Enhancing Network Communication Security Using Hybrid Cryptographic Techniques

Qing He

Department of Information Engineering, Southwest Jiaotong University Hope College, Chengdu 610400, China
E-mail: qinghe015@outlook.com, heqingcss@163.com

Received 16 March 2026; Accepted 29 May 2026

Abstract

The security requirements of networked systems have become increasingly critical due to the growing need for intelligent systems that can detect intrusions and protect data during transmission. This study presents a network security system that combines deep learning-based traffic trust assessment with two different cryptographic protection methods. The system employs a DenseNet–BiGRU design to capture network traffic patterns across different spatial and temporal dimensions, enabling the system to distinguish between normal and malicious traffic before the data is encrypted. The system uses Elliptic Curve Cryptography (ECC) to secure session establishment for trusted traffic, which enables key exchange and implements Advanced Encryption Standard (AES) for data encryption that requires low computational resources. The proposed framework reaches an accuracy of 94.5%, together with a precision of 88.8%, recall of 82.3%, F1-score of 85.4% and Matthews Correlation Coefficient (MCC) of 0.82, which demonstrates its ability to detect under conditions of class imbalance. The model demonstrates

Journal of Cyber Security and Mobility, Vol. 15_4, 1053–1086.
doi: 10.13052/jcsm2245-1439.15410

exceptional ability to differentiate between classes, which results in an ROC-AUC of 0.96 and PR-AUC of 0.93. The analysis of cryptographic performance shows that encryption and decryption process times remain minimal while system performance maintains consistent throughput, which increases with larger payloads. The framework demonstrates its ability to detect attacks in real time while maintaining secure communication, which makes it suitable for modern network protection and IoT security frameworks.

Keywords: Intrusion detection system, hybrid cryptography, deep learning, secure network communication, traffic trust verification.

1 Introduction

The rapid growth of networked systems, cloud computing, and Internet services has significantly increased the volume and complexity of transmitted data [1]. Although this expansion improves data sharing efficiency, it also introduces serious security threats [2]. Among the security threats are eavesdropping, data tampering, replay attacks, and sophisticated intrusion attempts [3]. Traditional security mechanisms typically focus either on attack detection or data encryption, but not both together. However, it is rare to find a security mechanism that offers the advantage of both intelligent traffic validation and secure data transmission under a unified framework [4].

Traditional cryptographic methods, while good at ensuring confidentiality and integrity, do not take into account whether the communication is trustworthy or not [5]. The encryption of harmful or already-targeted traffic is a waste of computational resources and might even aid the attackers by blocking security monitoring systems. Static rule-based intrusion detection systems and basic machine learning models face difficulties in handling modern cyber-attack patterns which evolve throughout time together with their subsequent network traffic patterns [6]. These issues indicate the necessity for a comprehensive security strategy that would bring together traffic intelligence and strong cryptographic protection [7].

The deep learning technique appears to be effective because it enables scientists to analyze network traffic patterns that exhibit complex behavior [8]. The system establishes its framework through the deployment of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), which it uses to construct network flow patterns that identify legitimate and malicious network communications [9]. The security of communication channels needs more than deep learning traffic analysis because

unauthorized data access and network transmission alterations can occur even when legitimate data moves through the system [10].

To overcome those obstacles, the research proposes a complete security solution that protects network communications through the combination of deep learning traffic analysis and hybrid cryptographic systems [11]. Practical communication scenarios are modelled using realistic network data from the CICIDS-2017 dataset [12]. DenseNet is used to detect spatial relationships between the traffic features, while BiGRU is applied to realize the temporal dependencies and the evolution of the communication behaviors [13]. The strategy of having two models guarantees that only the network flows that are authenticated are allowed to move on to the communication stage. After the traffic verification process, a hybrid cryptographic system is employed to secure the communication that has been authenticated [14]. Asymmetric cryptography is used for secure key exchange, while symmetric encryption protects the data content, and cryptographic hashing secures the integrity [15]. The proposed framework is able to carry out both efficient threat mitigation and strong communication security by decoupling the decision phase (traffic trust assessment) from the protection phase. The primary contribution of the paper is described below.

The proposed framework introduces a unique integration strategy by decoupling Traffic Trust Verification from the encryption process using a DenseNet–BiGRU-based spatial and temporal learning model. This selective encryption mechanism ensures that only validated traffic is processed for secure transmission, which improves efficiency and reduces unnecessary cryptographic overhead compared to conventional integrated security systems.

- Design a hybrid cryptographic security mechanism that uses ECC to secure session keys and Advanced Encryption Standard (AES) to encrypt data, which provides strong protection but requires less processing power.
- Implement a selective encryption system that protects only trusted network traffic to deliver better encryption efficiency while decreasing network delays and stopping attackers from using encrypted paths to conduct their operations.
- Evaluate the integrated ECC–AES encryption framework to demonstrate its effectiveness for real-time network and IoT communication environments through its security performance, encryption/decryption speed, and throughput capacity.

The remainder of this paper is organized as follows. Section 2 reviews related work on intrusion detection systems, deep learning-based traffic analysis, and hybrid cryptographic techniques for secure network communication. Section 3 describes the proposed methodology in detail, including data preprocessing, Traffic Trust Verification using the DenseNet–BiGRU model, and the hybrid encryption mechanism based on ECC and AES. Section 4 presents the experimental results, which include detection accuracy and comparative analysis, along with cryptographic performance metrics for latency and throughput. Section 5 concludes the paper by presenting essential findings and suggesting possible research paths for future studies.

2 Literature Review

Rathod and Kotari [16] proposed a secure message transmission framework for MANETs by joining multipath routing with hybrid cryptography, thereby improving confidentiality and resistance against route failures. The method, however, does not include smart traffic analysis to tell apart the good and bad communications. Sharmila et al. [17] suggested a secure hybrid data transmission protocol for WSNs with a focus on the key management and message authentication that are efficient, making it appropriate for the resource-limited environments, but it does not have adaptive intelligence to cope with changing threats.

Nandanwar and Katarya [18] proposed a hybrid blockchain-based framework that secured intrusion detection systems in IoT networks and increased the trust and data integrity of IDS outputs. However, the framework has not directly considered secure data transmission mechanisms, although its reliability has been improved. Khagga et al. [19] merged sophisticated deep learning models with cryptographic techniques to create a QoS-aware secure routing protocol, thereby improving both routing efficiency and security awareness. However, the solution is primarily focused on routing optimization rather than end-to-end communication security.

Ye et al. [20] have done research on hybrid encryption and watermarking methods for safeguarding social images and came up with a solution that is very difficult to tamper with or access by unauthorized people; however, the technique is restricted to multimedia data types. Priya et al. [21] put forward an adaptive clustering and optimization-driven security scheme for MANETs, which makes the network more efficient and durable, although cryptographic protection is considered as a secondary measure rather than a primary security mechanism.

Ganesh et al. [22] proposed a user authentication system based on deep learning integrated with hybrid encryption for secure blockchain-assisted mobile edge computing, thereby smartly merging cryptographic protection with intelligence, but still giving priority to the security of authentication and storage. Kanneboina and Sundaram [23] used hybrid metaheuristic optimization in the Internet of Medical Things scenarios to raise the security performance, making the detection more accurate without directly dealing with secure communication channels. Mahdi et al. [24] devised a dynamic intrusion detection framework based on incremental learning and blockchain to deal with changing attacks in IoT networks, guaranteeing trust and scalability of the system, but not providing integrated encryption for data transfer. Fu et al. [25] introduced a powerful hybrid encryption system that unites AES and ECC to protect communication through smart service platforms, attaining the equilibrium between security level and computational cost, albeit not having intelligent traffic validation before encryption. Xiao [26] proposed a malware cyber threat intelligence system for IoT using machine learning to detect and analyze malicious activities in connected environments, thereby improving threat detection capability in IoT networks. However, the approach does not incorporate hybrid cryptographic mechanisms or selective encryption for secure data transmission. Gudivaka et al. [27] introduces an IIoT intrusion detection system using blockchain, ELCG-DSA signatures, PEC-Tiger hashing, and LWS-BiOLSTM for attack classification, achieving high accuracy. This work informs the proposed method by showing the value of combining blockchain security, cryptographic authentication, and deep learning for improved IIoT intrusion detection.

