Journal of Engineering and Technology for Industrial Applications

ITEGAM-JETIA

Manaus, v.12 n.58, p. 399-410. March/April, 2026.

DOI: https://doi.org/10.5935/jetia.v12i58.3086

ISSN ONLINE: 2447-0228

RESEARCH ARTICLE OPEN ACCESS

AES-ENHANCED BLOCKCHAIN INTRUSION DETECTION SYSTEM FOR SECURE NETWORKS

Hemanth Uppala*[1], R. Renuga Devi[2]

[1, 2]Department of Computer Science and Applications, SRM Institute of Science and Technology Ramapuram Campus, Chennai, India.

[1]https://orcid.org/0009-0002-0708-7025, [2]https://orcid.org/0000-0002-6197-236X

Email: * uppala.hemanth@gmail.com, renugadr@srmist.edu.in

ARTICLE INFO

Article History

Received: December 9, 2025

Reviewed: January 1, 2026

Accepted: January 14, 2026

Published: March 31, 2026

Keywords:

Blockchain-based IDS

Z-Score Normalization

Decentralized Identity (DID)

Smart Contracts,

Consensus-drivenKeyVerification,

AES Encryption

Intrusion Detection System

Secure Data Communication

MLPNN

SVM.

ABSTRACT

Blockchain is a distributed ledger technology that can securely, transparently, and tamper-proofly record transactions across a network. IDS on blockchain monitors nodes and activities in transactions to detect malicious or unusual patterns, thereby improving network integrity. However, blockchain also has disadvantages such as high computational overhead, vulnerability to specific attacks, and limited scalability. As a method to enhance IDS performance, the Z-score is used to normalize feature values, stabilizing the model and facilitating convergence. A combination of Support Vector Machines (SVM) and Decentralized Identification (DID) architecture can effectively and reliably distinguish between normal node behavior and malicious node behavior. Key verification uses consensus-driven transaction authentication, and a Multi-Layer Perceptron Neural Network (MLPNN) can detect complex attack patterns. All sensitive information is protected using the Advanced Encryption Standard (AES) to provide robust encryption for both stored and transmitted data. By combining these methods, blockchain applications can offer a comprehensive, secure, and scalable network intrusion detection system, addressing the limitations of detection accuracy, computational efficiency, and privacy in existing data and traditional blockchain implementations, to achieve an accuracy of 91%.

I.INTRODUCTION

Intrusion detection in distributed and decentralized systems has become a hot research topic, especially when combined with blockchain technology to provide transparency, security, and coordination to participating nodes [1], which allows for decentralized learning without compromising data privacy [2]. Similarly, automotive edge applications and blockchain combined with federated learning have been used to implement collaborative and scalable intrusion detection architectures [3].

In addition to the transportation and automotive fields, security intrusion detection algorithms have also been proposed in the drone cyber field, including blockchain and radial-based functional neural networks to provide platform security and manage trust levels in real time [4]. Additionally, blockchain-based collaborative detection systems have been designed to improve the resilience and responsiveness of intrusion detection systems through challenges [5]. In summary, the above strategies demonstrate that blockchain with the help of advanced learning methods has great potential to improve the scalability, security and efficiency of intrusion detection in various application areas.

- Objective: To design a decentralized intrusion detection system combining DIT, smart contracts, consensus-based key verification and AES encryption to detect intrusions in real-time.
- Problem Definition: Reliance on centralized trust, identity spoofing, key management attacks, and insecure data transmission (including network transmission) pose many threats to traditional IDS.

- Key Contribution: Proposes a multi-stage process for tamper-proof node authentication, decentralized key verification, and encrypted sharing of IDS data to ensure its security, reliability, and scalability.
- Reason for investigation: Traditional intrusion detection methods have their limitations, and the proposed solution using blockchain and cryptography technologies should address these issues to provide transparency, decentralization, and resilience in detecting intrusions in contemporary distributed networks.
- Work Structure: This paper is divided into sections including workflow design, DID generation, key verification, AES encryption, performance metrics, results and discussion, and conclusion for in-depth analysis and evaluation.

II. LITE8RATURE SURVEY

Vehicle communication systems are usually managed by a controller area network (CAN). By broadcasting data packets over a bus, CAN facilitates communication between electronic control units (ECUs) that coordinate, monitor, and control components in a vehicle [6]. However, automakers owners may want to keep confidential the important information needed to train the models.

