International Journal of Engineering Trends and Technology
ISSN: 2231-5381/ https://doi.org/10.14445/22315381/IJETT-V73I3P115

Volume 73 Issue 3, 198-211, March 2025

Original Article

IoT Edge Computing Security Framework Powered by LSA, MLKEM and GLSKM

A. Anandhavalli[1], A. Bhuvaneswari[2]

[1,2]*Department of Computer Science, Cauvery College for Women (Autonomous), Affiliated to Bharathidasan University, Trichy.*

[1]*Corresponding Author : anandhavalli.ca@cauverycollege.ac.in*

Received: 21 August 2024 Revised: 16 January 2025 Accepted: 7 February 2025 Published: 28 March 2025

Abstract - *The Internet of Things (IoT) Edge computing represents a paradigm shift in data processing and analytics, where data processing occurs closer to the source of data generation rather than being transmitted to centralized cloud servers. This approach addresses several critical challenges in traditional IoT architectures, including latency, bandwidth consumption, security, and reliability. By leveraging edge devices-such as sensors, gateways, and local servers-IoT Edge enables real-time data analysis, rapid decision-making, and localized actions essential for applications requiring immediate responses. IoT Edge faces challenges such as the need for robust edge device management, scalability issues, and the requirement for sophisticated security measures to protect distributed data. A multi-security scheme-based framework titled "IoT Edge Computing Security Framework powered by LSA, MLKEM and GLSKM (IECSF)" is submitted here to address the current challenges in IoT wireless sensor network edge computing environments. Lightweight Security Algorithm, Masked Location-based Key Exchange Mechanism, and Game of Life-based Security Key Mechanism are the background works behind the IECSF method. The proposed IECSF method has three novel functional modules, namely Least Hop Cluster Manager, Intra Cluster Data Distributor, and Inter-Cluster Data Aggregator. Performance metrics such as Throughput, Latency, End-to-End Delay, Packet Delivery Rate, Security and Energy consumption of the proposed method are compared with the performance of the most recent existing works.*

Keywords - *Edge computing, Game-of-life based Security key exchange, Lightweight security, Masked location key, IoT, WSN.*

1. Introduction

The core idea behind IoT is to create a smarter, more responsive environment by integrating the physical world with the digital world. IoT wireless sensor device can monitor their surroundings, process information, and communicate with each other, allowing for real-time data collection, analysis, and decision-making [1]. There is a list of various day-to-day activity examples, such as smart thermostats that can adjust home temperatures based on occupancy patterns, while connected medical devices can monitor patient health and send alerts to healthcare providers in case anomalies are in practice witnessed by current human generation [2, 3]. Integrating the Internet of Things-powered Wireless Sensor Network is pivotal in environmental monitoring, offering innovative solutions to track, manage, and analyze various environmental parameters in real-time [4]. This synergy enhances the capability to monitor ecosystems, detect environmental changes, and respond to potential hazards more effectively [5]. Several notorious key benefits include Real-time data collection, Remote monitoring, Early warning systems, domain-specific customized network environments, pollution control, and Controlled biodiversity care achieved by using IoT-WSN edge network environment. The deployment of IoT-WSN for environmental monitoring has several challenges, including data security and privacy concerns, the need for robust and reliable communication networks, and the management of large volumes of data [6]. Addressing these challenges requires ongoing research, technological innovation, and collaborative efforts among governments, organizations, and communities. IoT edge computing brings numerous advantages, including reduced latency, improved bandwidth management, and enhanced data privacy [7, 8]. However, it also introduces several challenges that must be addressed to fully realize its potential. Securing data at the edge is crucial since sensitive information is processed locally. Ensuring robust encryption and data protection measures is essential to prevent unauthorized access and breaches. Edge devices are often less secure than centralized systems, making them more vulnerable to attacks [9]. Ensuring the security of these devices against hacking, malware, and physical tampering is one of the critical challenges. The lack of standardization in IoT-Edge computing can lead to compatibility issues between devices from different manufacturers. Developing and adopting common standards and protocols is necessary for seamless integration. Ensuring that new IoT-Edge solutions can

integrate with existing legacy systems can be difficult, requiring significant customization and adaptation. Security key generation is crucial in ensuring data integrity, confidentiality, authentication, access control, secure communication establishment, energy efficiency, and reliability against several attacks [10]. The challenges are relatively complex due to the resource-constrained nature of heterogeneous IoT-WSN devices. The IECSF enhances security and efficiency in IoT wireless sensor network edge computing by integrating three novel mechanisms: LSA, MLKEM, and GLSKM. It introduces three functional modules-Least Hop Cluster Manager, Intra Cluster Data Distributor, and Inter-Cluster Data Aggregator-to optimize data processing, minimize latency, and enhance reliability. The framework is evaluated against recent existing methods based on key performance metrics, including Throughput, Latency, Packet Delivery Rate, Security, and Energy Consumption, demonstrating its effectiveness. This multi-layered security and data management approach significantly improves real-time decision-making, scalability, and security in IoT edge environments.

2. Existing Methods

Most recent security-energy balanced protocols have been chosen from both highly secure, energy-intensive protocols and energy-efficient, low-security protocols to provide a clear evaluation comparison with the proposed method. They are Machine-Learning-Based IoT-Edge Computing Healthcare Solutions [11], Brainyedge: An ai-enabled framework for IoT edge computing [12], an efficient IoT group association and data sharing mechanism in edge computing paradigm [13], and Node Selection Algorithm for Federated Learning Based on Deep Reinforcement Learning for Edge Computing in IoT [14].

2.1. Machine-Learning-Based IoT-Edge Computing Healthcare Solutions (MLIEC)

In 2023, Abdulrahman K. Alnaim et al. proposed the MLIEC method to achieve optimized data transfer in IoT edge networks by incorporating distributed computing and task-level parallelism. A distributed Edge computing-based IoT framework backed by machine learning algorithms is used to combine the cloud and edge computing applicability in real-time environments. The design of edge-based computing to collect patients' data is critical for modern healthcare systems. Securing communication between edge nodes and ensuring the privacy of patient data are paramount. A new hybrid model, which consists of IoT and edge nodes, is being applied to predict and mitigate cyberattacks in medical healthcare systems. Additionally, new machine learning algorithms are being developed specifically for use on edge devices with limited resources. Privacy and security concerns surrounding collecting and transmitting personal health data are being thoroughly investigated. Studies are examining the effectiveness of IoT–edge-computing-based solutions for improving patient outcomes and reducing healthcare costs. The development of new IoT devices and sensors for healthcare applications is also being explored. Integrating IoT–edge computing with other technologies, such as 5G networks, aims to improve data transmission and processing capabilities. Different edge computing architectures, including fog computing and cloudlets, are being compared to assess their suitability for healthcare applications. The scalability and reliability of IoT–edge-computing-based solutions for healthcare are under investigation, alongside the development of models for data fusion and data analytics specifically tailored for healthcare applications.

A handful of dedicated architectures are introduced in MLIEC work for different functionalities such as medical healthcare systems, security and privacy preservation, edge node-based IoT healthcare systems, edge distributed ledger models, public edge distributed ledger infrastructure, cloud-based edge distributed ledger and privacy-centric strategy. The MLIEC method achieves higher accuracy, precision, and recall, effectively eliminating various attacks. However, using a stacked multiple cryptography procedure, specifically Blowfish, RSA, and AES, results in high processing time, adversely affecting overall throughput. This increased computational overhead is recognized as a key limitation of the MLIEC method. The trade-off between security and efficiency remains challenging despite its strong security measures.

2.2. Brainyedge: An AI-Enabled Framework for IoT Edge Computing

Brainyedge work was published in 2023 through ScienceDirect by Kim-Hung Le et al. to provide artificial intelligence for IoT-Edge computing environment. The authors state that the blending of AI along with IoT edge improves the Quality of Experience (QoE) altogether. A set of AI models are taken, and the authors of Brainyedge work carry out several enhancements. In the Bootstrap phase of the Brainyedge model, the selected model is packaged as a docker container with its environment. This micro-service is prepared for deployment to edge nodes using techniques such as initiating suitable machine learning platforms such as TensorFlow and PytorchandKeras, injecting necessary libraries, optimizing the model size, and defining access rights policies for edge device collaboration. In the Deployment phase, the model package is compressed and stored in a shared repository accessible to authorized edge nodes. Compression and pruning techniques are employed to minimize network costs during deployment. During the Operating and Learning phases, each edge node unpacks the model package, creates a container using docker files, and retrains the model with local, edge-private data upon its initial operation after booting. This approach enhances the model's specificity to edge contexts and significantly improves inference accuracy. Additionally, incremental learning updates the model with recent data to maintain efficiency in response to changes in edge contexts. The BrainyEdge work achieves higher model accuracy and

reduced computational delay, making it a significant advantage. However, the model is tested in a computational environment with abundant resources, leading to a higher spike in deployment cost, which is identified as its limitation. While the approach enhances performance, its cost-effectiveness remains a challenge.

