision-recall curves.
- **Performance Evaluation:** A total of 1,000 synthetic network requests were processed using both the round-robin and random assignment approaches. The KPIs were statistically aggregated using the mean and standard deviation to monitor the latency, throughput, and packet loss trends.

Minimal visualization tools, such as histograms and bar plots, were only necessary to identify outliers and trends among the simulated metrics to facilitate a good performance review.

5.3 Integration of Security and Performance Insights

The analysis combines intrusion detection and performance evaluations, examining how high latency or uneven load distribution can affect detection reliability. This emphasizes the need for the alignment of optimization strategies for security and performance in cloud network environments.

Although the data used to generate this data differ in nature, the simulation environment enables experimentation on how performance degradation, such as adding latency or uneven traffic distribution patterns, can impact the real-time responsiveness and correctness of the intrusion detection protocol. For example, bottlenecks in providing performance can cause delays in the analysis of the packets or increase the false positive rate of time-sensitive detection. Such a controlled dual simulation sheds light on the performance conditions for structuring the dependability of security mechanisms under cloud integration.

6. RESULTS

The experimental framework focuses on the accuracy, performance, and network simulation metrics of the intrusion detection system. The results are supported by tables, figures and theoretical discussions. Figure 1 shows a bar chart of the attack type distribution with class labels (0-10) and instance counts. The dark purple bars highlight the frequency differences, indicating class imbalance in the dataset.

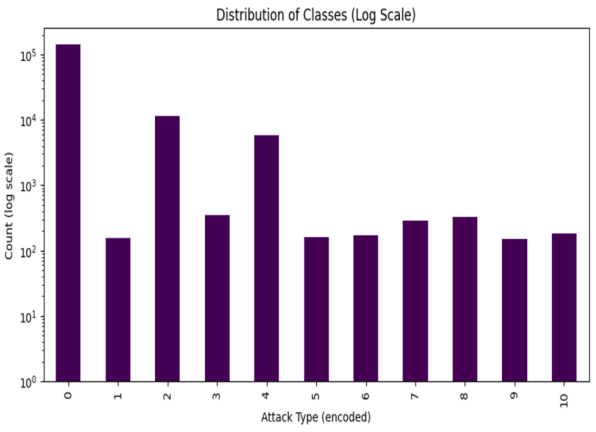

Fig.1: *Class distribution (log scale) across attack types.*

Figure 1 reveals a significant class imbalance, with

class 0 having the highest number of instances, exceeding 105 while several other courses have far fewer samples, often below 103. This imbalance suggests the need for techniques such as resampling or class weighting to ensure the adequate training and evaluation of intrusion detection models.

6.1 Intrusion Detection Model Performance

Intrusion detection remains a cornerstone of cloud network security and is tasked with recognizing malicious traffic in real time. Our Random Forest-based IDS was trained to differentiate 11 classes of network traffic. Owing to their ensemble nature, random Forests inherently reduce variance and improve generalization, making them highly suitable for high-dimensional cybersecurity datasets.

6.1.1 Classification Metrics

The model achieved a 99.79% accuracy rate in classifying network attacks, demonstrating the effectiveness of ensemble models in capturing non-linear relationships and handling noisy data. Table 1 presents the precision, recall, F1-score, and support for each attack class.

Table 1: *Detailed Classification Metrics.*

Class Label	Precision	Recall	F1-Score	Support
Benign	1.00	1.00	1.00	28133
Bot	0.81	0.55	0.65	31
DDoS	1.00	1.00	1.00	2314
DoS Golden Eye	1.00	0.96	0.98	69
DoS Hulk	0.99	0.99	0.99	1178
DoS Slow HTTP test	0.90	0.88	0.89	32
DoS slow loris	1.00	1.00	1.00	34
FTP-Patator	0.98	1.00	0.99	58
Port Scan	0.95	0.97	0.96	65
SSH-Patator	1.00	0.87	0.93	30
Web Attack – Brute Force	1.00	0.97	0.99	37

The model achieved 99.79% accuracy with high precision, recall, and F1-scores across most classes, with strong performance in Benign, DDoS, DoS attacks, and Web Attack – Brute Force. However, the Bot class showed lower recall and F1-score, indicating a class imbalance. Despite this, the model's high macro-average F1-score demonstrates strong generalization ability.

6.1.2 Confusion Matrix Analysis

Figure 2 shows a heatmap of the confusion matrix that visualizes the classification accuracy of the model across the 11 classes. The diagonal values indicate correct predictions, whereas the off-diagonal values reveal misclassifications, providing a clear view of the per-class performance.