A conditional generative adversarial network (CGAN) based collaborative intrusion detection algorithm is proposed and combined with blockchain-enabled distributed federated learning to solve the above problems [7]. However, many consensus algorithms, especially proof of work, require a large amount of computing power and energy.

Zero Trust Architecture (ZTA) has recently emerged as a new security model where a breach mindset dominates the threat model [8]. Blockchain to share and store the blacklist, and to create a duplicate verification function in the blockchain layer. However, relying on consensus mechanisms across all nodes makes B-IDS less efficient than centralized systems.

The Cooperative Intelligent Transportation System (C-ITS) is a promising technology designed to enhance traditional traffic management systems [9]. However, as the network grows, performance decreases as the amount of data each node must process and store increases.

Table 1: IDS in Cyber Security using Blockchain Technology.

Author/Year	Type of Techniques	Proposed Improvement	Key Contribution
Abou El Houda et al., (2021) [10]	Blockchain-enabled Federated Learning	Improves security and scalability by enabling decentralized and privacy-preserving learning.	Enhanced collaborative intrusion detection in vehicular edge computing with blockchain integration.
Moulahi, et al., (2022) [11]	Privacy-preserving Federated Learning	Strengthens privacy while maintaining detection accuracy in large-scale transport networks.	Designed cyber-threat detection framework for intelligent transport systems with blockchain-based security.
Pranto et al., (2022) [12]	Blockchain + Machine Learning	Enhances trust and adaptability while reducing data exposure in decentralized detection.	Developed a fraud detection system ensuring privacy and adaptive incentive mechanisms.
Truong & Le et al. (2024) [13]	MetaCIDS (Federated Learning + Blockchain)	Introduces fairness and real-time adaptability in blockchain-based intrusion detection.	Privacy-preserving collaborative intrusion detection for metaverse applications.
Cui et al. (2022) [14]	Federated Deep Learning (SDVN)	Ensures fair contribution of nodes and boosts performance in federated detection models.	Proposed a fairness-aware collaborative intrusion detection system for software-defined vehicular networks.
Said et al. (2021) [15]	CNN-BiLSTM + Hybrid Feature Selection	Improves accuracy and reduces computational overhead through optimized feature selection.	Hybrid deep learning framework for network intrusion detection in SDN.
Ferrag et al. (2021) [16]	Machine Learning (ML)	Provides benchmark datasets and highlights future research directions in agricultural cybersecurity.	ML-based intrusion detection solutions and datasets.

Source: Authors, (2026).

Table 1 demonstrates the IDS in Cyber Security using blockchain technology author's previous structure, including the type of Algorithm, proposed improvement, and key contributions.

Distributed Denial of Service (DDoS) attacks continue to grow rapidly, silently affecting Internet Service Providers (ISPs) and individuals alike [17]. Traditional IDS typically perform detection based on signatures and are therefore vulnerable to zero-day attacks. However, relying on consensus mechanisms across all nodes makes B-IDS less efficient than centralized systems. Cyber physical systems (CPS) are the integration of physical processes with processing and data transmission [18]. To solve this problem, this paper proposes a blockchain-based data transmission security intrusion detection system and classification model for CPS in the healthcare industry. However, the initial investment in setting up the infrastructure and hiring professional blockchain developers is significant, which can be a barrier for many companies. Large power systems have regional applications whose assets transmit sensor readings in real time. [19]. Introduces fairness and real-time adaptability in blockchain-based intrusion detection. However, this centralized sharing mechanism becomes a significant bottleneck due to operational limitations. Blockchain-based Collaborative IDS (PC-IDS), which uses blockchain technology to connect multiple IDSs [20]. Structure of BC-IDS: 1) Use IDS to create a list of malicious IP addresses; 2) Use blockchain to share and store the blacklist; 3) Create a duplicate verification function in the blockchain layer. However, integrating decentralized blockchain systems with existing centralized legacy infrastructure is technically complex and costly.

III. PROPOSED METHODOLOGY

The section product combines several methods to improve the security of the blockchain. Z-score normalization provides reliable results because identical scales are used throughout the data analysis. Support vector machine (SVM) is a useful tool for distinguishing between normal and malicious behaviors and improving intrusion detection performance. Decentralized identity (DIT) follows node identity, which is secure and verifiable, while transaction verification using consensus requires cooperation to verify transactions to eliminate unauthorized use. However, Multilayer Perceptron Neural Network (MLPNN) will be used to identify nonlinear patterns and improve the prediction capabilities of complex attacks. Finally, Advanced Encryption Standard (AES) is used for data encryption to ensure confidentiality and integrity of confidential information. All of these methods combine to create a powerful and robust blockchain intrusion detection system.