2.3. An Efficient IoT Group Association and Data Sharing Mechanism in Edge Computing Paradigm (IGADSM)

In 2023, author Haowen Tan published IGADSM work to provide secure data communication in IoT Edge environments using group association and updating mechanisms. Slightly modified Edge Intelligence (EI) is used in IGADSM work to improve the group key generation and revocation processes.

A key update flag-based revocation process is followed in the IGADSM work to improve the security features. A specialized certificate authentication procedure is introduced in IGADSM work to improve communication in the IoT Edge environment. Different phases of authentication processes, such as registration, verification, and key distribution, are defined by legacy procedures in IGADSM work. Trusted User (TU) registration and certain non-trivial key initialization preparations occur during offline registration.

All TUs must register with Edge Intelligence (EI) before accessing the edge networks. Customized certificateless authentication methodology and the bilinear pairing strategy ensure security against intruder attacks. The author states that the IGADSM is suitable for practical circumstances in complex environments for edge communication. The IGADSM work enhances security by providing resilience against several attacks, which is recognized as its advantage. However, some IoT nodes among trusted users may undergo constant decryption procedures, allowing only limited key revocation. This limitation could lead to compromised nodes during the key update process, potentially impacting overall security.

2.4. Node Selection Algorithm for Federated Learning Based on Deep Reinforcement Learning for Edge Computing in IoT (NSAFL)

Shuai Yan et al. presented NSAFL work in 2023 for steadfast node selection based on the Deep Reinforcement Learning method. Personal privacy and data leakage are major concerns in IoT edge computing environments. A federated learning-based approach has been used in NSAFL work to improve privacy and eliminate data leakages. Achieving these in the heterogeneous network environment is a complicated task that needs to be resolved using the NSAFL method. Various elements of the NSAFL method, such as Feature extraction, computational model design, communication model design, data quality model, equipment enumeration, network policy, and probability layer pattern, are discussed vividly in the document. NSAFL work significantly improves training accuracy by 30% in heterogeneous device IoT environments while ensuring the efficient participation of diverse devices. The findings underscore the potential impact of these advancements on enhancing privacy protection measures in IoT edge computing, offering innovative solutions that could be widely applicable in practical settings. The findings from this research highlight the transformative potential of federated learning in IoT edge computing, particularly in privacy-sensitive applications such as healthcare, finance, and smart cities. By strengthening privacy protection measures, NSAFL offers a pioneering solution that could be widely applicable in real-world IoT deployments, ensuring data confidentiality, security, and efficiency. Furthermore, NSAFL addresses potential challenges in federated learning, such as communication overhead, device synchronization, and adversarial attacks, making it a robust and scalable solution. Improved accuracy is found to be the advantage of NSAFL work, whereas unnoticed performance depletion in terms of overall throughput is observed as the limitation. A summary of the methodologies, advantages and limitations of the discussed existing methods is given in Table 1.

Table 1. Existing methods outline

Authors	Work	Year	Methodology	Advantages	Limitations
Abdulrahman K. Alnaim et.al.,	Machine-Learning-Based IoT–Edge Computing Healthcare Solutions	2023	Resource-constrained Machine Learning	Higher Accuracy and Precision in attack detection	diminished throughput
Kim-Hung Le et al.,	BrainyEdge: An AI-enabled framework for IoT edge computing	2023	AI and IoT Edge blending	Higher accuracy, lesser computational delay	Higher implementation cost
Haowen Tan,	An efficient IoT group association and data sharing mechanism in the edge computing paradigm	2023	Edge Intelligence and Certificateless authentication	Improved security	Compromised node vulnerability
Yan S et al.,	A. Node Selection Algorithm for Federated Learning Based on Deep Reinforcement Learning for Edge Computing	2023	Federated Learning based on Deep Reinforcement Learning	Higher Accuracy	Lower Throughput

3. Background

The proposed IECSF framework consists of a Lightweight Security Algorithm, a Masked Location-based Key Exchange Mechanism, and a Game of Life-based Security Key Mechanism as the core component. A brief explanation of these functional components is given in this section.

3.1. Lightweight Security Algorithm (LSA)

The LSA model comprises two main functional components. The first module, MAC Address XOR key exchange authentication (MAXOR), initializes communication sessions between nodes. The second module, Legacy Random Number Generator (LRG), handles session key annihilation and updates. Together, these modules constitute LSA for IoT-based wireless sensor networks, facilitating secure and efficient communication setup and maintenance. The session key initialization process is streamlined to avoid power-intensive exponential operations and complex calculations using the MAXOR model.

Utilizing the inherent logical XOR operation within processor architecture ensures seamless initiation of sessions. In essence, MAXOR is a lightweight, hardware-friendly key generation procedure. Generating truly random numbers is critical for security protocols. Equally important is ensuring that different devices generate the same random numbers in a networked environment. This synchronization is essential for consistent session key updates during communication between nodes. Session key updates are necessary under various circumstances, such as session timeouts, high latency or packet loss, and during security threats like intruder attacks, to maintain network security. The Legacy Random Number Generator (LRG) achieves this using the Combined Linear Congruential Method with legacy seeds.

3.2. Masked Location Based Key Exchange Mechanism (MLKEM)

MLKEM method comprises two main components, namely, Dual Rosenberg Pairing Location Masker (DRPLM) and Fuzzy Miller's Elliptic Curve Key Exchange (FMECKE).

Together, they form the MLKEM, where DRPemployed for uniquely initializing communication session keys, and FMECKE is utilized for securely distributing these keys. DRLPM is devised with dedicated bitwise and block-wise diffusion procedures to mask geographical location-based hash and physical media access control address hash.

Rosenberg's strong pairing function combines the two different hash addresses into a single entity. FMECKE module performs the secure key exchange process in the MLKEM model. MLKEM is an energy-efficient security key handling procedure that reduces communication delays. MLKEM has a wide range of real-time IoT-WSN applications due to its evident performance.

3.3. Game of Life-Based Security Key Mechanism (GLSKM)

Random Seed and Iteration Limit Selector (RSILS) and Game of Life-based Key Exchange Mechanism (GoLKEM) are the Primary components of the GLSKM model. In an IoT network, devices vary widely in their hardware capabilities, which affects their ability to perform computational tasks. This heterogeneity means that some devices are not as powerful as others and are dedicated to specific functions. Balancing computational capabilities against performance and cost is crucial. For example, sensor nodes focus on measuring environmental parameters and don't require high computational power. In contrast, data processing nodes need significant computational power to manage large volumes of data.

Key agreement procedures biased towards specific nodes can strain low-powered devices or fail to provide adequate security in high-powered device zones. The RSILS module introduces a seamless bridging process to facilitate communication between devices of different categories. Traditional methods calculate computational power based on hardware characteristics such as processor frequency and memory. In RSILS, a simplified count-based procedure is introduced to quickly assess the computational power index of a node. Based on the power index, the security key size and the number of iterations to be performed are determined effectively using the GLSKM model.

4. Proposed Method

Least Hop Cluster Manager, Intra Cluster Data Distributor, and Inter-Cluster Data Aggregator are the essential core component modules of the proposed IECSF method. The algorithms and methodologies used to construct these modules and their services are explained in this section.