2.1 Problem Statement

Modern communication networks are constantly threatened by security issues due to changing traffic patterns and advanced cyberattacks [28]. The current cryptographic approaches provide data confidentiality and integrity with high efficiency, but do so blindfolded without knowing if the traffic is trustworthy, which sometimes leads to the encryption of the malicious traffic, thus incurring unnecessary computational overhead [29]. Deep learning-based intrusion detection systems focus on detecting malicious activities but fail to protect legitimate data transmissions. The systems fail to protect real-world environments because their traffic intelligence lacks integration with communication security systems [30]. An integrated system is required to perform network traffic assessment while using hybrid cryptography for trusted communications to achieve complete network security.

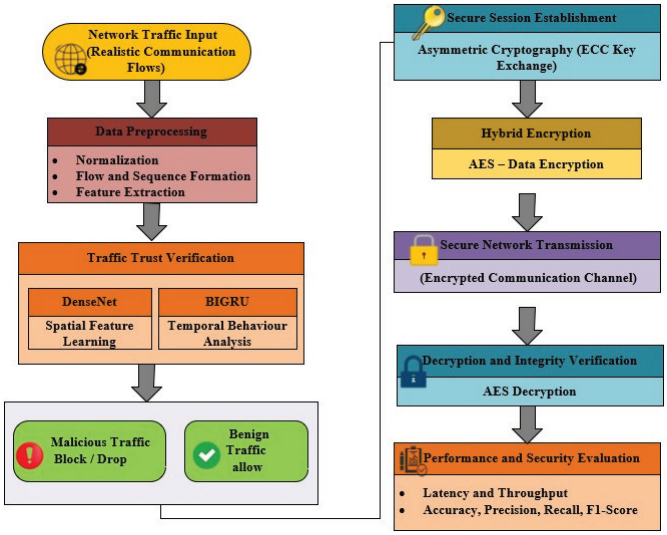

Figure 1 Proposed hybrid deep learning-driven secure network communication framework.

3 Proposed Methodology

The suggested approach fuses Traffic Trust Verification with encrypted data transfer to strengthen network security. First, network traffic is analyzed to detect malicious activity, thereby avoiding unnecessary encryption and reducing computational overhead. Next, the verified traffic is protected using hybrid cryptographic methods involving secure key exchange, fast encryption, and data integrity verification, ensuring efficient and secure communication. Figure 1 illustrates a complete secure network system that combines deep learning-based traffic trust assessment with dual cryptographic protection. The system shows how DenseNet–BiGRU intrusion detection operates with ECC key exchange and AES encryption to achieve both intelligent attack detection and secure data transmission.

3.1 Data Collection

The Network Intrusion Dataset is a modified network traffic dataset suitable for intrusion and anomaly detection tasks [31]. It consists of flow-level network features along with labels denoting normal and malicious activities. It can be used for both supervised and unsupervised learning experiments. The dataset is delivered in CSV format and is a light, research-friendly version of the intrusion detection data that is often used for model training and testing.

Table 1 Normal and anomalous traffic distribution across the CIC-IDS2017 dataset

Class Label	Description	Number of Instances	Percentage (%)
0	Normal (Benign) Traffic	≈2,273,000	≈80%
1	Anomalous (Attack)	≈557,000	≈20%
Total	—	≈2,830,000	100%

The total size of the download on Kaggle is approximately 100–200 MB, which varies with the version and preprocessing, thus making it easy to handle for quick testing as opposed to the full-scale raw network traffic datasets (https://www.kaggle.com/datasets/chethuhn/network-intrusion-dataset). Table 1 displays the normal and anomalous traffic distribution across the CIC-IDS2017 dataset while demonstrating the class imbalance that exists in actual network environments.

Table 1 shows the label-wise distribution of the CIC-IDS2017 dataset under a binary classification setting, including the number and percentage of Normal (Benign) and Anomalous (Attack) traffic instances. The dataset exhibits a severe class imbalance because normal traffic makes up most of the samples, which demonstrate actual network behavior. The study uses imbalance-aware evaluation metrics, including F1-score, Matthews Correlation Coefficient (MCC), ROC-AUC, and PR-AUC, to evaluate results because of this distribution pattern.

3.2 Data Preprocessing

CIC-IDS2017 network traffic is subjected to data preprocessing, which entails cleaning by getting rid of duplicates, missing values, and irrelevant features. Categorical attributes are transformed into a numerical form, while feature scaling is done to bring all data ranges to a common level. In order to tackle the issue of class imbalance between normal and attack traffic, the application of suitable balancing techniques are applied, and the processed data is divided into training and testing sets for the purpose of effective model learning and evaluation.

3.2.1 Normalization

The procedure of min-max normalization is utilized for this research work to bring all the network traffic features to the same range, which helps the deep learning models to be more stable in learning and quicker in convergence. This method retains the data distribution in its original form but, at the same time, it prevents features with very large numeric ranges from having the

upper hand during the training process. The normalization is defined as shown in Equation (1)

$$X_{\text{norm}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{1}$$

where X denotes the original feature value, and $X_{\min}$ and $X_{\max}$ are the minimum and maximum values of that feature, respectively, in the dataset. By using this normalization, all network attributes have the same scale, which helps facilitate the accurate detection of intrusions and the quick making of cryptographic decisions in the downstream process.

3.2.2 Flow and sequence formation

The outlined system handles network traffic in an integrated way that combines secure communication with intelligent intrusion detection. Initially, the network traffic $T = \{t_1, t_2, \ldots, t_n\}$ obtained from the CIC-IDS2017 dataset of the Canadian Institute for Cybersecurity is subjected to a cleaning and scaling procedure to facilitate consistent feature representation. The normalization by min-max method is utilized for each feature as defined in Equation (2)

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{2}$$

The generation of normalized feature vectors that are appropriate for deep learning analysis. Subsequently, the normalized traffic is passed through a deep learning classifier, which recognizes spatial and temporal patterns and outputs a classification as in Equation (3)

$$y = f(X', \theta) \tag{3}$$

where y is an element of $\{\text{Normal, Anomaly}\}$ and θ is the set of parameters of the trained model. Considering the classification result, a trust decision mechanism allows only legitimate traffic to flow and malicious traffic to be removed, as defined in Equation (4)

$$T_{\text{secure}} = \{t_i \mid y_i = \text{Normal}\} \quad \text{or} \quad T_{\text{secure}} = \{t_i \mid y_i = \text{Anomaly}\} \tag{4}$$

In the case of trusted traffic, the security of data transmission is guaranteed through the use of a hybrid cryptographic scheme, which mathematically exhibits the encryption of the message and the secure key exchange, as shown in Equation (5)

$$C = \text{Enc}_{\text{sym}}(M, K_s), \quad K_s = \text{Enc}_{\text{asym}}(K_s, K_{\text{pub}}) \tag{5}$$

3.2.3 Feature extraction

In this paper, the emphasis of feature extraction is placed on the conversion of raw network traffic into representations that are capable of differentiating the two classes, and that should spatially and temporally be very close to both normal and anomalous behavior. A feature vector $X_i = [x_1, x_2, \ldots, x_m]$ is created from each network flow t_i that includes statistical, temporal, and protocol-based properties like packet count, flow duration, byte rate, and flag information. A neural transformation is used for deep feature extraction that assists the learning of representations by mapping the original feature space into a higher-level latent space, as explained in Equation (6)

$$H_i = \phi(WX_i + b) \tag{6}$$

where *W* and *b* refer to the parameters that can be adjusted during training and $\phi(\cdot)$ is a nonlinear activation function. A bidirectional recurrent representation is used, as shown in Equation (7), to capture the temporal dependencies between flows for sequential traffic modelling

$$Z_i = \overrightarrow{h_i} \oplus \overleftarrow{h_i} \tag{7}$$

where $\overrightarrow{h_i}$ and $\overleftarrow{h_i}$ symbolize forward and backwards hidden states, respectively, and $\oplus$ signifies concatenation. The deep feature vector Z_i which has been extracted is then passed on to the classification layer, which allows for accurate distinction between normal and anomalous network traffic and at the same time trusts subsequent cryptographic enforcement. The pre-processed dataset is divided into training, validation, and testing subsets to ensure unbiased learning and reliable performance evaluation. Model training uses 70% of the data while 15% serves as validation to adjust hyperparameters and stop overfitting, and the last 15% functions as testing. Stratified splitting keeps the initial class percentages of normal and anomalous traffic intact for all subsets because the dataset has built-in class imbalance. This splitting strategy enables fair evaluation of models while maintaining their ability to generalize to new network traffic.

