Contents lists available at ScienceDirect

Digital Communications and Networks

journal homepage: www.keaipublishing.com/dcan

GRADE: Deep learning and garlic routing-based secure data sharing framework for IIoT beyond 5G

Nilesh Kumar Jadav [a], Riya Kakkar [a], Harsh Mankodiya [a], Rajesh Gupta [a], Sudeep Tanwar [a,*], Smita Agrawal [a], Ravi Sharma [b]

[a] *Department of Computer Science and Engineering, Institute of Technology, Nirma University, Ahmedabad, Gujarat, India*
[b] *Centre for Inter-Disciplinary Research and Innovation, University of Petroleum and Energy Studies, P.O. Bidholi Via-Prem Nagar, Dehradun, India*

A R T I C L E I N F O

Keywords:
Garlic routing
Blockchain
I2P
LSTM
Artificial intelligence
Onion routing

A B S T R A C T

The rise of automation with Machine-Type Communication (MTC) holds great potential in developing Industrial Internet of Things (IIoT)-based applications such as smart cities, Intelligent Transportation Systems (ITS), supply chains, and smart industries without any human intervention. However, MTC has to cope with significant security challenges due to heterogeneous data, public network connectivity, and inadequate security mechanism. To overcome the aforementioned issues, we have proposed a blockchain and garlic-routing-based secure data exchange framework, i.e., *GRADE*, which alleviates the security constraints and maintains the stable connection in MTC. First, the Long-Short-Term Memory (LSTM)-based Nadam optimizer efficiently predicts the class label, i.e., malicious and non-malicious, and forwards the non-malicious data requests of MTC to the Garlic Routing (GR) network. The GR network assigns a unique ElGamal encrypted session tag to each machine partaking in MTC. Then, an Advanced Encryption Standard (AES) is applied to encrypt the MTC data requests. Further, the Inter-Planetary File System (IPFS)-based blockchain is employed to store the machine's session tags, which increases the scalability of the proposed *GRADE* framework. Additionally, the proposed framework has utilized the indispensable benefits of the 6G network to enhance the network performance of MTC. Lastly, the proposed *GRADE* framework is evaluated against different performance metrics such as scalability, packet loss, accuracy, and compromised rate of the MTC data request. The results show that the *GRADE* framework outperforms the baseline methods in terms of accuracy, i.e., 98.9%, compromised rate, i.e., 18.5%, scalability, i.e., 47.2%, and packet loss ratio, i.e., 24.3%.

1. Introduction

The rise in industrial automation propels the Internet of Things (IoT) to interconnect smart devices intelligently for effective predictive services in the Industrial Internet of Things (IIoT). The interconnection between devices requires pervasive communication, such as Machine-Type Communication (MTC), where a machine communicates with another machine to execute a shared task of IIoT without human interference. A machine collects the sensitive data from the industry environment and transmits it via a Wi-Fi or cellular network interface to other machines. Then, a computing model interprets the data for stringent decision-making that can trigger automated actions. However, the communication between machines uses a public network interface, i.e., the Internet, where MTC systems suffer from various security issues such as intrusion attacks, network exploitation, injection attacks, and resource starvation attacks. Moreover, multiple devices in the MTC system acquire a colossal amount of heterogeneous data (industrial-critical data) that requires constant supervision, which otherwise maneuvers into data breaches and data-tampering attacks.

To overcome the security threats and facilitate secure communication, researchers have utilized convincing solutions such as in Ref. [1] authors have used a novel hybrid key management infrastructure to secure the data exchange and authentication schemes for Machine-to-Machine communication (M2M)-based microgrids. Later, the authors of [2] have applied cryptographic solutions in an M2M-based smart healthcare system. To confront network attacks on the handover mechanism between the base station and smart devices, they have applied a group-based secrecy-aware authentication scheme to protect

* Corresponding author.
E-mail addresses: 21ftphde53@nirmauni.ac.in (N.K. Jadav), 21ftphde56@nirmauni.ac.in (R. Kakkar), 19bce117@nirmauni.ac.in (H. Mankodiya), rajesh.gupta@nirmauni.ac.in (R. Gupta), sudeep.tanwar@nirmauni.ac.in (S. Tanwar), smita.agrawal@nirmauni.ac.in (S. Agrawal), ravisharmacidri@gmail.com (R. Sharma).

https://doi.org/10.1016/j.dcan.2022.11.004
Received 2 March 2022; Received in revised form 20 October 2022; Accepted 1 November 2022
Available online 19 November 2022

the handover protocol and essential principles of cryptography, i.e., data integrity, confidentiality, and availability. Their result outperforms other baseline handover protocols in terms of computational overhead, storage, and security.

However, with the modern processing power, it is evident that one can break the cryptographic solutions in a transient time. Additionally, the data exchange between machines propagates through the public Internet, where a malicious attacker can passively view the routing information and accordingly plan the attack. Therefore, researchers have adopted an Onion Routing (OR)-based MTC, where the data packets are encrypted multiple times to preserve security and privacy concerns. Gupta et al. in Ref. [3] have investigated security, privacy, and trust issues in the combat operation. They embrace the anonymous network, i.e., onion-routing and blockchain network, to address the aforementioned issues in the battlefield operation. Their result shows that their proposed architecture attains effective latency and scalability and alleviates malicious security obstructions. Further, in Ref. [4], the authors have examined the susceptible security loopholes in e-healthcare systems. To deal with that, they have adopted onion encryption, where the medical data is transferred via encrypted onion routers; additionally, blockchain-based smart contracts are enabled to validate the forged data from the malicious intruders.

Nevertheless, the OR has an intricate design configuration and has a deceptive exit routing policy, making the protocol unmanageable against attacks such as passive eavesdropping, traffic analysis, tor exit node attacks, and replay attacks. Therefore, the Garlic Routing (GR) network has emerged as a promising solution with a few similar functionalities as OR; however, it is far better and more secure than OR. This is because it has intuitive cryptographic algorithms and encrypted tunnels that help in securing the garlic-wrapped message from the MTC devices. The significant advantage of the GR network lies in its end-end encryption without any centralized system. In Ref. [5], the authors discussed the privacy shortcomings in securing sensitive data. Thus, they adopted the integration of garlic and OR into a combinatorial onion-garlic routing framework. Later, in Ref. [6] the authors considered a private content sharing scheme to preserve the privacy and confidentiality of the content. They mentioned that proxies and conventional cryptographic solutions are infeasible in securing the content. Hence, they utilized a decentralized and anonymous network, i.e., the GR network, which assists the users in finding private proxies wherein a dispersal algorithm relays the content securely. However, it is computationally expensive because the complexity of the GR network increases as the data packet increases. Therefore, it entails having a mechanism where only legitimate packets are processed by the onion routers, discarding other illicit data packets. Opportunistically, the indispensable attributes of Artificial Intelligence (AI) algorithms present a plausible solution for the above-mentioned problem by classifying the normal and attack data. Toward this goal, the authors of [7] proposed a federated learning model to offload and efficiently manage the surplus amount of heterogeneous data at each device by employing reinforcement learning, where they classify the devices based on high-quality data. Later, in Ref. [8], the authors discussed the shortcomings of false alarm attacks in IIoT systems that delude the industrial applications by forging the incorrect sensor measurement. AI-based autoencoders are used to detect such attacks, which significantly recover the original data from the falsified data.

However, AI-based solutions are not feasible in protecting the privacy of the data. This is because there is no secure storage provision with AI models once it classifies the normal data [9]. Therefore, even though the data is recovered from the attacks, there is still a possibility that an attacker will leverage the recovered data into data tampering attacks. The blockchain blended with smart contracts can be a robust solution for preserving the machine's data privacy [10]. In this context, the authors of [11] use a blockchain-based sharding mechanism to enhance the throughput and scalability of the IIoT systems. However, the attackers deteriorate the performance of the sharding technology, resulting in ineffective throughput and unreliable scalability of the IIoT system. To alleviate this issue, vector optimization is used in the blockchain sharding mechanism, improving the overall throughput and scalability of the blockchain-assisted IIoT systems. In Ref. [12] a combinatorial framework is proposed to tackle storage issues, access control, and detection of malicious intruders in the IIoT system. First, it permits authenticated users to store their public keys, revocation, and user-specific data. Later, the authorities validate the user's data from the already stored data in the blockchain to remark them as authenticated or malicious users. However, most of the blockchain-enabled solutions rely on the Ethereum-based platform, which is a public blockchain [13]. Hence, storing industry-critical data inside the public blockchain persuades insider attacks that need to be overseen.

To address the aforementioned security issues, we have proposed a DL and GR-based secure data sharing framework, i.e., *GRADE*, which secures the data exchange between machines in MTC. First, we bifurcate the malicious and non-malicious data requests using the Long Short-Term Memory (LSTM) model to reduce the computational overhead of the GR network. In addition, to satisfy the computational requirement of the intelligence layer, we have used edge servers where all computations are executed in order to reduce the overall cost of the proposed *GRADE* framework. Afterward, only non-malicious data requests are transferred to the anonymous peer-to-peer network, i.e., the GR network. All the data packets are layered-encrypted and tunneled from one machine to another, making it challenging for an attacker to find helpful information from the obfuscated data requests. For that, the ElGamal algorithm is used to encrypt the session tags of the participating MTC machines, and an Advanced Encryption Atandard (AES) is applied to encrypt the MTC data request payload to prevent it from the attackers. In addition, we have used an Inter-Planetary File System (IPFS)-based blockchain to store the session tags of the MTC machines, which lessens the effects of privacy concerns. We have incorporated a 6G network to maximize the performance of the proposed *GRADE* framework in terms of latency and packet loss ratio [14]. Furthermore, the proposed framework has been simulated by configuring the GR network in the I2P framework highlighting the performance of the trained LSTM models in terms of precision, recall, and F1-score. Lastly, the LSTM model has been trained with different optimizers in which Nadam attains the highest accuracy of 98.9% compared with other optimizers. Moreover, the *GRADE* outperforms the existing approaches in terms of data compromised rate, i.e., 18.5%, scalability, i.e., 47.2%, and packet loss ratio, i.e., 24.3%.

1.1. Contributions

- We propose a blockchain and GR-based secure and privacy-preserved framework, i.e., *GRADE* to enable the secure data requests exchange for MTC in the IIoT environment. Furthermore, the employed IPFS-based blockchain over the 6G network makes the framework scalable and reliable.
- We employ an AI-based LSTM model to efficiently bifurcate the malicious and non-malicious data requests from a time-series dataset to predict the possibility of an attack in the future.
- The proposed *GRADE* framework is examined by considering various parameters such as precision, recall, and F1-score for the AI-based LSTM model. It is then plotted for the loss curve and accuracy curve, revealing that the Nadam optimizer outperforms other LSTM-based optimizers in terms of accuracy, i.e., 98.9%.
- Lastly, the GR network is configured inside the I2P framework by manually setting up the Transport Control Protocol(TCP)-based inbound and outbound tunnels. Then, the GR network is tested against several security attacks such as DDoS, injection, session hijacking, etc., to verify the data requests compromise rate, which reveals that it has a lower compromised rate, i.e., 18.5%, lower packet loss ratio, i.e., 24.3%, and high scalability, i.e., 47.2% compared to other baseline works.