4.1. Least Hop Cluster Manager (LHCM)

The purpose of the LHCM module is to select cluster heads and construct a set of clusters based on the number of hops between the nodes. A novel Multi-criteria Dynamic Cluster Head Selection Algorithm (MDCHSA) is put forward in the LHCM module. The member nodes are assigned to the selected cluster heads based on the hop count. Residual energy, Computational power, Storage capacity, and Energy efficiency are the criteria used in the MDCHS algorithm. Measuring the residual energy of an IoT device typically involves assessing the amount of remaining battery capacity or power available for operation. IoT devices include circuitry to directly measure the voltage across the battery terminals. Battery voltage can provide an approximate indication of the remaining charge, assuming a known discharge characteristic. Let r_x be the representation symbol of residual energy of node x. In LHCM, the computational power is quantized based on the clock speed of the IoT device. The processor's clock speed indicates how fast it can execute instructions per second. Higher clock speeds generally imply faster processing capabilities. Let c_x be the symbol for representing the

computational power (clock speed) of node x. IoT devices are used to be equipped with both volatile and non-volatile memories. The Storage capacity s_x of an IoT node x is computed using the following equation.

$$s_x = \frac{s_{x_v} + s_{x_n}}{2} \quad (1)$$

Where s_{x_v} refers to the volatile memory of nodes x, and s_{x_n} denotes the non-volatile memory, both measured in Bytes (B). Energy efficiency modes in IoT devices are designed to optimize power consumption based on different operational states and usage scenarios.

These modes help extend battery life, reduce energy consumption, and manage resources effectively. LHCM uses a number of power modes such as Active mode, Sleep Mode, Idle Mode, Stand-by mode, Power saving mode, Hibernate mode, Doze mode, Low power wireless mode, and Active power management mode provided to an IoT node. Let e_x be the energy efficiency symbol for node x.

A cluster head selection coefficient Γ is computed by Equation 2 given below.

$$\Gamma_x = \frac{\frac{r_x}{max(r)} + \frac{c_x}{max(c)} + \frac{s_x}{max(s)} + \frac{e_x}{max(e)}}{4} \quad (2)$$

Where max () refers to the maximum quantity of a particular resource available in the network.

Algorithm 1: Multi-criteria Dynamic Cluster Head Selection (MDCHS)

Input: $\forall i = 1 \rightarrow n \coloneqq r_i, c_i, s_i, and\ e_i$
($where\ n\ is\ the\ number\ of\ nodes\ in\ the\ network$)
Output: Cluster heads

Step 1: $\forall i = 1 \rightarrow n \coloneqq read\ r_i, c_i, s_i, e_i$
Step 2: Determine $\max(r)$, $\max(c)$, $\max(s)$, $\max(e)$
Step 3: $\forall i = 1 \rightarrow n \coloneqq compute\ \Gamma_i$ by Equation 2
Step 4: Create a node array $A \exists n$ number of elements
Step 5: $\forall i = 1 \rightarrow n \coloneqq A(i) = i$
Step 6: $\forall i = 1 \rightarrow n :: \forall j = 1 \rightarrow (n-1) \Leftrightarrow i \neq j$
Step 7: $if\ (\Gamma_i < \Gamma_j)$ then $A[i] \oplus A[j] \oplus (A[j] = A[i]$
Step 8: end $\forall i$
Step 9: Let d_i be the Euclidean distance between Node i and the Base station
Step 10: Compute distance weight index as $\forall i = 1 \rightarrow n \coloneqq w_i = \frac{d_i}{d_{max}}$
Step 11: Select Cluster heads from A based on Distance based Weighted Cluster algorithm

Let $CH_1, CH_1, \ldots CH_{n_{ch}}$ be the selected cluster heads where n_{ch} refers to the number of cluster heads. Let $H_1, H_2, \ldots H_{n_{ch}}$ be the Hop sets are directly mapped to the cluster heads $CH_1, CH_1, \ldots CH_{n_{ch}}$ in order. The maximum number of permitted hops is set to 5 since a greater number of hops is inappropriate for including edge computing medium-range IoT applications.

Therefore, any Hop set $H_x$can contain up to 5-member node sets, such as $\{h_1, h_1, \cdots h_5\}$. The member nodes for all cluster heads are allocated through $H_i$using LHCM algorithm

Algorithm 2: Least Hop Cluster Manager (LHCM)

Input: Nodes and Cluster head information
Output: Least Hop Cluster Manager

Step 1: Load network environment
Step 2: $\forall i = 1 \rightarrow n_{ch} \coloneqq$
Step 3: Initialize $H_i = \emptyset$
Step 4:$\forall j = 1 \rightarrow 5 \coloneqq$
Step 5: $(h_j \in H_i) = \emptyset$
Step6: $\forall k = 1 \rightarrow n \coloneqq (h_j \in H_i) \cup k \Leftrightarrow Hopcount(CH_i, k) \equiv j$
Step 7: end $\forall k$
Step 8: end $\forall j$
Step 9: end $\forall i$
Step 10: return $\{H_1, H_2, \ldots H_{n_{ch}}\}$

An illustration of a typical LHCM network cluster circumstance is given in Figure 1. In this way, the proposed LHCM method gathers and arranges the cluster heads and nodes into hop-based clusters.

4.2. Intra Cluster Data Distributor (ICDD)

The process can be handled in an edge computing environment through methods such as Local execution, Task offloading, Hierarchical distribution, Dynamic Orchestration, Microservices Architecture, Data Localization, and Collaborative processing.

The dynamic Orchestration method is applied in ICDD due to surrounding awareness of network congestion, device availability, and resource utilization.

ICDD estimates the process into three categories: Simple, Moderate and High, based on the computational time complexity. The incoming processes are stored in a process Queue and served based on First-In First-Out (FIFO) in ICDD. Each process $p_x$is fetched sequentially from memory, and its complexity label λ_i is designated as defined in Equation 3.

$$\lambda_i = \begin{cases} Simple\ if\ Complexity\ (p_x) < O(1) \\ Moderate\ if\ O(1) \leq Complexity\ (p_x) < O(\log n) \\ High\ if\ Complexity\ (p_x) \geq O(n) \end{cases} \quad (3)$$

The “Simple” category processes are split into tasks and allocated to operate with $Nodes \in h_1$ clusters, the “Moderate” category processes are split into tasks and allocated to operate with $Nodes \in h_2 \cup h_3$.