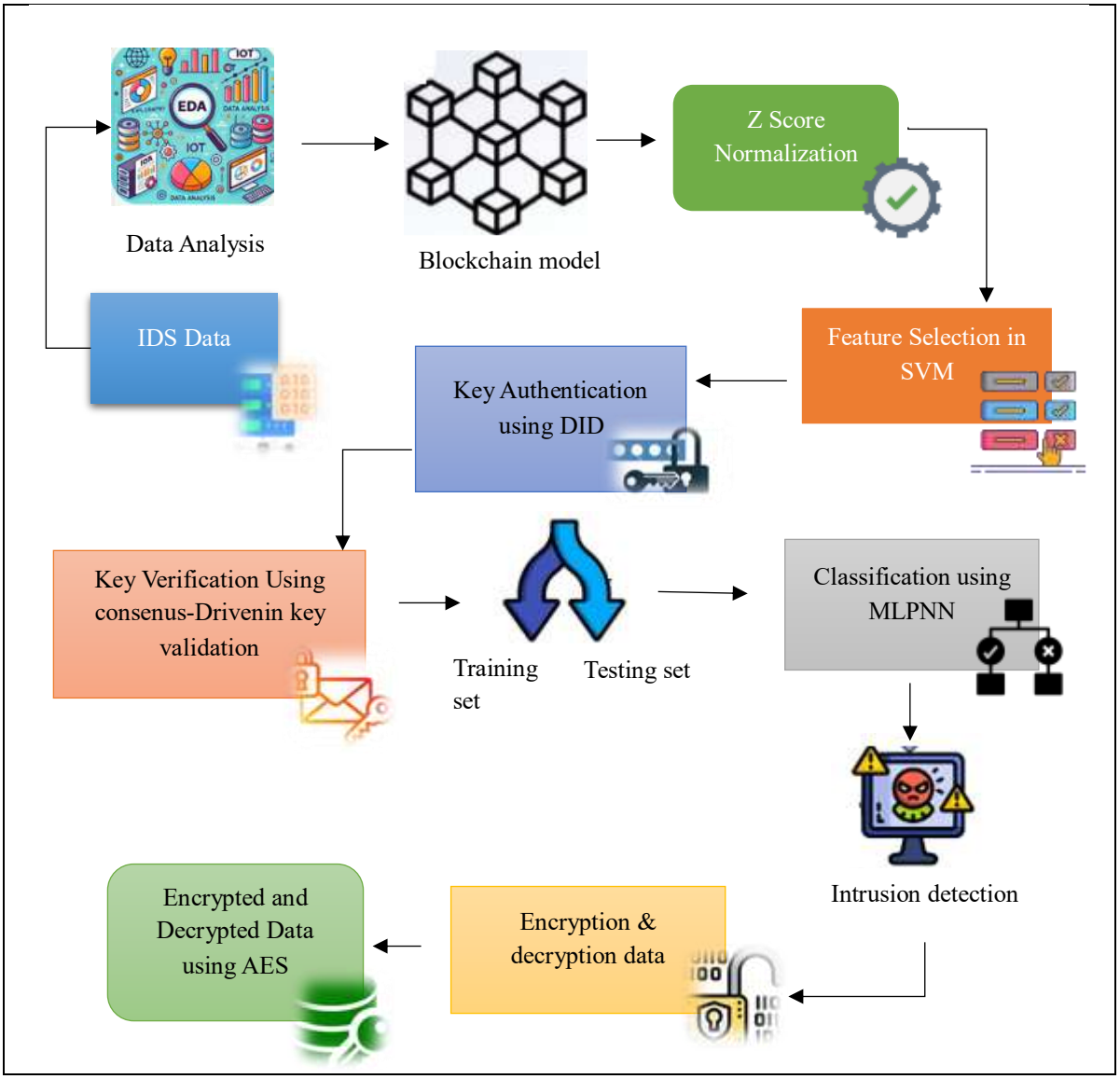

Figure 1: Architecture Diagram of AES-Enhanced Blockchain Intrusion Detection.
Source: Authors, (2026).

The figure 1 show it Z-Score Normalization Z-score normalization normalizes transaction characteristics by ensuring scaling consistency and improving model consistency. Support vector machines (SVMs) help classify behavior as normal or malicious, with the aim of increasing detection accuracy. DID uses verifiable consensus to securely identify nodes, CDKV uses network consensus to verify transactions so unauthorized parties cannot access them. Multilayer Perceptron Neural Networks (MLPNNs) offer linear modeling capabilities. Finally, Advanced Encryption Standards (AES) provide data security, preventing leakage of sensitive information during storage and transmission. Collectively, these methods create an effective, secure, and efficient blockchain intrusion detection system.

