Journal of Information Systems Engineering and Management
2024, 9(4)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

A Multi-Layered Security Approach: Protecting MQTT Protocols, Enhancing Sensor Data via DWT, and Safeguarding Cloud Data Integrity

N Musrat Sultana[1,2*], K. Srinivas[1]
[1*]Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Hyderabad -500075, Telangana, India.
[2]Department of Computer Science and Engineering, Mahatma Gandhi Institute of Technology, Hyderabad, Telangana -500075, India.
[1]Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation Hyderabad -500075, Telangana, India

ARTICLE INFO

Received: 07 Nov 2024

Revised: 18 Dec 2024

Accepted: 26 Dec 2024

ABSTRACT

The paper examines novel approaches to improve the security, effectiveness, and fault diagnosis of Internet of Things systems, with an emphasis on MQTT protocol security and discrete wavelet transform (DWT) signal processing. Among the main concerns addressed are data integrity problems in cloud storage, security flaws in MQTT-based communications, and noise reduction in various IoT contexts. Lebesgue Hash Message Double Authentication fused Deep Learning (LHMDA-DL) is a hybrid deep learning technique that is suggested to protect MQTT message transport from malicious variations and Denial-of-Service (DoS) attacks. It uses Hellinger Distributed Stochastic Neighbor Embedding for data preprocessing, Gaussian Mixture Models with Lebesgue measures for feature extraction, and Hash Message Double Authentication Rules for secure message transmission over Long Short-Term Memory (LSTM) networks. Through simulation, the efficacy of the framework is confirmed, demonstrating notable gains over current methods in processing time, traffic overhead reduction, message transmission accuracy, and error rate. In order to reduce risks and maintain user data security, sophisticated encryption techniques and hierarchical key management are highlighted as ways for guaranteeing data integrity in cloud storage systems. This thorough investigation shows how deep learning models, IoT security standards, and sophisticated wavelet signal analysis work together to create a strong foundation for safe and effective cloud and IoT operations.

1. INTRODUCTION

1.1. Discrete Wavelet Transform in IoT

The Internet of Things, sometimes known as IoT, is a rapidly growing network of numerous smart devices with a wide range of uses in industries such as manufacturing, transportation, healthcare, and agriculture. IoT networks promise unparalleled coverage, flexibility, and quality of service (QoS). Such an ecosystem can only be achieved by combining many low size, weight, and power (SWaP) components (such relays, sensors, actuators, etc.) across a range of environments. The devices inside the network will use a range of signaling formats to send large amounts of data at the same time. Transmitted signals are dispersed, blocked, distorted by intrinsic interference, and contaminated by noise on their approach to the IoT gateway or the receiving devices. Interference arises from the superposition of signals broadcast simultaneously over many access channels, and noise can be caused by external factors, internal electronics of the devices, or frequency-dependent abnormalities [1, 2, 3].

Denoising is the process of eliminating noise from a signal while preserving the transmitted signal's originality and resolution. Filtering (low or band-pass) is the fundamental denoising technique, although it only removes out-of-band noise [4]. Denoising is a very challenging task since noise has a wide range of origins, features, and natures [3].

The process of dividing a signal into a group of wavelets is known as wavelet analysis. This is achieved via convolution of the signal with an appropriate wavelet function. Wavelet Transform offers high resolution for time-dependent features and frequency-dependent characteristics [3]. The two fundamental features of WT are scaling and shifting [3, 5].

The process of Discrete Wavelet Transform (DWT) involves breaking down a signal into frequency components using multi-rate filter banks. The signal is passed through a low-pass and a high-pass filter, separating it into two sub-bands. The low-frequency sub-band is further decomposed iteratively, resulting in narrower sub-bands. This allows the signal to be analyzed at different resolutions. DWT yields large-magnitude coefficients that represent the significant components of the signal, while small coefficients correspond to noise [3].

The final step in denoising is to reconstruct the signal using the "Inverse DWT (IDWT)" approach [3, 6]. IDWT recovers the signal from the changed detail coefficients by combining the lifting and folding techniques.

1.2. Attacks in IoT Protocol

MQTT's architecture enables reliable real-time messaging for remote nodes with minimal bandwidth and code. It operates on a publish-subscribe model with three primary roles: publisher, subscriber, and broker. MQTT control packets consist of a fixed header, with optional variable headers and payload. Communication is established via TCP, using message types like PINGREQ and PINGRESP. Quality of Service (QoS) levels define message delivery guarantees:

- **QoS 0:** At-most-once delivery (no acknowledgments).
- **QoS 1:** At-least-once delivery (acknowledgment required).
- **QoS 2:** Exactly-once delivery (highest overhead due to a four-step handshake)

IoT systems face a multifaceted threat landscape
Physical Attacks: Direct interference with devices (e.g., node tampering, RF interference, jamming, physical damage, sleep deprivation attacks, malicious code injection).
Network Attacks: Targeting communication infrastructure (e.g., RFID spoofing/cloning, traffic analysis, unauthorized access, denial-of-service (DoS), man-in-the-middle (MiTM), sinkhole attacks).
Software Attacks: Exploiting software vulnerabilities (e.g., viruses, worms, Trojan horses, spyware, malicious scripts, adware).
Encryption Attacks: Compromising data security (e.g., cryptanalysis attacks like ciphertext-only, known-plaintext, chosen-plaintext/ciphertext, and side-channel attacks).

1.3. Vulnerabilities in Large Scale Cloud Storage

The dynamic and flexible Service Level Agreement (SLA)-based negotiated services that provide customers access to practically infinite computer resources make cloud computing appealing [9, 10]. It follows a pay-per-use approach, allowing on-demand, accessible, and adjustable network access to shared resources with less management work and service provider involvement [9, 11].
There are several types of cloud computing: community, hybrid, private, and public. Services are divided into three categories: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). IaaS providers include Windows Azure Virtual Machines, Google Compute Engine, and Amazon Elastic Cloud Compute, which offer computational, storage, and network resources to meet client needs while reducing maintenance expenses [9, 12, 13].
Data owners are hesitant to transfer sensitive data to cloud service providers (CSPs) because of the shared nature of cloud storage and concerns over CSP integrity [9, 14, 15]. Cloud data privacy is a major security concern, worsened by malicious users, leading to issues with data integrity and confidentiality. In Database Management Systems (DBMS), data integrity (completeness, correctness, and consistency) is critical. Problems arise when CSPs cannot guarantee that the information they provide to clients is accurate and complete [9, 16].

Key vulnerabilities in cloud storage include:

- **Inability of CSP:** Potential data loss due to lack of computational capacity, failure to meet user requirements, and absence of a user-friendly data serialization standard.
- **Lack of Physical Scalability:** Inability to seamlessly combine hardware resources, potentially benefiting unauthorized access and alteration.
- **Absence of Performance Monitoring:** Monitoring data stored in a shared public pool may not suit users wanting control over workload distribution.
- **Data Threat:** Inadequate protection strategies from CSPs can lead to data loss or damage.
- **Malicious Vendor:** Lack of transparency and access control can lead to selling user data to third parties.
- **Data Sharing:** While a key cloud feature, resource sharing can violate confidentiality and recovery policies.

2. LITERATURE SURVEY

Encryption cryptography converts messages into an incomprehensible format for unauthorized parties while allowing authorized individuals to decode them. This study focuses on two encryption techniques: symmetric Advanced Encryption Standard (AES) and asymmetric Rivest-Shamir-Adleman (RSA). AES uses a single key for encryption/decryption with key lengths of 128, 192, or 256 bits and a fixed block size of 128 bits. Longer keys enhance

security but increase computational time. RSA, a public key encryption method, supports key sizes up to 2048 bits. To protect diagnostic text data within medical images in IoT healthcare, this study proposes a hybrid model combining AES, RSA, and steganography (2D-DWT-1L or 2D-DWT-2L). The encrypted data is embedded into a cover image to securely conceal sensitive information [23].
Denoising signals in IoT networks is challenging due to node diversity, noise characteristics, and application requirements. A generalized approach combining energy correlation analysis with wavelet packet transform (WPT) is proposed. WPT decomposes signals into high/low-frequency components. Noise is identified, filtered, and signals reconstructed using inverse WPT. Comparing reconstructed energy with the original signal adjusts properties to recreate the signal while eliminating noise. Results show improved error probability by ~3 dB and 7 dB over DWT and WPT methods [3].
DWT denoising involves three steps: forward transformation, reduction of wavelet coefficients, and inverse transformation. Key considerations include wavelet type, thresholding method, and application of thresholds. Among 22 wavelets (Haar, Daublets, Coiflets, Symmlets), the MDL method proved most effective. Thresholding techniques such as TI, level-dependent, and global (hard, soft, garrote, firm) were assessed using white Gaussian noise. Results indicated MDL reliably identifies optimal wavelet types and thresholds, and TI produces consistent results. These DWT techniques outperformed Savitzky-Golay and Fourier transform denoising on IR and HPLC data [5].
MQTT protocol is widely adopted for device-cloud interactions but poses security risks. Few studies offer thorough evaluations of MQTT security, and many reviews are cursory or omit key elements [7].