1.2. Organization

The rest of the paper is organized as follows. Section 2 discusses the related work and the comparative analysis of the proposed work with the existing state-of-the-art work. Section 3 introduces the system model and problem formulation. Section 4 presents the proposed collaborative intelligence framework for secure MTC in IIoT environments. Section 5 presents the results and discussion. Section 6 presents an exhaustive discussion on *GRADE* and its potential impact on tackling security threats. Finally, Section 7 gives the concluding remarks. Table 1 presents the abbreviations used in the proposed *GRADE* framework.

2. Related works

Many researchers have given solutions to strengthen the security and privacy of the data exchange between MTC devices in the IIoT system. However, according to the literature, most of the proposed solutions have adopted the OR protocol that works on the principle of multi-layer encryption for single data requests. In addition, it is imperiled by time analysis and exit node vulnerability that raises the data compromisation rate of the OR network. For example, the authors in Ref. [15] discussed the OR method in an anonymous peer-to-peer network to secure the user identity utilizing the applied cryptography techniques. However, they have not explored the potential of OR protocol in real-time applications [16].

To further mitigate the security threats, Zhang et al. in Ref. [17] presented a secure and traceable onion chain integrated with the blockchain to preserve the communication in Vehicular Ad Hoc Networks (VANETs). Similarly, the authors in Ref. [18] also adapted the OR protocol to maintain anonymity in VANETs. Later, Ali et al. [19] adopted the Ethereum blockchain with the Tor network, i.e., OR mechanism, to facilitate the security of patient data in the remote health monitoring systems. They have performed the security analysis of the system considering the attacks such as Man-in-the-Middle (MiTM), Denial of Service (DoS), social engineering, etc. Moreover, many researchers have employed AI techniques to classify and predict the security attack before it executes on real-world applications. In that direction, the authors of [20] have applied the AI-based URL classification for Internet communication. They have embraced the encryption scheme of the OR mechanism to perform the webpage classification that attains improved accuracy compared to the conventional network. Further, to lessen the impact of the aforementioned security attacks, Tiwari et al. [21] incorporated Deep Learning (DL) models to provide secure cloud communication by comparing Tor and non-Tor networks. However, AI techniques have no provision of secure data storage; hence they can suffer from data tampering and data manipulation attacks. A few of the researchers have tried to improve the data security with blockchain network [17,19], where they securely store the data in an immutable ledger. However, they have used an Ethereum-based blockchain network, raising data storage costs. Moreover, we observed that there is not much literature available that utilizes the integration of all technologies, i.e., AI, blockchain, anonymous and encrypted networks, and there is no literature available that highlights the benefits of the GR network. Motivated by this, we have proposed a blockchain and AI-based GR framework, i.e., *GRADE* for secure data exchange between machines of the IIoT systems. In this framework, the data request of MTC machines is first classified by the LSTM model, i.e., whether the data request is malicious or non-malicious.

Only the non-malicious data requests are forwarded to the GR network to reduce the computational overhead of the proposed framework. Then, in the GR network, each outbound and inbound tunnel is associated with the blockchain, where the session tags of each data request are securely stored. Each data request is verified by the session tags stored inside the blockchain. Once validated, the GR network encrypts the multiple data requests in multi-layer encryption like a garlic bulb. Upon receiving the encrypted data requests, the receiver validates the session tag; if the session tag matches, the encrypted data request is decrypted securely at the destination. This approach enhances the security and reliability of the proposed GR network. The system model is implemented inside the Invisible Internet Project (I2P) framework, where we configure ElGamal and AES-based encrypted tunnel to transmit the data request securely. The proposed framework is evaluated with different performance evaluation metrics, such as accuracy, scalability, packet loss ratio, and data compromised rate, which shows that the proposed framework outperforms other baseline works. Table 2 shows the comparative analysis of the state-of-the-art data sharing schemes with the proposed scheme for IIoT applications.

Table 1
Abbreviation table.

Abbreviation	Definition
AES	Advanced Encryption Standard
AI	Artificial Intelligence
DoS	Denial of Service
DES	Data Encryption Standard
DDoS	Distributed Denial-of-Service
DPI	Deep Packet Inspection
DHT	Distributed Hash Table
ECC	Elliptic Curve Cryptography
GR	Garlic Routing
ITS	Intelligent Transportation System
IPFS	Interplanetary File System
IIoT	Industrial Internet of Things
IoT	Internet of Things
I2P	Invisible Internet Project
LSTM	Long Short-Term Memory
MTC	Machine-Type Communication
M2M	Machine-to-Machine Communication
MITM	Man-in-the-Middle
OR	Onion Routing
PLCs	Programmable Logic Controllers
RSA	Rivest-Shamir-Adleman
SCADA	Supervisory Control and Data Acquisition
TTL	Time To Live
TCP	Transport Control Protocol
VANETs	Vehicular Ad Hoc Networks
VPN	Virtual Private Network

3. System model and problem formulation

In this section, we have considered the machine-type-communication known as MTC, which offers communication between different machines of IIoT systems. In the considered scenario, bundle of messages, i.e., $\{D_{m_1}, D_{m_2}, \ldots, D_{m_n}\} \in B_{m_i}$ can be transferred between number of machines $\{M_1, M_2, \ldots, M_y\}$. Now, bundling of messages can be performed in two ways, i.e., source machine M_x can decide to exchange the multiple messages with the destination machine M_y. Another way can be that multiple source machines M_x can come together to exchange the bundled message with the destination machine M_y. The source machine M_x can start communicating with the destination machine M_y by sending the data message requests $D_{m_i} \in \{D_{m_1}, D_{m_2}, \ldots, \ldots, D_{m_n}\}$. The communication between source machine and the destination machine in the form of message requests can be represented as follows.

$$M_x \in \{M_1, M_2, \ldots, M_y\} \overset{D_{m_i}}{\rightarrow} M_y \in \{M_1, M_2, \ldots, M_y\} \tag{1}$$

Various conventional encryption algorithms such as Data Encryption Standard (DES), Elliptic Curve Cryptography (ECC), and Rivest-Shamir-Adleman (RSA) functions to secure the communication of messages in MTC. But, these security algorithms can be easily compromised by the modern technological aspects and capabilities. So, OR concept has been used to deal with these security issues. The concept involves multiple layers of encryption to secure the message communication. But, there are some security breaches associated with the OR due to the involvement of a centralized server. Therefore, the security and anonymity issues can be

Table 2
Comparative analysis of various state-of-the-art data sharing schemes with the proposed framework.

Author	Year	Objective	Block-chain	Pros	Cons
Balasubramanian et al. [15]	2019	Presented an onion routing protocol to secure the user's identity in a peer-to-peer network	No	Improved anonymity and interoperability	Security issues against single point of failure, data manipulation, and eavesdropping attack
Zhang et al. [17]	2019	Investigated a blockchain-based secure onion chain in VANETs	Yes	Enhanced traceability, throughput, and cost-efficient	No discussion of scalability and interoperability
Haghighi et al. [18]	2020	Proposed an onion routing mechanism to ensure the privacy and anonymity in VANETs	No	Improved latency and delivery ratio	Security issues against data tampering, data manipulation, and MITM attack
Ali et al. [19]	2020	Discussed blockchain network with onion routing mechanism for secure data sharing in remote health monitoring system	Yes	Secure against MITM, DoS, and social engineering attack	Scalability and interoperability issues
Liu et al. [20]	2020	Applied AI-based URL classification in the wireless communication	No	Improved accuracy	Security issues against data tampering and data manipulation attack
Tiwari et al. [21]	2021	Presented DL models to facilitate secure cloud communication	No	Secure against eavesdropping attack, high efficiency, and flexibility	Security concerns against data manipulation, data tampering, and DoS attack
Jadav et al. [22]	2022	Discussed a DL and onion routing-based framework for smart homes	Yes	Low latency, improved delivery ratio, secure against DDoS and eavesdropping attack	No focus to secure against impersonation attack
The proposed framework	2022	Proposed a DL and garlic routing-based secure data sharing framework for IIoT	Yes	Highly secure, scalable, and accuracy	

mitigated by using a multi-layered encryption protocol, i.e., GR. It provides the flexibility over OR to send messages in the bundled form instead of a single message. Also, bundled messages that need to be transferred between the MTC machines are passed through the inbound and outbound tunnels of the GR network. The concept of tunneling in the GR ensures secure and anonymous communication between the number of source and destination machines.

Further, we have employed an AI-based LSTM model to tackle various security attacks associated with the data message requests of the source machine in the IIoT environment. It basically works to discard the malicious data and send the non-malicious data requests to the destination machine. Now, the employed GR can be employed after the arrival of non-malicious data requests D_{m_i} of different machines, i.e., source machine M_x that needs to be forwarded to the destination machine M_y. GR utilizes multi-layered encryption to reduce the security or traffic analysis attacks on the data message requests with the usage of the inbound tunnel and outbound tunnel. Now, the message requests of the source machine arriving for the destination machine should be arranged in a sequence so that the destination machine can acquire the messages in a similar order. For that, sequence numbers $\mathcal{S}_i \in \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \ldots, \mathcal{S}_n\}$ can be assigned to the forwarded data requests D_{m_i} of the source machine using the TCP. After appending sequence numbers to the message requests, GR as an encryption protocol utilizes the ElGamal algorithm to generate multiple encrypted session keys for the upcoming source machine message requests to the destination machine. The above-mentioned association of data message requests D_{m_i} with the sequence numbers can be represented as follows.

$$\sum_{i=1}^{y} D_{m_i} \overset{\text{TCP}}{\rightarrow} \sum_{i=1}^{n} \mathcal{S}_i \tag{2}$$

In GR, different source machines M_x can request to forward their bundle of message request D_{m_i} to the destination machine M_y. For that, the ElGamal algorithm can be used to assign the multiple session tags S_{t_i} for the multiple messages exchanged between the MTC machines. For that, ElGamal algorithm generates pair of public keys and private keys, i.e., $\{P_{m_x}, Pr_{m_x}\}$ for each machine in the routing as shown in Fig. 1. Equations (3) and (4) highlight the bundle of public keys and private keys generated to append session keys to the multiple data requests D_{m_i}. The data message requests at the source machine side can be encrypted with the generated public key P_{M_x} so that the destination machine can decrypt the message request with the help of private key Pr_{M_y}. The generation of

Source machine (M_x)	**Destination machine (M_y)**
# Public and private key generation Select$\langle I \rangle$ s.t. $I \rightarrow$ largest integer Generate$\langle \mathbb{G}_I \rangle$ s.t. $G_I \rightarrow$ cyclic group Choose $\langle g_p, g_q \rangle$ from G_I s.t. $\langle g_p, g_q \rangle \in G_I$ GCD$(g_q, I) = 1$ Compute $\beta = g_p^{g_q}$ Secret key of M_x $\mathbb{S}_{M_x} = g_q$ Generate public key $\mathbb{P}_{M_x}$ $\mathbb{P}_{M_x} = \langle \mathbb{G}_{\mathbb{I}}, \beta, I, g_p \rangle$	
$\xrightarrow[\textbf{Share } \mathbb{P}_{M_x} \textbf{ with } M_y]{\langle \mathbb{G}_{\mathbb{I}}, \beta, I, g_p \rangle}$	
	# Encrypt D_{m_x} using $\mathbb{P}_{M_x}$ Choose $\langle g_z \rangle \in \mathbb{G}_{\mathbb{I}}$ s.t. GCD$(g_z, I) = 1$ Compute shared secret key $\beta' = g_p^{g_z}$ Compute $\gamma = \beta^{g_z}$ Calculate $\gamma \times D_{m_x}$ Sends $\langle \beta', \gamma, \gamma \times D_{m_x} \rangle$
	$\xleftarrow[\textbf{Share with } M_x]{\langle \beta', \gamma, \gamma \times D_{M_x} \rangle}$
# Decrypt received message $\gamma' = \beta'^{g_q}$, i.e., $\gamma' \equiv \gamma$ $D_{m_x} = \frac{\gamma \times D_{m_x}}{\gamma'}$ D_{m_x} is now a secret data.	