3.3 Traffic Trust Verification

The Traffic Trust Verification layer is in charge of deciding if network traffic is good or bad before putting the cryptographic protection on. A trust score is calculated for each traffic instance based on the deep features extracted from network flows to quantify its reliability. The softmax function, as described

in Equation (8), is applied to the classifier output to get the likelihood of a traffic flow being normal

$$P(y_i = k) = \frac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}} \tag{8}$$

Here, the vector of extracted features is denoted by Z_i and K indicates the number of categories. A trust score is determined for every flow according to this probability, as shown in Equation (9)

$$\mathcal{T}_i = P(y_i = \text{Normal} \mid Z_i) \tag{9}$$

A predefined threshold τ is then used to verify the trustworthiness of traffic. Traffic flows satisfying the trust condition are allowed for secure transmission, while others are rejected, as formulated in Equation (10)

$$\text{Decision}_i = \begin{cases} \text{Trusted}, & \text{if } \mathcal{T}_i \geq \tau \\ \text{Untrusted}, & \text{if } \mathcal{T}_i < \tau \end{cases} \tag{10}$$

The trust verification process guarantees that only the authenticated traffic of the network moves to the encryption phase, thus minimizing the computational power needed and also blocking the communication of the malicious activities that were previously hidden in the encrypted communications. The DenseNet–BiGRU hybrid system shown in the Figure 2 uses network traffic data to extract both spatial and temporal features which it then uses to identify normal and malicious network activity.

Figure 2 displays a hybrid DenseNet–BiGRU model which performs network traffic classification. DenseNet uses convolution and pooling layers to extract spatial features from raw traffic data, which it transforms into a compact feature representation. The BiGRU layer processes these features to capture temporal behavior which it tracks in both forward and backward directions. The final output layer classifies the traffic as normal or malicious, enabling accurate and context-aware intrusion detection.

3.3.1 DenseNet

DenseNet for Spatial Feature Learning has been utilized in this study to seize the rich spatial relationships among network traffic features and to facilitate feature reuse across layers. In contrast to conventional deep networks, DenseNet makes direct connections between all layers residing within a dense block, thus allowing each layer to collect feature maps from all the layers that

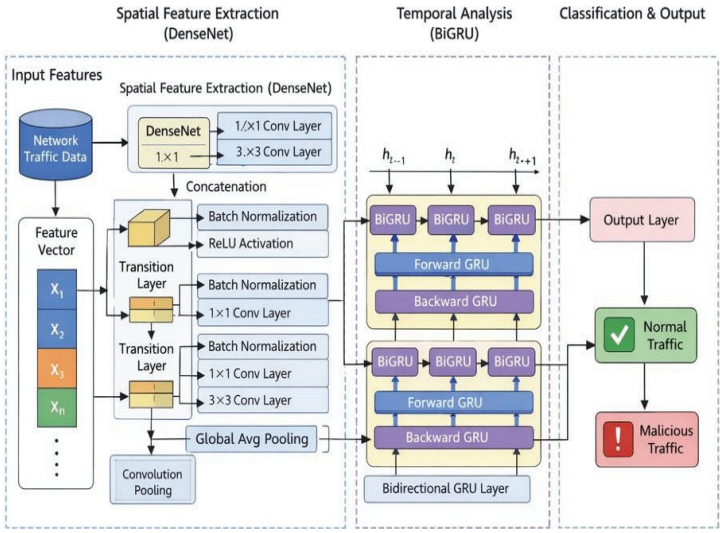

Figure 2 Architecture of the proposed DenseNet–BiGRU based secure network traffic classification framework.

are prior to it. Such dense connectivity enhances gradient flow, mitigates the vanishing gradient problem, and reinforces spatial feature propagation at the same time. The function is composed of the lth layer in DenseNet is defined in Equation (11) as

$$H_l = \mathcal{F}_l([H_0, H_1, \ldots, H_{l-1}]) \tag{11}$$

where $[\cdot]$ indicates the combination of feature maps, H_0 is the initial feature map, and $F_l(\cdot)$ shows a combination of batch normalization, nonlinear activation, and convolution operations. By this method, DenseNet has been able to differentiate normal and anomalous behaviors in network traffic patterns, which are vital for trust verification and encryption, by learning the discriminative spatial representations of such patterns effectively.

3.3.2 BiGRU

BiGRU for Temporal Behavior Analysis is used to capture the sequential dependencies and the temporal dynamics present in the network traffic flows. By processing the traffic sequences in both forward and backward directions, BiGRU can get past and future contextual information, which in turn makes it possible to have a full understanding of the attack patterns over time. The hidden state update of a GRU unit is influenced by the update and reset gates

as defined in Equations (12) and (13)

$$z_t = \sigma(W_z x_t + U_z h_{t-1}) \tag{12}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1}) \tag{13}$$

where x_t refers to the input at the time t, h_{t-1} indicates the earlier hidden state, and $\sigma(\cdot)$ represents the sigmoid activation function. The potential hidden state and final hidden state are derived as described in Equations (14) and (15), respectively

$$\tilde{h}_t = \tanh(W_h x_t + U_h(r_t \odot h_{t-1})) \tag{14}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \tag{15}$$

In the bidirectional network, forward and backwards hidden states are combined to form the temporal representation as expressed in Equation (16)

$$H_t = \overrightarrow{h_t} \oplus \overleftarrow{h_t} \tag{16}$$

The use of temporal modelling based on BiGRU makes the detection of evolving and stealthy attack behaviors very effective, as it learns the time-dependent traffic patterns.

Algorithm 1: DenseNet–BiGRU Based Network Traffic Classification for Secure Communication

Input: Network traffic dataset D = {X, Y}
Output: Predicted class labels $\hat{Y}$ (Normal/Malicious)

1: Preprocess dataset D
2: Handle missing values
3: Normalize features
4: Convert traffic flows into sequences
5: Split the dataset into training, validation, and testing sets
6: Initialize DenseNet–BiGRU model parametersv 7: for each training epoch do
8: for each batch B in the training set do
9: # DenseNet Spatial Feature Extraction
10: F_dense ← DenseBlock(B)
11: F_trans ← TransitionLayer(F_dense)
12: F_pool ← GlobalAveragePooling(F_trans)
13: # BiGRU Temporal Modelling
14: H_forward ← GRU_forward(F_pool)
15: H_backward ← GRU_backward(F_pool)
16: H_concat ← Concatenate(H_forward, H_backward)

17: # Classification Layer
18: $\hat{Y}$_batch ← Softmax(Dense(H_concat))
19: Compute loss L using cross-entropy
20: Update weights using backpropagation
21: end for
22: end for
23: Evaluate trained model on test set
24: Compute accuracy, precision, recall, F1-score, MCC, ROC-AUC
25: Return final predictions $\hat{Y}$

3.4 Secure Session Establishment

The process of Secure Session Establishment by Asymmetric Cryptography (ECC-Based Key Exchange) is the one that makes it possible to safely produce and share the symmetric session key needed to encrypt data. Although Elliptic Curve Cryptography (ECC) gives very strong security even with small keys, it is still a good choice for dynamic network environments. Let's consider the elliptic curve over a finite field as defined in Equation (17)

$$D : y^2 = x^3 + ax + b(modp) \tag{17}$$

where a, b, and p represent the parameters of the curve. The sender and the receiver choose their private keys d_A and d_B, respectively, and generate their public keys through scalar multiplication, as shown in Equation (18)

$$Q_A = d_A G, \quad Q_B = d_B G \tag{18}$$

where G denotes the point on the curve considered as the base point. Subsequently, a shared secret is generated separately by each party utilizing the public key of the other, as shown in Equation (19)

$$S = d_A Q_B = d_B Q_A \tag{19}$$

The symmetric session key K_s for AES encryption is then derived from the shared secret S. This ECC-based key exchange guarantees secure session establishment, it also protects against eavesdropping, and this enables the proposed secure network communication framework to employ efficient hybrid encryption.