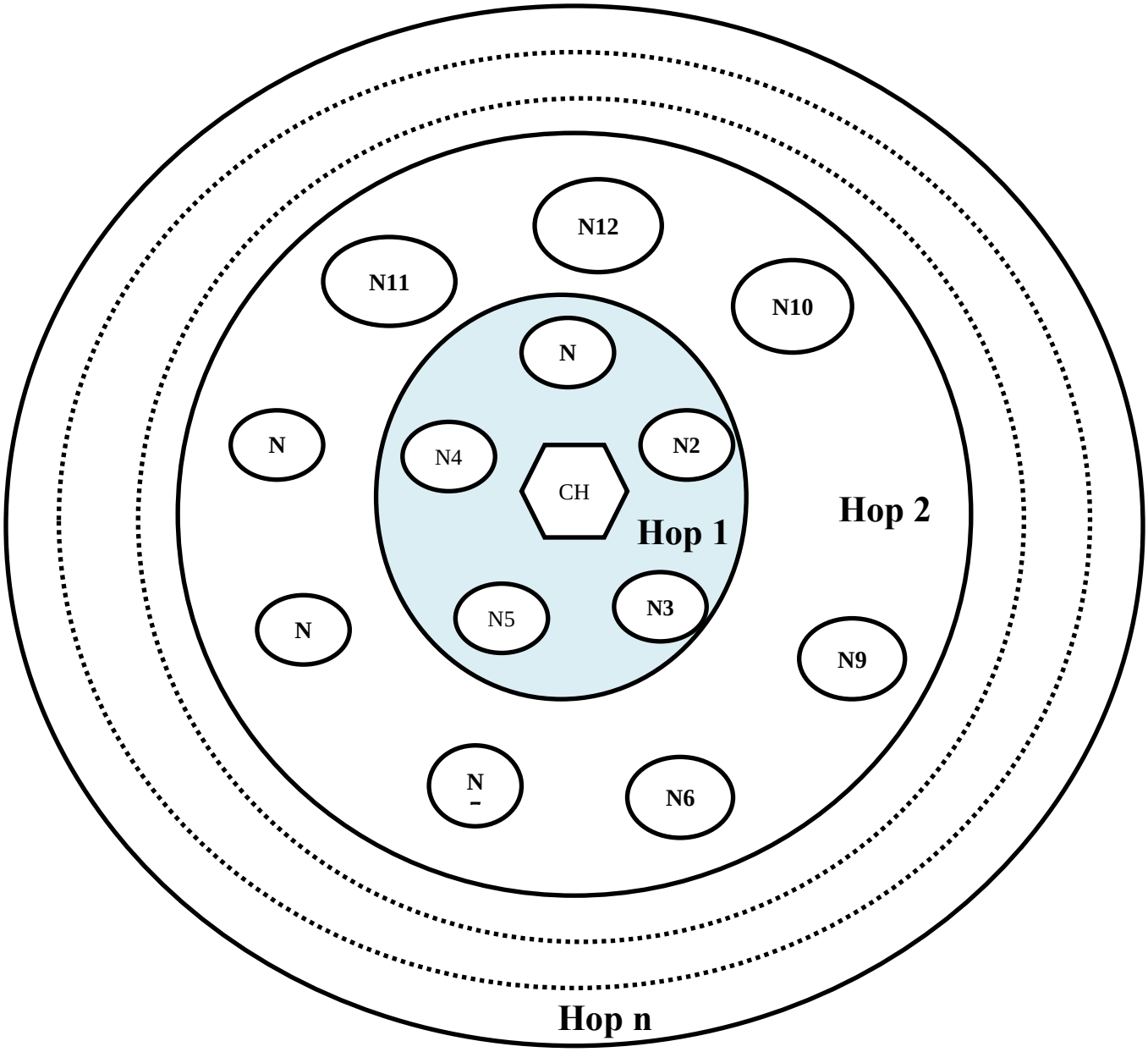

Fig. 1 LHCM Hop-based cluster formation

Similarly, the "High" category processes are fragmented into different tasks and assigned to operate with $Nodes \in h_3 \cup h_4$. In the ICDD method, each task is assigned with a 8 byte unique ID constructed as 8 bits (cluster head id), 8 bits (Node ID), 5 bits (Day), 4 bits (Month), 12 bits (Year), 5 bits (Hour), 6 bits (Minutes), 6 bits (Seconds), and 10 bits (Milliseconds). The Millisecond precision ensures that every task could be assigned with a different ID.

The secured communication between the cluster head and the designated nodes is furnished by the framework elements. That is, the secured communication is established through LSA for $Nodes \in h_1$ clusters, MLKEM for $Nodes \in h_2 \cup h_3$ clusters, and GLSKM for $Nodes \in h_3 \cup h_4$. A typical Task ID was generated in cluster head 3 for node 4 on 01-02-2024 at 06:45:55:369, and its segmentation is presented in Table 2.

Table 2. Task ID segmentation model

Cluster Head ID								**Node ID**								**Day**					**Month**				**Year**											
0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	1	1	1	1	1	1	0	1	0	0	0
Hour								**Minute**								**Second**									**Millisecond**											
0	0	1	1	0	1	0	1	1	0	1	1	1	0	1	1	1	0	1	0	1	1	1	0	0	0	1	0	0	1	1	0	1	0	1	1	0

Each node will return the completed task data to the cluster head after successful task accomplishment; if any node fails to return an executed task, data will be pinged by the cluster head for the current mode state of the node. If the node is active, the cluster head will wait for the node to complete the data. If the node is dead or not reachable beyond the TTL (Time-To-Live) instruction, then that particular task will be reallocated to a different node. ICDD ensures optimal performance and reliability during the data distribution phase by performing these task deployment strategies.

4.3. Inter Cluster Data Aggregator (ICDA)

The process details are stockpiled in the base station, where the ICDA module is deployed. Each process has a number of sub-tasks, which are managed as sets in the ICDA approach. Let $p_x = \{\tau_1, \tau_1 \dots \tau_n\}$ is the process set where $\tau_1 \dots \tau_n$ refers to the subtasks. ICDA has access to ample computational resources since the base stations are relatively more equipped than other nodes in the network. ICDA acquires the fundamental information about the process details, such as process category and the number of sub-tasks involved in the completion of the process. For every process, the ICDA module allocates sufficient memory in the heap for every segmented task in the process. Heap memory is a region of a device memory used for dynamic allocation, allowing flexible runtime memory management for variable-sized data structures. This flexibility can lead to fragmentation and requires careful handling to avoid memory leaks. Whenever ICDA receives task data from cluster heads, it identifies the process information based on the cluster head ID and Node ID. It can also identify the position to store the processed data in the heap based on the date and time fetched from the task ID. A typical representation of the ICDA memory heap is given in Figure 2. Whenever all the task memories of a process are filled in the heap memory, the situation refers that the particular process is completed. Those process details are further moved to non-volatile storage, and the allocated heap will be freed up for further processing. This is how the ICDA module aggregates the task information dynamically using heap memory. ICDA algorithm is given below.

Algorithm 3: Inter-Cluster Data Aggregator

Input: Processes information $p_1, p_2 \dots p_n$
Output: Aggregated processed data

Step 1: Read input informationπ
Step 2: If $\pi \in process$
Step 3: Read π header and create process p
Step 4: Allocate heap memory to fitall$\tau \in p$
Step 5: Else If $\pi \in Tasks$
Step 6: Parse task header and find the associated process
Step7: Insert task data in the appropriate process memory segment
Step 8: If all memory segments for process p are filled
Step 9: Relocate Process Information to non-volatile memory
Step 10: Release heap memory of p
Step 11: End if // Step 8
Step 12: End if // Step 2
Step 13: Start over from Step 1