Fig. 1. ElGamal secure data exchange algorithm between M_x and M_y.

public keys and private keys for a number of machines M_x can be represented as follows.

$$\sum_{o=1}^{n'} P_{M_o} \overset{ElGamal}{\longrightarrow} \{P_{M_1}, P_{M_2}, \ldots, \ldots, P_{M_{n'}}\} \tag{3}$$

$$\sum_{o=1}^{n'} S_{M_o} \overset{ElGamal}{\longrightarrow} \{S_{M_1}, S_{M_2}, \ldots, \ldots, S_{M_{n'}}\} \tag{4}$$

Therefore, the ElGamal algorithm can be used to append encrypted multiple session tags to the forwarded multiple data requests D_{m_i} of the machines in the GR. After appending the session tags to the D_{m_i}, the bundle of messages at the source machine M_x needs to be encrypted before forwarding it to the destination machine M_y. GR uses AES symmetric encryption algorithm to encrypt the D_{m_i} combined with the generated session tags S_{t_i} and sequence numbers $\mathcal{S}_i$ corresponding to each message requests. However, AES is a symmetric cryptographic algorithm that mainly relies on the same secret key for both the source machine and destination machine. So, we need to perform the secret key exchange between the source object O_x and destination object O_y using the Diffie-Hellman algorithm to secure the communication between machines as shown in Fig. 2. Here, the object can be any router or node, or machine communicating with each other. The Diffie-Hellman uses prime number P and prime root R that can be shared with O_y. Further, secret key S_{O_y} for O_y is being computed to generate the public key P_{O_y} to share it with the O_x. Then, the secret key S_{O_x} for O_x can be calculated to compute the public key P_{O_x} to share it with the O_y.

Now, AES algorithm can encrypt and decrypt the D_{m_i} using the shared secret key S'_{O_i} during the communication between M_x and M_y as shown in Fig. 2. There can be a scenario in which D_{m_i} can be compromised by any malicious intermediate machine or route while being exchanged between

Source object (O_x)	**Destination object (O_y)**
# Select prime number and prime root Generate$\langle P, R\rangle$	
$\xrightarrow{\hspace{3em}}$ **Share**$\langle P, R\rangle$ **with** O_y	
	# Compute O_y secret key S_{O_y} $S_{O_y} = (\zeta_y * \theta + k_y) mod m_y$ s.t. $m_y > 0, 0 < \zeta_y < m_y$ Generate public key of O_y $P_{O_y} = R^{S_{O_j}}$ mod P
	$\xleftarrow{\hspace{3em}}$ **Share** P_{O_y}
# Compute secret key of O_x $S_{O_x} = (\zeta_x * \theta_x + k_x)$ mod m_x s.t. $m_x > 0, 0 < \zeta_x < m_x$ Compute O_x public key $P_{O_x} = R^{S_{O_x}}$ mod P	
$\xrightarrow{\hspace{3em}}$ **Share** P_{O_x} **with** O_y	
	# Generate share secret key $S'_{O_y} = (P_{O_x})^{S_{O_y}}$ mod P
# Generate shared secret key $S'_{O_x} = (P_{O_y})^{S_{O_x}}$ mod P	

Fig. 2. Diffie-Hellman secret key exchange algorithm between O_x and O_y.

M_x and M_y. So, it is necessary to ensure the security and privacy of the message requests to prevent the modification by the malicious attacker, which can be represented in the form of an objective function Δ^{M_x,M_y} as follows.

$$\Delta^{M_x,M_y}(D_{m_i}, S_{T_i}, \mathcal{S}_i) = \max \sum_{i=1}^{n} \text{Secure}(M_i, D_{m_i}) \tag{5}$$

Therefore, communication between the source and destination machines needs to be secure from the data modification attacks in the GR to improve the security and privacy of the MTC in the IIoT ecosystem.

4. The proposed *GRADE* framework

This section describes the working of the proposed *GRADE* framework that comprises three distinct layers, i.e., application, intelligence, and security layer, as illustrated in Fig. 3. A summarized explanation of each layer is as follows.

4.1. Application layer

In the proposed *GRADE* framework, different machines of MTC are partaking in data exchange to assist applications such as remote monitoring, healthcare systems, automotive, smart home, and many more. Due to its varied applications, a machine has to confront heterogeneous data and different network interfaces that shrink the strength of the first line of defense, i.e., firewalls, detection systems, access controls, and a Virtual Private Network (VPN) against nefarious attacks. An attacker can target MTC in three ways, i.e., a physical attack, where the attacker specifically influences the MTC hardware, like flipping the bits of function register to thwart the regular operation of the IIoT application. Logical attack, where attackers are not in favor of modifying the data but instead, the attacker passively exploits the MTC such as impersonation, spoofing, and replay attack. Lastly, the data attacks primarily target the data exchange between different entities of MTC. The *GRADE* framework primarily focuses on tackling logical and data attacks. Therefore, in the application layer, we incorporate different MTC machines that continuously send and receive IIoT data by embracing the paramount features of a 6G network. A machine M_x shares the data request D_{m_i} with another machine M_y. However, the attacker $M_z \notin \{M_1, M_2, \ldots, M_y\}$ is inquisitive in fetching D_{m_i} for his own benefit. M_z is capable of manipulating the D_{m_i} such as it can forge the D_{m_i}, find vulnerabilities in other devices, or can perform an identity-based attack to erode the performance of MTC devices. Therefore, it necessitates an intelligence layer that helps in bifurcating the normal and attack traffic of MTC. Before that, we need a mechanism of request picker that assists the intelligence layer in selecting the high priority D_{m_i}. As a result, we have used a priority-based selection mechanism that selects the D_{m_i} based on its priority. For that, the priority $\{\varrho_1, \varrho_2, \ldots, \varrho_n\} \in \Lambda$ (priority) of each D_{m_i} is calculated.

$$\forall\ D_{m_i}\ \exists\ B_t, A_t, \text{and}\ \Lambda \tag{6}$$

When D_{m_i} arrives at the request picker, it selects D_{m_i} with the highest priority. For example, in Eq. (7) D_{m_1} has the highest priority compared to D_{m_2}, hence it is selected by the request picker.

$$\text{Request picker} \overset{\text{selects}}{\longrightarrow} D_{m_1} \tag{7}$$

$$\text{s.t. } \varrho(D_{m_1}) \gg \varrho(D_{m_2}) \tag{8}$$

The matter of concern in the application layer is to validate the authenticity of D_{m_i}. This is because there is a possibility that the enroute D_{m_i} is manipulated or exploited by the attacker. Thus, the objective of the intelligence layer is to categorize the malicious D_{m_i} and non-malicious D'_{m_i} from the MTC, as shown in Eq. 9

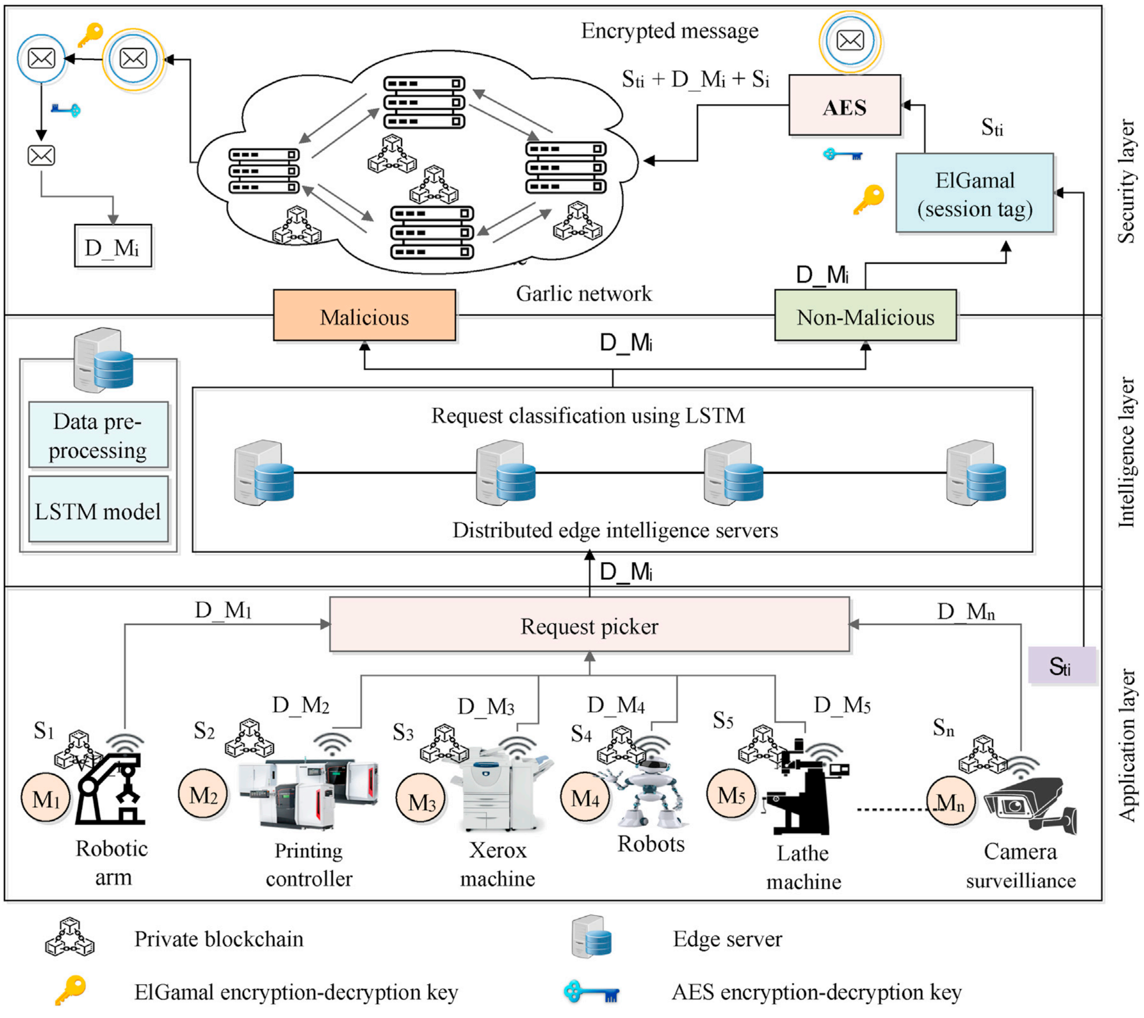


Fig. 3. *GRADE*: Blockchain and garlic routing-based secure data exchange framework for MTC.

$$\forall(D_{m_i}, D'_{m_i}) \exists \mathbb{C} \overset{\text{classifies}}{\rightarrow} D_{m_i} = 0 \text{ and } D'_{m_i} = 1 \tag{9}$$

where, D'_{m_i} is a malicious data request, D_{m_i} is a non-malicious data request, $\mathbb{C}$ is the classifier that classifies the non-malicious $D_{m_i} = 0$ and malicious $D'_{m_i} = 1$.