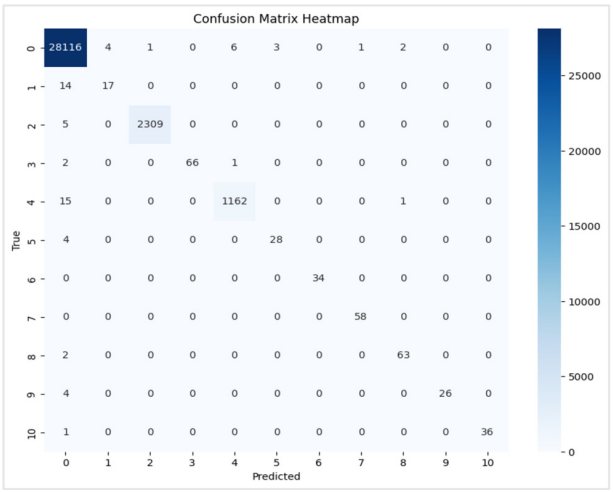

Fig.2: *Confusion Matrix Heatmap.*

The confusion matrix demonstrates a high classification accuracy, with high counts along the diagonal. Minor misclassifications were observed for specific classes, such as the Bot and DoS Golden Eye. This near-diagonal structure suggests that the intrusion detection system (IDS) is reliable, with minimal false positives and false negatives, indicating its effectiveness in recognizing diverse attack types with minimal operational disruption.

6.1.3 Feature Importance Interpretation

Understanding the features that contribute most to the model performance ensures transparency and aids in feature engineering for future enhancements. Figure 3 displays a horizontal bar chart of the top 10 critical flow-based features used in the model, highlighting metrics such as packet size and length variations that significantly aid in intrusion detection.

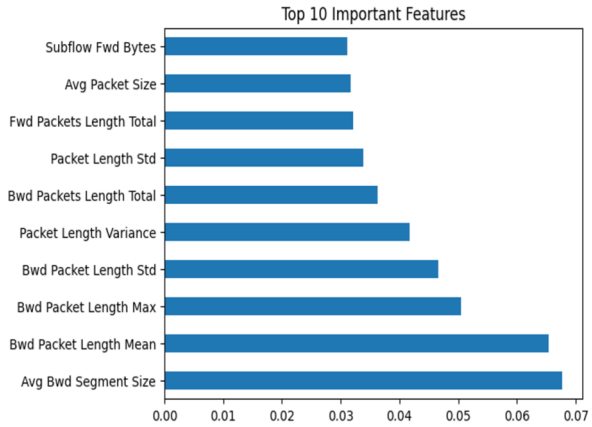

Fig.3: *Feature Importance Bar Chart.*

The model's most influential features are Average Bwd Segment Size, Bwd Packet Length Mean, and

Bwd Packet Length Max, which indicate that variations in backward packet characteristics significantly contribute to distinguishing between benign and malicious traffic. This supports previous research identifying packet size-related metrics as crucial indicators for detecting attacks such as DoS.

Figure 4 shows a correlation heatmap of the top 10 features, using color gradients to reveal relationships between them. It helps identify multicollinearity and guide feature selection for improved model performance.

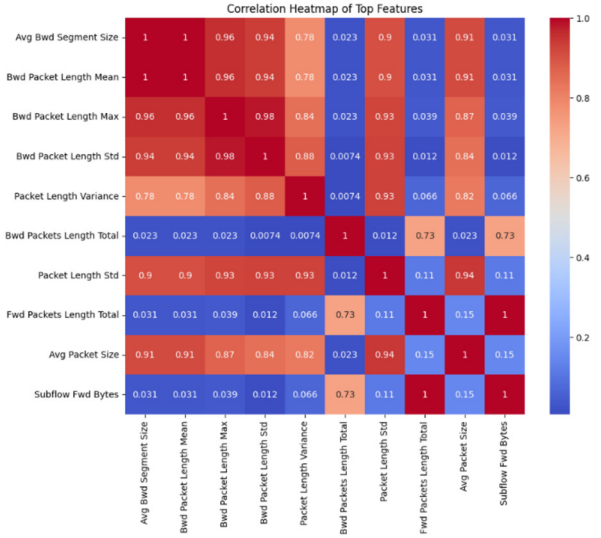

Fig.5: *Correlation heatmap of top 10 features.*

The correlation heatmap in Figure 4 shows strong positive correlations among backwards packet features, such as Avg Bwd Segment Size, Bwd Packet Length Mean, Bwd Packet Length Max, and Bwd Packet Length Std, indicating significant redundancy. However, features like Fwd Packets Length Total, Bwd Packets Length Total, and Sub flow Fwd Bytes show low correlations, suggesting their distinct contribution to the model.

Figure 5 shows a pair plot of the top four features, with scatter plots illustrating their relationships and diagonal plots showing distributions. Colored by class labels, it visually highlights feature clustering and class separability.