Location-based services are crucial in IoT. GPS accuracy drops when signals are weak, so opportunistic signals can locate devices. A residual neural network-based approach uses Wi-Fi, geomagnetic, temperature, pressure, humidity, and light signals for localization. Tests with three datasets showed noise-resistance and higher accuracy than existing techniques [24].

DDoS attack detection has mostly targeted traffic from traditional devices. IoT introduces large numbers of unsecured devices, now exploited to generate DDoS traffic in smart home and workplace settings [25].

Cloud computing is popular for cost, reliability, performance, and scalability. However, concerns about data security and accuracy keep some businesses using traditional storage. Data integrity is crucial, and researchers are developing new algorithms for integrity verification [9].

Cloud computing also enables privacy-preserving computation and ensures data integrity across data centers. However, maintaining security and integrity remains challenging. Sensitive information must be processed using privacy-preserving and integrity-protecting methods. Recent advances include non-intrusive person recognition using face recognition for privacy preservation [26].

3. PROBLEM STATEMENT

Key security challenges in MQTT-based IoT systems include:

- **Unauthorized Access:** Exploitation of servers to post/subscribe using wildcards (#, +), accessing sensitive system information ($SYS/#) and compromising data.
- **Risks to Configuration and Personal Information:** Plaintext transmission of identifiers (names, emails) enables spear-phishing. Displayed device configurations allow attackers to manipulate functionality.
- **Poor Credentials:** Weak passwords enable dictionary attacks, leading to unauthorized access, data modification, or exfiltration.
- **Unsecured Transmission:** Unencrypted data is vulnerable to sniffing and MiTM attacks, compromising confidentiality, integrity, and authenticity.
- **Denial-of-Service Attacks:** Susceptible to TCP-based attacks (SYN flooding), protocol-specific attacks (SlowITe, SlowTT), QoS exploitation (especially level 2), and resource exhaustion via large payloads or Will messages.

4. METHODOLOGY

To protect MQTT messages against DoS attacks in DWT-based IoT transactions, we propose the **Lebesgue Hash Message Double Authentication fused Deep Learning (LHMDA-DL)** hybrid technique. It preprocesses data to discard extraneous instances, extracts features using a Lebesgue measure Gaussian Mixture, and secures transmission via a hybrid LSTM and Hash Message Double Authentication Rules network.

4.1 Dataset Description

The MQTTset dataset simulates a real industrial IoT network using the MQTT protocol. An Eclipse Mosquitto broker connects eight sensors with different IPs, monitoring parameters like Temperature (T), Light Intensity (LI), Humidity (H), CO-Gas (CO), Motion (Room 1 & 2), Smoke (S), Door Lock (DL), Fan Speed (FS), and Fan Speed Controller (FSC), each with unique message frequencies.

- **PCAP files:** raw, authentic, and attack traffic (DoS, Flooding, Bruteforce, SlowlTe, distorted).
- **CSV files:** authentic and attack data (Bruteforce, SlowlTe, Flooding, DoS, corrupted).
- **Final datasets:** Train70, Test30, and their reduced/augmented versions.

Data is captured via the broker, processed with *tshark* into CSV files, and formatted into integrated, reduced, or augmented sets for training and testing.

4.2 Hellinger Distributed Stochastic Neighbor Embedding – Based Preprocessing

Acceptable data in various patterns must be processed before use to protect MQTT messages from DoS attack variants. The behavior of 25,000 samples is examined using the MQTT protocol. Superfluous instances are removed using **Hellinger Distributed Stochastic Neighbor Embedding** (H-DSNE) to minimize dataset dimensions. H-DSNE embeds high-dimensional data in low-dimensional spasce so that dissimilar objects are modeled by distant locations, and similar objects (e.g., sensors) by adjacent locations. The raw data is first prepared as a network sample vector matrix:

$$NS = [F_1D_1\ F_1D_2 \ldots F_1D_n\ F_2D_1\ F_2D_2 \ldots F_2D_n \ldots \ldots \ldots \ldots F_mD_1\ F_mD_2 \ldots F_m\ ] \quad (1)$$

Where feature set *F* and data subset *D* build the network sample vector matrix *NS*. H-DSNE assesses probabilities proportional to object similarity (e.g., temperature and humidity sensors). Because both subscribe to the humidity topic, they receive the same data and operate in tandem for humidity control.

$$Prob_{j|i} = \frac{\exp[-\frac{(S_i-S_j)^2}{2\sigma_i^2}]}{\sum_{k\neq i}\exp[-\frac{(S_i-S_k)^2}{2\sigma_i^2}]} \quad (2)$$

According to equation above, the conditional probability "Prob(j|i)" that "Sj" would choose "Si" as its neighbor if neighbors were chosen according to their probability density under Gaussian is the similarity between data points or sensors "Si" (i.e., humidity sensor) and "Sj" (i.e., temperature sensor).

$$Prob_{i|j} = \frac{Prob_{i|j}+Prob_{j|i}}{2N} \quad (3)$$

The Hellinger distance, which quantifies the similarity between two probability distributions and is expressed mathematically as follows, is then minimized to find the locations of data points or sensors.

$$DRNS = H^2(S_i, S_j) = \frac{1}{2}\int(\sqrt{S_i(F)} - \sqrt{S_j(F)})^2\alpha\,(dF) \quad (4)$$

The probability distribution of two sensors (temperature and humidity) being positioned is represented by the symbols "Si (F)" and "Sj (F)" in equation above. When tracking and location data are supplied, the frequency of data transmission would be significantly higher. A map that takes into account the similarities between high dimensional inputs is produced by using Hellinger distance in relation to the neighboring points, which further reduces the dimensionality. The Hellinger Distributed Stochastic Neighbor Embedding-based Preprocessing pseudo code representation is shown below.

```
Input: Dataset ‘$DS$’, Features ‘$F = \{F_1, F_2, \ldots, F_m\}$’, Data ‘$D = \{D_1, D_2, \ldots, D_n\}$’, Sensor ‘$S = \{S_1, S_2, \ldots, S_s\}$’
Output: dimensionality reduced network samples ‘$x \in DRNS$’
1: Initialize ‘$m$’, ‘$n$’, Network Samples ‘$NS$’, ‘$s$’ 2: Begin
4: Formulate sample vector matrix as given in (1)
5: Measure probability of object similarity positioned in a room as given in (2) 6: Evaluate similarity of data point or sensors as given in (3)
7: Measure Hellinger distance with which similarity between two probability distributions is quantified as given in (4)
8: Return dimensionality reduced network samples ‘$DRNS$’ 9: End for
10: End
3: For each Dataset ‘$DS$’ with Features ‘$F$’, Data ‘$D$’, Sensor ‘$S$’ and Network Samples ‘$NS$’
```

Algorithm - Hellinger Distributed Stochastic Neighbor Embedding-based Preprocessing

As stated in the technique above, two different functions are used in order to reduce the dataset's dimensions for safe data exchange in DWT signal based IoT and eliminate superfluous instances from the MQTTset dataset. The first step is to perform dimension reduction using the raw data that was collected as input. The Distributed Stochastic Neighbor Embedding function is used in situations where distant points have a high possibility of modeling distinct things, but adjacent points process comparable items or sensors, eliminating duplicate instances. The similarity between two objects or sensors is then further assessed by using Hellinger distance, which further reduces the dimensionality. This in turn protects the MATT messages from identified DoS attack variations in IoT even at an early stage, hence reducing the training time significantly (temperature and humidity sensors, for instance, have a strong correlation with one another).