4.2. Intelligence layer

An attack can be detected by integrating a distinct intelligence layer that comprises pre-trained AI models based on the data sent by various machines. To improve the performance and reduce the resource-intensive nature of the AI model, we have utilized edge servers where all computations are executed. This layer consists of a control system that performs three primary functions: input data preparation, data pre-processing, and LSTM predictions. The enhanced AI framework for intrusion attack detection on a network comprising diverse machine requests is shown in Fig. 4.

4.2.1. Dataset description

A dataset called X-IIOTID: connectivity and device-agnostic intrusion dataset for the IIoT [23] is used to train the LSTM-based sequence model for intrusion detection in the suggested system architecture. The dataset's final version includes 820,834 training instances and a feature space with a size of 68. Normal and attack, normal and sub-category attack, and normal and sub-sub-category attack are the three types of target labels available. Algorithm 1 displays the algorithmic flow of data

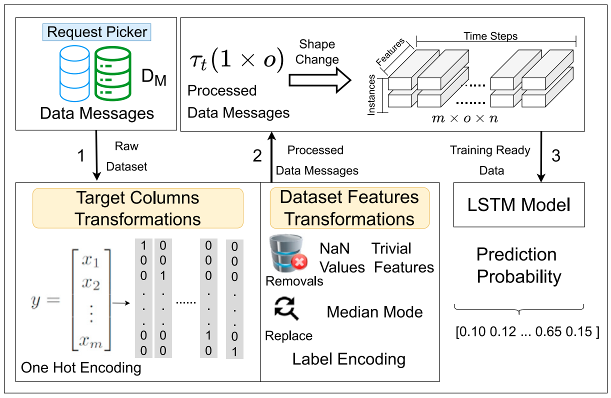


Fig. 4. Refined edge intelligence framework for smart predictions.

preprocessing used in the intelligence layer.

4.2.2. Dataset preprocessing

The collected dataset is then preprocessed to allow for training and prediction. Some non-essential columns, such as IP addresses, dates, and ids, are removed from the dataset. The median of the associated columns is used to replace all null and NaN values [24]. Label encoding is used to transform string values into numeric representation in columns with string values. Consider a dataset with replaced null values and columns eliminated, denoted by $\phi_{m \times n}$(m: total instances and n: total remaining features). The group by transformation G is applied to each class label l.

$$\phi_l = G_l(\phi_{\text{instances} \times \text{features}}) \quad \forall l \varepsilon C \tag{10}$$

Where C denotes a collection of distinct target classes. Additionally, ϕ_l can be broken into many timesteps, each with a size of o. $\Phi_{m \times o \times n}$ can be written as $\Phi_{m \times o \times n} = \bigcup_C \phi_l$ for the final training set. Each target class has a string data type that must be transformed into a one-hot encoded vector. Consider the original target class column y, which is taken from the dataset containing C unique classes. It is clear that y has the shape $(m, 1)$ and the data-type string. Using OHE as one-hot encoding transformation on y,

$$y_{m,C} := OHE(y_{m,1}) \tag{11}$$

where $:=$ denotes the replacement operator. $y_{m,C}$ is one hot encoded vector that can be represented as follows.

$$y_{m,C} = \begin{bmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 1 \\ 0 \end{bmatrix} \cdots \begin{bmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ 1 \end{bmatrix} \tag{12}$$

Algorithm 1. Data preprocessing algorithmic flow.

```
Input: Raw machine requests $D$, Target Values $y_{m,1}$
Output: $\Phi_{m \times o \times n}, y_{m,C}$
$y_{m,C} \leftarrow$ OHE($y_{m,1}$)
D ← drop(D, columns = [IP, date, timestamps, ids])
if $D_{\text{columns}}$.isNull() and $D_{\text{columns}}$.dataType in [string, int] **then**
    $D_{\text{columns}} \leftarrow D_{\text{columns}}$.fillNull($D_{\text{columns}}$.mode())
**Else**
    $D_{\text{columns}} \leftarrow D_{\text{columns}}$.fillNull($D_{\text{columns}}$.median())
end if
if $D_{\text{columns}}$.dataType is string **then**
    $D_{\text{columns}}$ = label Encoder($D_{\text{columns}}$)
end if
$\phi_{m \times n} \leftarrow D$
$\phi_l \leftarrow$ groupBy($l$)
$\Phi_{m \times o \times n} \leftarrow \bigcup_C \phi_l$
return $\Phi_{m \times o \times n}, y_{m,C}$
```

4.2.3. *Data preparation*

Before the LSTM model can be applied to incoming machine requests, it must first be feature engineered to ensure that the data is of the correct shape, size, and distribution. Consider a data burst of machine requests that occurs in real-time. D,

$$D = \{D_{M_1}, D_{M_2}, D_{M_3}, \ldots D_{M_i} \ldots, D_{M_m}\} \tag{13}$$

$$D_{M_i} = \{a_1, a_2, a_3, \ldots a_j \ldots, a_n\} \tag{14}$$

$$\forall 1 \le i \le m$$
$$\forall 1 \le j \le n$$

Where a_j denotes a single feature/attribute of data transmitted by the machine D_{M_i}, and n denotes the total number of features. Allow o active computers to generate a request to transmit to the destination node for a single time interval of t units. These requests can be acquired from the transmitting machines τ as follows.

$$\begin{aligned} &\tau_t = \{t_1, t_2, \ldots t_k \ldots, t_o\} \\ &\quad \forall 1 \le k \le o \\ &\quad \tau_t \subseteq D \end{aligned} \tag{15}$$

Shapes of the feature spaces of the corresponding machine data are given as follows.

$$D_{M_i}, t_k = 1 \times n \tag{16}$$

$$\tau_t = \{(1 \times n), (1 \times n), \ldots\}_{1 \times o} \tag{17}$$

$o \cdot n$ can be used to provide the size of the set τ_t. As a result, τ_t is reshaped to $1 \times o \times n$ to obtain the final dataset suitable for model predictions. The total time step and several attributes are represented by the second and third dimensions in the transformed dataset τ_t, respectively. Data preparation is implemented to satisfy the model requirements of the specific input shape. It's worth noting that the actual size of the transmitting dataset, τ_t, remains constant, i.e., $1 \cdot o \cdot n$.

4.2.4. *Long short-term memory prediction*

A sequential model can be trained to examine temporal sequences at interdependent timestamps based on the total number of users transmitting at a given instance. Fig. 4 shows the structure of a single LSTM cell, which is a particular recurrent neural network structure that may be utilized for sequential prediction on a time-series dataset [25]. A single hidden LSTM unit is made up of multiple gates, each of which performs a different task. The input and output of the gates are controlled using activation functions such as tanh and sigmoid. LSTMs can map and learn long-range temporal data by training the combination activation function and cell gates. The *sig* (sigmoid) and *tanh* [26] functions used inside the LSTM cell for a given input vector x can be given as follows.

$$sig(x) = \frac{1}{1 + e^{-x}} \tag{18}$$

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{19}$$

4.2.5. *Model training and predictions*

The above-mentioned LSTM model can be trained on a three-dimensional train dataset, with the time axis as the second dimension and the feature axis as the third. The overall model is shown in Fig. 5, which is trained and used for predictions. The proposed framework of the intelligence layer is described layer by layer as follows.

1. Input Layer: The input shape for each dataset varies depending on the feature space and number of timesteps used for training. In our scenario, there are $o = 15$ time steps and $n = 55$ feature space dimension; thus the input layer is changed accordingly.
2. Batch Normalization: The uncontrolled change in the distribution in the hidden units [27] is the chief problem for the deep neural

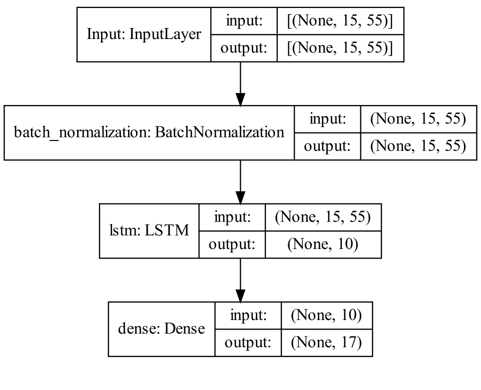


Fig. 5. LSTM architecture.

network. Utilizing batch normalization, a model can dynamically re-scale and revive the hidden layer distribution change caused by exploding gradients. For an input batch B of dimensions $b \times o \times n$, consider the batch normalization transformation BN with momentum value m', scaling factor γ, control offset β, and small constant ε.

$$\bar{x}_{\text{moving}}(B) = \bar{x}_{\text{moving}}(B) \times m' \times \overline{(B)} \times (1 - m') \tag{20}$$

$$\sigma^2_{\text{moving}}(B) = \sigma^2_{\text{moving}}(B) \times m' \times \sigma^2(B) \times (1 - m') \tag{21}$$

$$BN(B) = \gamma \frac{B - \bar{x}_{\text{moving}}(B)}{\sqrt{\sigma^2_{\text{moving}}(B) + \varepsilon}} + \beta \tag{22}$$

Where $\bar{x}$ is a mean calculation function and σ^2 is a batch variance calculation function. The training settings for momentum m' and small constant ε were 0.99 and 0.001, respectively.

3. LSTM Layer: The LSTM Layer is where actual learning and training take place. This layer has ten hidden units and uses tanh to activate their output. The symbol L indicates the net loss function. For example, consider a B mini-batch that has already undergone the BN transformation. Algorithm 2 shows the algorithmic flow of the LSTM prediction.

$$B := BN(B) \tag{23}$$

$$\hat{y} = \theta_1^T B + \theta_0 \tag{24}$$

$$L(\theta_1, \theta_0) = \sum_{l=1}^{C} -y log(\hat{y}) \tag{25}$$

subject to the constraints

$$\theta_1 = \theta_1 - \alpha \frac{\partial L}{\theta_1} \tag{26}$$

$$\theta_0 = \theta_0 - \alpha \frac{\partial L}{\theta_0} \tag{27}$$

Where θ_1 stands for model weights, θ_0 for bias, C for total target classes, y for actual target classes, $\hat{y}$ predicted target classes for batch B, α for learning rate, and ∂ for partial differential notation. Total epochs, loss function used, batch size, and other training configuration parameters for the proposed framework can be found in Table 3. Note that multiple optimizers were used to test the models. The Nadam optimizer was chosen for final predictions since it outperformed the others. The results will be discussed in the next section.