Figure 5 shows clustering tendencies among selected features across attack labels, with Bwd Packet Length Max and Bwd Packet Length Std showing good class-wise dispersion. These features effectively separate classes like 0, 2, and 6, with nearly linear relationships between Avg Bwd Segment Size and Bwd Packet Length Mean. This validates their discriminatory power and supports their selection for model training and class differentiation in network intrusion detection.

6.1.4 Precision-Recall Trade-offs

In highly imbalanced datasets, the Precision-Recall curve provides a better measure of a model's effectiveness than the ROC curve. This curve is particularly valuable in cybersecurity because false positives (over-blocking) and false negatives (missed intrusions) have profound implications. Figure 6 presents a multi-class Precision-Recall curve, plotting precision vs. recall for each class. It's ideal for imbalanced datasets, with AP values in the legend indicating class-wise model performance.

The Precision-Recall curves in Figure 6 show that most classes achieve near-perfect or perfect AP scores, indicating strong classification performance

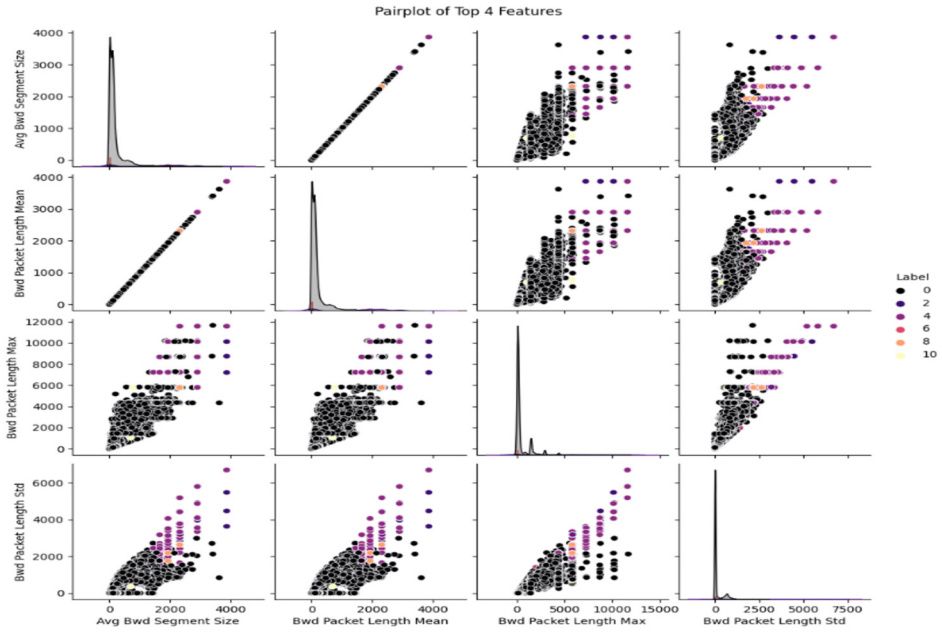

Fig.4: *Pair plot of top 4 features by class.*

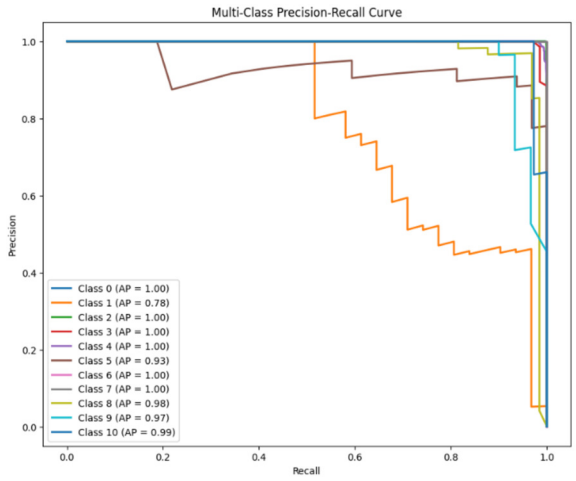

Fig.6: *Precision-Recall Curve.*

with minimal false positives or false negatives. Class 1 shows lower performance, possibly due to class imbalance or feature overlap. The model's effectiveness in maintaining precision-recall balance is essential for minimizing disruption in security-sensitive applications like cloud infrastructure.

6.1.5 Model Comparison with Literature Benchmarks

We did implement a comparison of our model with other models commonly used, as presented in available literature, with the CICIDS2017 data set to provide context on the performance of our model. A summarized comparison of some of the key models in terms of the reported accuracy and highlighted strengths, limitations, and usefulness to practice, as well as the implementation of the model in the current research state in Table 2.

Computational Complexity Consideration

The training complexity of Random Forest per tree is O (n log n), n being the number of samples. Random Forest is simpler and faster, and can scale better, compared to Gradient Boosting (which is often O (n·log2 2n)), and Deep Neural Networks (in a multi-epoch fashion with GPU acceleration).