4.3 Lebegue measure Gaussian Mixture based Feature Extraction

Local neighborhood histograms are measured for each target feature from the DRNS. These histograms capture quantitative neighborhood features. Gaussian Mixture Models (GMMs) represent these histograms efficiently for similarity assessment. The probability density Prob(F) of a random feature is modeled by the GMM. The Lebesgue measure identifies the fittest GMM, determining the likelihood Likelihood(F) of extracting the most relevant feature (closer to 1 indicates higher likelihood). This process extracts 18 relevant features from the MQTTset (e.g., TCP flags, time_delta, MQTT msgtype, QoS, retain).

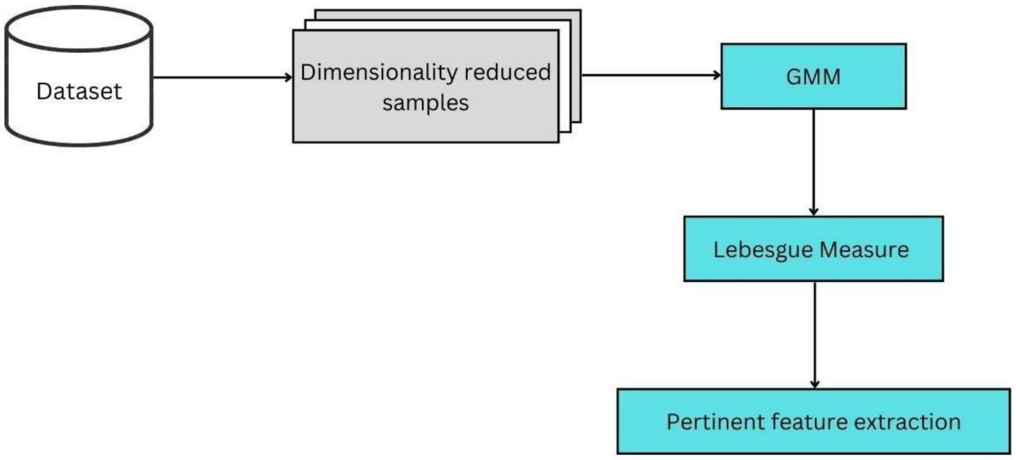

Figure - Structure of Lebesgue measure Gaussian Mixture based Feature Extraction

The purpose of using GMMs is to represent local neighborhood histograms, as they are faster and easier for similarity assessment to extract relevant characteristics.

$$Pr(F)=\Sigma wi*N(F|\mu i,\sigma i) \quad (5)$$

The probability density Prob(F) is calculated by considering random feature F, number of Gaussian components m, weight wi, mean μi, and standard deviation σi. The Lebesgue measure determines the fittest GMM:

$$D=CM2\log(0.5) \quad (6)$$

$$Likelihood\ (F) = \exp(\Delta 2D) \quad (7)$$

The Lebesgue correlative measure "$CM2$" is used to determine the likelihood of deriving the most relevant feature "$Likelihood\ (F)$" from the a fore mentioned equations. This makes it evident that the likelihood has a range of "[0,1]" for any "Δ." The feature being retrieved is more likely the higher the likelihood, and vice versa. The table lists the 18 features that were retrieved from the MQTTset dataset using the Gaussian Mixture and Lebesgue measure.

4.4 Selecting Features using HMDAR-LSTM hybrid network for secure MQTT message transmission from detected DoS variants

The most difficult jobs for secure MQTT message transmission from detection DoS variations are obtaining dimensionality reduced network samples with relevant features, then extracting the most valuable network features. To address this, a number of unique network flow characteristics were used. HMDAR-LSTM (Hash Message Double Authentication Rules and Long Short-Term Memory), a deep learning-driven hybrid model, is suggested to choose selective features and secure MQTT message delivery from identified DWT signal based IoT variants. Features used to generate the Hash Message Authentication Code are chosen from the network trace dataset after relevant features have been retrieved. First, a feature selection model based on Hash Message Double Authentication Rules is used. Devices get necessary features (Connect, ConnectAck, ConnectRate, PublishMessage, ConnAct, DisconnectReq) from network traffic during this process, in addition to the extracted characteristics listed in Table.
The proposed work also considers network traffic features Connect and ConnectAck of CONNECT and CONNACK messages for training. Two variables, Connection Message Ratio (CMR) and Connection Acknowledgement Message Ratio (CAMR), are obtained.

$$CMR = \frac{N_{connect}}{N} \quad (8)$$

The Connection Message Ratio (CMR) is the ratio of the count of connect feature (Nconnect) to total MQTT messages (N).

$$CAMR = \frac{N_{conAck}}{N} \quad (9)$$

The Connection Acknowledgment Message Ratio (CAMR) is the ratio of connection acknowledgement messages from the MQTT broker (NConnAck) to total MQTT messages (N).

HMDAR is then used by the HMAC inference engine to assess each MQTT client's anomaly measurement. The output variable anomaly assesses the message's authenticity using CMR and CAMR. The MQTT broker executes HMDAR if anomalies are found; otherwise, the subscriber receives standard data packets. The HMAC inference engine creates authentication rules as IF-THEN statements:

- IF CMR = LOW and CAMR = LOW → anomaly = Normal
- IF CMR = LOW and CAMR = MEDIUM → anomaly = Abnormal
- IF CMR = HIGH and CAMR = MEDIUM → anomaly = Attack

Hash Message Authentication Code Double Authentication rules are developed based on the above triplet:

$$HM(SK, NS) = Hash((SK' \oplus OPAD)||Hash((SK' \oplus IPAD)||NS)) \ (10)$$

$$HMAC(SK, NS) = Hash((SK' \oplus OPAD)||Hash((SK' \oplus IPAD)||NS)) \ (10)$$

$$SK' = \begin{cases} Hash(SK), \text{if } SK \text{ is larger than block size} \\ SK, Otherwise \end{cases} \quad (11)$$

The cryptographic hash function "Hash" is applied to network samples "NS" using secret key "SK" with outer (OPAD) and inner padding (IPAD) for double authentication. The HMDAR-LSTM structure is depicted in Figure.

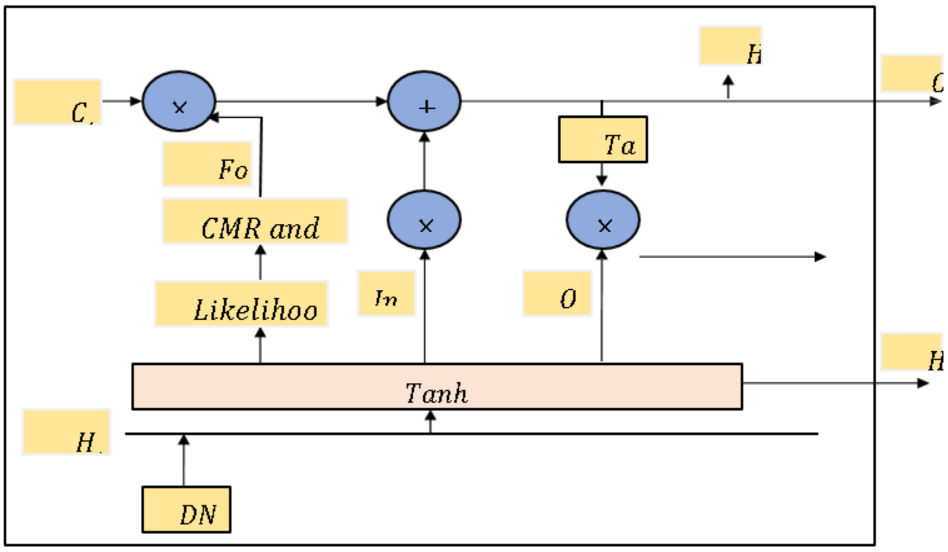

Figure - Structure of Hash Message Double Authentication Rules via Long Short

Term Memory model

Determining which features the model will keep in the cell state and which will be discarded is the first stage in the HMDAR-LSTM process for protecting MQTT messages against identified DoS variants in DWT signal-based IoT data, as seen in the image. The forgetting gate formula below helps make this decision. The "UFor" function receives as input the features retrieved and selected using the Lebesgue measure and Gaussian Mixture with Hash Message Double Authentication criteria. Through weight "WMFor" and bias "BFor," the forget gate reads "UFor" and "H(t-1)" and outputs a value between 0 and 1 in cell state "C(t-1)" (i.e., 1 denotes retain and 0 denotes discard).