III.1 DATASET DESCRIPTIONS

The section dataset is provided for the audit, which contains various simulated intrusions conducted in a military network environment. It provides a view that captures raw TCP/IP dump traffic of the network by emulating a standard U.S. Air Force LAN. This LAN is centralized, similar to a real-world environment, and is subject to various attacks. A connection is a series of TCP packets, beginning and ending within a defined time period, that flow from a source IP address to a destination IP address within a defined protocol. Each attachment is defined as either a normal attachment or a single attack type. All connection records are approximately 100 bytes in size. For each TCP/IP connection, 41 quantitative and qualitative features (3 qualitative factors and 38 quantitative factors) are read from the normal data and the attack data. Class variables are divided into two types: normal and abnormal.

III.2 Z-SCORE NORMALIZATION

The data preprocessing technique called Z-score normalization transforms numerical values in CSV datasets to achieve standard normal distribution by computing them from mean and standard deviation parameters. Each data point receives its adjustment based on the mean and standard deviation of the feature it belongs to. The dataset achieves normalized statistics as features obtain an average value of zero with a standard deviation set at one, which enables acceptance by learning algorithms sensitive to scale variation. Z-score normalization provides optimal results in feature normalization because it allows for uniform importance between variables while maintaining standardization regardless of original measurement units.

$$Z_i = \frac{X_i - \mu}{\sigma} \tag{1}$$

The normalization process of CSV dataset points described by Equation 1 includes both mean μ and standard deviation σ adjustment to obtain standardized value Z_i. The calculated value Z_i determines the number of differences in standard deviations between a value and the mean. The X_i represents an initial value from the dataset. At the same time, μ refers to the mean column value, and σ represents the standard deviation column amount, along with Z_i representing the normalized standardized value of raw data. This normalization method enables different numerical features to contribute proportionally, ensuring that large values do not control training algorithms.

$$\mu = \frac{1}{N}\sum_{i=1}^{N} X_i \tag{2}$$

The μ value of a dataset column represents the calculated average obtained by adding all individual values in the column. The mean calculation requires a sum of all values X_i followed by division by N, which represents the total count. This formula shows how the variable N counts all data points from the column, while X_i represents the sum of all column values. The discussed equation is the main distribution center for characteristic values needed in Z-score transformations.

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(X_i - \mu)^2} \tag{3}$$

The σ indicates data point distribution about the mean value. Higher values of standard deviation represent wide dispersion of data points so data exists farther from the mean whereas lower values show data points tightly grouping near the mean.

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(X_i - \mu)^2} \tag{4}$$

This equation is selected when working with sample data from a larger population instead of complete population data. The main distinction from the initial formula involves substituting N with $N-1$ in the denominator to improve accuracy when measuring the population standard deviation from a group. The Z-score normalization process divides data in CSV files into several independent groups since CSV files consist of multiple numerical columns. The general formula for applying Z-score normalization to an entire dataset is,

$$Z_{ij} = \frac{X_{ij} - \mu_j}{\sigma_j} \tag{5}$$

The equation uses X_{ij} as the original value from row i and column j. At the same time, μ_j represents column mean values and σ_j represents the standard deviation of the columns, leading to Z_{ij} as the normalized value. The formula establishes independent standardization techniques that preserve data scaling consistency across all columns within the CSV dataset.

$$X' = X_i \text{ (if } X_i \text{ is not missing), else } X' = \mu \tag{6}$$

The analysis of actual datasets shows missing values (NaN values). The standard procedure for dealing with missing values X_i is replacing them with column mean μ. Missing values do not interrupt the normalization procedure through the implementation of this preventional measure.

$$X_i = Z_i.\sigma + \mu \tag{7}$$

The model interpretation requires the return of normalized values back to their original scale after completing the training. The procedure returns standardized value Z_i to its original scale by reversing normalization transformation.