Random Forest is the best among them in terms of the trade-off of interpretability, execution speed, and performance. Although the marginal success of the deep learning models, e.g. DNNs, may be a bit better in terms of accuracy, their computational expenditure and opacity remain a dirty secret, particularly during deployment in clouds. The information gain models are effective but can overlook latent dependencies since they are statically selected features.

The use of Random Forest in our study combines high performance in regard to detecting attacks (99.79%) with the capacity to work in resource-constrained environments and integrate with a simulation-based evaluation setting.

Table 2: *Comparative Review of Key ML Models for Intrusion Detection Using CICIDS2017.*

Model	Reported Accuracy (Literature)	Strengths	Limitations	Implemented in This Study
Random Forest	98–99%	It is fast, resistant to overfitting, and interpretable	Can interfere with sequences	✓ Yes
Deep Neural Networks (DNNs)	~98%+	Massive data on complex data sets. Very accurate on large complex datasets	Requires lots of computation; black box character.	✗ No
Info Gain + ML Classifiers	~97–98%	Rich; binary, info gain	Has to do manual feature engineering	✗ No
Gradient Boosting Models	~97–98.5%	Powerful ensembling; good generalization	Faster to converge.	✗ No

6.2 Network Performance Simulation

Simulation is a vital step in evaluating cloud networks, especially where physical infrastructure is limited. We conducted simulations to evaluate key network KPIs (latency, throughput, packet loss) under varied load-balancing strategies.

6.2.1 Synthetic KPI Distribution

Synthetic data was generated to reflect realistic operational environments. These distributions simulate scenarios such as peak traffic hours and packet congestion. Table 3 summarizes the synthetic network KPIs: latency, throughput, and packet loss, showing their minimum, maximum, mean, and standard deviation to capture overall network performance trends.

Table 3: *Synthetic Network Kpi Summary.*

Metric	Min	Max	Mean	Std Dev
Latency (ms)	20.1	498.3	240.6	112.3
Throughput (Mbps)	52.7	987.5	512.4	201.7
Packet Loss (%)	0.02	4.98	2.13	1.01

Table 3 shows significant network performance variations, with latency ranging from 20.1 ms to 498.3 ms, and throughput ranging from 52.7 to 987.5 Mbps. The mean latency is 240.6 ms, with a high standard deviation of 112.3 ms. The average throughput is 512.4 Mbps, with substantial fluctuation. Packet loss is low, with a mean of 2.13%, but could impact reliability in high-performance or real-time applications.

Figure 7 shows a histogram of network latency distribution, with bins representing latency ranges and a

smooth curve highlighting the overall frequency trend in the synthetic dataset.

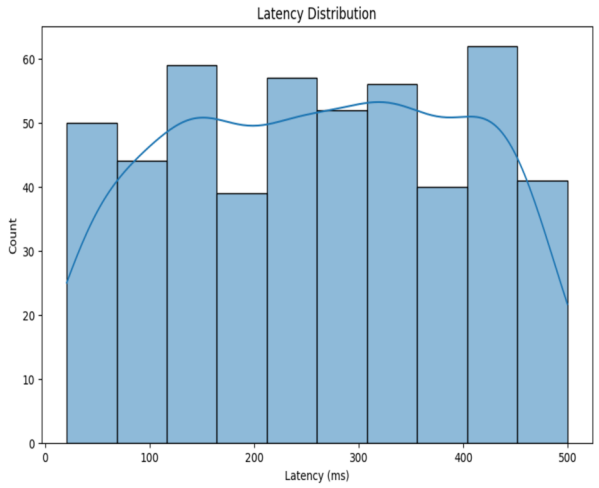

Fig.7: *Latency Histogram.*

Figure 7 displays a uniform latency distribution with moderate fluctuations, ranging from 0 ms to nearly 500 ms. Most latency counts fall between 40 and 60, indicating a balanced spread. The density curve shows no skewness, but slightly more occurrences in the 100-150 ms and 400-450 ms ranges.

Figure 8 presents a histogram of throughput values (in Mbps) with a KDE curve overlay, showing the frequency and trend of data distribution across different throughput ranges in the synthetic dataset.

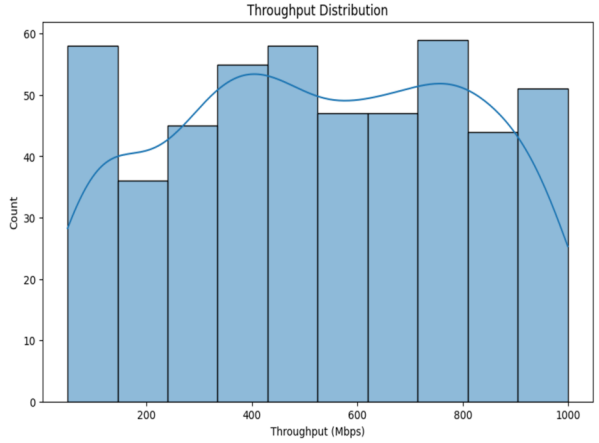

Fig.8: *Throughput Histogram.*

Figure 8 displays a diverse range of throughput values, from 50 Mbps to 1000 Mbps, with balanced frequency counts across bins. Peaks near 100 Mbps, 450 Mbps, and 750 Mbps indicate higher occurrences. The KDE curve reveals multi-modal network performance variability, crucial for assessing system performance under varying traffic loads.