Algorithm 2: Elliptic Curve Cryptography-Based Secure Key Exchange for Session Establishment

Input: Elliptic curve parameters (E, G, n), where
E ← elliptic curve
G ← base point

n ← order of G
Output: Shared secret key K
1. Sender selects a private key:
a ← random integer in [1, n-1]
2. Sender computes public key:
A ← a · G
3. Receiver selects a private key:
b ← random integer in [1, n-1]
4. Receiver computes public key:
B ← b · G
5. Exchange public keys A and B over the network
6. Sender computes shared secret:
K_sender ← a · B
7. Receiver computes shared secret:
K_receiver ← b · A
8. Since:
a · B = a · (b · G) = b · (a · G) = b · A
9. Therefore:
K ← K_sender = K_receiver
10. Derive session encryption key:
SessionKey ← Hash(K)
Return SessionKey

3.5 Hybrid Encryption

The implementation of the Hybrid Encryption Module using AES guarantees quick but secure protection of authenticated network data payloads. Upon the successful validation of the traffic by the trust verification layer, the data payload M gets encrypted using the AES and a symmetric session key K_s. AES works with data blocks of a predetermined size and performs several transformations per block to ensure strong confidentiality. The total AES encryption procedure is given in Equation (20) as

$$C = \mathrm{AES\_Enc}(M, K_s) \tag{20}$$

where C represents the output of the ciphertext. An AES round is made up of substitution, permutation, and mixing operations, starting with the addition of the initial key. The transformation of the core round is given by Equation (21)

$$S_r = \mathrm{MixColumns}(\mathrm{ShiftRows}(\mathrm{SubBytes}(S_{r-1}))) \oplus K_r \tag{21}$$

where S_r is the state matrix at round r and K_r is the round key that comes from the key schedule. AES stands out for its minimal computational cost,

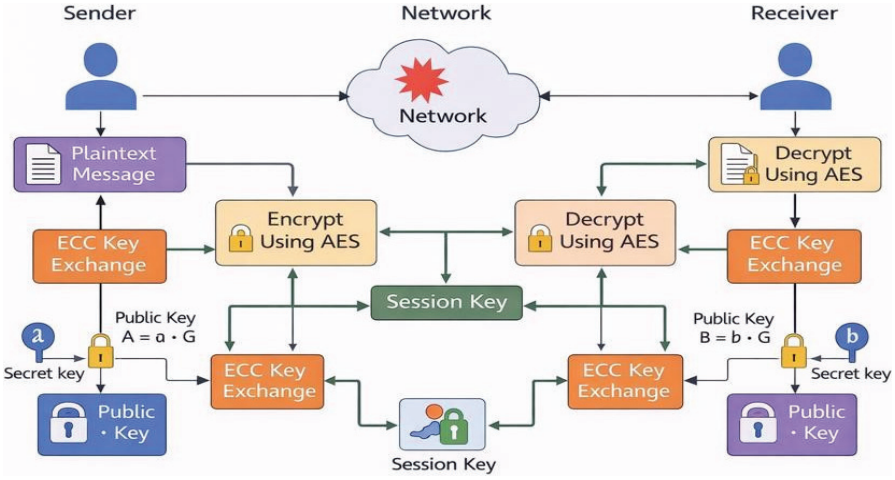

Figure 3 Hybrid ECC–AES encryption architecture for secure network communication.

strong resistance to cryptanalysis, and the ability to operate in real-time, thus being the best choice for encrypting trusted data payloads in the proposed hybrid security framework with the least possible impact on the network. Figure 3 illustrates a secure communication process which creates a shared session key through ECC and uses that key to encrypt and decrypt messages between sender and receiver through AES.

Figure 3 depicts a hybrid encryption workflow that uses both ECC and AES encryption methods to create secure communication between a sender and a receiver. First, both parties generate their ECC private keys and corresponding public keys, which they exchange through the network to create a shared session key through elliptic curve key agreement. The session key remains secret because both parties create it through independent methods that protect its content. The sender uses the session key that he obtained to secure his plaintext message through AES encryption, which acts as a rapid symmetric encryption method that works well with extensive data streams. The network transmits the encrypted information in a secure, protected state. The receiver uses the same ECC-derived session key to decrypt the ciphertext through AES decryption, which retrieves the original message. This integration uses ECC to secure key exchange while implementing AES for effective data encryption which results in both strong security and high performance for network communication.

Algorithm 3: Advanced Encryption Standard-Based Secure Data Encryption

Input: Plaintext message M, Secret key K
Output: Ciphertext C
1. Expand the secret key:
RoundKeys ← KeyExpansion(K)

2. Convert plaintext into a state matrix:
State ← Block(M)
3. Initial round:
State ← AddRoundKey(State, RoundKeys[0])
4. for round = 1 to Nr-1 do
5. State ← SubBytes(State)
6. State ← ShiftRows(State)
7. State ← MixColumns(State)
8. State ← AddRoundKey(State, RoundKeys[round])
9. end for
10. Final round:
State ← SubBytes(State)
State ← ShiftRows(State)
State ← AddRoundKey(State, RoundKeys[Nr])
11. Convert state matrix to ciphertext:
C ← Combine(State)
Return C

3.6 Secure Network Transmission

The Secure Network Transmission (Encrypted Communication Channel) guarantees that data, after being authenticated and encrypted, is delivered over the network securely, thus maintaining confidentiality and integrity. The encrypted AFS-based payload is making the ciphertext C go to the communication channel, where the integrity information is sent together to reveal any unauthorized modifications. The integrity of the message is tested by means of a cryptographic hash function, which is computed as shown in Equation (22)

$$H = \mathrm{Hash}(C) \tag{22}$$

where H stands for the message digest. At the receiving end, the integrity of the accepted ciphertext C' is checked by recalculating the hash and contrasting it with the sent digest, as shown in Equation (23)

$$\mathrm{Verify} = \begin{cases} \mathrm{Valid}, & \text{if } \mathrm{Hash}(C') = H \\ \mathrm{Invalid}, & \mathrm{otherwise} \end{cases} \tag{23}$$

When the user is authenticated, the original message is deciphered using AES, defined in Equation (24)

$$M = \text{AES\_Dec}(C', K_s) \tag{24}$$

The method of transmitting encrypted messages helps make communication end-to-end secure by safeguarding the information from spying, alteration, and replay attacks while it is transmitted through the network.

3.7 Decryption and Integrity Verification

The receiver carries out Decryption and Integrity Verification using AES in order to obtain secure and reliable data recovery. After the encrypted payload C' has been received through the secure channel, its integrity is initially authenticated to assert that the ciphertext has not been changed while sending it. The integrity verification is done by checking the present hash against the recomputed hash, as it is mentioned in Equation (25)

$$\text{Verify} = \begin{cases} \text{Valid}, & \text{if Hash}(C') = H \\ \text{Invalid}, & \text{otherwise} \end{cases} \tag{25}$$

The AES session key K_s that is shared is only used to decrypt the ciphertext if the verification is successful. The AES decryption process is mathematically represented in (26)

$$M = \text{AES\_Dec}(C', K_s) \tag{26}$$

Here, M denotes the plaintext message that has been recovered. The AES decryption process uses reverse transformations which match the encryption round transformations that were performed during the encryption process to restore the data to its original state. The proposed framework uses joint decryption and integrity verification to protect network communication through confidentiality and authenticity and reliability.

4 Results

The proposed framework was developed by using machine learning, deep learning, and cryptographic development tools to create accurate models and evaluate protected communication systems. The team selected Python as their primary programming language because it offers flexible capabilities together

with complete support for data science work. The team used Pandas and NumPy for data preprocessing and feature engineering and dataset management, whereas they used Matplotlib and Seaborn to visualize traffic patterns and performance graphs. The development of DenseNet–BiGRU deep learning architecture occurred through TensorFlow/Keras, which provided the framework for efficient model training, GPU acceleration, and large-scale traffic classification. Scikit-learn supported performance evaluation through its ability to generate accuracy, precision, recall, F1-score, ROC-AUC, and confusion matrix results.

The cryptographic layer required developers to build secure communication modules using Python cryptography libraries which included PyCryptodome and OpenSSL bindings to implement AES encryption and ECC key exchange security. The research team conducted virtual environment testing through network simulation tests to examine encryption operations and network performance which used actual traffic patterns. The experiments and virtual environment testing were conducted on a system equipped with an Intel Core i7 processor, 16 GB RAM, and NVIDIA GPU support, running on a Windows/Linux operating system. The implementation was carried out using Python with TensorFlow/Keras and cryptographic libraries such as PyCryptodome and OpenSSL, ensuring efficient model training and secure communication evaluation. The integration of deep learning frameworks with cryptographic toolkits created a system which enabled the combination of intrusion detection and secure transmission functions. The research tools enabled researchers to replicate their experiments while achieving optimal processing power and successfully testing the hybrid security framework.

4.1 Model Training

The DenseNet–BiGRU model is trained using the preprocessed CIC-IDS2017 dataset to learn both spatial and temporal traffic patterns. The DenseNet component consists of multiple dense blocks, which include batch normalization and ReLU activation together with transition layers that help in efficient feature reuse and protection against vanishing gradients. The BiGRU layer contains bidirectional gated recurrent units, which enable it to analyze traffic sequences by processing temporal information in both forward and backward directions. The model uses the Adam optimizer for training while it employs categorical cross-entropy loss together with an adaptive learning rate schedule. The system uses dropout regularization and early stopping to

enhance its ability to generalize. A stratified k-fold cross-validation strategy is used to preserve class imbalance during training, and hyperparameters such as learning rate, batch size, number of GRU units, and dense block depth are tuned through grid search to achieve optimal detection performance.