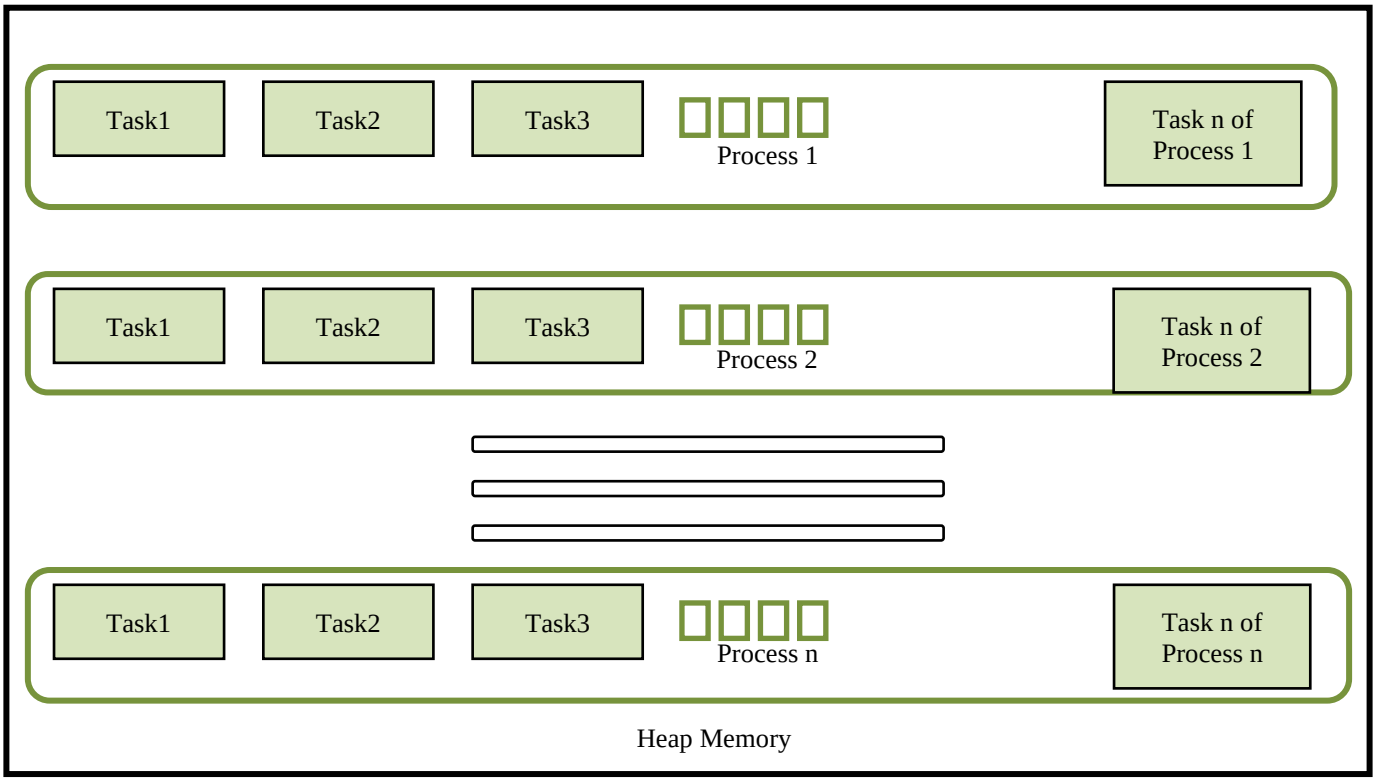

Fig. 2 ICDA heap memory organization

4.3.1. Applicability

The proposed IECSF framework is highly applicable across various domains requiring secure, real-time edge computing. In smart cities, it enhances traffic management, smart grids, and energy security, while in healthcare, it ensures secure remote patient monitoring and medical device protection. Industrial IoT and manufacturing benefit from predictive maintenance and supply chain security, whereas autonomous vehicles leverage their low-latency decision-making and secure V2X communication. The defense sector utilizes it for battlefield surveillance and cybersecurity, and in agriculture, it optimizes smart irrigation and livestock tracking. Smart homes and consumer IoT gain enhanced home automation security and energy efficiency, while financial services use it for fraud detection and ATM security. Retail and supply chain management benefit from secure in-store analytics, shipment tracking, and smart energy; they protect renewable assets and optimize grid performance. IECSF provides a robust, scalable security framework for diverse IoT-driven industries, ensuring efficiency, privacy, and resilience in edge computing environments.

5. Experimental Setup

The evaluation process uses the OPNET [15] network simulator, which can simulate various network components, protocols, nodes, and architectures in a realistic environment. This helps validate measures like operational performance, application troubleshooting, network planning and design, hardware validation, protocol modeling, and traffic modeling. Essential network performance metrics like throughput, communication delays, packet delivery rate, energy consumption, and security are measured during simulations, which can be conducted over different time periods and with varying numbers of nodes. OPNET has a comprehensive library of network components supporting many modern network communication devices, especially in the IoT and WSN categories. A user interface (UI) was created using Visual Studio, with communication scripts coded in C++20.0 [16]. This UI manages loading network environments into OPNET, inputting details of network components, protocols, and security mechanisms, and receiving and plotting simulation results from OPNET. The simulation involves deploying between 100 to 1000 nodes randomly in a 10,000 square meter environment, using nodes like ESP-32 [17], ESP-8266 [18], NodeMCU [19], and LoRa [20] IoT. These simulations run for 168 real-world hours, with nodes having RF ranges from 100 to 1000 meters.

6. Results and Analysis

The simulation process measures important network performance metrics such as throughput, end-to-end delay, latency, packet delivery ratio, average energy consumption, and security for networks with 100 to 1000 nodes in steps of 100. This section analyzes and compares the performance evaluations of existing and proposed methods.

6.1. Throughput

Throughput is a key network evaluation parameter that significantly influences overall user experience. It refers to the rate at which data communication messages are transferred from a source to a destination through a communication channel, commonly measured in bits per second (bps). A good network should support high throughput rates to ensure an improved user experience. Throughput values are logged for different node densities during the experiments, as shown in Table 3. An improvement of 8.56% in throughput is a notable improvement achieved by the proposed IECSF method.

6.2. End-To-End Delay (E2ED)

End-to-End Delay is the total time taken, including Transmission delay, Propagation delay, Processing delay, Queuing delay and jitter for all hops for a data packet to travel from the source to the destination. End-to-End delay is measured in milliseconds (mS) nits. End-to-End delay has a direct impact on several network performance factors, such as user experience, application performance, data synchronization, and overall user experience. A well-designed network infrastructure should minimize this end-to-end delay to prevent communication lags. The recorded end-to-end delay values are provided in Table 4.

Table 3. Throughput

Throughput (kbps)					
Nodes	**MLIEC**	**BREDGE**	**IGADSM**	**NSAFL**	**IECSF**
100	33116	34169	33566	34421	37368
200	31962	33328	32539	33146	36714
300	30783	32180	31690	32047	35440
400	29896	31233	30646	31168	34394
500	28774	30408	29888	30457	33646
600	28129	29067	28582	29254	32502
700	26877	28187	27842	28126	31866
800	25843	27316	26635	27432	30788
900	24884	26341	25875	26302	29736
1000	24147	25249	24696	25127	28890

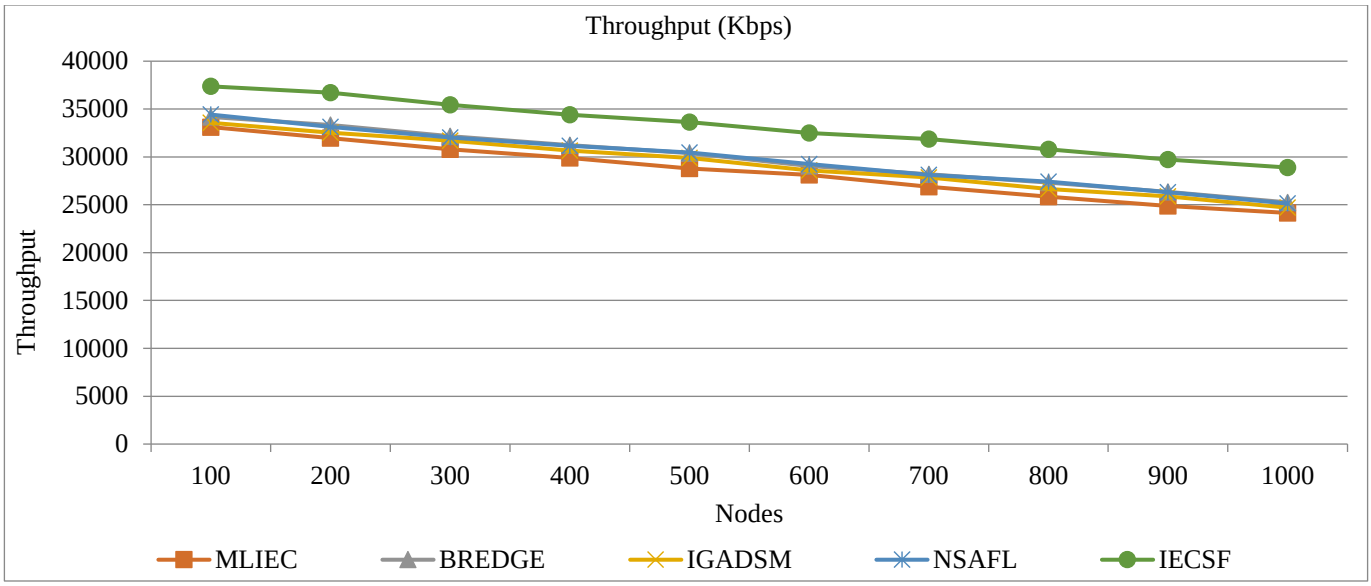

Fig. 3 Throughput

Table 4. End-to-end delay

End-to-End Delay (mS)					
Nodes	**MLIEC**	**BREDGE**	**IGADSM**	**NSAFL**	**IECSF**
100	214	175	197	166	59
200	256	206	235	213	83
300	299	248	266	253	129
400	331	282	304	285	167
500	372	312	331	310	194
600	395	361	379	354	236
700	441	393	406	395	259
800	478	425	450	421	298
900	513	460	477	462	337
1000	540	500	520	504	368