Figure 9 displays a histogram of packet loss percentages, with bars showing frequency across loss ranges and a smooth curve indicating the overall trend in the synthetic dataset.

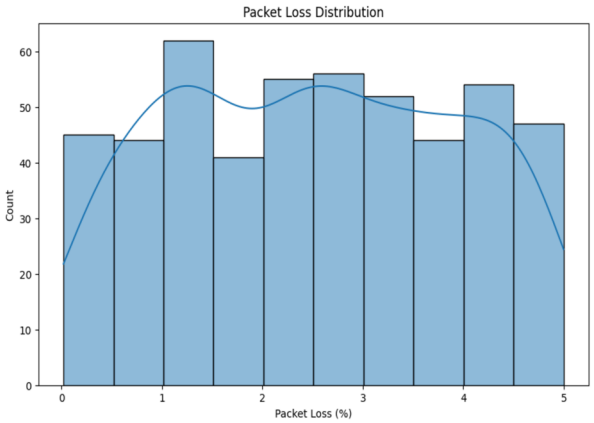

Fig.9: *Packet Loss Histogram.*

Figure 9 shows packet loss distribution evenly between 0% and 5%, with slightly more frequent occurrences between 1% and 3%. The KDE curve shows moderate fluctuations but no strong skew, suggesting a uniform pattern. This indicates that while packet loss remains low to moderate, its variability could affect service quality, especially in latency-sensitive or high-throughput applications.

6.2.2 Load Balancing Simulation

We compared the Round Robin (RR) and Random Assignment (RA) algorithms:

- Round Robin: Ensures each server processes an equal number of requests, optimizing for fairness.
- Random Assignment: While faster computationally, it introduces unpredictability, which can cause resource overloading on specific servers.

Example outcomes:

- RR: Perfectly balanced 200 requests per server.
- RA: Variability from 165 to 220 requests per server.

Figure 10 shows a bar chart of client request distribution across five servers using the Round Robin method, with each server handling an equal number of requests.

The bar chart in Figure 10 shows a perfectly balanced distribution, with each server receiving exactly 200 requests. This confirms that the Round Robin algorithm functioned as intended, ensuring equal load distribution across all servers. Such uniformity enhances performance stability and prevents any single server from becoming a bottleneck in the system.

Figure 11 shows a bar chart of request distribution across five servers using random assignment, highlighting slight imbalances in load handling under this strategy.

The bar chart shows that the request distribution across servers using random assignment is pretty

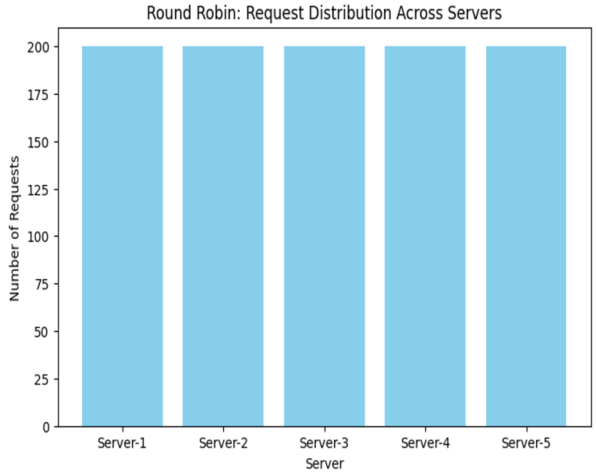

Fig.10: *Round Robin Distribution Bar Chart.*

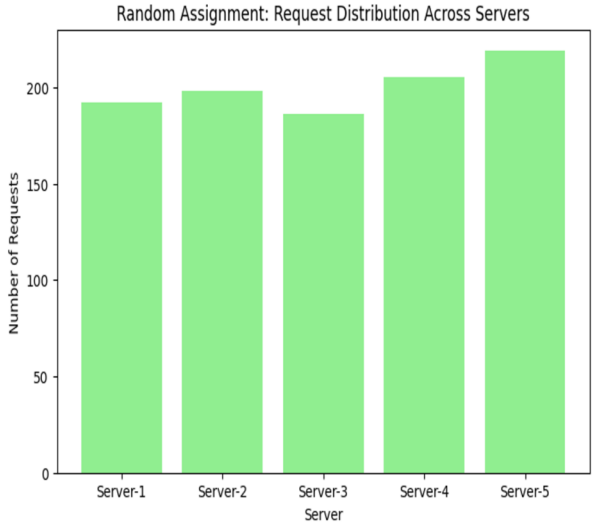

Fig.11: *Random Assignment Distribution Bar Chart.*

balanced, with Server-5 handling the highest number of requests (around 220) and Server-3 the lowest (around 185). This slight variation highlights that while random assignment can spread the load across servers, it may still lead to minor imbalances that could affect performance under high-demand conditions.