$$For_t = \tanh x\,(WM_{For}DRNS_t + U_{For}H_{t-1} + B_{For}) \qquad (12)$$

The second stage is to use the forget gate results to determine how much new information should be added to the cell state. The information that has to be updated is determined by the hyperbolic tangent function, which is expressed mathematically as follows.

$$In_t = \tanh x\,(WM_{In}DRNS_t + U_{In}H_{t-1} + B_{In}) \qquad (13)$$

The hyperbolic tangent function then produces the update vector shown below.

$$Out_t = \tanh x\,(WM_{Out}DRNS_t + U_{Out}H_{t-1} + B_{Out}) \qquad (14)$$

$$Cell_t = \tanh Cell\,(WM_{Cell}DRNS_t + U_{Cell}H_{t-1} + B_{Cell}) \qquad (15)$$

The two components are then merged to create the cell state update shown below using the two resultant values mentioned above.

$$c_t = For_t \odot c_{t-1} + In_t \odot Cell_t \qquad (16)$$

Lastly, the output of the cell is represented by the output gate in equation. The cell state is processed using a hyperbolic tangent function to produce a value between "-1" and "1," as shown below, after first evaluating the cell portion that needs to be exported.

$$H_t = Out_t \odot \tanh H\,(C_t) \qquad (17)$$

The input vector is obtained from the dimensionality reduced network samples (DRNS) using the equations above. Activation vectors for the forget gate, input gate, output gate, and cell input gate are denoted as "Fort," "Int," "Outt," and "Cellt," respectively. The hyperbolic tangent function "tanh" activates these vectors according to the gates' operation. The weight matrix (WM) and bias vector (B) are learned during training. Double authentication ensures message transmission protection, securing MQTT messages (DRNS) from identified DoS variants in DWT signal-based IoT data. The following pseudo code represents HMDAR-LSTM for protecting MQTT messages using double authentication.

```
Input: Dataset 'DS', Features '$F = \{F_1, F_2, \dots, F_m\}$', Data '$D = \{D_1, D_2, \dots, D_n\}$', Sensor '$S = \{S_1, S_2, \dots, S_s\}$'
Output: Computationally efficient secure message transmission with detected DoS variants
Step 1: Initialize dimensionality reduced network samples '$DRNS$', '$m$', '$n$', '$s$', count of the connect feature '$N_{connect}$', total number of MQTT messages '$N$', connection acknowledgement messages from the MQTT broker
Step 2: Begin
Step 3: For each Dataset '$DS$' with Features '$F$', Data '$D$', Sensor '$S$' and dimensionality reduced network samples '$DRNS$'
//Feature extraction
Step 4: Formulate Gaussian mixture model to extract the dimensionality reduced features as given in equation (5)
Step 5: Obtain the fittest GMM using the Lebesgue measure as given in equations (6) and (7)
Step 6: If '$Likelihood\ (F) > 0.5$'
Step 7: Return features extracted '$FE$'
Step 8: End if
Step 9: If '$Likelihood\ (F) \leq 0.5$'
Step 10: Non trivial features
Step 11: Go to step 27
Step 12: End if
//Feature selection
Step 13: Evaluate Connection Message Ratio as given in equation (8)
Step 14: Evaluate Connection Acknowledgement Message Ratio as given in equation (9)
Step 15: If '$CMR = LOW\ and\ CAMR = LOW$'
Step 16: Go to step 18
Step 17: End if
Step 18: Formulate Hash Message Authentication Code Double Authentication rules as given in equations (10) and (11)
Step 19: If '$CMR = LOW\ and\ CAMR = MEDIUM$'
Step 20: Go to step 18
Step 21: End if
Step 22: If '$CMR = HIGH\ and\ CAMR = MEDIUM$'
Step 23: Go to step 18
Step 24: End if
//Secure message transmission
Step 25: Formulate forgetting gate as given in equation (12)
Step 26: Update the information according to the resultant forgetting gate activation vector as given in equation (13)
Step 27: Obtain output gate's activation vector and cell input gate's activation vector as given in equations (14) and (15)
Step 28: Update cell state as given in equations (16) and (17)
Step 29: End for Step 30: End
```

Journal of Information Systems Engineering and Management
2024, 9(4)
e-ISSN: 2468-4376
https://www.jisem-journal.com/
Research Article

Algorithm - Hash Message Double Authentication Rules and Long Short-Term Memory
According to the algorithm, relevant characteristics are first retrieved using the Gaussian Mixture Model with dimensionality reduced network samples and associated MQTT client or sensor data packets. High-likelihood features are then extracted using the Lebesgue measure and the fittest GMM. Based on the Connection Message Ratio and Connection Acknowledgment Message Ratio, the most selective features are chosen. MQTT messages from DoS variations are then subjected to Hash Message Authentication Code Double Authentication rules. Finally, message transmission protection is ensured using the hyperbolic tangent function.

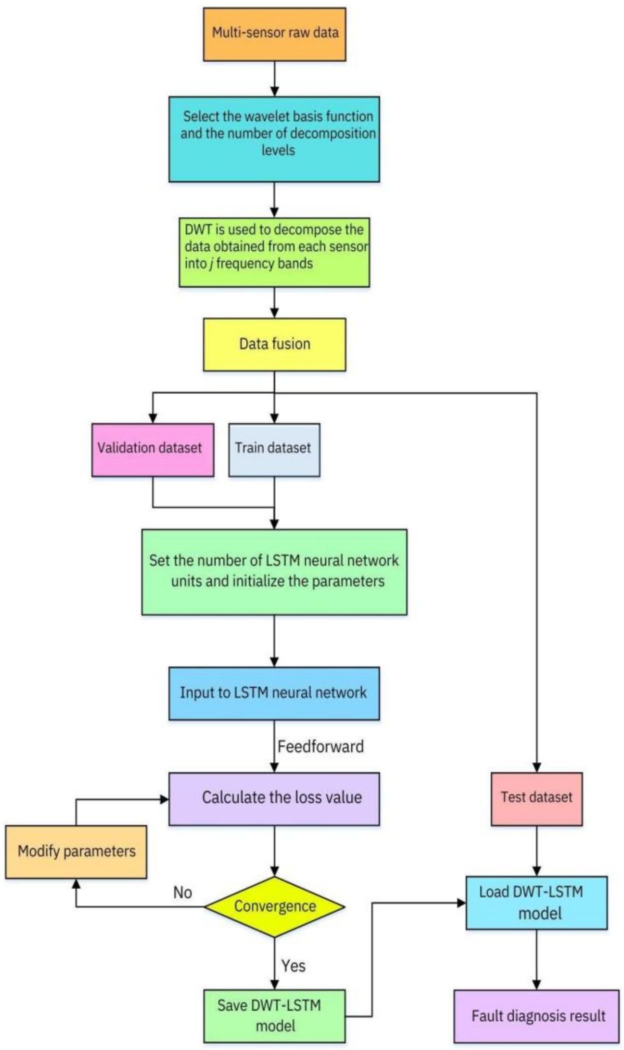
Multi-sensor raw data
Select the wavelet basis function and the number of decomposition levels
DWT is used to decompose the data obtained from each sensor into j frequency bands
Data fusion
Validation dataset
Train dataset
Set the number of LSTM neural network units and initialize the parameters
Input to LSTM neural network
Feedforward
Calculate the loss value
Modify parameters
No
Convergence
Yes
Save DWT-LSTM model
Test dataset
Load DWT-LSTM model
Fault diagnosis result

Figure – Fault Diagnosis using DWT-LSTM model

This Figure outlines a workflow for fault diagnosis using a DWT-LSTM model. Raw data from multiple sensors (vibration, temperature, pressure) is processed using wavelet decomposition, selecting a suitable wavelet basis and decomposition levels. DWT decomposes the data into frequency bands, capturing both time and frequency information. Data fusion combines information from different sensors into a unified format, then split into training and validation datasets.
The fused data is fed into an LSTM network, initialized with specified units and parameters. During training, the network processes input data, calculates loss by comparing predictions with actual outcomes, and adjusts parameters iteratively until convergence. The trained model is then saved.