4.2 Cryptographic Evaluation

The evaluative process of the cryptographic module uses ECC as its method for session key establishment and AES as its method for encrypting and decrypting payloads. The researchers execute their tests in a controlled setting to assess how different payload weights affect their computational efficiency. The researchers conduct several tests to measure encryption and decryption delays, which they need to establish statistical validity. The researchers measure throughput by comparing the size of encrypted data to the time taken for execution, which enables them to assess performance across multiple traffic conditions. The evaluation setup uses real-time communication conditions to test whether the hybrid ECC–AES framework can maintain secure transmission without adding any significant delays.

4.3 Performance Metrics

The proposed cyber-attack detection framework undergoes performance evaluation through standard classification metrics which help assess reliability when faced with unequal class distribution. Detection accuracy assessment relies on precision and recall measurements which are defined mathematically in Equation (27)

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN} \tag{27}$$

The F1-score, which combines precision with recall, is shown in Equation (28) as its mathematical representation

$$F1\text{-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{28}$$

MCC serves as an effective assessment tool that measures performance while handling class distribution differences. Moreover, ROC-AUC and PR-AUC are introduced to evaluate the discriminative power, while latency during inference measures the real-time usage. These metrics, used together, validate the robustness and potential for deployment of the suggested framework.

The proposed intrusion detection framework evaluation requires two main assessment metrics, which are accuracy and recall. The system measures accuracy through its ability to identify both normal traffic and attack traffic as accurate test results. The model's overall accuracy can be measured using the formula provided as given in Equation (29)

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{29}$$

where TP and TN denote true positives and true negatives, while FP and FN represent false positives and false negatives.

The system performance test assesses its capacity to identify real attack occurrences through its recall measurement. The criticality of this measurement in cybersecurity systems arises from its potential to overlook actual attacks. Recall is defined in Equation (30) as

$$\text{Recall} = \frac{TP}{TP + FN} \tag{30}$$

These were related to measuring the overall correctness of the proposed framework in detecting attacks and evaluating the attacking performance.

4.4 Performance Analysis and Visualization

The numerical interpretation of the experimental results obtained from the proposed framework is presented in this section in detail. The graphics show the distribution of correct and incorrect classifications, the capacity of the model to differentiate between classes at various thresholds, and the detection accuracy, along with false alarms. These visual evaluations give a detailed picture of the effectiveness of the framework in the area of normal and malicious network traffic, of course, but also in the area of consistency and stability under class-imbalanced conditions. Figure 4 illustrates the precision–recall curves, that the DenseNet–BiGRU model maintains the highest precision across increasing recall levels, achieving the best PR-AUC and demonstrating strong robustness under class imbalance. The model shows better anomaly detection performance because its detection curve exceeds all other models in the plot.

Figure 4 shows how precision and recall interact through different deep learning models which operate at multiple decision thresholds in class-imbalanced intrusion detection scenarios. The curve for DenseNet–BiGRU not only prevails but also ensures the highest PR-AUC score (0.9677), indicating the best ability to keep precision high even when recall is increased.

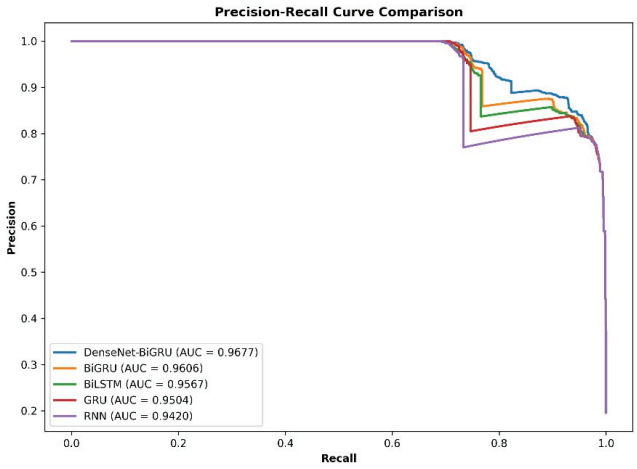

Figure 4 Precision–recall curve comparison for intrusion detection under class imbalance.

BiGRU and BiLSTM are quite similar in their performance, but with slightly lower PR-AUC values, showing a little bit of a drop in precision due to the increase in recall. On the other hand, GRU and RNN are characterized by more rapid drops in precision that signify lesser discrimination between attack and normal traffic. All in all, the graph supports the assertion that the proposed DenseNet–BiGRU model offers stronger and more trustworthy detection performance over a wide range of operating points.

Figure 5 shows how different deep learning models can tell apart normal network traffic from the malicious one by looking at the true positive rate and false positive rate trade-off. The DenseNet–BiGRU model has the highest discriminative power since its curve is always the closest to the top-left corner, and it also gets the maximum AUC value of 0.9911, which is a sign of great classification performance. BiGRU and BiLSTM are next in line with slightly lower AUC values representing reliable detection performance, but with somewhat higher false positives at certain thresholds. On the other hand, GRU and RNN have lower AUC scores, which shows less sensitivity in separating attack traffic from benign data. Therefore, the graph proves the strength and better separability of the proposed DenseNet–BiGRU framework.

Figure 6 presents the average cryptographic latency during the encryption and decryption processes of the suggested hybrid security framework. The two phases are characterized by almost the same latency values, where encryption represents a slightly longer average delay than decryption. The small error bars signify low variance, hence, reliable and steady

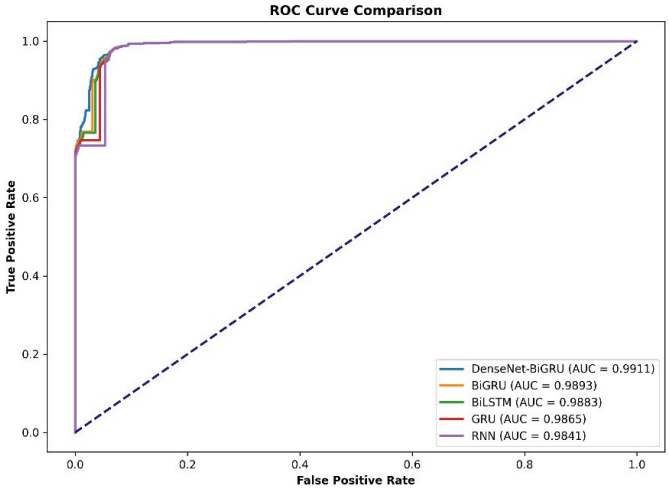

Figure 5 Receiver operating characteristic curve and AUC comparison of detection models.

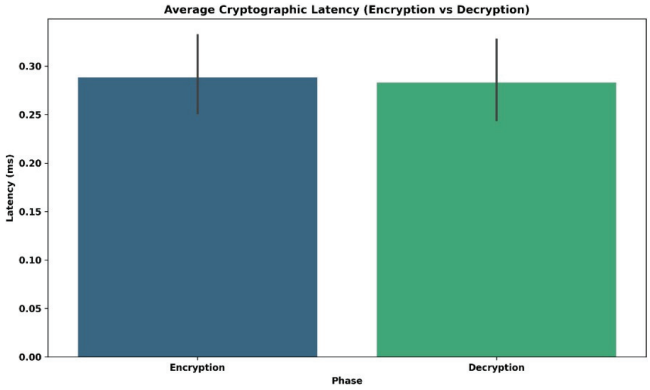

Figure 6 Average encryption and decryption latency of the hybrid AES–ECC scheme.

cryptographic processing over many transmissions. This almost symmetric latency behavior illustrates that the used encryption and decryption methods only add a slight amount of computation, thus allowing the secure communication system to be real-time applicable without a huge impact on the network performance. Table 2 shows that the ECC–AES hybrid framework, which they proposed, achieves high-performance encryption because it combines AES efficiency with ECC strong key security.