Algorithm 2. Prediction algorithmic flow

```
    Input: $\Phi_{m \times o \times n}$
    Output: $\hat{y}_{m \times o \times n}$
    Initialization: $C = 17$
model = Sequential()
model.add(inputLayer(shape = (o, n)))
model.add(batchNormalization())
model.add(LSTM(hiddenUnits = 10, activation = tanh))
model.add(Dense($C$, activation = softMax))
model.compile(optimizer = nadam, loss = crossEntropy)
model.loadWeights()
$\hat{y}_{m \times o \times n}$ = model.predict($\Phi$)
```

4.3. *Security layer*

The analysis layer bifurcates the normal and attack data requests from several IIoT machines. In this context, an AI-based LSTM model is utilized for training on 17 different types of security attacks, such as a dictionary, exfiltration, scanning, fuzzing, etc., to learn and terminate the forged data request in the future from the attacker machine M_t. Due to this learning, LSTM discards the forged data request and forwards the normal data requests, i.e., D_{m_i}, to the destination machine M_y. Nevertheless, sending the D_{m_i} instantly to the M_y through an open wireless interface persuades the attackers to exploit the D_{m_i}. The manipulation of D_{m_i} raises the security and privacy concerns in the IIoT environment. Consequently, there is a requirement for an encrypted network, i.e., GR encryption, which enhances the security and reliability of MTC in IIoT systems.

Table 3
Training configuration for the prediction model.

Parameters	Values
Epochs	5
Learning rate	0.001
Batch size	100
Loss function	Categorical cross-entropy
Optimizer	Nadam

The GR network is a variant of the OR network that aims to apply layered encryption on the incoming message. However, the key difference between them is that OR encrypts a single message, and GR encrypts a bundle of messages known as "bulbs" using layered encryption, as shown in Eqs. (28) and (29).

$$\mathcal{E}_1(\mathcal{E}_2(\mathcal{E}_3(D_{m_i}))) \tag{28}$$

$$\mathcal{E}_1(\mathcal{E}_2(\mathcal{E}_3(D_{m_1}, D_{m_2} \ldots D_{m_n}))) \tag{29}$$

where $\mathcal{E}_1$, $\mathcal{E}_2$, $\mathcal{E}_3$ are the layered encryption of GR. The layer consists of multiple non-malicious data requests of different machines arriving for garlic encryption. Each D'_{m_i} belongs to a particular machine of the IIoT system.

$$D_{m_1}, D_{m_2}, \ldots, D_{m_n} \in M_1, M_2, \ldots, M_y \tag{30}$$

$$\text{s.t. } D_{m_i} \overset{\text{associated to}}{\longrightarrow} M_i \tag{31}$$

where M_x and M_y are the sending and receiving machines, forwarding their own data requests to perform a shared task. Garlic encryption uses randomness inside their routing patterns to obfuscate the network traffic; this approach reduces the traffic analysis attacks on the GR network. To perfectly acquire the respective D_{m_i} at the destination machine M_y, the GR network uses TCP sequence numbers $\mathcal{S}_i$ that append with each D_{m_i}. GR operates on symmetric and asymmetric cryptography, i.e., ElGamal and AES algorithms, to authenticate and prevent the integrity of D_{m_i}. Furthermore, the ElGamal algorithm formulates an encrypted session between sending and receiving machines for data request sharing. As the GR network uses multiple messages, it has multiple session tags S_{t_i} associated with each data request between machine x and y.

$$(M_x \xleftrightarrow[D_{m_i}]{\text{Data request}} M_y) \xrightarrow[\text{a session tag}]{\text{is associated with}} S_{T_i} \tag{32}$$

Next, an AES algorithm encrypts the D_{m_i} which is destined for the destination machine M_y. The combined message, i.e., ElGamal encrypted session block ζ_{enc} and AES encrypted data request block ψ_{enc}, is relayed through the GR network to reach the destination securely. To reduce the complexity of garlic encryption, we assume a machine M_x poses a single encrypted session tag S_{t_i} and data request D_{m_i} along with its sequence number $\mathcal{S}_i$.

$$M_x \xrightarrow{\text{poses}} \zeta_{enc}(S_{t_i}) + \psi_{enc}(D_{m_i} + \mathcal{S}_i) \tag{33}$$

The Elgamal block is 222-byte in size and comprises a 32-byte of the session key, 32-byte of initialization vector, and 158-bytes of random padding. Contrary, the AES block is of variable size but always in multiples of 16-bytes. It encrypts and decrypts the D_{m_i} using the shared secret key S'_{o_i} as shown in Fig. 2 shared between M_x and M_y. Next, the combined encrypted block $\mathbb{E}_d$ comprises multiple messages of different machines and is forwarded to the GR network to improvise the security and reliability of MTC.

$$\mathbb{E}_d = \zeta_{enc}\left(\sum_{i=0}^{y} S_{t_i}\right) + \psi_{enc}\left(\sum_{i=0}^{y} D'_{m_i} + \sum_{i=0}^{y} S_i\right) \tag{34}$$

where, $\zeta_{enc}(\sum_{i=0}^{y} S_{t_i})$ is an ElGamal encrypted block that includes all session tags of machines, $\psi_{enc}(\sum_{i=0}^{y} D'_{m_i} + \sum_{i=0}^{y} S_i)$ is an AES encrypted block that consists of data request and sequence number of all different machines MTC. Moreover, the GR network consists of routers, and each router has a unidirectional virtual tunnel to send and receive the data requests. Router identity is a cryptographic identity assigned to each router in the GR network. A router communicates with other routers using transport layer protocols such as TCP and User Datagram Protocol (UDP). The sending machine M_x can connect and lease these routers for sending and receiving data request D_{m_i}. In addition, the GR network also maintains a secure Network Database (NetworkDb) using a modified Kademlia algorithm to store each Router's Information (RouterInfos) and the routers which are on a lease (LeaseSets). The information has a Time To Live (TTL); after a particular TTL, the information is dropped to make room for newer entries. While the data request traverses from one router to another, it parallelly computes the Distributed Hash Table (DHT), which assists them in having an optimized and secure path from the source and destination machine. Consequently, all routers have multiple paths to transit a data request. However, we need an approach that finds the best router from the set of routers in the GR network. Therefore, we have incorporated a concept of profiling ϖ, where a router nominates the other router based on his previous performance.

Furthermore, the GR network has ElGamal and AES-based encrypted tunnel to secure the garlic wrapped message as shown in Fig. 6. Here, the sending machine M_x operates on an outbound tunnel Υ_{out} to shove their data request, and receiving machine M_y uses an inbound tunnel Υ_{in} to fetch the incoming data request of M_x. Likewise, when M_y replies back to the M_x, it utilizes his outbound tunnel, and M_x uses his inbound tunnel to fetch the reply back message from M_y. The Υ_{out} consists of outbound gateway O_g, outbound participants O_p, and outbound endpoint O_e; similarly, the Υ_{in} consists of inbound gateway I_g, inbound participants I_p, inbound endpoint I_e.

$$O_g, O_p, O_e \in \Upsilon_{out} \tag{35}$$

$$I_g, I_p, I_e \in \Upsilon_{in} \tag{36}$$

The combined garlic-wrapped encrypted block $\mathbb{E}_d$ reaches the outbound gateway O_g. Then, it nominates the next router as outbound participants using peer selection. This happens because each router in the GR network maintains the statistical profile - "profiling" ϖ of other routers. The profile information includes the router's speed, duration to accept the request, failure time, last connection and disconnection time, etc., to select appropriate routers that provide complete anonymity, effective performance, and better reliability.

$$\mathbb{E}_d \overset{\text{reaches}}{\longrightarrow} \Upsilon_{out} \tag{37}$$

$$\Upsilon_{out} \overset{\text{selects the best}}{\longrightarrow} O_{g_1}, O_{g_2}, \ldots, O_{g_l} \tag{38}$$

$$\text{s.t.} \varpi(O_{g_d}) \gg \varpi(O_{g_e}) \tag{39}$$

Likewise, the O_{g_d} finds the next suitable outbound participants O_{p_d} to efficiently transit the $\mathbb{E}_d$ to the inbound gateway of destination machine M_y. This makes the GR network competent compared with the OR network since there is no directory server that explicitly provides the fixed routers between source and destination in the GR network. Instead, here a secure and dynamic path is created using profiling ϖ of the routers. Once the $\mathbb{E}_d$ reaches the destination machine M_y, it immediately sends an acknowledgment message to the sending machine M_x through his outbound tunnel Υ_{out} to enhance the reliability of the GR network. Further, the M_y maintains the storage of session tags S_{t_i} from the previous communication inside the immutable blockchain. Storing S_{t_i} inside an IPFS-based blockchain reduces the data integrity and injection attacks, as any minute change in the data request by attackers gets recorded in the blockchain transaction and is overseen by the authorized blockchain members. If the data request is manipulated, the blockchain discards the transaction and removes it from the decentralized ledger.

Upon receiving the $\mathbb{E}_d$, the M_y checks the first 32-bytes of the S_{t_i} to compare it with the available session tags. Storing a session tag benefits the M_x and M_y. This is because, next time, the M_x wants to share D_{m_i} to M_y, instead of generating a new session tag S_{t_i} and encrypting it using ElGamal encryption ζ, it uses the previously generated session tag S'_{t_i} and encrypts the D_{m_i} using the AES algorithm like before. It is inevitable to store the session tags inside the blockchain; if not, then an attacker can fetch the session tag and try to exploit the data request of machines.