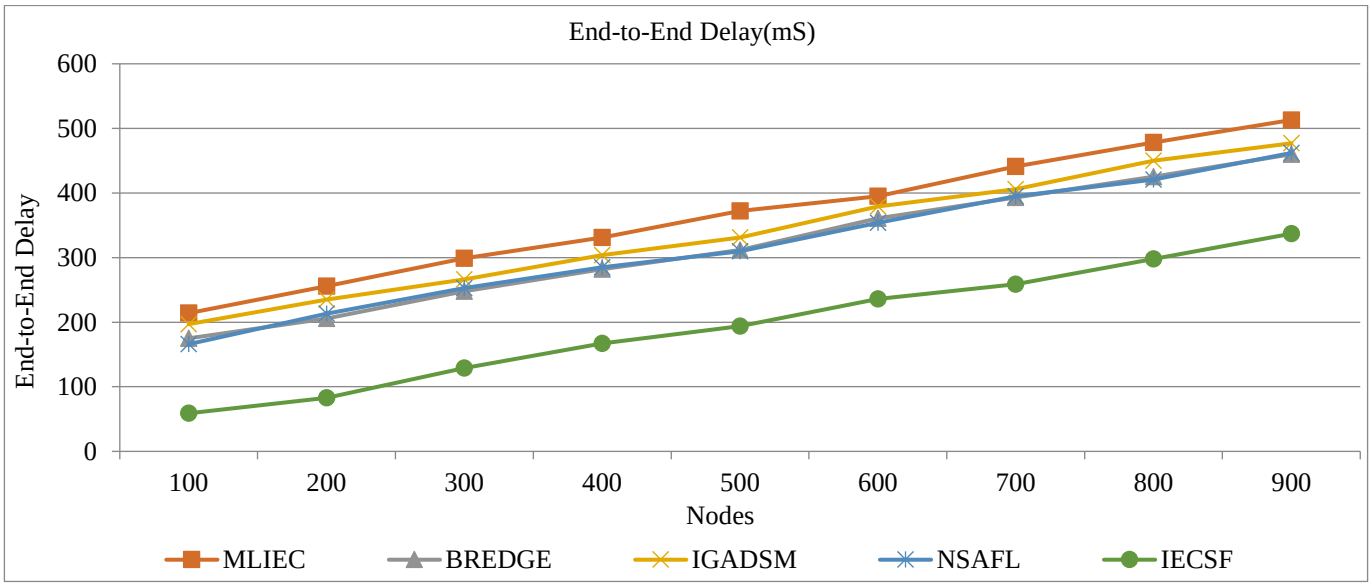

Fig. 4 End-to-end delay

The observed results show that a drop of 36.64 % in the end-to-end delay average is achieved by the IECSF method. The performance rank lineup of the assessed methods is IECSF, BREDGE, NSAFL, IGADSM, and MLIEC, with end-to-end delay averages of 213 mS, 336.2 mS, 336.3 mS, 356.5 mS, and 383.9 mS chronologically listed from the best. A comparison graph for the end-to-end delay of compared methods is given in Figure 4.

6.3. Latency

Latency in an IoT edge network refers to the time delay between when data is generated by IoT devices at the network's edge and when it reaches its destination, typically a central server or cloud service for processing and analysis. Low latency is critical in IoT edge networks for real-time applications like industrial automation, autonomous vehicles, and remote healthcare monitoring. Minimizing latency enhances responsiveness and improves IoT systems' overall efficiency and reliability, ensuring timely data insights and actionable responses. Metered latency measures of the evaluated methods are given in Table 5, and a comparison graph is provided in Figure 5. The results indicate that the IECSF method has achieved a significant average reduction of 36.85 % in end-to-end delay. The performance sequence of analyzed techniques is IECSF, NSAFL, BREDGE, IGADSM, and MLIEC, with the latency averages of 48.5 mS, 76.8 mS, 76.9 mS, 81.6 mS, and 87.7 mS listed from the optimal.

Table 5. Latency

Latency (mS)					
Nodes	**MLIEC**	**BREDGE**	**IGADSM**	**NSAFL**	**IECSF**
100	49	40	45	38	13
200	58	47	54	48	19
300	68	57	61	58	29
400	76	64	69	65	38
500	85	71	76	71	44
600	90	83	87	81	54
700	101	90	93	90	59
800	109	97	103	96	68
900	117	105	109	106	77
1000	124	115	119	115	84

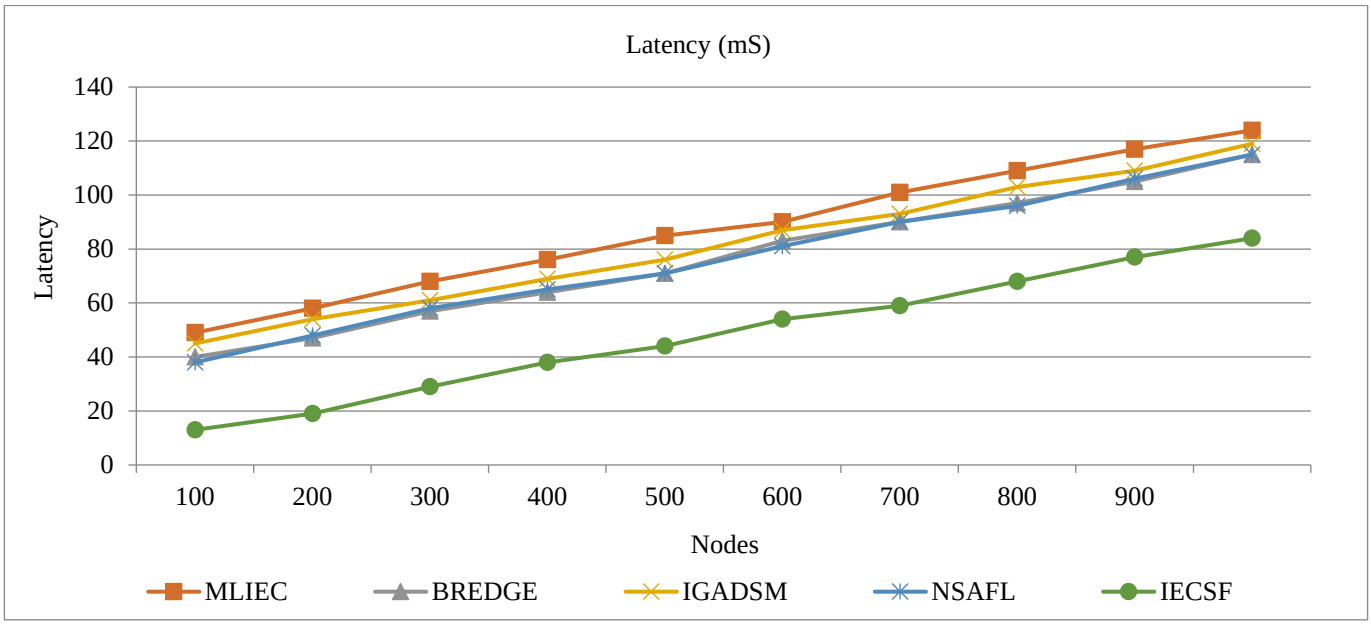

Fig. 5 Latency

6.4. Packet Delivery Ratio

Packet Delivery Ratio (PDR) in IoT edge networks refers to the ratio of successfully delivered data packets to the total number of packets sent by the source. It is a key performance metric that indicates the reliability and efficiency of the network. PDR is calculated using the formula,

$$PDR = \frac{Number\ of\ Packets\ Received}{Number\ pf\ Packets\ transmitted} \times 100.$$

A high PDR signifies that the network is effectively delivering data packets with minimal loss, which is crucial for the reliable functioning of IoT applications. Factors affecting PDR in IoT edge networks include network congestion, signal interference, packet collisions, and the efficiency of routing protocols. Maintaining a high PDR ensures that IoT devices communicate effectively, essential for real-time data processing and decision-making in edge computing environments. PDR for the compared methods for different node densities are charted in Table 6. The highest packet delivery ratio value during the entire simulation process is 99.05%, achieved by the IECSF method. The rank sequence for the performance in terms of PDR is IECSF, NSAFL, BREDGE, IGADSM, and MLIEC, with the PDR averages of 87.8282%, 78.852%, 78.8515%, 77.3885%, and 75.3878%.

Notably, the performance in terms of PDR of NSAFL and BREDGE are nearly identical. The highest PDR average, excluding the proposed method, is 78.852%, achieved by the NSAFL method. The proposed IECSF has achieved 87.8282%, an 11.38% improvement over the next in line. A comparison graph for PDR is provided in Figure 6.