Table 2 shows the main characteristics of AES, RSA, ECC, and the new ECC–AES hybrid system. The AES system delivers rapid symmetric

Table 2 Cryptographic feature comparison of conventional and proposed hybrid encryption framework

Algorithm	Type	Key Size	No. of Rounds/ Operation	Block Size	Security Level
AES	Symmetric	128/192/256 bits	10/12/14 rounds	128 bits	Very High
RSA	Asymmetric	1024–4096 bits	1 modular exponentiation	Key-size dependent	High
ECC	Asymmetric	256–521 bits	1 scalar multiplication	Key-size dependent	Very High
Proposed ECC–AES Hybrid	Hybrid	ECC (256–521) + AES (128–256)	ECC + AES rounds	128 bits	Very High

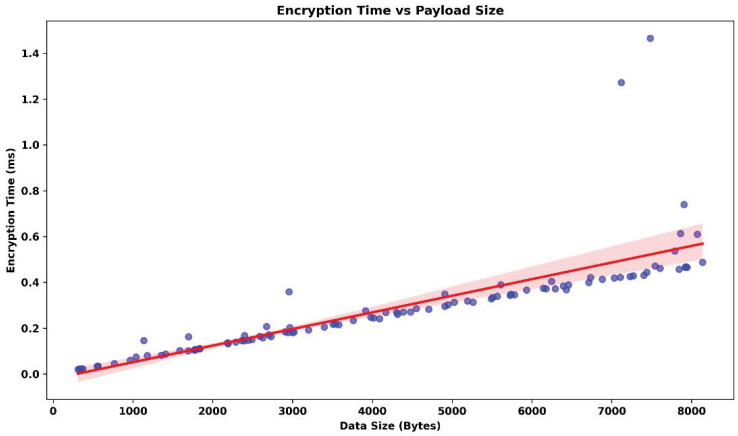

Figure 7 Encryption time variation with respect to payload size.

encryption through its 128-bit block system, which requires multiple processing rounds to achieve strong security protection. RSA and ECC serve as asymmetric methods that protect key exchange through their complex mathematical operations that do not depend on cypher rounds, and ECC requires smaller keys to achieve better security than RSA. The hybrid system uses ECC for secure key establishment and AES for efficient data encryption to achieve maximum security while maintaining fast performance during secure network communication.

Figure 7 shows the relationship between encryption time and payload size, and it also shows the scale of cryptographic processing overhead with the increase in data volume. A definite positive linear trend is visible, showing that the time for encryption increases with payload size at the same rate. The data points that are tightly packed around the regression line represent

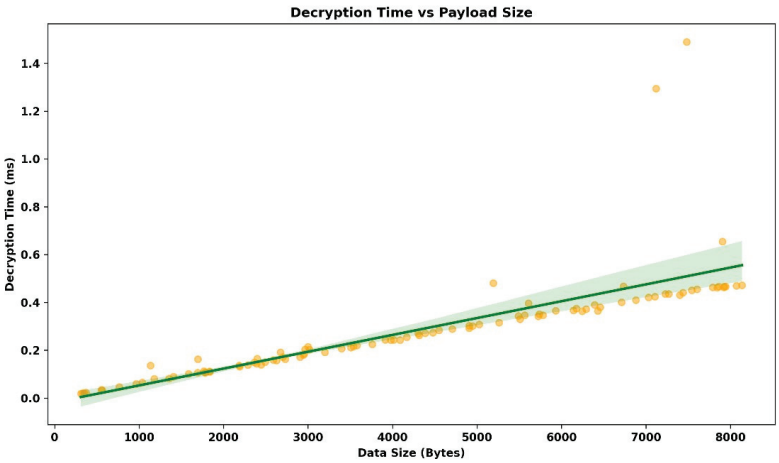

Figure 8 Decryption time variation with respect to payload size.

consistent and reliable performance, whereas the few outliers with higher latency in the case of large payload sizes signify processing variations that occur occasionally. In general, findings validate that the encryption method proposed can handle the increase in data size in an efficient manner, still giving low latency even for larger payloads and, thus, supporting the argument that it is suitable for secure communication in networks.

Figure 8 demonstrates how decryption time changes with payload size, revealing the computational behavior of the decryption process as data volumes increase. A significant linear correlation is found where the decryption time increases slowly along with the payload size. The fitted trend line is very close to the majority of data points, which means that decryption performance is stable and predictable, but the few data points at larger payload sizes with higher latencies show that there was an occasional delay in processing. In summary, the scaling of the decryption mechanism with data size and its ability to maintain low latency have been confirmed by the results, thus making it fit for secure and real-time network communication.

Figure 9 shows the distribution of relative encryption overhead, which reveals the occurrence of different overhead values during the encryption process. The large number of values in a very tight range means that the encryption overhead is nearly the same regardless of the payload size, and there are no extreme deviations or long tails. Such a distribution points to the stable computational behavior and predictable encryption costs, thus confirming that the cryptographic mechanism used results in an overhead that is

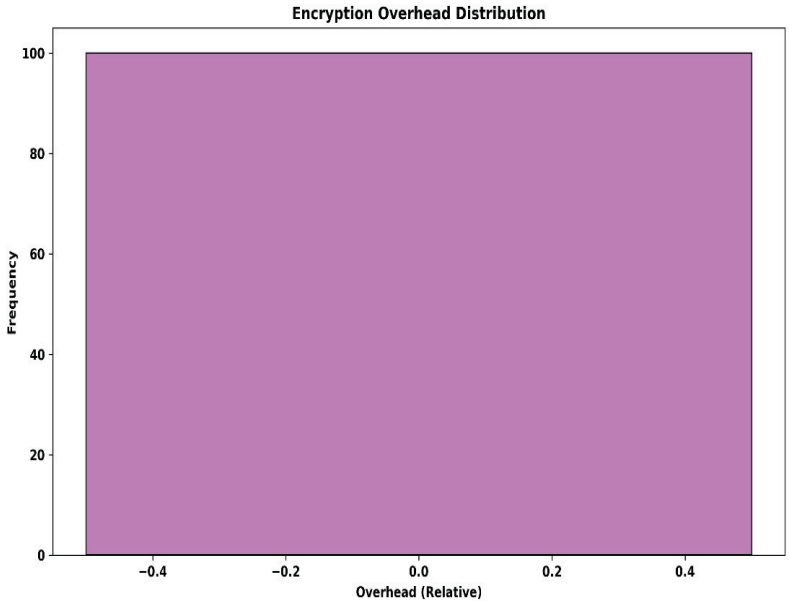

Figure 9 Distribution of encryption overhead in secure network communication.

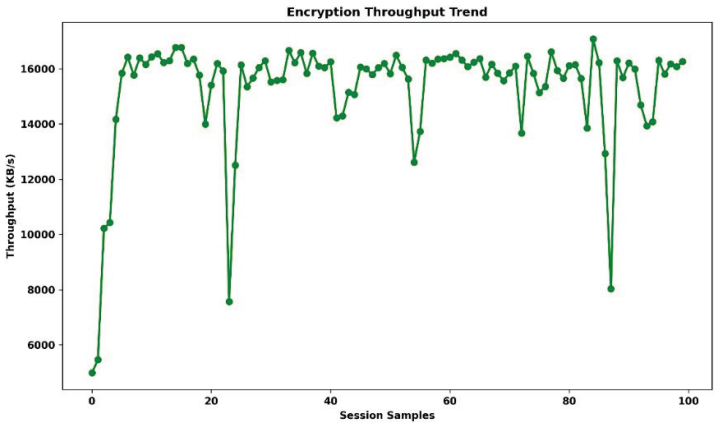

Figure 10 Encryption throughput performance over time.

minimal and well-controlled and, hence, it is appropriate for the applications that are sensitive to latency and require real-time secure networking.

Figure 10 depicts the change in encryption throughput across different session samples, thereby showing the data processing capacity of the system over the period. There was an initial warm-up period with low throughput,

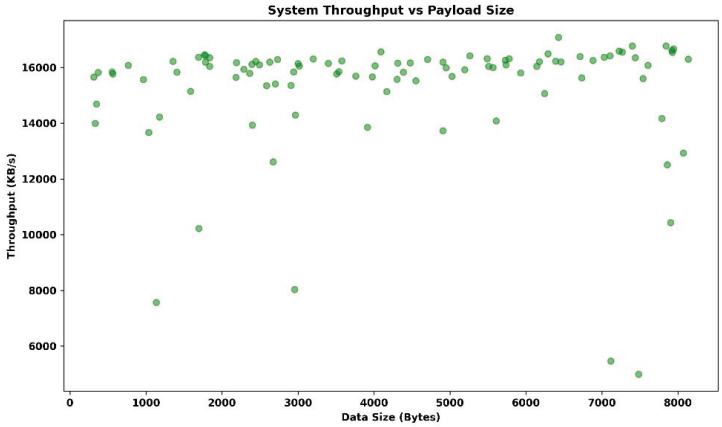

Figure 11 End-to-end system throughput versus payload size.

but very soon the encryption process stabilized at a high throughput level of mainly 15,000–16,500 KB/s. There are sharp drops from time to time that can be referred to as temporary system overheads, like context switching, buffer reallocation, or changes in payload characteristics. However, the overall trend is steady, indicating that the encryption module is consistently providing high throughput and is thus able to support real-time network environments with the continuous high-volume secure data transmitted being consumed.