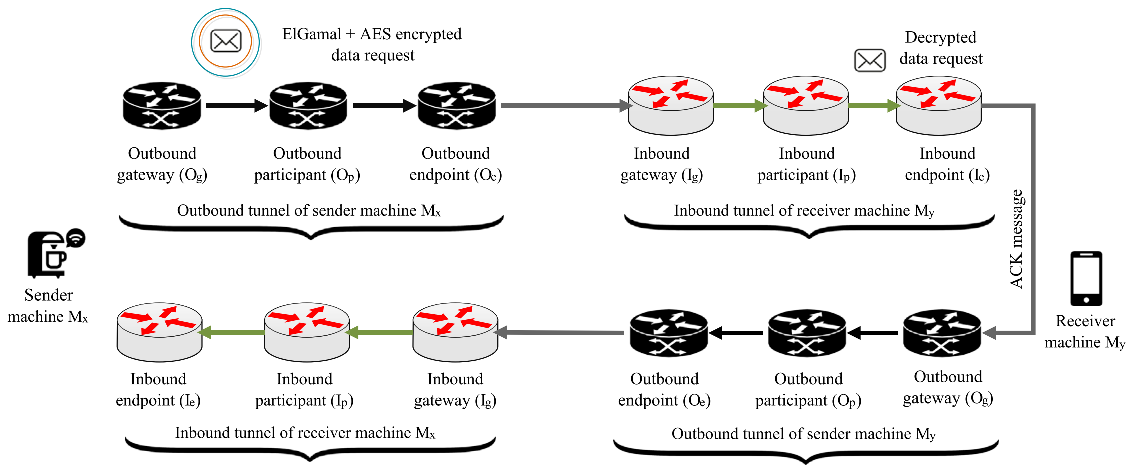


Fig. 6. Garlic routing outbound and inbound tunnels.

$$\mathbb{E}_d = \sum_{i=0}^{y} S'_{t_i} + \psi_{enc}\left(\sum_{i=0}^{y} D'_{m_i} + \sum_{i=0}^{y} \mathcal{S}_i\right) \tag{40}$$

where S'_{t_i} is a previously delivered session tag, it is then prepended without encryption with AES encrypted block ψ_{enc}. When the $\mathbb{E}_d$ is received by the M_y, it simply verifies the session tag; if the new S_{t_i} matches with the available session tag S'_{t_i}, then M_y directly decrypts the ψ_{enc} using his private key. Otherwise, the new S_{t_i} is decrypted using ElGamal private key and stored inside the blockchain.

$$\mathbb{E}_d \overset{\text{reaches}}{\longrightarrow} M_y \tag{41}$$

$$\mathbb{D}_d = \begin{cases} S_{t_i} = S'_{t_i}, & \text{decrypts the } \psi_{enc} \\ S_{t_i} \neq S'_{t_i}, & \text{decrypts the } \zeta_{enc} \text{ and } \psi_{enc} \end{cases} \tag{42}$$

$$\mathbb{D}_d = (D_{m_1}, D_{m_1}, \ldots, D_{m_n}) + (\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_n) \tag{43}$$

$$\text{s.t. } D_{m_i} \overset{\text{associated to}}{\longrightarrow} \mathcal{S}_i \tag{44}$$

where $\mathbb{D}_d$ is the decrypted message acquired from the inbound gateway I_g of the destination machine M_y. The $\mathbb{D}_d$ consists of all the data requests from different devices of MTC along with its sequence number. With the help of the TCP sequence number, each destination machine finds its data request from the set of the data request message to accomplish the desired task of the IIoT system. As we can observe that Eq. (44) efficiently secures the D_{m_i} that has been formulated as an objective function in Eq. (5).

Fig. 7 depicts the sequential procedure of the proposed framework *GRADE* that involves the application layer, intelligence layer, and security layer.

5. Results and discussion

In this part, the test dataset is used to evaluate our proposed LSTM-based model framework. A few typical evaluation metrics are used to get the results. To acquire the optimal working model, the generated results with varied parameters are compared to one another. Next, we have discussed and illustrated results based on the GR network, blockchain scalability, data request compromised, and the 6G network packet loss ratio.

5.1. LSTM-based results

Model evaluation and validation by the metrics are required to investigate the performance of the trained LSTM models. Precision, recall, and F1-score [28] are the metrics used to evaluate the model's performance. The fraction of actually predicted positive values over actual possible ones is reflected in the recall score. On the other hand, the precision score can be defined as the percentage of actually anticipated positive values that outnumber the actual ones. For prediction performance [29], the F1-score maintains the overall balance between precision and recall levels.

On the training and test sets, the performance of models with different optimizers is compared in Table 4. For training purposes, a constant learning rate of 0.001 is used for all optimizers. Models trained with the Nadam and adam optimizers perform equally well on both the train and test sets. However, the metrics score for Stochastic Gradient Descent (SGD), adagrad, and adadelta optimizer is lower than Adam and Nadam. It is worth noting that the findings mentioned above are obtained by maintaining the overall model architecture, the number of epochs, and the batch size constant. In addition, curves of Loss vs. Iterations and Accuracy vs. Iterations are plotted to examine the degree of convergence

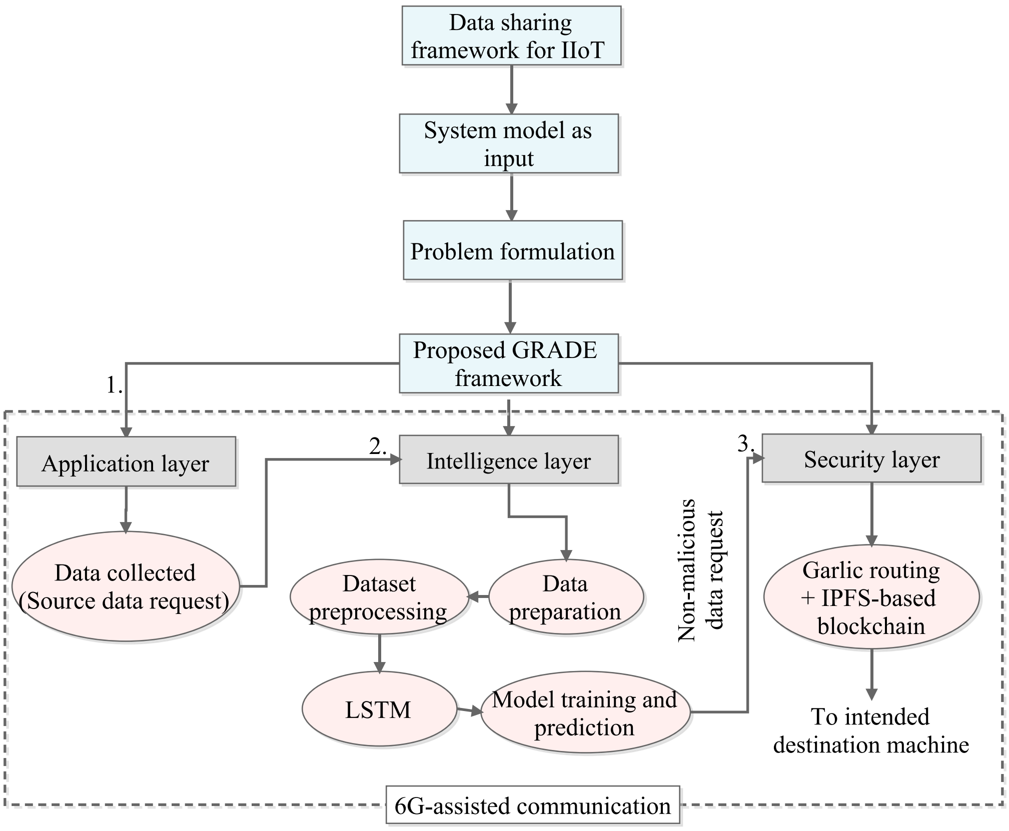

Data sharing framework for IIoT
System model as input
Problem formulation
Proposed GRADE framework
1.
Application layer
2.
Intelligence layer
3.
Security layer
Data collected (Source data request)
Dataset preprocessing
Data preparation
Non-malicious data request
Garlic routing + IPFS-based blockchain
LSTM
Model training and prediction
To intended destination machine
6G-assisted communication

Fig. 7. *GRADE*: Sequential flow of the *GRADE*.

Table 4
Comparison of model performance for different optimizers.

Optimizer	Test dataset			Training dataset		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score
Adam	0.9844	0.9795	0.9815	0.9853	0.9806	0.9825
Adadelta	0.2244	0.0789	0.1103	0.2131	0.0725	0.1016
Nadam	0.9863	0.9834	0.9839	0.9871	0.9842	0.9847
Adagrad	0.8694	0.7112	0.7679	0.8678	0.7097	0.7675
SGD	0.9321	0.7238	0.8019	0.9313	0.7256	0.8029

of models over multiple iterations.

Fig. 8 shows a comparison of train and test set accuracy for the trained LSTM model with the nadam optimizer. The train and test sets are randomly sampled in 10 batches each of size 512, which are then used to calculate prediction accuracy scores. The cross-validation batch number and accuracy scores are represented on the x-axis and y-axis. The graph clearly illustrates that the model's prediction accuracy on the train and test sets are nearly equal. This is determined by computing the absolute difference between each point's values. The difference between the test and train accuracy scores for batch number 0 is determined to be roughly 0.01 (train accuracy = 0.989, test accuracy = 0.979 and difference = 0.01). There is a small difference between test and train accuracy ratings for batches 1–9. As a result, it can be understood that the trained model does not overfit the dataset.

Fig. 9(a) shows loss curve during the training. The model with adam and Nadam as optimizers converges faster than the other optimizers for a given number of iterations used in training. Similarly, for Fig. 9(b), the accuracy score tends to increase faster for Nadam and adam, while adagrad, adadelta, and SGD show a comparatively slower convergence rate.

Table 5 presents a classification report for the trained LSTM model with Nadam optimizer. Class-wise precision, recall, and F1-score values are obtained to effectively evaluate the model and tackle the overfitting problem. It can be clearly seen that all the other classes have considerably better scores except for two, i.e., Reverse Shell and Man in the Middle. This is because of the low support count of Reverse Shell and Man in the Middle class. A low support count denotes insufficient data for class to train the model; hence, considerably low classification scores are produced.

In Fig. 9(c), a bar plot is used to assess the confidence scores of class prediction for different test samples. Seven samples are chosen at random

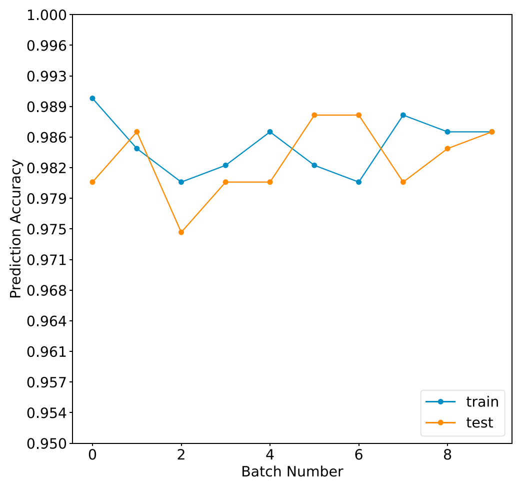


Fig. 8. On the LSTM model with nadam optimizer, comparing prediction accuracies for train and test datasets.

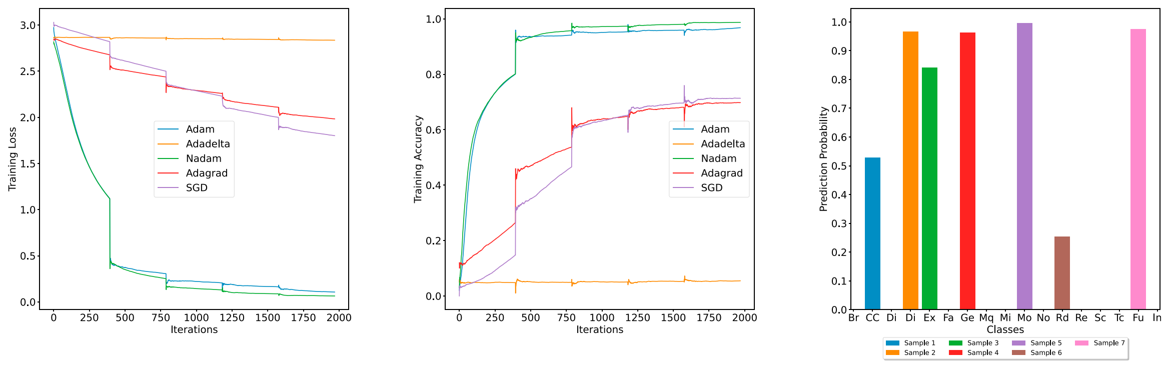
along with the LSTM model with a nadam optimizer from the test set for class prediction. Class-wise probability values are calculated for each sample, creating a bar plot. The y-axis is the confidence score of the predicted class. The x-axis represents the target classes that are taken into account. As a result, the x-axis represents the predicted class for each of the samples considered.