Table 6. Packet delivery ratio

PDR (%)					
Nodes	**MLIEC**	**BREDGE**	**IGADSM**	**NSAFL**	**IECSF**
100	87.77948	90.57063	88.97228	91.23859	99.05011
200	84.72061	88.34142	86.25005	87.85899	97.31657
300	81.59547	85.29845	83.99963	84.94592	93.93963
400	79.24433	82.78827	81.23233	82.61598	91.16703
500	76.27028	80.60147	79.22312	80.73135	89.18433
600	74.5606	77.04693	75.76135	77.5426	86.15197
700	71.24197	74.71434	73.79986	74.55265	84.46614
800	68.50118	72.40561	70.60051	72.71309	81.60873
900	65.95919	69.82121	68.586	69.71783	78.82022
1000	64.00565	66.92668	65.46086	66.6033	76.57776

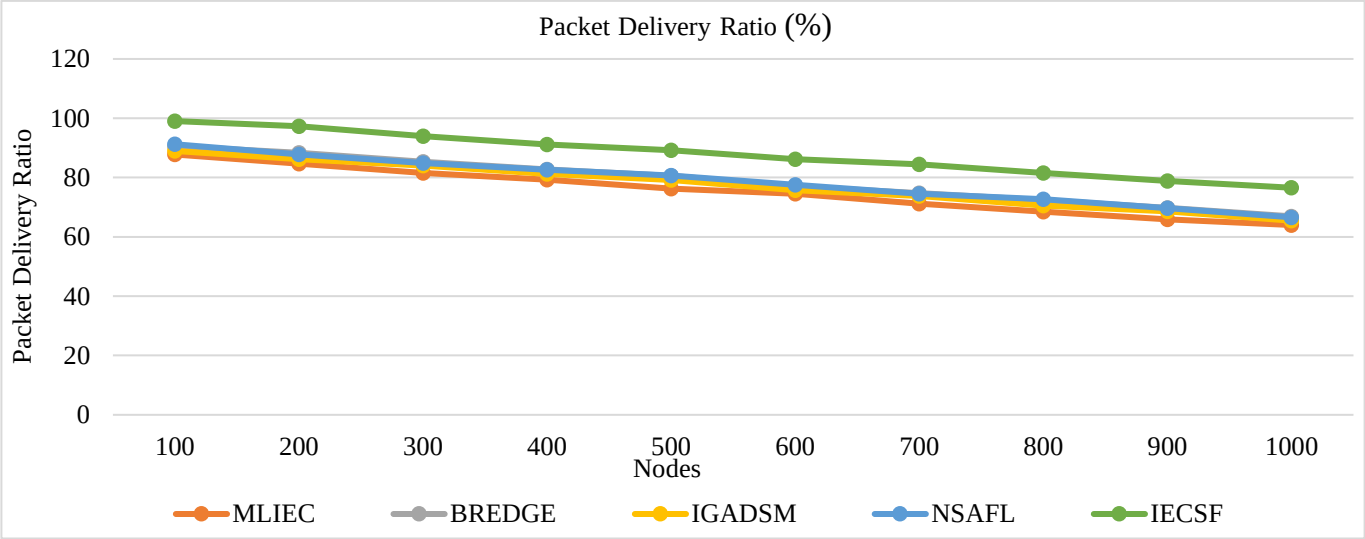

Fig. 6 Packet delivery ratio

6.5. Average Energy Consumption

Average Energy Consumption in IoT edge networks refers to the mean amount of energy consumed by IoT devices, sensors, and edge nodes over a specific period of time. This metric is critical for evaluating the efficiency and sustainability of IoT systems, particularly in environments where devices are battery-powered or where energy resources are limited. Metered average energy consumption during the simulation for the compared methods is enumerated in Table 7. Average energy consumption is typically measured in units such as Joules (J). It is calculated by taking the total energy consumed by all devices and dividing it by the number of devices or the duration of operation.

The performance hierarchy for Average Energy Consumption is led by IECSF with an average of 312.4 uJ, followed by NSAFL with 431 uJ, BREDGE with 450.7 uJ, IGADSM with 467.9 uJ, and MLIEC with 505 uJ. A reduction of 27.52% in average energy consumption is achieved by the proposed IECSF method, highlighting the overall energy efficiency of the proposed method. A comparison graph of average energy consumption values is illustrated in Figure 7.

Table 7. Average energy consumption

Energy (uJ)					
Nodes	**MLIEC**	**BREDGE**	**IGADSM**	**NSAFL**	**IECSF**
100	443	389	413	376	254
200	454	402	420	380	258
300	471	411	434	399	270
400	485	437	447	405	285
500	506	444	463	432	303
600	512	449	464	429	321
700	516	470	487	455	338
800	537	492	509	468	354
900	552	498	515	472	359
1000	574	515	527	494	382

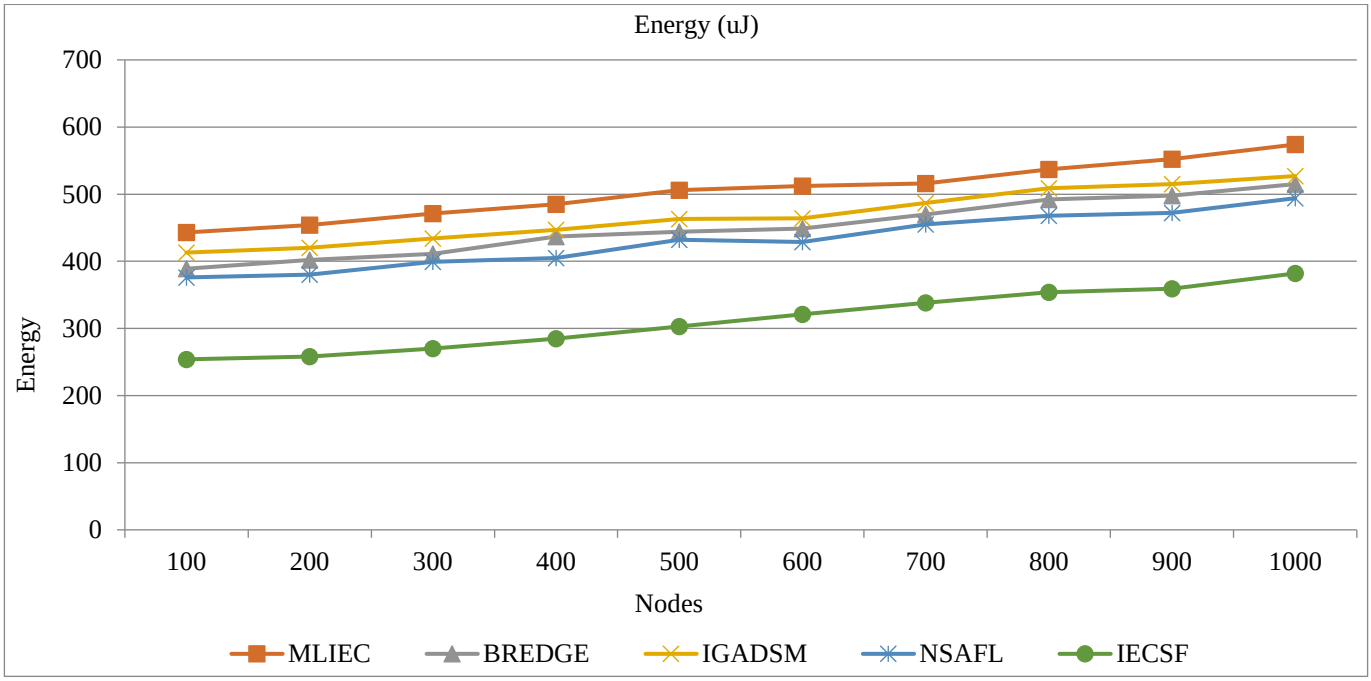

Fig. 7 Average energy consumption

6.6. Security

Security in IoT edge networks refers to the measures and protocols implemented to protect data, devices, and the network infrastructure from unauthorized access, cyber-attacks, and other security threats. Due to the distributed and often resource-constrained nature of IoT devices, securing these networks poses unique challenges. Security for IoT edge networks is measured in OPNET through various metrics and techniques that evaluate the effectiveness of security measures and identify potential vulnerabilities.

The security scores of compared techniques during the evaluation session are logged in Table 8. The order of performance in terms of Security is IECSF, NSAFL, BREDGE, IGADSM, and MLIEC, with the corresponding scores 99.65%, 98.46%, 97.62%, 97.25%, and 96.95% sorted from the best. An improvement of 1.21% achievement in terms of the proposed IECSF method is a deserving augmentation of proposed functional modules. A comparison graph for security scores of compared methods is presented in Figure 8.