Finally, the trained DWT-LSTM model is tested on a separate dataset to detect anomalies and provide fault diagnosis. This approach combines DWT's frequency-domain analysis with LSTM's time-series learning, making it effective for diagnosing faults in dynamic systems.

4.5 Key management strategies concerning cloud storage level

To enhance cloud data security, a technique combining encryption and data distribution is proposed. Key management strategies include:

- **Hierarchical Key Technique:** Uses secret sharing and key hierarchy derivation for improved key security.
- **Private Key Update Technique:** Identity-based encryption updates keys for non-revoked users, simplifying PKI.
- **Key Separation Technique**: Preserves privacy for shared sensitive data.
- **Attribute-based Encryption (ABE) Key Technique:** Achieves semantic security for confidentiality without revealing decryption keys.
- **Multiple Key Technique:** Uses k-NN query-based methods where Data Owner (DO) and users hold distinct keys.

The data integrity methodology involves three actors: Data Owner (DO), Cloud Service Provider (CSP), and optional Third-Party Auditor (TPA). It consists of three phases:

- **Data Processing Phase:** Files are split into blocks, encrypted, digests are created, keys are generated, and blocks are signed before being sent to CSP.
- **Acknowledgment Phase (Optional):** CSP acknowledges receipt, though often omitted to reduce overhead.
- **Integrity Verification Phase:** DO/TPA sends a challenge to CSP, which responds with proof metadata. TPA/DO verifies integrity, and the result is sent to DO.

4.6 Stages of the data integrity methodology

In cloud storage, data integrity ensures accuracy and consistency, giving customers confidence in outsourcing data. This design involves three actors: the Data Owner (DO), Cloud Storage/Service Provider (CSP), and an optional Third-Party Auditor (TPA) [9, 34]. The DO creates data and uploads it to cloud storage. The CSP provides Infrastructure as a Service (IaaS). The TPA verifies the accuracy and integrity of outsourced data, reducing the DO's computational and transmission overhead [9, 35, 36]. Sometimes, the DO verifies data integrity without TPA. The following describes the three stages of a data integrity approach.

4.6.1 Data processing phase: Data files are divided into blocks [37], encrypted [38], hashed, masked [39], keys are generated [40], and encrypted blocks signed [41]. Encrypted or obfuscated data is then sent to cloud storage.
4.6.2 Phase of Acknowledgment: Optional but important if CSP unintentionally deletes data or hides data loss [39]. Most research omits this phase to reduce computational overhead.
4.6.3 Phase of integrity verification: DO/TPA sends a challenge to CSP, which replies with metadata for verification. If verification is completed, the audit result is sent to DO.

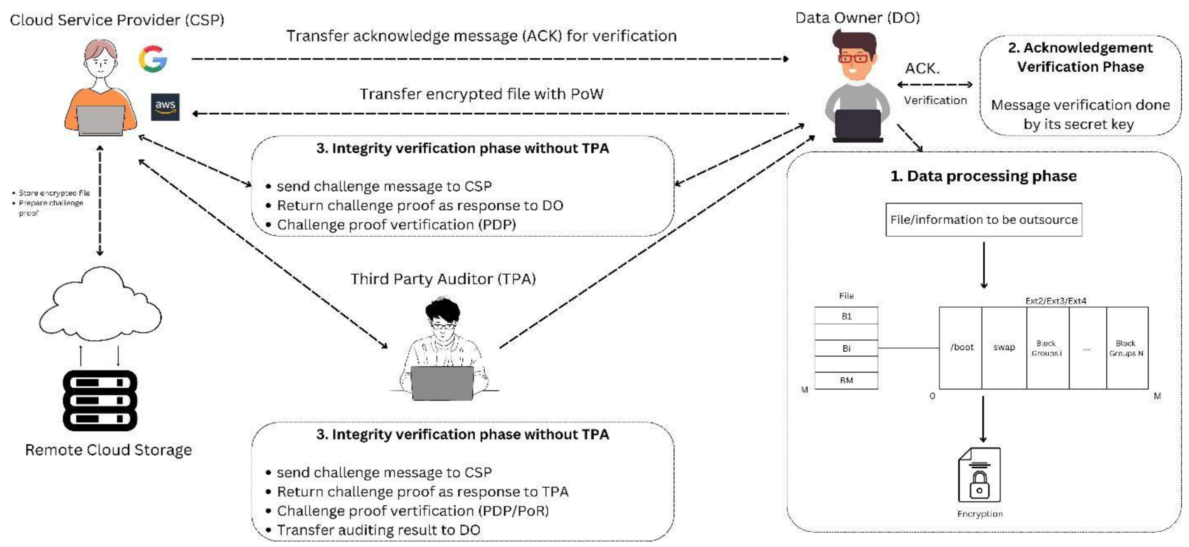
Cloud Service Provider (CSP)
Transfer acknowledge message (ACK) for verification
Data Owner (DO)
aws
Transfer encrypted file with PoW
ACK.
Verification
2. Acknowledgement Verification Phase
Message verification done by its secret key
3. Integrity verification phase without TPA
• send challenge message to CSP
• Return challenge proof as response to DO
• Challenge proof verification (PDP)
• Store encrypted file
• Prepare challenge proof
1. Data processing phase
File/information to be outsource
Third Party Auditor (TPA)
File
B1
Bi
BM
M
Ext2/Ext3/Ext4
/boot
swap
Block Groups i
...
Block Groups N
0
M
Encryption
Remote Cloud Storage
3. Integrity verification phase without TPA
• send challenge message to CSP
• Return challenge proof as response to TPA
• Challenge proof verification (PDP/PoR)
• Transfer auditing result to DO

Figure – Cycle of Data Integrity Method

5 RESULTS AND DISCUSSION

5.2 Security performance evaluation in relation to processing time and traffic overhead

This section measures the security measure in terms of overhead and processing time. The following is a mathematical expression for the processing time.

$$PT = \sum_{i=1}^{N} NS_i \cdot Time[Pub + Sub] \qquad (18)$$

The processing time (PT) is calculated using equation (18) above, accounting for the network samples (NSi) used for simulation and the time spent on publishing (Pub) and subscribing (Sub), respectively. Using the suggested LHMDA-DL approach and the current approaches [42] and [43], Table shows the processing time results by changing the values in equation (18).

Table - Processing time using DWT signal based IoT Monitoring Systems, SEC-RMC, LHMDA-DL, and Deep Learning

Network samples	**Processing time (ms)**		
	LHMDA-DL	**SEC-RMC**	**Deep Learning and DWT signal based IoT Monitoring System**
2500	875	1075	1375
5000	995	1135	1425
7500	1035	1235	1565
10000	1123	1345	1735
12500	1235	1515	1825
15000	1115	1435	1715
17500	1035	1315	1635
20000	1000	1225	1525
22500	985	1145	1415
25000	1035	1215	1535

The total processing time is recorded on both the Publisher and Subscriber sides of MQTT nodes, measured from message publication by the MQTT client to receipt by the subscriber. Processing occurs in two stages: the publication stage (client to broker) and the subscription stage (broker to recipient client). Experiments [42, 43] show the LHMDA-DL technique is faster than traditional methods because relevant features are first extracted using the Lebesgue measure Gaussian Mixture, then the most selective features are obtained based on Connection Message Ratio and Connection Acknowledgment Message Ratio. This increases overhead to achieve security but provides protection against DoS attacks. The figure shows total processing times for traditional approaches [42, 43] and the proposed LHMDA-DL method.