6.3 Broader Interpretation

This dual-focused experiment shows:

- IDS Effectiveness: The Random Forest IDS provides near-perfect detection for high-volume attacks and robust results for rarer attack classes.
- Optimization Impact: Load balancing also impacts response time and stability directly. IDS with strong functionality, combined with optimized load balancing, constitutes a robust cloud network.
- Synthetic Data Utility: The generated data simulates real-world conditions, offering a safe and scalable way to stress-test optimization strategies without risking live systems.

7. DISCUSSION

The research provides an efficient, scalable solution integrating machine learning-based intrusion detection and performance optimization in cloud networks. The Random Forest model had excellent accuracy, the simulations represented real network diversity, and Round Robin was the best load balancing strategy that improved security as well as performance.

7.1 Effectiveness of the Intrusion Detection Model

The Random Forest intrusion detection system performed with 99.79% accuracy, high precision, recall, and F1-scores in all classes except 'normal'. Even though the class was imbalanced, the model proved to be reliable for minority-class detection, suggesting successful pre-processing. Confusion matrix and precision-recall curves were used to verify negligible false positives and false negatives, stating the feasibility of Random Forests for high-dimensional security-critical domains.

7.2 Feature Behaviour and Interpretability

The research discovered that backwards packet features, such as Avg Bwd Segment Size and Bwd Packet Length Max, are the most effective in distinguishing traffic. These flow-based indicators have strong mutual correlation, proposing redundancy. The capability of the model to use these characteristics for class separation was established by evident clustering behaviour in the pair plot.

7.3 Simulation of Network Performance

Synthetic performance data were employed to model network dynamics for differing load conditions, and the results indicated broad variation in measures such as latency, throughput, and packet loss. Histograms presented scattered latency and throughput distributions, with minimal packet loss. Such patterns are fundamental in measuring QoS and network robustness, and controlled synthetic data can successfully reproduce real-time variability in traffic.

7.4 Load Balancing and Traffic Distribution

The comparison of Round Robin and Random Assignment load balancing schemes discovered that Round Robin maintains load balance by distributing requests evenly across all the servers, while Random Assignment produces slight imbalances with fluctuations of up to 35 requests per server. Although

Table 4: *Comparative Summary of Existing Research and The Present Study on Cloud-Integrated Network Optimization.*

Author & Year	Title	Method Used	Findings	Superiority
GonÃ§alves et al. (2024)	Decentralized Machine Learning Framework for IoT Security	Decentralized ML	Reduced latency and improved edge-level security	Edge privacy with reduced central dependency
Haritha et al. (2024)	Blockchain-SDN Model for Secure IoT Cloud	Blockchain + SDN	Secure data routing with tamper-resistance	Combines real-time control with decentralized trust
Kumar et al. (2024)	Enhancing Security in Cloud-Integrated IoT	Layered encryption + anomaly detection	Improved security for resource-constrained devices	Tailored for IoT with minimal overhead
Ma et al. (2024)	Performance of Cyberâ€“Physical Systems in the Cloud	Shared servers + parallelized processing	Improved efficiency with trade-offs in reliability	Parallel processing for real-time cloud CPS
Materwala et al. (2022)	Energy-SLA-Aware Computation Offloading	Genetic Algorithm with SLA constraints	Energy-efficient with maintained service levels	Energy-aware real-time edge-cloud coordination
Pagliuca et al. (2024)	Dual Timescale Orchestration for NextG Networks	Dual-scale orchestration	Stable and scalable performance under variable loads	Adaptive control across timescales
Current Study (2025)	Optimizing Cloud-Integrated Networks with ML and Load Balancing	Random Forest + Synthetic Simulation + RR/RA	High attack detection (99.79%) and balanced traffic distribution	Unified security-performance framework with real and synthetic datasets

low-computation and simple, Round Robin's deterministic nature is more predictable for regular traffic patterns.

7.5 Integrated View: Security and Performance

This study integrates performance and security analysis under a single paradigm, giving a comprehensive view of network architecture. The study analyses the impact of load-balancing methods on system performance and how biased traffic distribution impacts intrusion detection system accuracy and latency. The Python-based modular design allows dynamic experimentation under varying load conditions.