III.3 SUPPORT VECTOR MACHINE SVM METHOD

The section SVM is a supervise machine learning technique that is mostly used in classifiers and regression. Concerning feature selection, SVM advantage uses the functional ability to determine the significance of each feature in terms of its contribution to the decision boundary of the hyperplane. Features are ordered according to the measure of how they impact an SVM's decision function. In this method, features are gradually eliminated one by one according to their weight in this SVM model until the minimal set is obtained. In this way, only significant features are kept with the method, decreasing dataset complexity, enhancing model solutions, and preventing overfitting. The number of kernel functions available in SVM can be linear, polynomial, or radial basis functions, and they can identify the linear or non-linear relations of the features. In the equation 8 perform SVM decision function,

$$f(i) = u^T i + b \tag{8}$$

Let assume, i as input feature vector, u as weight vector, b as bias term. This equation defines a hyperplane that separates different classes. The goal is to find u and b that maximize the margin between the two classes. By following in equation 9 maximize the margin,

$$\gamma = \frac{1}{\|u\|} \tag{9}$$

Here, the $\|u\|$ is represented for the magnitude of the weight vector, this equation is used to maximize this margin to ensure better generalization. The margin is the distance between the closest points (support vectors) of the different classes to the hyperplane. Then we optimization the problem for SVM through equation 10,11.

$$\min_{u,b} \frac{1}{2}\|u\|^2 \tag{10}$$

subject to the constraints:

$$j_x(u^T i_x + b) \geq 1 \text{ for all } x = 1,2,\dots,n \tag{11}$$

Let assume, i as training sample, j as the label, an n as total number of samples. The equation is used to find the optimal hyperplane that maximizes the margin while ensuring that the data points are classified correctly. By following we support vectors through equation 12, which the data points that lie closest to the decision boundary (hyperplane) and play a critical role in defining the hyperplane.

$$j_x(u^T i_x + b) = 1 \tag{12}$$

This equation used in the model to construct the decision function. This model contributes to the decision boundary, and their associated weights provide information about which features are most important. Then we perform kernel trick through equation 13, to map the data into a higher-dimensional space in cases where the data is not linearly separable,

$$f(i) = \sum_{x=1}^{n} \alpha_x j_x Q(i_x, i) + b \tag{13}$$

Let assume, Q as kernel function, α_x as Lagrange multipliers, it is used to assign u to each support vector, and j_x as class label of support vector. This equation is used to map the original data into a new higher-dimensional space where there exists a linear boundary in order to solve non-linear classification problems. The significance of each of the features can be determined by the examination of the u or the coefficients relating to the respective feature in the model produced. In linear cases, the magnitude of weight coefficients is used to determine the feature importance of $|u_y|$. We identified that those attributes which have a higher value in terms of weightage are important Features with higher magnitude weights are considered more important. This is why SVM is frequently used in image recognition, disease prediction, and other high-dimension problems: SVM is capable of identifying features that should be important and at the same time, it can sort data quickly and efficiently. This mathematical formulation, such as the decision function and the optimization process, and the kernel method guarantee that SVM as an algorithm is a reliable algorithm in machine learning, especially for feature selection in big and demand data.

III.4 DECENTRALIZED IDENTITY (DID)

The key authentication process uses decentralized identities (DITs) with smart contracts to verify that only legitimate nodes or organizations are allowed to join a blockchain-based intrusion detection system. In this approach, each node generates a DID and stores it on the blockchain as a permanent and auditable digital identity. Whenever a node tries to join or communicate with the system, the smart contract verifies the authenticity of the DID through the blockchain ledger. This eliminates the centralization of certificate authorities and the risk of identity spoofing or unauthorized access. The DID-to-public key relationship is established and validated by the smart contract, verifying that the requested identity is indeed associated with the node.

This mechanism provides a convenient, transparent and tamper-proof key authentication solution by improving the overall reliability of intrusion detection in distributed environments through a decentralized verification and automated trust process. The equation 14 binding process binds not only the node's identity, but also its encryption keys inextricably into a single, indestructible value. This DID is embedded in the blockchain, making it a permanent and immutable identifier. Therefore, hashing the public key and identity preserves the confidentiality of the original identity and the proof of the node's true identity during the authentication process. Let assume the DID_{node} −decentrlized identifier, ID_{node}-unique identify of the node, PK −public key of the node, H −cryptographic hash function, $||$ −concatenation.

$$DID_{node} = H(DID_{node}||PK_{node}) \tag{14}$$

The equation 15 DIT is registered, the smart contract will perform an authentication process to ensure that the submitted DIT is in the blockchain ledger. If the DID can be found, the contract executes the validation routine correctly associated with the DID. When both conditions are met, the authentication variable becomes 1, indicating that authentication is successful. Otherwise, the node is excluded from the system. The advantage of this is that it ensures that only genuine and pre-registered nodes can be authenticated and that there is no identity spoofing or unauthorized access. Let assume the $Auth_{node}$ − autheitcation values, $ledger$ −blockchain stored DID.