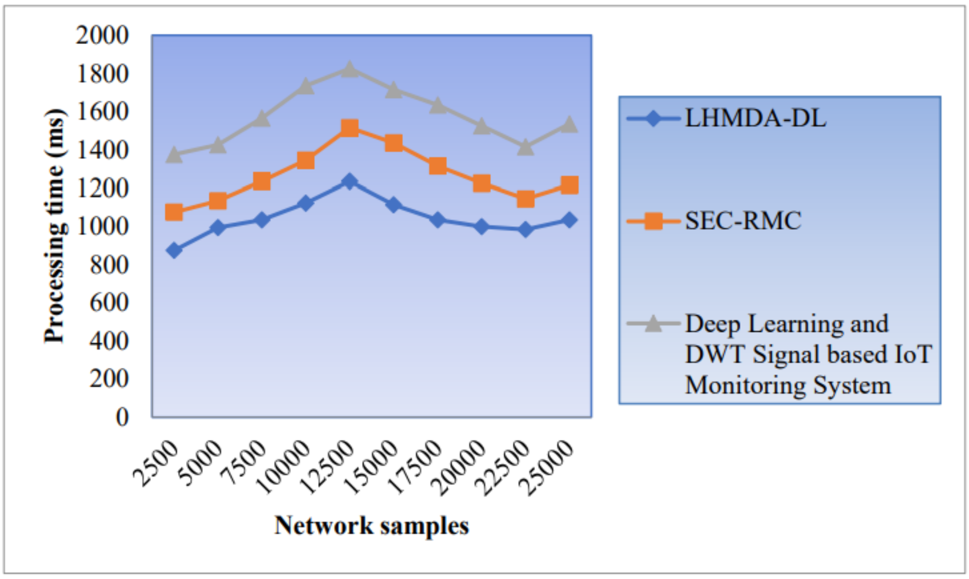

Figure – Graphical Representation of Processing time

MQTT requires less processing time as it lacks default security options, though both publishing and subscribing times are measured. Message size varies with the MQTT payload. Processing time per message is set for both publisher and subscriber, as the publisher uses the Lebesgue measure to select the appropriate GMM for secure transmission. Publishing messages takes longer than subscribing since the publisher ensures security. The LHMDA-DL approach reduced processing times by 17% compared to [42] and 34% compared to [43].

5.3 Traffic Overhead

This section discusses the traffic overhead, which is related to the recommended techniques for enhancing MQTT interaction security. The following is a mathematical formulation of the traffic overhead.

$$TO = \sum_{i=1}^{N} NS_i * Mem\ [Pub + Sub] \qquad (19)$$

The traffic overhead (TO) is calculated using equation (19) above, accounting for the network samples used in the simulation process (NSi) and the memory used by the respective MQTT clients for publishing (Pub) and subscribing to the messages (Sub). Kilo Bytes (KB) are used as the unit of measurement. By replacing the values in equation (19) with the suggested LHMDA-DL approach and the current methods [42] and [43], Table shows the traffic overhead results.

Table - Utilizing LHMDA-DL, SEC-RMC, and Deep Learning and DWT signal based IoT Monitoring Systems to monitor traffic overhead

Network samples	**Traffic overhead (KB)**		
	LHMDA-DL	**SEC-RMC**	**Deep Learning and DWT signal based IoT Monitoring System**
2500	125	175	225
5000	155	215	255
7500	175	235	285
10000	200	255	315
12500	225	285	335
15000	185	250	300
17500	155	225	275
20000	195	245	260
22500	225	295	315
25000	250	300	335

As previously mentioned, an excessive MQTT data packet density results in little control overhead. The traffic overhead of the suggested approach was calculated using the full size of the data packet. The number of interacting data packets in the MQTT network is determined by the features that are chosen; the features that are chosen will increase both the number of data packets and the size of the data. This is due to the fact that obtaining the most relevant and selected characteristics will also significantly lower the traffic overhead.

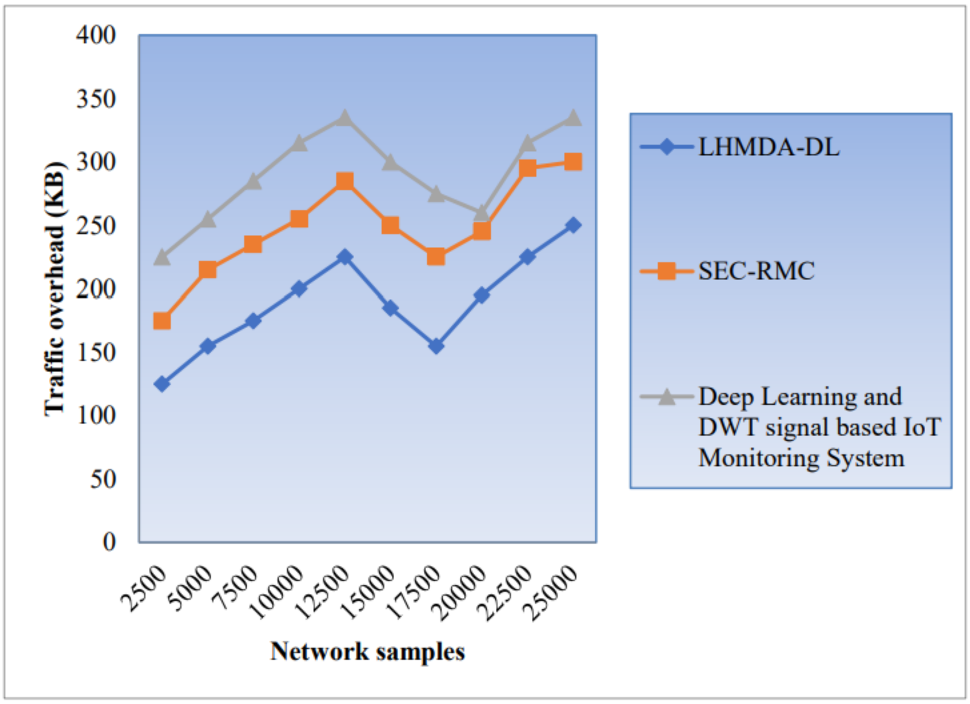

Figure - Graphical Representation of Traffic Overload

The suggested solution increases total data packet size linearly with network samples, but traffic overhead decreases as samples grow, demonstrating higher security. Using 2500 samples, the method caused 125KB traffic overhead per communication, compared to 175KB and 225KB for [42] and [43]. The approach adds some overhead but reduces overall MQTT traffic, confirmed on both publisher and subscriber sides. HMDAR uses Connection Message Ratio and Connection Acknowledgment Message Ratio to prevent DoS attacks. Traditional methods [42, 43] have higher overhead due to lack of security. LHMDA-DL reduces traffic overhead by 24% and 35% compared to [42] and [43], respectively.

5.4 Performance evaluation of the method's effectiveness in terms of message transmission error rate and accuracy

This section presents the message transmission accuracy for evaluating the method's efficacy and efficiency. The correctness of the message transmission is assessed as follows.

$$MTA = \sum \frac{NS_{recvd}}{NS_{sent}} * 100 \tag{20}$$

The message transmission accuracy (MTA) is calculated using equation (20) above, accounting for the network samples delivered (NSsent) and received (NSrecvd). The percentage (%) is used to measure it. By replacing the values in equation (20) with the suggested LHMDA-DL approach and current methods [42] and [43], Table shows the message transmission accuracy.