5.2. Garlic routing evaluation

The GR network in the proposed *GRADE* framework is a peer-to-peer network that facilitates an encrypted tunnel to securely traverse sensitive information of MTC. We want to mention that there is hardly any literature, tools, or simulation available that helps us to create the GR network. Therefore, we have utilized a GR-based framework, i.e., Invisible Internet Project (I2P) [30], to configure ElGamal and AES-based encrypted tunnel. We have manually set up the TCP-based outbound and inbound tunnel by changing the design configuration, network parameters, and tunnel type between sending machine M_x and receiving machine M_y in the tunnel wizard of the I2P framework. We have used the SOCKS5 proxy tunnel that permits both TCP and UDP functions for a seamless exchange of MTC data. Once the tunnel gets ready, we have forwarded the garlic-wrapped messages between multiple clients. However, the GR encryption's consistency needs to be evaluated; therefore, we have performed several attacks, such as Distributed Denial-of-Service (DDoS), injection, session hijacking, and privilege escalation attacks on the GR network. In the backend, we have used sniffers that log the attack and normal traffic of the GR.

The packet capture file from the sniffer is hard to interpret as it is entirely encrypted. Therefore, we utilized Zeek [31] - an open-source Deep Packet Inspection (DPI) tool to analyze the packets at their root level that includes packet headers, protocols, defined criteria of the packet, and other behavioral activities of the packet. This helps us to understand the robustness and reliability of the GR network. Fig. 10(a) shows the probability of the data request being compromised when the tunnel participants are increasing. From the graph, we can depict that as the number of encrypted tunnels increases, the rate of compromising data requests of machines decreases. This happens because the data requests are traversing from the ElGamal and AES-based encrypted tunnel; hence, there is a slight possibility that an attacker can exploit the data request.

Further, the proposed framework is evaluated with other baseline works, i.e., [3,22]. The authors of [3] proposed an OR network to secure mission-critical data of battlefield operations. However, they have not utilized the benefits of AI intelligence, which implies that there is no classification between malicious and non-malicious data requests. Therefore, the OR network has to process both malicious and non-malicious data requests where the malicious data request can compromise the OR circuit and manipulate the original data request. Jadav et al. in Ref. [22] overcome the issue of [3] and employs AI intelligence in the framework, which means the OR network has to process only the non-malicious data requests. However, the major challenge of the OR network is time analysis and exit node vulnerability attacks that raise the chances of data compromisation. Therefore, the proposed work employs both GR network and AI intelligence to overcome the aforementioned issues. This is because the GR network has TCP-based inbound



(a) Loss curve. (b) Accuracy curve. (c) Prediction probabilities.

Fig. 9. Loss and accuracy curves for different model optimizers during the training process. Prediction probabilities of a few random test samples w.r.t. target class labels.

Table 5
Classification report for the trained LSTM with Nadam optimizer.

Target Classes	Precision	Recall	F1-Score
Brute Force	0.99	0.90	0.94
Command-And-Control	0.90	1.00	0.95
Dictionary	1.00	1.00	1.00
Discovering Resources	0.99	0.98	0.99
Exfiltration	1.00	1.00	1.00
False Data Injection	1.00	0.98	0.99
Generic Scanning	1.00	0.89	0.94
MQTT Cloud Broker Subscription	1.00	1.00	1.00
Man-in-the-Middle	0.00	0.00	0.00
Modbus Register Reading	1.00	1.00	1.00
Normal	1.00	1.00	1.00
Ransom Denial of Service	1.00	1.00	1.00
Reverse Shell	0.62	1.00	0.76
Scanning Vulnerability	0.79	0.98	0.88
Transmission Control Protocol Relay	0.96	1.00	0.98
Fuzzing	0.86	1.00	0.93
Insider Malicious	1.00	1.00	1.00

and outbound tunnels, which employ ElGamal and AES-based encryption to encrypt the data requests. As these tunnels are increasing in the *GRADE* the chances of getting compromised get reduced. From the graph, we can observe that the blue line (work of [3]) has a high compromised rate than the red line (work of [22]). Needless to say, the *GRADE* has the lowest compromised rate, i.e., 18.5% than [3,22] because it uses the indispensable characteristics of GR network and AI intelligence.

To enable a 6G network in the I2P framework, we first have simulated the 6G network in the MATLAB by configuring the parameters of the 5G toolbox; once we achieved the nearby result of the 6G network, we took those parameters and inserted them into the network manager of the I2P framework. Fig. 10(b) shows the probability of packet loss comparison between cellular network interfaces such as 4G, 5G, and 6G. The essential features of a 6G network like high throughput (1 Tbps), high mobility support (1000 km/h), processing delay (< 10 ns), and reliability (10^{-9}) make the *GRADE* framework highly scalable and efficacious. The essential benefits of a 6G network enhance the latency and data rates of the proposed framework, resulting in minimal time to forward the data request from one machine to another. Further, the 6G-based GR network reduces the packet loss ratio because the data requests are not buffered at the inbound/outbound tunnel, and it is seamlessly forwarded to the next tunnel due to the high data rates of a 6G network. The conventional 5G network has lower data rates (1 Gbps), high latency(10 ms), and low reliability (99.999%) compared to a 6G network. Therefore, from the graph, it is clear that due to the staggering features of the 6G network, there is a minimal packet loss, i.e., 24.3%, when compared with the 5G and 4G networks.

Next, an IPFS protocol has been used in the *GRADE* framework to store the ElGamal encrypted session tags associated with multiple data requests forwarded to the destination machine. Fig. 10(c) shows the scalability comparison of the IPFS-based proposed scheme with the traditional blockchain scheme. The IPFS-based blockchain converts the raw session tags into the hashed session tags, resulting in the faster fetching of the hashed tags compared to the raw session tags. This implies that the proposed framework has a better latency due to the involvement of IPFS and a 6G network. Moreover, due to the low latency, the proposed framework can process larger number of data requests, resulting in a highly scalable framework. Fig. 10(c) shows the improved scalability, i.e., 47.2% of the IPFS-based proposed framework when compared with the scalability, i.e., 24.1% of the traditional blockchain-based storage. This is due to the property of IPFS, which involves storing data in the form of hash instead of storing complete raw data in the blockchain.

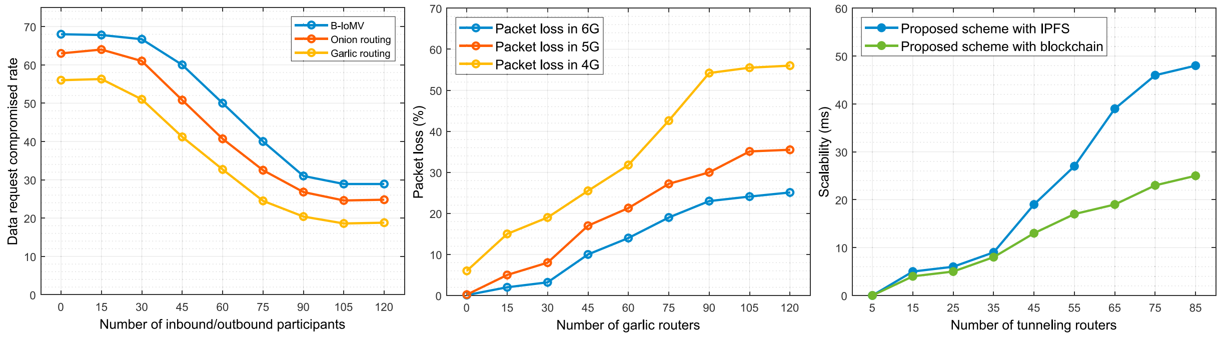


Fig. 10. (a) Probability of data request being compromised, (b) Packet loss comparison, and (c) Scalability comparison.

6. Discussion

The advent of IoT technology enables remote devices, such as sensors, actuators, connectors, and controllers, to interconnect with each other, effectively sense the environment, and intelligently collect the data. The expansion of IoT technology in all avenues boosts the nation's economic growth, especially in industrial sectors. IIoT is one such technological advancement that offers better operational efficiencies, improved monitoring and maintenance, reduced delay, increased productivity, and many more in the IoT environment. MTC is a subset of IIoT technology that offers interactions between machines without human intervention to accomplish the shared objective of the IIoT system, such as sharing the telemetry data from the car to the centralized computer, forwards the energy usage to the energy provider in smart grids, tracks the asset using MTC-enabled GPS trackers, and many more. However, it is hurdled by security threats that hinder the massive roll-out of IIoT applications. MTC uses the conventional Internet, which is pervasive to numerous hardware and software attacks. For instance, an attacker can manipulate the Supervisory Control and Data Acquisition (SCADA) systems to maneuver the IIoT operations, execute the malware on Programmable Logic Controllers (PLCs) to exploit the functions of applications, such as water treatment plants, nuclear power stations, smart grids, and many more.

In addition, the attackers can counterfeit sensitive data of MTC, raising the data integrity questions in the IIoT environment. Further, the conventional Internet impedes precarious routing protocols that leave various footprints, such as IP addresses, port numbers, applications name, hardware addresses, etc., resulting in passive attacks like social engineering attacks, sniffing, and phishing. Therefore, there is a stringent requirement for a robust and intelligent framework, i.e., garlic routing and AI techniques that protect MTC from nefarious security threats. The involvement of garlic routing is because it offers multi-layer encryption and high-level anonymity to the machines of MTC. On the contrary, onion routing is also a variant of garlic routing, which delivers similar services, but it suffers from time analysis and exit node vulnerability. The multi-layer encryption of garlic routing is highly intricate to the attacker that is performing attacks like time analysis, node exploitation, sniffing, and many more; therefore, it is feasible to incorporate garlic routing instead of onion routing. Toward this goal, the proposed framework *GRADE* utilizes garlic routing encryption to encrypt the data requests of multiple machines of the IIoT systems. The GR encryption is a robust encryption mechanism that is difficult to break by the attackers; this can be seen by the data request compromised rate, i.e., 18.5% as shown in Fig. 10(a), which is far better than the baseline approaches, such as [3, 22].