$$Auth_{node} = \begin{cases} 1, DID_{node} \in ledger \\ 0, \ otherwise \end{cases} \tag{15}$$

The equation 16 policy can be role-based or attribute-based, and when the smart contract rules are followed, the node is authorized to join or communicate with the IDS network. Otherwise, access is denied. Through this step, decentralized and transparent access control is implemented in the access control mechanism, where even authorized nodes require full rights to participate in accordance with the blockchain-mediated security policy. Let assume the $Access_{node}$ −final access status of node, SC_{policy} −smart contract policy

$$Access_{node} = Auth_{node} \times SC_{policy} \tag{16}$$

Identity and Public Key Generation DID generation and record cryptographic hashing combine both to provide each node with its own identity and public key. The generated DID is stored on the blockchain, hence incorruptible and decentralized. The result of this process is secure and verified node authentication without the need for a central authority.

III.5 CONSENSUS-DRIVEN KEY VALIDATION

Key Verification The key verification process uses each connected device to verify or deny the authenticity of the key before it is communicated within the intrusion detection workflow in a consensus-driven manner. When a node provides its public key, the proposed binding of the key to a decentralized identity (DID) is broadcast on the blockchain network. That is, instead of a single point of authority, multiple validator nodes participate in verifying that the key is correctly associated with the DID and has not been compromised. Each validator individually verifies the key binding and uses a consensus mechanism (such as proof-of-stake, practical Byzantine fault tolerance, or proof-of-representation) to reach consensus on whether the key is valid. When a majority of validators prove it is correct, the key is validated and permanently written to the blockchain ledger. Such a process supports transparent key verification, key tamper resistance and single point failure protection, improving the reliability and robustness of the intrusion detection mechanism. The equation 17 contains the node's decentralized identity, sharing this request with the blockchain network, the node asks the system to confirm or deny that the provided key is actually its DID. This request triggers consensus-driven verification, so that when a node requests verification, its keys are verified in a decentralized and transparent manner that considers all nodes in the decentralized network before passing the request to an acceptable state during the intrusion detection process. Let assume the Req_{key} −key verification, DID_{node} −decentrized identify of the node, PK_{node} −public key of the node.

$$Req_{key} = (DID_{node,}PK_{node}) \tag{17}$$

The equation 18 validation request is broadcast, validator nodes in the blockchain network independently evaluate the binding of the DIT and the public key. Verifier's Result: If the verify method Verify (·) verifies that the DID is correctly associated with the public key on the blockchain, the validator will output 1 (valid). Otherwise, it will output 0 (invalid). This is to ensure that validation is distributed across multiple validators to avoid single point manipulation and improve validation efficiency. Let assume the v_i − decision validator, $verify(.)$ − function that checks in public key

$$v_i(Req_{key}) = \begin{cases} 1, if \ verify(.)(DID_{node,}PK_{node}) = true \\ 0, \qquad\qquad otherwise \end{cases} \tag{18}$$

The equation 19 validity of the key is greater than or equal to, then the key is considered valid. Otherwise, the key is declared invalid and rejected. This makes key validation not dependent on a single validator, but rather the consensus of the network, and therefore can be considered non-tampering and acceptable. Let assume the $valid(pk_{node})$ −final decision on the validity of the node, N −total number of validators, θ −cpnsenus threshold

$$valid(pk_{node}) = \begin{cases} 1, \ \ if \ \frac{\sum_{i=1}^{n} v_i}{N} \geq \theta \\ 0, otherwise \end{cases} \tag{19}$$

Key verification using consensus ensures that a node's key is verified by multiple blockchain validators rather than by an individual. Each validator validates the key binding based on the node's DID, which is determined by a consensus threshold. This decentralized system prevents any kind of tampering and improves the reliability of the verification process.