Table - Accuracy of message transmission with LHMDA-DL, SEC-RMC, and DWT signal based IoT Monitoring System and Deep Learning

Network samples	**Message transmission accuracy (%)**		
	LHMDA-DL	**SEC-RMC**	**Deep Learning and DWT signal based IoT Monitoring System**
2500	93	91.4	88.6
5000	92.15	90	86.35
7500	92	89	85
10000	91.35	88.25	84.25
12500	91	88	83
15000	90	86	82
17500	90	86	82
20000	91.35	87.35	83.15
22500	93	88	84
25000	93.35	90	85

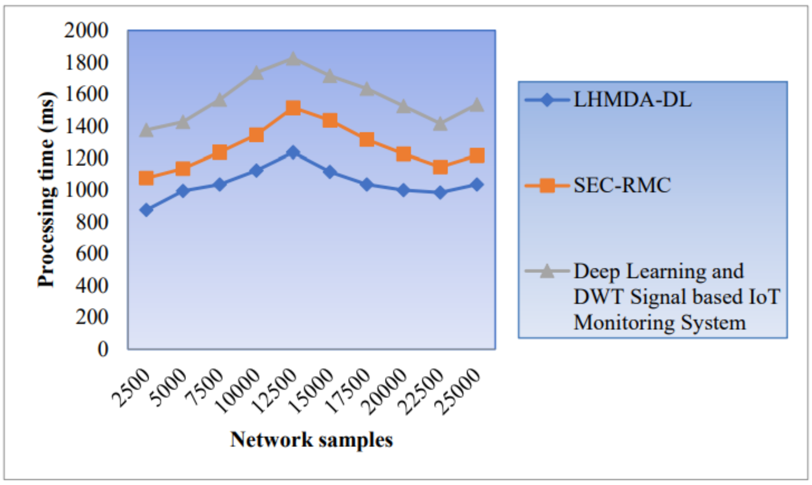

Figure 6 shows message transmission accuracy for 2,500–25,000 network samples over ten simulation iterations. Sample size did not affect outcomes. With 2,500 samples, LHMDA-DL achieved 93% accuracy, compared to 91.4% [42] and 88.6% [43]. The improvement comes from extracting relevant features using Gaussian Mixture and Lebesgue measure, then selecting the most relevant features via HMDAR-LSTM using Connection Message Ratio and Connection Acknowledgment Message Ratio. This distinguishes DoS from non-DoS variants, protecting MQTT messages and increasing accuracy by 4% over [42] and 9% over [43].

5.5 Message Transmission Error Rate

The message transmission error rate is finally determined in this part. This part assesses the message transmission error rate to confirm the method's effectiveness.

$$MTE = \sum \frac{NS_{missed}}{NS_{sent}} * 100 \tag{21}$$

According to equation (21) above, the message transmission error rate (MTE) is calculated by securely calculating the network samples that were transmitted by the publisher (called "NSsent") and received by the subscriber (called "NSmissed"). The percentage (%) is used to measure it. Using the suggested LHMDA-DL approach and current techniques [42] and [43], Table shows the message transmission error rate by changing the values in equation (21).

Table - Error rate of message transmission utilizing SEC-RMC, LHMDA-DL, and Deep Learning and DWT signal based IoT Monitoring System

Network samples	Message transmission error rate (%)		
	LHMDA-DL	**SEC-RMC**	**Deep Learning and DWT signal based IoT Monitoring System**
2500	7	8.6	11.4
5000	7.35	9.35	12
7500	7.85	10.15	12.35
10000	8	11	13
12500	8.25	11.35	13.55
15000	7.25	11	13
17500	7	10.35	12.15
20000	6.35	10	11.65
22500	7	10.25	11.85
25000	7.35	10.85	12.15

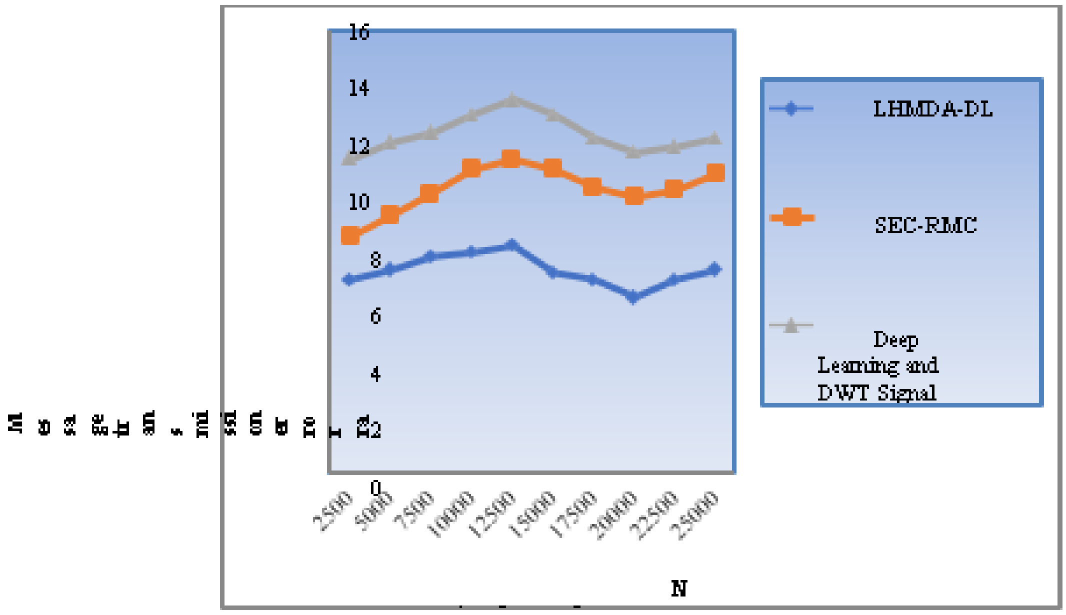

Figure – Graphical Representation of message transmission error rate

Figure shows the message transmission error rate for LHMDA-DL, SEC-RMC [42], and Deep Learning and IoT-Based Monitoring System [43]. The LHMDA-DL method (blue line) has a significantly lower error rate. With 2,500 network samples, LHMDA-DL missed 175 packets (7% error), compared to 215 packets (8.6%) [42] and 285 packets (11.4%) [43]. Using Connection Message Ratio and Connection Acknowledgment Message Ratio with HMDAR-LSTM, MQTT messages are protected from DoS variants. LHMDA-DL reduces the error rate by 28% compared to [42] and 40% compared to [43].

6 CONCLUSION

The important developments in tackling issues of security, effectiveness, and dependability in cloud computing and IoT systems. The paper offers a strong defense against risks like Denial-of- Service (DoS) attacks for MQTT-based IoT transactions by presenting the Lebesgue Hash Message Double Authentication integrated Deep Learning (LHMDA-DL) framework. High message transmission accuracy and low error rates are ensured by the use of Long Short-Term Memory (LSTM) networks for authentication, Gaussian Mixture Models for feature extraction, and Discrete Wavelet Transform (DWT) for noise reduction. By reducing processing times and traffic overhead while retaining superior performance in protecting IoT connections, the research shows how effective the suggested method is. This demonstrates how well-suited the method is for real-time industrial applications where scalability and dependability are essential. The study also examines thorough approaches to guaranteeing cloud data integrity, with a focus on encryption and cutting- edge key management technologies. By addressing flaws in cloud storage systems, these strategies improve data security and uphold user and service provider confidence. To sum up, combining cutting-edge signal processing, machine learning, and cryptography methods provides a scalable and effective means to address issues with the Internet of Things and the cloud. The results highlight how these techniques may enhance system security and operational effectiveness, which makes them extremely applicable to a variety of cloud computing and IoT applications.

REFERENCES

[1] Huang, T. S. (1988). *Advances in computer vision & image processing*. JAI Press, Inc.Rao, V. S., Prasad, R. V., Prabhakar, T. V., Sarkar, C., Koppal, M., & Niemegeers, I. (2019). Understanding and improving the performance of constructive interference using destructive interference in WSNs. *IEEE/ACM Transactions on Networking*, *27*(2), 505- 517.

[2] Dey, I. (2023). Wavelet Transform for IoT--A Signal Processing Perspective. *Authorea Preprints*.

[3] Fischer-Hwang, I., Ochoa, I., Weissman, T., & Hernaez, M. (2019). Denoising of aligned genomic data. *Scientific reports*, *9*(1), 15067.

[4] Cai, C., & Harrington, P. D. B. (1998). Different discrete wavelet transforms applied to denoising analytical data. *Journal of chemical information and computer sciences*, *38*(6), 1161-1170.

[5] Çavuslu, M. A., & Karakaya, F. (2010, April). Hardware implementation of Discrete Wavelet Transform and Inverse Discrete Wavelet Transform on FPGA. In *2010 IEEE 18th Signal Processing and Communications Applications Conference* (pp. 141-144). IEEE.

[6] Li, W., Manickam, S., Nanda, P., Al-Ani, A. K., & Karuppayah, S. (2024). Securing MQTT Ecosystem: Exploring Vulnerabilities, Mitigations, and Future Trajectories. *IEEE Access*.

[7] Hamid, H. G., & Alisa, Z. T. (2021). Survey on IoT application layer protocols. *Indonesian Journal of Electrical Engineering and Computer Science*, *21*(3), 1663-1672.