Table 8. Security score

Security (%)					
Nodes	**MLIEC**	**BREDGE**	**IGADSM**	**NSAFL**	**IECSF**
100	96.79091	97.57273	97.45454	98.2	99.63568
200	96.88182	97.39091	97.09091	98.38182	99.5706
300	97.06364	97.39091	97.45454	98.47273	99.68893
400	96.88182	97.84545	97.27273	98.56364	99.47

500	97.24545	97.75455	97.45454	98.47273	99.83095
600	96.88182	97.66364	97.45454	98.74545	99.77177
700	96.79091	97.75455	97.09091	98.2	99.58243
800	96.97273	97.48182	97.09091	98.74545	99.71852
900	96.97273	97.75455	97.09091	98.65455	99.69486
1000	97.06364	97.57273	97.09091	98.2	99.60609

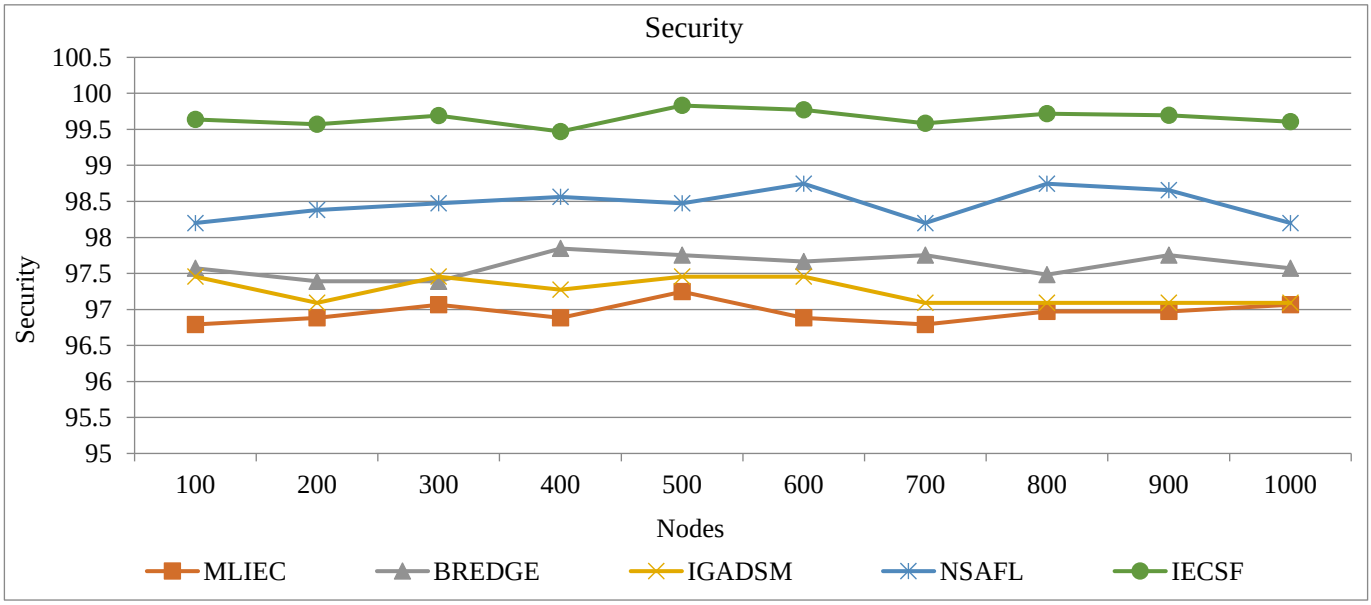

Fig. 8 Security score

7. Conclusion

In conclusion, the proposed IECSF has successfully demonstrated significant advancements in performance metrics such as Throughput, End-to-End Delay, Latency, Packet Delivery Ratio, Average Energy Consumption, and Security within IoT edge networks. Integrating the novel proposed functional modules, namely Least Hop Cluster Manager, Intra Cluster Data Distributor, and Inter-Cluster Data Aggregator, is a vivid accomplishment.

This study has underscored the importance of efficient data processing at the edge, optimizing communication protocols, and enhancing device-to-cloud interactions by evaluating various methodologies and techniques. The findings reveal notable improvements in latency reduction, throughput enhancement, and reliability, paving the way for more responsive and scalable IoT applications. Moving forward, continued research and innovation in this field promise to further elevate the capabilities of IoT edge networks, ensuring they meet the increasingly demanding requirements of modern connected systems.

Source Availability

The complete source code for this work is available on GitHub, and the link will be shared by way of an e-mail request to the authors.

References

[1] Marc Schmitt, "Securing the Digital World: Protecting Smart Infrastructures and Digital Industries with Artificial Intelligence (AI)-Enabled Malware and Intrusion Detection," *Journal of Industrial Information Integration*, vol. 36, pp. 1-12, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[2] Ajibike Eunice Akin-Ponnle et al., "Home Chimney Pinwheels (HCP) as Steh and Remote Monitoring for Smart Building IoT and WSN Applications," *Sensors*, vol. 23, no. 5, pp. 1-26, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[3] Divya Upadhyay et al., "A Linear Quadratic Regression-Based Synchronised Health Monitoring System (SHMS) for IoT Applications," *Electronics*, vol. 12, no. 2, pp. 1-16, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[4] Kamalrulnizam Bin Abu Bakar et al., "A Review on the Immediate Advancement of the Internet of Things in Wireless Telecommunications," *IEEE Access*, vol. 11, pp. 21020-21048, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[5] Jiu Li Chong et al., "Internet of Things (IoT)-Based Environmental Monitoring and Control System for Home-Based Mushroom Cultivation," *Biosensors*, vol. 13, no. 1, pp. 1-24, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[6] Tayyaba Anees et al., "The Integration of WoT and Edge Computing: Issues and Challenges," *Sustainability*, vol. 15, no. 7, pp. 1-27, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[7] Panagiotis Gkonis et al., "A Survey on IoT-Edge-Cloud Continuum Systems: Status, Challenges, Use Cases, and Open Issues," *Future Internet*, vol. 15, no. 12, pp. 1-27, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[8] Arif Ullah, Saman Yasin, Tanweer Alam, "RETRACTED ARTICLE: Latency Aware Smart Health Care System Using Edge and Fog Computing," *Multimedia Tools and Applications*, vol. 83, no. 11, pp. 34055-34081, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[9] Lukman Adewale Ajao, and Simon Tooswem Apeh, "Secure Edge Computing Vulnerabilities in Smart Cities Sustainability using Petri Net and Genetic Algorithm-Based Reinforcement Learning," *Intelligent Systems with Applications*, vol. 18, pp. 1-21, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[10] Chintan Patel et al., "EBAKE-SE: A Novel ECC-Based Authenticated Key Exchange between Industrial IoT Devices using Secure Element," *Digital Communications and Networks*, vol. 9, no. 2, pp. 358-366, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[11] Abdulrahman K. Alnaim, and Ahmed M. Alwakeel, "Machine-Learning-Based IoT-Edge Computing Healthcare Solutions," *Electronics*, vol. 12, no. 4, pp. 1-16, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[12] Kim-Hung Le, Khanh-Hoi Le-Minh, and Huy-Tan Thai, "BrainyEdge: An AI-enabled Framework for IoT Edge Computing," *ICT Express*, vol. 9, no. 2, pp. 211-221, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[13] Haowen Tan, "An efficient IoT Group Association and Data Sharing Mechanism in Edge Computing Paradigm," *Cyber Security and Applications*, vol. 1, pp. 1-6, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[14] Shuai Yan et al., "Node Selection Algorithm for Federated Learning Based on Deep Reinforcement Learning for Edge Computing in IoT," *Electronics*, vol. 12, no. 11, pp. 1-14, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[15] Zhihai Zhuo et al., "Research on Communication Stability of Inter-Cannonball Network Based on OPNET," *Applied Sciences*, vol. 13, no. 7, pp. 1-16, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[16] Cppreference.com, 2019. [Online]. Available: https://en.cppreference.com/w/

[17] Darko Hercog et al., "Design and Implementation of ESP32-Based IoT Devices," *Sensors*, vol. 23, no. 15, pp. 1-20, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[18] Nabil Litayem, and Ahmed Al-Sa'di, "Exploring the Programming Model, Security Vulnerabilities, and Usability of ESP8266 and ESP32 Platforms for IoT Development," *IEEE 3rd International Conference on Computer Systems*, Qingdao, China, pp. 150-157, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[19] NodeMcu - An Open-Source Firmware Based on ESP8266 Wifi-Soc, Nodemcu.com, 2018. [Online]. Available: https://www.nodemcu.com/index_en.html

[20] Aikaterini I. Griva et al., "LoRa-Based IoT Network Assessment in Rural and Urban Scenarios," *Sensors*, vol. 23, no. 3, pp. 1-24, 2023. [CrossRef] [Google Scholar] [Publisher Link]