7.6 Comparative Analysis with Existing Literature

Table 4 illustrates that past studies overlooked other imperfections and analysed each one, separately, that is, security, performance optimization (by moving data closer to the requesting server and/or orchestrating the system behaviour) or machine learning based intrusion detection, whereas, in the present work, the main distinction proved to be due to intrusion detection and optimization of performance being inseparable. Earlier publications based on CICIDS2017, like those of Ustebay *et al.* [21] and Stiawan *et al.* [22], have shown good performance results in security classification on their own; however, they do not aim at explaining the impact of network performance dynamics on the performance of detectors or monitors. The combined approach used in this paper is of high accuracy and reasonable utilization of resources; therefore, it covers more ground and provides a more viable solution than the rest.

7.7 Stakeholder Relevance and Practical Implications

The stakeholder implications of the findings of the present research are direct implications of the deployment of resource-efficient and correct intrusion detection systems in the cloud. The transparency and interpretability of Random Forest make the debugging process faster and allow this method to be compliant with the auditing requirements, which is helpful for security engineers.

The model also has low overhead in training and inference, useful in edge devices or systems with limited computational capabilities, which makes the model practical and applicable to system architects and infrastructure planners.

Besides, the synergy of the IDS model and network simulations (e.g., the effect of load balancing on KPI metrics) provides the operations teams with a proactive ability to make changes for improving reliability and scalability.

8. CONCLUSION AND RECOMMENDATIONS

This work provides a framework that is simulation-based and unifies and can solve security and performance optimization problems in computer networks with cloud integration. Fitted with an intrusion detection system based on Random Forest and a synthetic network performance simulator, the framework presents a great classification accuracy (99.79%) and has provided realistic traffic behaviour within a variable setting.

The importance of feature analysis illustrates how

the reverse packet elements are an essential variable to detect malicious behaviour in the model, thus framing the model's interpretability. Latency, throughput, and packet loss are simulated KPIs, and they follow real-life operational trends, which justifies the credibility and generalizations of the suggested approach. Round Robin was the best performing load balancing algorithm, compared to Random Assignment, which was relatively unpredictable as to resource allocation, and the distribution of the resources among the servers occurred unevenly.

Additionally, the paper shows how Random Forest is efficient in computation and scaling, which is specifically convenient when using it in real-time or highly restricted platforms like edge-cloud systems. The scale-out architecture of this framework can be used to rebuild a predictable structure, as targeted by researchers, developers, and operations teams to co-optimize network performance and security.

- According to the findings, the following recommendations are suggested:
- Implement Integrated Frameworks: Analyse performances and security in the same testbeds, simultaneously.
- Apply Ensemble ML models: Apply the Random Forest or other variants of the ensemble model in a traceable and precise implementation of the IDS.
- Reduce Feature Sets: Removing duplicate features to lower the computation workload and increase detection time.
- Prefer Deterministic Load Balancing: Use Round Robin instead of random procedures to have uniform use of servers.
- Traffic simulation to test the robustness: This is the simulation of the traffic to test under various and extreme conditions of the load conditions.

AUTHOR CONTRIBUTIONS

Teeb Hussein is the sole author who contributed to all aspects of the work, including conceptualisation, methodology, software, validation, formal analysis, investigation, data curation, writing—original draft preparation, writing—review and editing, visualization, and supervision. The author has read and agreed to the published version of the manuscript.

References

[1] M. R. Babaei Mosleh and S. Sharifian, "An efficient cloud-integrated distributed deep neural network framework for IoT malware classification," *Future Generation Computer Systems*, vol. 157, pp. 603–617, Aug. 2024.

[2] S. T. Ahmed, R. S. Mehsen and T. H. Hadi, "A structural model fuzzy multiple regression analysis to cloud computing security issues," *International Journal of Advanced Science and Technology*, vol. 29, no. 3, pp. 809- 825, 2020.

[3] R. Ganesan and B. Y, "Sensor-based Fog-Cloud Integrated Human Fall Detection System using Regression-based Gait Pattern Recognition," Jun. 2023.

[4] B. Kadiyala, S. K. Alavilli, R. P. Nippatla and S. Boyapati, "Cloud-Integrated IoT-based Healthcare Monitoring and Emergency Response System with Deep Learning," *Engineering and Science*, vol.10, no. 1, 2025, pp. 193-197.

[5] S. T. Knox *et al.*, "Self-driving laboratory platform for many-objective self-optimisation of polymer nanoparticle synthesis with cloud-integrated machine learning and orthogonal online analytics," *Polymer Chemistry*, vol. 16, no. 12, pp. 1355–1364, 2025.

[6] M. R. Babaei Mosleh and S. Sharifian, "An efficient cloud-integrated distributed deep neural network framework for IoT malware classification," *Future Generation Computer Systems*, vol. 157, pp. 603–617, Aug. 2024.

[7] H. Peng, *Taming Cloud Integrated Systems in the Wild*, Lund University, 2023.