[8] Goswami, P., Faujdar, N., Debnath, S., Khan, A. K., & Singh, G. (2024). Investigation on storage level data integrity strategies in cloud computing: classification, security obstructions, challenges and vulnerability. *Journal of Cloud Computing*, *13*(1), 45.

[9] Buyya, R., Broberg, J., & Goscinski, A. M. (Eds.). (2010). *Cloud computing: Principles and paradigms*. John Wiley & Sons.

[10] Mell, P. (2011). The NIST Definition of Cloud Computing. *Recommendations of the National Institute of Standards and Technology*.

[11] Wu, C., Buyya, R., & Ramamohanarao, K. (2019). Cloud pricing models: Taxonomy, survey, and interdisciplinary challenges. *ACM Computing Surveys (CSUR)*, *52*(6), 1-36.

[12] Dimitri, N. (2020). Pricing cloud IaaS computing services. *Journal of Cloud Computing*, *9*(1), 14.

[13] Klosterboer, L. (2011). *ITIL Capacity Management (paperback)*. Pearson Education.Dong, Y., Sun, L., Liu, D., Feng, M., & Miao, T. (2018, August). A survey on data integrity checking in cloud. In *2018 1st International Cognitive Cities Conference (IC3)* (pp. 109- 113). IEEE.

[14] Iqbal, A., & Saham, H. (2014). Data integrity issues in cloud servers. *International Journal of Computer Science Issues (IJCSI)*, *11*(3), 118. Tunc, C., Hariri, S., Merzouki, M., Mahmoudi, C., De Vaulx, F. J., Chbili, J., ... & Battou, A. (2017, September). Cloud security automation framework. In *2017 IEEE 2nd International Workshops on Foundations and Applications of Self* Systems (FAS* W)* (pp. 307-312). IEEE.

[15] Somasundaram, T. S., Prabha, V., & Arumugam, M. (2012, December). Scalability issues in cloud computing. In *2012 Fourth International Conference on Advanced Computing (ICoAC)* (pp. 1-5). IEEE.

[16] Natu, M., Ghosh, R. K., Shyamsundar, R. K., & Ranjan, R. (2016). Holistic performance monitoring of hybrid clouds: Complexities and future directions. *IEEE Cloud Computing*, *3*(1), 72-81.

[17] Rakotondravony, N., Taubmann, B., Mandarawi, W., Weishäupl, E., Xu, P., Kolosnjaji, B. ... & Reiser, H. P. (2017). Classifying malware attacks in IaaS cloud environments. *Journal of Cloud Computing*, *6*, 1-12.

[18] Mahajan, A., & Sharma, S. (2015). The malicious insiders threat in the cloud. International Journal of Engineering Research and General Science, 3(2), 245-256.

[19] Liao, X., Alrwais, S., Yuan, K., Xing, L., Wang, X., Hao, S., & Beyah, R. (2018). Cloud repository as a malicious service: challenge, identification and implication. *Cybersecurity*, *1*, 1-18.

[20] Niharika, V., Sathvika, A., Kumar, A. T., & Prasad, Y. (2024). Securing diagnostic text data in medical

photographs: A novel hybrid approach integrating AES-RSA encryption and 2D-DWT steganography. *Journal of Nonlinear Analysis and Optimization, 15*(1). ISSN: 1906-9685.

[21] Pandey, A., Tiwary, P., Kumar, S., & Das, S. K. (2020, August). Residual neural networks for heterogeneous smart device localization in IoT networks. In *2020 29th International Conference on Computer Communications and Networks (ICCCN)* (pp. 1-9). IEEECvitić, I., Peraković, D., Periša, M., & Botica, M. (2021). Novel approach for detection of IoT generated DDoS traffic. *Wireless Networks*, *27*(3), 1573-1586.

[22] Kumar, S., Singh, S. K., Singh, A. K., Tiwari, S., & Singh, R. S. (2018). Privacy preserving security using biometrics in cloud computing. *Multimedia Tools and Applications*, *77*, 11017-11039.

[23] Gebresilassie, S. K., Rafferty, J., Chen, L., Zhan, Z., & Abu-Tair, M. (2023). Transfer and CNN-based de-authentication (disassociation) DoS attack detection in IoT Wi-Fi networks. *Electronics*, September 2023.

[24] Song, H., Li, J., & Li, H. (2021). A cloud secure storage mechanism based on data dispersion and encryption. *IEEE Access*, *9*, 63745-63751.

[25] Zhang, Y., Yu, J., Hao, R., Wang, C., & Ren, K. (2018). Enabling efficient user revocation in identity-based cloud storage auditing for shared big data. *IEEE Transactions on Dependable and Secure computing*, *17*(3), 608-619.

[26] Zuo, C., Shao, J., Liu, J. K., Wei, G., & Ling, Y. (2017). Fine-grained two-factor protection mechanism for data sharing in cloud storage. *IEEE Transactions on Information Forensics and Security*, *13*(1), 186-196.

[27] Cui, H., Deng, R. H., Li, Y., & Wu, G. (2017). Attribute-based storage supporting secure deduplication of encrypted data in cloud. *IEEE transactions on big data*, *5*(3), 330-342.

[28] Sun, S., Ma, H., Song, Z., & Zhang, R. (2020). WebCloud: Web-based cloud storage for secure data sharing across platforms. *IEEE Transactions on Dependable and Secure Computing*, *19*(3), 1871-1884.

[29] Cheng, K., Wang, L., Shen, Y., Wang, H., Wang, Y., Jiang, X., & Zhong, H. (2017). Secure $ k $ k-NN query on encrypted cloud data with multiple keys. *IEEE Transactions on Big Data*, *7*(4), 689-702.

[30] Pujar, S. R., Chaudhari, S. S., & Aparna, R. (2020, July). Survey on data integrity and verification for cloud storage. In *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)* (pp. 1-7). IEEE.

[31] Wang, B., Li, B., & Li, H. (2014). Oruta: Privacy-preserving public auditing for shared data in the cloud. *IEEE transactions on cloud computing*, *2*(1), 43-56.

[32] Indhumathil, T., Aarthy, N., Devi, V. D., & Samyuktha, V. N. (2017, March). Third-party auditing for cloud service providers in multicloud environment. In *2017 Third International Conference on Science Technology Engineering & Management (ICONSTEM)* (pp. 347-352). IEEE.

[33] Rukavitsyn AN, Borisenko KA, Holod II, Shorov AV (2017) The method of ensuring confidentiality and integrity data in cloud computing. *In: 2017 XX IEEE International Conference on Soft Computing and Measurements (SCM)*. IEEE, pp 272–274.

[34] Shen W, Qin J, Yu J, Hao R, Hu J (2018) Enabling identity-based integrity auditing and data sharing with sensitive information hiding for secure cloud storage. *IEEE Trans Inf Forensic Secure* 14(2):331–346.

[35] Zhang Y, Xu C, Li H, Liang X (2016) Cryptographic public verification of data integrity for cloud storage systems. *IEEE Cloud Computing* 3(5):44–52.

[36] Hiremath S, Kunte S (2017) A novel data auditing approach to achieve data privacy and data integrity in cloud computing. *In: 2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*. IEEE, pp 306–310.

[37] Shao B, Bian G, Wang Y, Su S, Guo C (2018) Dynamic data integrity auditing method supporting privacy protection in vehicular cloud environment. *IEEE Access* 6:43785– 43797.

[38] Shilpa Va, Vidya Aa, Santosh Pattar, "MQTT based Secure Transport Layer Communication for Mutual Authentication in IoT Network", *Global Transitions Proceedings*, Elsevier, Apr 2022.

[39] Minh-Quang Tran, Mahmoud Elsisi, Meng-Kun Liu, Viet Q. Vu2 , Karar Mahmoud, Mohamed M. F. Darwish, Almoataz Y. Abdelaziz, Matti Lehtonen, "Reliable Deep Learning and IoT-Based Monitoring System for Secure Computer Numerical Control Machines Against Cyber-Attacks With Experimental Verification", *IEEE Access*, Mar 2022.

[40] N Musrat Sultana, K. Srinivas, "Data Privacy Protection in Cloud Computing Using Visual Cryptography", Multimedia Tools and Applications, August 2024, https://doi.org/10.1007/s11042-024-19963-6.