Figure 11 illustrates how system throughput and payload size are related, revealing the capability of the system to handle encrypted traffic efficiently as the data size goes up. For most of the payload sizes, the throughput is quite high and is in the range 15,000–16,500 KB/s, which indicates that the system is capable of scaling up with the increasing data volume and is able to maintain good processing performance. There are a few isolated points of low throughput at certain payload sizes, which might be due to temporary computational overheads, network buffering effects, or contention for system resources. In general, the throughput distribution shows that the payload size has a negligible negative effect on throughput, thereby confirming the strength and scalability of the proposed secure detection and transmission framework under different traffic loads.

The DenseNet–BiGRU model shows its classification results through Figure 12 which displays the normal traffic and attack traffic classification results. The system achieved high success rate for normal instance detection because it identified 7841 true negative cases and only misclassified 203 cases as attacks. The system detected 1609 true attack instances while 347 attacks

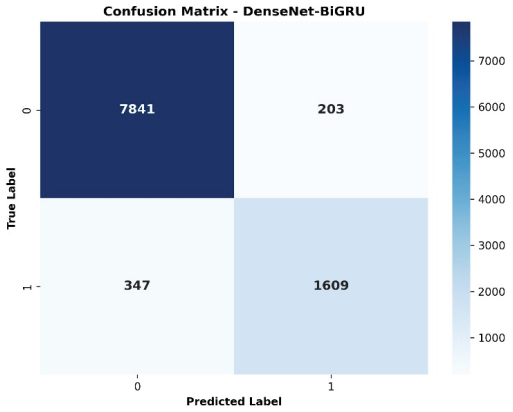

Figure 12 Confusion matrix of the proposed DenseNet–BiGRU intrusion detection model.

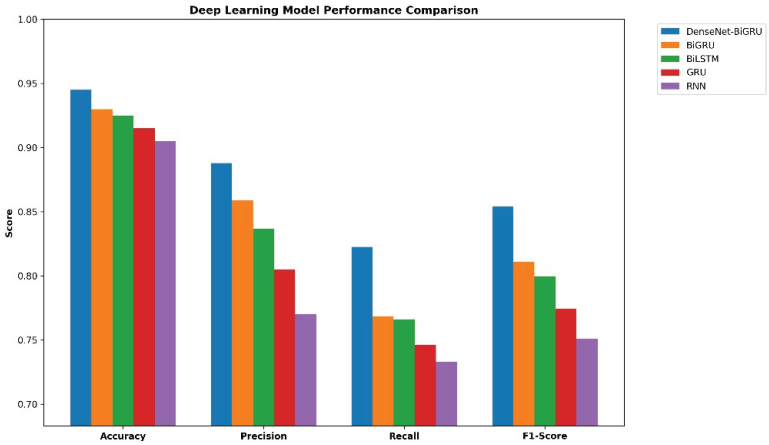

Figure 13 Comparative performance analysis of deep learning models.

were wrongly identified as normal traffic. The matrix results confirm the system classification accuracy and attack detection efficiency together with its ability to balance between false positive rates and detection failures which proves the system framework operates reliably.

Figure 13 shows a comparative analysis of multiple deep learning models through their key performance metrics which include accuracy, precision, recall, and F1-score measurements. The DenseNet–BiGRU model achieves its highest performance through all evaluation metrics because it can effectively track both spatial and temporal traffic patterns. The BiGRU and BiLSTM models show similar results but the latter two models demonstrate

Table 3 Latency and security comparison of proposed ECC scheme with traditional cryptographic methods

	Key Exchange Latency (ms)	Encryption Latency (ms)	Decryption Latency (ms)	Security Level (Bits)
Proposed (ECC-256)	1.2	0.29893	0.283558	128
Traditional (RSA-2048)	15.4	1.345185	3.402696	112
Traditional (DSA-2048)	18.2	1.554436	2.410243	112

slightly lower efficiency during recall and F1-score tests. GRU and RNN models show a dramatic decrease in their precision and recall rates because they cannot learn the intricate relationships present in network traffic data. The hybrid DenseNet–BiGRU architecture demonstrates its ability to produce accurate and unbiased classification results which exceed the performance of traditional recurrent and bidirectional models according to the graph. Table 3 demonstrates that the proposed ECC scheme achieves better security results through its lower latency performance compared to standard RSA and DSA methods which leads to faster and more effective secure communication.

Table 3 compares latency and security strength between the proposed ECC-based scheme and traditional RSA/DSA methods. The proposed ECC-256 framework achieves significantly lower key exchange, encryption, and decryption latency, demonstrating faster secure session establishment and data protection. The computational delays of RSA-2048 and DSA-2048 increase because their large key operations require more processing time which results in higher communication expenses. The proposed approach delivers stronger security with improved efficiency because ECC maintains a higher effective security level (128-bit equivalent) while operating with smaller keys.

4.5 Discussion

The findings validate that the suggested hybrid scheme does a great job in mingling the intrusion detection based on deep learning with safe cryptographic communication [32]. The DenseNet–BiGRU model exceeds the traditional RNN, GRU, BiGRU, and BiLSTM models in the aspect of accurately recognizing both spatial and temporal traffic patterns, thus resulting in higher accuracy, precision, recall, and F1-score on the CIC-IDS2017 dataset. The combination of confusion matrix and ROC and precision-recall analysis methods shows strong ability to distinguish between different classes without making incorrect predictions during class imbalance testing [33]. The

cryptographic assessment shows that AES encryption and ECC-based key exchange create minimal latency impact according to the findings of which show that encryption and decryption times increase with payload size but throughput remains constant [34]. Results show that the proposed system achieves an effective equilibrium between trustworthy intrusion detection and safe real-time network communication which makes it suitable for practical use. Further validation under real-world network environments and live cyber-attack scenarios is necessary to assess the robustness of the proposed framework. Future work will focus on deploying the model in real-time network settings to evaluate its performance against evolving and zero-day attacks, thereby enhancing its practical applicability.

5 Conclusion

This research proposes a security framework that protects network communication using hybrid deep learning for traffic trust assessment and advanced cryptographic mechanisms. The approach integrates a DenseNet–BiGRU intrusion detection system with ECC-based key exchange and AES-based encryption, ensuring that only verified traffic is encrypted, thereby improving both security and processing efficiency. The system was evaluated using standard intrusion detection datasets and achieved high detection accuracy with balanced precision, recall, F1-score, and MCC values, along with strong ROC-AUC and PR-AUC performance under class-imbalanced conditions. Cryptographic evaluation confirmed that the system operates efficiently in real time with minimal latency and stable throughput. Future improvements include extending the framework to multi-class attack classification and adapting it for federated and edge-based environments. Additionally, incorporating adaptive cryptographic schemes and online learning mechanisms can further enhance resilience against dynamic and zero-day attacks in IoT and next-generation networks.

References

[1] C. Pereira et al., Security and privacy in physical-digital environments: Trends and opportunities, Future Internet, vol. 17, no. 2, 83, 2025. DOI: https://doi.org/10.3390/fi17020083.

[2] V. Maurya et al., Blockchain-driven security for IoT networks: State of the art, challenges and future directions, Peer-to-Peer Netw. Appl., vol. 18, 53, 2024. https://doi.org/10.1007/s12083024-01812-w.

[3] A. A. Almuqren, Cybersecurity threats, countermeasures and mitigation techniques on the IoT: Future research directions, J. Cyber Security Risk Audit, vol. 1, no. 1, 1–11, 2025. DOI: https://doi.org/10.63180/jcsra.thestap.2025.1.1.

[4] U. Tariq and T. A. Ahanger, Enhancing intelligent transport systems through decentralized security frameworks in vehicle-to-everything networks, World Electric Vehicle Journal, vol. 16, no. 1, 24, 2025. DOI: https://doi.org/10.3390/wevj16010024.

[5] K. Mansoor, M. Afzal, W. Iqbal and Y. Abbas, Securing the future: Exploring post-quantum cryptography for authentication and user privacy in IoT devices, Cluster Comput, vol. 28, 93, 2024. https://doi.org/10.1007/s10586-024-04799-4.