III.6 MULTI-LAYER PERCEPTRON NEURAL NETWORK (MLPNN)

The section IDS prediction uses the Multi-Layer Perceptron Neural Network (MLPNN) to classify datasets as an artificial neural network. The network architecture includes an input layer followed by multiple hidden layers before reaching the output layer, and all of these layers use weighted connections. The supervised learning system of MLPNN depends on backpropagation and gradient descent for its optimization process. IDS prediction through MLPNN involves the analysis of historical IDS data, which includes price trends, trading volumes, and technical indicators to produce IDS classification results between bullish, bearish, and neutral statuses. The activation functions in ReLU or Sigmoid help the system create precise forecasts by simultaneously learning multiple complex patterns. The predictive power of MLPNN increases because it applies deep learning features and extraction techniques to become a beneficial predictor for financial decisions and investing approaches. In equation 20 weighted sum at each neuron,

$$z_j = \sum_{i=1}^{n} w_{ji} x_i + b_j \tag{20}$$

Calculating weighted sum input features occurs through Equation 23 across each neuron of hidden and output layers. The sentence includes various IDS -related features symbolized as x_i, encompassing opening price, closing price, volume, moving averages, and relative strength index (RSI). The overall model flexibility benefits from bias (b_j) adjustments while the weights (w_{ji}) control the importance of each feature. IDS relationships and price trend associations become learnable through activating the weighted input before it becomes the activation function input. By following we perform activation function, in equation 21 we perform sigmoid activation

$$f(z) = \frac{1}{1+e^{-z}} \tag{21}$$

Equation 22 finds practical use when analyzing binary classification issues because it predicts IDS value movement between bullish (1) and bearish (0). The function converts the weighted sum z into values ranging from 0 to 1 to enable forecasting IDS trends in bullish (1) or bearish (0). In equation 16 we perform Rectified Linear Unit (ReLU) function,

$$f(z) = \max(0, z) \tag{22}$$

IDS prediction systems employ Equation 23 as a standard implementation within MLPNN. The equation introduces nonlinearity to the analysis without causing gradient vanishing effects. Due to its limited positive value generation process, the network efficiently recognizes meaningful IDS shifts from past data patterns. Then we perform Hyperbolic Tangent ($\tan h$) through equation 17,

$$f(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}} \tag{23}$$

According to Equation 24, the alternative to the sigmoid activation produces output values spanning from -1 to +1. Zero-centered data enable improved prediction learning because it suits strongly bullish, slightly bullish, neutral, lightly bearish and strongly bearish multi-class IDS trend analysis.

$$y_j = f(z_j) \qquad 8 \tag{24}$$

Starting from the weighted sum result, the activation function generates output y_j for the following layer. The softmax activation function helps the final output layer classify stocks into three categories: Buy, Hold, and Sell. The output layer should employ linear activation since it deals with continuous IDS price predictions. By following we compute the Mean Squared Error (MSE) through equation 25 for regression,

$$\mathbb{E} = \frac{1}{N}\sum_{k=1}^{N}(y_k - \hat{y}_k)^2 \tag{25}$$

The relationship determines the distance between predicted IDS values $\hat{y}_k$ and actual IDS values y_k. Model accuracy improves as the MSE decreases in the MLPNN predictive method. Future IDS predictions use this loss function that operates on historical data. Then we compute the cross-entropy loss through equation 26 to classification,

$$\mathbb{E} = -\sum_{k=1}^{N}[y_k \log(\hat{y}_k) + (1 - y_k)\log(1 - \hat{y}_k)] \tag{26}$$

For IDS category classification (Buy, Hold, or Sell), equation 29 replaces Mean Squared Error (MSE). Equation 27 computes the distinction between registered values and calculated probability estimates. The model receives low loss when it correctly predicts a high probability value, which optimizes MLPNN for improved IDS trading decisions. In equation 21 we perform backpropagation for update the weight by utilizing gradient descent,

$$w_{ji} \leftarrow w_{ji} - \eta \frac{\partial \mathbb{E}}{\partial w_{ji}} \tag{27}$$

Equation 27 enables the MLPNN to minimize errors through weight adjustments in the training phase. The gradient descent algorithm applies derivative computation on loss function ($\mathbb{E}$) to update weights w_{ji}. The learning rate η sets the amount by which weights adjust in each iteration.

The selection of an accurate learning rate in IDS prediction ensures efficient model learning, which avoids excessive step sizes and slow convergence rates. The system utilizes IDS inputs to perform weight-based transformations while including activation functions to introduce non-linearities that compute the loss difference between actual and predicted values before employing algorithm-enabled weight updates. The architecture of MLPNN makes it possible to identify IDS patterns for financial investors who need correct trading information.