Moreover, GR is associated with the IPFS-based blockchain that securely stores the session tags of the data request. The involvement of IPFS-based blockchain reduces the data integrity attacks and raises the scalability, i.e., 47.2% as shown in Fig. 10(c) of the *GRADE*. This happens because the IPFS converts the session tags into hashed session tags that are easy and faster to fetch from the blockchain network than the raw session tag. Furthermore, the overwhelming learning property of the LSTM model with the time-series dataset makes it a competent contender against other AI models. The LSTM model uses various optimizers, such as Nadam, Adam, Adadelta, Adagrad, and SGD, to predict the class labels of malicious and non-malicious data requests, wherein the Nadam optimizer outperforms other optimizers in terms of accuracy, i.e., 98.9% as shown in Fig. 9(b) because it requires fewer parameters and has faster computation time than others. The LSTM model is deployed and executed on the edge servers to reduce the computational cost of the *GRADE*. Further, the *GRADE* utilizes the essential properties of a 6G network, such as peak data rates ($1 \geq$ Tbps), higher mobility ($\geq$1000 km/h), low latency (10–100 μs), high end-user data rate (1 Gb/s), dense connectivity (10^7 devices/km^2), and higher availability (99.99999%) that offers high scalability and lower latency in the IIoT systems. Hence, the adopted 6G network shows a remarkable improvement in the packet loss ratio, i.e., 24.3% in Fig. 10(b) of the *GRADE* framework.

Finally, the computational complexity of *GRADE* is analyzed by observing a few points in the intelligence and security layer of the *GRADE*. First, we reduce the computational overhead of the GR network by effectively classifying the malicious and non-malicious data requests of the MTC machines. Only the non-malicious machine data requests are forwarded to the GR network. This relieves the overhead of the GR network, which otherwise has to process both non-malicious and malicious data requests. Also, we want to mention that the execution of the LSTM model is inside the edge servers, which likely reduces the computational cost of the *GRADE*. Unlike the work of [3] that has not adopted the crucial benefits of AI techniques; hence, their scheme has to process both malicious and non-malicious data requests, which raises the computational complexity of their scheme. Secondly, IPFS-based blockchain uses the hashed session tags instead of the raw session tags of the machine's data requests. The hashed data is faster in transit due to its smaller size than the original raw data, resulting in rapid access to the data requests. Additionally, it reduces the data storage cost because the Ethereum-based blockchain is highly costly; for instance, storing a word of 256-bit costs approximately $528.3 dollar. However, the IPFS-based blockchain offers free storage to store the data. Hence, the incorporation of IPFS benefits the *GRADE* and reduces the computational complexity and data storage costs. By contrast, the work of [22], which has not used IPFS in the blockchain network, has to store the raw data instead of hashed data. This results in high computational complexity that deteriorates the overall performance of the proposed framework. From the aforementioned discussion and the result analysis of the proposed work showcase that the *GRADE* has the potential to efficiently detect security attacks and restrict them from MTC in the IIoT environment.

7. Conclusion

Encompassing a broad range of skillful applications, MTC emerged as a robust technology that enables proactive mechanisms to develop cutting-edge smart IIoT applications. To study the security aspects of MTC, in this paper, we considered a blockchain and GR-based secure data exchange framework, i.e., *GRADE* for IIoT applications. First, we have highlighted the security and privacy concerns that imperiled the performance of the MTC devices. Thus, we have used an intelligence layer which bifurcates the non-malicious and malicious data requests of MTC. The AI-based LSTM model adroitly assisted the intelligence layer in predicting the output label, which is then evaluated using different performance metrics such as precision, recall, and F1-score. Further, to securely route the non-malicious data requests between MTC devices, we have introduced an anonymous and encrypted network, i.e., the GR network that uses ElGamal and AES-based encryption. Further, the GR network employs the IPFS-based blockchain that validates the session tags of each data request that raises the reliability and scalability of the *GRADE*. The communication between MTC devices is incorporated with a 6G network interface due to its indispensable features. To evaluate the performance of the *GRADE* it is simulated inside the I2P framework where it is assessed against the data request compromising rate that remarks that as the tunnel routers increase, the comprising rate decreases. The *GRADE* outperforms other baselines works in terms of accuracy, i.e., 98.9%, compromised rate, i.e., 18.5%, scalability, i.e., 47.2%, and packet loss ratio, i.e., 24.3%.

For future work, we will formulate a real-time attack scenario where adversarial attacks and malware threats are peculiarly explored to evaluate the performance of the proposed *GRADE* framework along with its computational complexity.

Declaration of competing interest

There is no Conflict of Interest.

References

[1] V. Bolgouras, C. Ntantogian, E. Panaousis, C. Xenakis, Distributed key management in microgrids, IEEE Trans. Ind. Inf. 16 (3) (2020) 2125–2133.

[2] S. Gupta, B.L. Parne, N.S. Chaudhari, ISAG: IoT-enabled and Secrecy Aware Group-based handover scheme for e-health services in M2M communication network, Future Generat. Comput. Syst. 125 (2021) 168–187.

[3] R. Gupta, S. Tanwar, N. Kumar, B-IoMV: blockchain-based onion routing protocol for D2D communication in an IoMV environment beyond 5G, Vehicular Commun. 33 (2022), 100401.

[4] R. Attarian, S. Hashemi, An anonymity communication protocol for security and privacy of clients in IoT-based mobile health transactions, Comput. Network. 190 (2021), 107976.

[5] R.M. Parizi, S. Homayoun, A. Yazdinejad, A. Dehghantanha, K.-K.R. Choo, Integrating privacy enhancing techniques into blockchains using sidechains, in: 2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE), 2019, pp. 1–4.

[6] C. Qian, J. Shi, Z. Yu, Y. Yu, S. Zhong, Garlic cast: lightweight and decentralized anonymous content sharing, in: 2016 IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS), 2016, pp. 216–223.

[7] P. Zhang, C. Wang, C. Jiang, Z. Han, Deep reinforcement learning assisted federated learning algorithm for data management of IIoT, IEEE Trans. Ind. Inf. 17 (12) (2021) 8475–8484.

[8] M.M.N. Aboelwafa, K.G. Seddik, M.H. Eldefrawy, Y. Gadallah, M. Gidlund, A machine-learning-based technique for false data injection attacks detection in industrial IoT, IEEE Internet Things J. 7 (9) (2020) 8462–8471.

[9] S. Garg, K. Kaur, N. Kumar, J.J.P.C. Rodrigues, Hybrid deep-learning-based anomaly detection scheme for suspicious flow detection in SDN: a social multimedia perspective, IEEE Trans. Multimed. 21 (3) (2019) 566–578.

[10] R. Kakkar, R. Gupta, S. Tanwar, J.J.P.C. Rodrigues, Coalition game and blockchain-based optimal data pricing scheme for ride sharing beyond 5G, IEEE Syst. J. 16 (4) (2022) 6321–6327.

[11] X. Cai, S. Geng, J. Zhang, D. Wu, Z. Cui, W. Zhang, J. Chen, A sharding scheme-based many-objective optimization algorithm for enhancing security in blockchain-enabled industrial Internet of Things, IEEE Trans. Ind. Inf. 17 (11) (2021) 7650–7658.

[12] K. Yu, L. Tan, M. Aloqaily, H. Yang, Y. Jararweh, Blockchain-enhanced data sharing with traceable and direct revocation in IIoT, IEEE Trans. Ind. Inf. 17 (11) (2021) 7669–7678.

[13] G. Zhang, X. Zhang, M. Bilal, W. Dou, X. Xu, J.J. Rodrigues, Identifying fraud in medical insurance based on blockchain and deep learning, Future Generat. Comput. Syst. 130 (2022) 140–154.

[14] I. Budhiraja, S. Tyagi, S. Tanwar, N. Kumar, J.J.P.C. Rodrigues, DIYA: tactile Internet driven delay assessment NOMA-based scheme for D2D communication, IEEE Trans. Ind. Inf. 15 (12) (2019) 6354–6366.

[15] K. Balasubramanian, S. Kannan, Onion routing in anonymous network, Appl. Math. 13 (S1) (2019) 247–253.

[16] J. Hur, D.K. Noh, Efficient and secure identity-based onion routing, J. Res. Pract. Inf. Technol. 46 (2014) 1–6.

[17] Y. Zhang, J. Weng, J. Weng, M. Li, W. Luo, Onionchain: towards Balancing Privacy and Traceability of Blockchain-Based Applications, 2019, pp. 1–13.

[18] M. Sayad Haghighi, Z. Aziminejad, Highly anonymous mobility-tolerant location-based onion routing for VANETs, IEEE Internet Things J. 7 (4) (2020) 2582–2590.

[19] M.S. Ali, M. Vecchio, G.D. Putra, S.S. Kanhere, F. Antonelli, A decentralized peer-to-peer remote health monitoring system, Sensors 20 (6) (2020) 1–18.

[20] I.-H. Liu, Y.-L. Chang, J.-S. Li, C.-G. Liu, Differential privacy protection with group onion routing based on ai-based url classification, in: Proceedings of the 2020 ACM International Conference on Intelligent Computing and its Emerging Applications, ACM ICEA '20, 2016, pp. 1–6.

[21] S. Tiwari, An ensemble deep neural network model for onion-routed traffic detection to boost cloud security, Int. J. Grid High Perform. Comput. (IJGHPC) 13 (1) (2021) 1–17.

[22] N.K. Jadav, R. Gupta, M.D. Alshehri, H. Mankodiya, S. Tanwar, N. Kumar, Deep learning and onion routing-based collaborative intelligence framework for smart homes underlying 6G networks, IEEE Transact, Network Service Manage. 19 (3) (2022) 3401–3412.

[23] M. Al-Hawawreh, E. Sitnikova, N. Aboutorab, X-IIoTID: A connectivity-and device-agnostic intrusion dataset for industrial Internet of Things, IEEE Internet Things J. 9 (5) (2022) 3962–3977.

[24] G. Rani, M. Oza, V. Dhaka, N. Pradhan, S. Verma, J. Rodrigues, Applying deep learning-based multi-modal for detection of coronavirus, Multimed. Syst. 28 (2022) 1251–1262.

[25] J. Donahue, L.A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, T. Darrell, Long-term recurrent convolutional networks for visual recognition and description, IEEE Trans. Pattern Anal. Mach. Intell. 39 (4) (2017) 677–691.

[26] M. Ma, Z. Mao, Deep-convolution-based LSTM network for remaining useful life prediction, IEEE Trans. Ind. Inf. 17 (3) (2020) 1658–1667.

[27] S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, in: Proceedings of the 32nd International Conference on International Conference on Machine Learning, ICML'15, 2015, pp. 448–456.

[28] S.-H. Ou, C.-H. Lee, V.S. Somayazulu, Y.-K. Chen, S.-Y. Chien, On-line multi-view video summarization for wireless video sensor network, IEEE J. Select. Topics Signal Process. 9 (1) (2014) 165–179.

[29] X. Zhou, Y. Hu, W. Liang, J. Ma, Q. Jin, Variational LSTM enhanced anomaly detection for industrial big data, IEEE Trans. Ind. Inf. 17 (5) (2020) 3469–3477.

[30] I2P - the invisible Internet Project. https://geti2p.net/en/, 2021. (Accessed 21 December 2021).

[31] Zeek - an open source network security monitoring tool. https://zeek.org/, 2021. (Accessed 21 December 2021).