[6] A. G. Filho, E. K. Viegas, A. O. Santin and J. Geremias, A dynamic network intrusion detection model for infrastructure-as-code deployed environments, Journal of Network System Management, vol. 33, no. 4, 75, 2025. https://doi.org/10.1007/s10922-025-09940-1.

[7] V. Leiva and C. Castro, Artificial intelligence and blockchain in clinical trials: Enhancing data governance efficiency, integrity, and transparency, Bioanalysis, vol. 17, no. 3, pp. 161–176, 2025. https://doi.org/10.1080/17576180.2025.2452774.

[8] S. Pasupathi, R. Kumar and L. K. Pavithra, Proactive DDoS detection: Integrating packet marking, traffic analysis, and machine learning for enhanced network security, Cluster Computing, vol. 28, no. 3, 210, 2025. https://doi.org/10.1007/s10586-024-04849-x.

[9] A. Hidri et al., Opinion mining and analysis using hybrid deep neural networks, Technologies, vol. 13, no. 5, 175, 2025. https://doi.org/10.3390/technologies13050175.

[10] H. Alqahtani and G. Kumar, Deep learning-based intrusion detection system for in-vehicle networks with knowledge graph and statistical methods, Int. J. Mach. Learn. & Cyber., vol. 16, no. 5, pp. 3539–3555, 2025. https://doi.org/10.1007/s13042-024-02465-0.

[11] J. Alotaibi, A hybrid software-defined networking approach for enhancing IoT cybersecurity with deep learning and blockchain in smart cities, Peer-to-Peer Netw. Appl., vol. 18, no. 3, 123, 2025. https://doi.org/10.1007/s12083-025-01935-8.

[12] I. Bibers et al., Ensemble-IDS: An ensemble learning framework for enhancing AI-based network intrusion detection tasks, Applied Sciences, Enhancing vol. 15, no. 19, 10579, 2025. https://doi.org/10.3390/app151910579.

[13] A. Bahuguna, M. C. Govil and G. Bhaumik, A hybrid approach for static hand gesture recognition with integrated BiGRU-BiLSTM and sequential self-attention mechanism, SIViP, vol. 19, no. 6, 486, 2025. https://doi.org/10.1007/s11760-025-04071-1.

[14] K. Soares and A. A. Shinde, Authentication-based VANET for data transfer: Unveiling the ability of deep learning models for attack classification, Multimedia Tools Application, vol. 84, no. 27, pp. 33041–33070, 2025. https://doi.org/10.1007/s11042-024-20489-0.

[15] M. Ozaif, M. Alam, S. Mustajab, M. Mustaqeem and N. Khan, A secure and efficient identity-based RFID mutual authentication scheme for IoT using elliptic curve cryptography, International Journal of Computations and Application, vol. 47, no. 5, pp. 424–437, 2025.

[16] J. A. Rathod and M. Kotari, Secure and efficient message transmission in MANET using hybrid cryptography and multipath routing technique, Multimed Tools Appl., vol. 84, no. 13, pp. 12633–12656, 2025. https://doi.org/10.1007/s11042-024-19542-9.

[17] A. Sharmila, V. Rishiwal, P. Kumar, M. Yadav and P. Yadav, Secure hybrid data transmission protocol for WSN with key management and message authentication, SN Computer Science, vol. 6, no. 5, 401, 2025. https://doi.org/10.1007/s42979-025-03946-x.

[18] H. Nandanwar and R. Katarya, A hybrid blockchain-based framework for securing intrusion detection systems in Internet of Things, Cluster Comput, vol. 28, no. 7, 471, 2025. https://doi.org/10.1007/s10586-025-05135-0.

[19] V. Khagga, N. S. Priya and A. M. Prasad, Enhanced QoS-aware secure routing protocol for WAHNs using advanced fast double-decker new binary Archimedes-Kepler pure convolutional transformer network and cryptographic techniques, Peer-to-Peer Netw. Appl., vol. 18, no. 4, 216, 2025. https://doi.org/10.1007/s12083-025-02035-3.

[20] C. Ye et al., Social image security with encryption and watermarking in hybrid domains, Entropy, vol. 27, no. 3, 276, 2025. https://doi.org/10.3390/e27030276.

[21] S. S. Priya, R. Vijayabhasker and A. Rajaram, Advanced security and efficiency framework for mobile ad hoc networks using adaptive clustering and optimization techniques, J. Electr. Eng. Technol., vol. 20, no. 3, pp. 1815–1826, 2025. https://doi.org/10.1007/s42835-024-02119-9.

[22] N. S. G. Ganesh, V. Balasubramanian, D. V. V. Prasad and S. S. Velan, Deep learning-based user authentication with hybrid encryption for secured blockchain-aided data storage and optimal task offloading in

mobile edge computing, Wireless Netw, vol. 31, no. 3, pp. 2389–2417, 2025. https://doi.org/10.1007/s11276-024-03886-z.

[23] A. Kanneboina and G. Sundaram, Improving security performance of Internet of Medical Things using hybrid metaheuristic model, Multimed Tools Appl, vol. 84, no. 9, pp. 6403–6428, 2025. https://doi.org/10.1007/s11042-024-19188-7.

[24] Z. S. Mahdi, R. M. Zaki and L. Alzubaidi, A secure and adaptive framework for enhancing intrusion detection in IoT networks using incremental learning and blockchain, Security Privacy, vol. 8, e70071, 2025. https://doi.org/10.1002/spy2.70071.

[25] B. Fu, T. Fang, L. Zhang, Y. Zhou and H. Xiao, Communication security of intelligent information service platform combining AES and ECC algorithms, Journal of Cyber Security Technology, vol. 9, no. 3, pp. 209–226, 2025. https://doi.org/10.1080/23742917.2024.2371053.

[26] P. Xiao, Malware cyber threat intelligence system for Internet of Things (IoT) using machine learning, JCSANDM, vol. 13, no. 1, pp. 53–89, 2024. https://doi.org/10.13052/jcsm2245-1439.1313.

[27] B. R. Gudivaka, R. L. Gudivaka, R. K. Gudivaka, D. K. R. Basani, S. H. Grandhi, S. Murugesan and M. M. Kamruzzaman, A predominant intrusion detection system in IIoT using ELCG-DSA and LWS-BiOLSTM with blockchain, Sustainable Computing: Information and System, vol. 46, 101127, 2025. https://doi.org/10.1016/j.suscom.2025.101127.

[28] C. Hazman, A. Guezzaz, S. Benkirane and M. Azrour, A smart model integrating LSTM and XGBoost for improving IoT-enabled smart cities security, Cluster Comput, vol. 28, no. 1, 70, 2024. https://doi.org/10.1007/s10586-024-04780-1.

[29] M. K. Chandol and M. K. Rao, Blockchain-based cryptographic approach for privacy-enabled data integrity model for IoT healthcare, Journal of Experimental & Technological Artificial Intelligence, vol. 37, no. 1, pp. 53–74, 2025. https://doi.org/10.1080/0952813X.2023.2183268.

[30] A. Kodituwakku and J. Gregor, InDepth: A distributed data collection system for modern computer networks, Electronics, vol. 14, no. 10, 1974, 2025. https://doi.org/10.3390/electronics14101974.

[31] H. N. Chethu, Network Intrusion Dataset (CIC-IDS-2017), Kaggle Dataset, Jan. 2026. Available: https://www.kaggle.com/datasets/chethuhn/network-intrusion-dataset (accessed Jan. 23, 2026).

[32] E. Braschi et al., Changing magma dynamics and plumbing system architecture at an explosive-effusive transition: The case of Nisyros

volcano (Greece), Eur. J. Mineral., vol. 37, no. 5, pp. 793–817, 2025. https://doi.org/10.5194/ejm-37-793-2025.

[33] Y. Lahraoui, S. Lazaar, Y. Amal and A. Nitaj, A novel ECC-based method for secure image encryption, Algorithms, vol. 18, no. 8, 514, 2025. https://doi.org/10.3390/a18080514.

[34] M. Bacevicius, A. Paulauskaite-Taraseviciene, G. Zokaityte, L. Kersys and A. Moleikaityte, Comparative analysis of perturbation techniques in LIME for intrusion detection enhancement, Machine Learning and Knowledge Extraction, vol. 7, no. 1, 21, 2025. https://doi.org/10.3390/make7010021.

Biography

Qing He, female, a postgraduate student and an associate professor, is currently employed as a faculty member in the Department of Information Engineering at Hope College, Southwest Jiaotong University, China, specializing in the domain of computer and communication signals. Her research interests encompass network communication and big data. Presently, her research focuses on network communication security and privacy protection.