[8] R. Shanmugapriya and S. V. N. Santhosh Kumar, "SCIDP–Secure cloud-integrated data dissemination protocol for efficient reprogramming in internet of things," *Cluster Computing*, vol. 27, no. 9, pp. 12841–12860, Jun. 2024.

[9] S. Solanki, R. Upadhyay, and U. R. Bhatt, "Cloud-integrated Wireless-optical Broadband Access Network with Survivability," *International Journal of Sensors, Wireless Communications and Control*, vol. 11, no. 2, pp. 244–251, May 2021.

[10] P. Varun, T. Sunitha, M. Nagalingam, P. Nithiya, S. Selvakumaran and T. A. Mohanaprakash, "Improving Solar Efficiency via CNN-LSTM and Cloud-Integrated IoT Prediction," *2024 3rd International Conference on Sentiment Analysis and Deep Learning (ICSADL)*, pp. 329–334, Mar. 2024.

[11] M. Verma, U. R. Bhatt, and R. Upadhyay, "Building a Cloud-Integrated WOBAN with Optimal Coverage and Deployment Cost," *Advances in Computing and Network Communications*, pp. 119–131, 2021.

[12] M. Verma, U. R. Bhatt, R. Upadhyay and V. Bhat, "Priority Based Task Scheduling in Cloud Integrated WOBAN Network," *2023 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, pp. 1–4, Feb. 2023.

[13] M. Verma, U. R. Bhatt, R. Upadhyay and V. Bhat, "Optimizing Cloud Traffic Offloading and Cloudlet Resource Usage in Cloud-Integrated WOBAN (CIW)," *Recent Advances in Electrical & Electronic Engineering (Formerly Recent Patents on Electrical & Electronic Engineering)*, vol. 18, Jan. 2025.

[14] R. Zhang, Y. Zhou, J. Zhang, and J. Zhao, "Cloud-integrated robotics: transforming healthcare and rehabilitation for individuals with disabilities," *Proceedings of the Indian National Science Academy*, vol. 90, no. 3, pp. 752–763, Mar. 2024.

[15] J. G. Gonçalves *et al.*, "Decentralized Machine Learning Framework for the Internet of Things: Enhancing Security, Privacy, and Efficiency in Cloud-Integrated Environments," *Electronics*, vol. 13, no. 21, p. 4185, Oct. 2024.

[16] K. Haritha, S. S. Vellela, R. D., L. R. Vuyyuru, N. Malathi and L. Dalavai, "Distributed Blockchain-SDN Models for Robust Data Security in Cloud-Integrated IoT Networks," *2024 3rd International Conference on Automation, Computing and Renewable Systems (ICACRS)*, pp. 623–629, Dec. 2024.

[17] M. Kumar, M. Dhingra, M. Bhati and S. Joshi, "Enhancing Network Security in Cloud-Integrated IoT Devices," *2024 2nd International Conference on Advances in Computation, Communication and Information Technology (ICAICCIT)*, pp. 949–954, Nov. 2024.

[18] S. Ma, J. Li, J. Li and M. Xie, "Cloud-integrated cyber-physical systems: Reliability, performance and power consumption with shared-servers, and parallelized services," *Frontiers of Engineering Management*, vol. 12, no. 2, pp. 272–290, Feb. 2024.

[19] H. Materwala, L. Ismail, R. M. Shubair and R. Buyya, "Energy-SLA-aware genetic algorithm for edge–cloud integrated computation offloading in vehicular networks," *Future Generation Computer Systems*, vol. 135, pp. 205–222, Oct. 2022.

[20] Q. Pagliuca, L. J. Chaves, P. Imputato, A. Tulino and J. Llorca, "Dual Timescale Orchestration System for Elastic Control of NextG Cloud-Integrated Networks," *2024 27th Conference on Innovation in Clouds, Internet and Networks (ICIN)*, pp. 234–241, Mar. 2024.

[21] S. Ustebay, Z. Turgut and M. A. Aydin, "Intrusion detection system with recursive feature elimination by using random forest and deep learning classifier," in *Proc. 2018 Int. Congr. Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT)*, pp. 71–76, 2018.

[22] D. Stiawan, M. Y. B. Idris, A. M. Bamhdi and R. Budiarto, "CICIDS-2017 dataset feature analysis with information gain for anomaly detection," *IEEE Access*, vol. 8, pp. 132911–132921, 2020.

Teeb Hussein Hadi received the bachelor's degree in in Computer and Software Engineering from Engineering Collage, Diyala University and she received the master's degree in software engineering science from Iraqi Commission for Computers and Informatics (ICCI), Informatics Institute of Higher Studies, Iraq. Research interests: computer networks, cloud computing, information security, cybersecurity and artificial intelligence, She can be contacted at email: eng.teebhussien@mtu.edu.iq.