III.7 ADVANCED ENCRYPTION STANDARD (AES)

The Advanced Encryption Standard (AES) is used for the encryption part of the blockchain-based intrusion detection process, which ensures confidentiality of communications and sensitive data shared between nodes. Each node encrypts the data using a key (a symmetric key, a session key derived from a decentralized identity (DIT), or a smart contract policy negotiation) before transmission. Plaintext data generated by an intrusion detection sensor or IDS alarm is encrypted into ciphertext using a series of permutations, permutations, and combination changes. This makes the information technically unintelligible to other entities. If the cipher is successfully authenticated and verified as belonging to the receiving node through the DIT and key verification process, the same symmetric key can be used to decrypt the cipher. Incorporating AES managed by blockchain control provides system confidentiality, machine integrity and secure communication, as only authorized and verified nodes can access IDS data in an untampered decentralized system. The equation 28, each node is equipped with its decentralized identity and a symmetric session key generated based on the access or session policy established in the smart contract. The key derivation function ensures that each node has a unique session key associated with its identity and permissions, and denies access to each unauthorized node. By linking the key to the blockchain's governance policy, the system implements access control, ensuring that only nodes cleared and authorized via DID can encrypt or decrypt IDS data using the key. Once the plaintext data is created, the generated session key is ignored. Let assume the $K_{session}$ −Key symmetric, DID_{node}-Decentralized Identify of the node, SC_{policy} −session policy, f −key derivation function.

$$K_{session} = f(DID_{node,}SC_{policy}) \tag{28}$$

The equation 29 data represented (C) consists of intrusion detection alerts or sensor output messages, and this data is encrypted using the AES cipher and the derived session key, resulting in a ciphertext unintelligible to any unauthorized node. This prevention measure ensures that data traffic cannot be read or tampered with, even if it is sent over an unsecured network. The collection process becomes part of the blockchain because only nodes that know each other using their DITs and session policies can generate valid ciphers and/or decrypt them. Let assume the c −ciphertext, p −plaintext data, AES_{Enc} − encryption function

$$c = AES_{Enc}(p, k_{session}) \tag{29}$$

The equation 30 target node, the authorized parties receive the same session key, so that only nodes that have authenticated, verified, and assigned the correct session key through the smart contract and the mechanisms provided by the DID system can access the initial data of the IDS. This step preserves the confidentiality and integrity of the intrusion detection system, enabling real-time and secure data sharing between decentralized nodes.let assume the P −deccrypted plaintext data, c −recevied ciphertext, AES_{Dec} −decryption function

$$P = AES_{Dec}(c, k_{session}) \tag{30}$$

Encryption of IDS data is achieved by encrypting the plaintext into a ciphertext using a symmetric session key derived from the node DID and smart contract policy. Only authorized nodes possessing the hard-coded session key can decrypt information. It provides confidentiality, integrity and controlled access within a decentralized IDS system.

IV. RESULT AND DISCUSSION

The section provides robust verification of public keys, rejecting any invalid or malicious keys through the moderator matching protocol, thereby increasing trust and reliability. All transmitted IDS data is encrypted using AES encryption, so that only the intended node can decrypt the sensitive information using its authorized session key, thus guaranteeing confidentiality and information integrity. Performance measurements demonstrate high detection accuracy, low false positive rate, and moderate computational overhead in key authentication and encryption tasks, indicating that the protocol is feasible in one-shot and test-in-test implementation scenarios. Overall interoperability, decentralized authentication, joint key verification, and strong cryptographic processes provide an effective and scalable intrusion detection framework that addresses the limitations of traditional IDS.

Table 2: Simulation Parameter.

Parameters	Values
Blockchain platform	Ethereum
Smart contract development	Chain code
Cryptography libraries	PyCryptodome
IDS framework	TensorFlow, PyTorch
Programming languages	Python
Network communication tools	REST API, WebSockets

Source: Authors, (2026).

This table 2 represents the core of hardware and software solutions needed to build a blockchain-based IDS implementation. Blockchain consists of software programs, including smart contracts and cryptographic libraries, to handle secure authentication and key verification and data encryption. Machine learning algorithms and databases are used to detect possible conflicts and store IDS logs. Collectively, these resources ensure that an intrusion detection system is secure, effective, and scalable.

IV.1 DISCUSSION PART

Blockchain-based IDS enhance efficiency by optimizing key aspects of workflow and resource utilization. Utilizing sophisticated machine learning models and anomaly detection algorithms, combined with real-time data analysis, can improve detection accuracy and reduce false favorable rates. Key verification times can be reduced by using efficient consensus mechanisms such as Practical Byzantine Fault Tolerance (PBFT) or Lightweight Proof-of-Stake. Encryption and decryption speed can be tuned by hardware acceleration of the AES implementation to find a speed/security tradeoff. In short, the concept of algorithm development and decentralized resource management ensures that the system is more efficient, responsive, and reliable in detecting attacks.

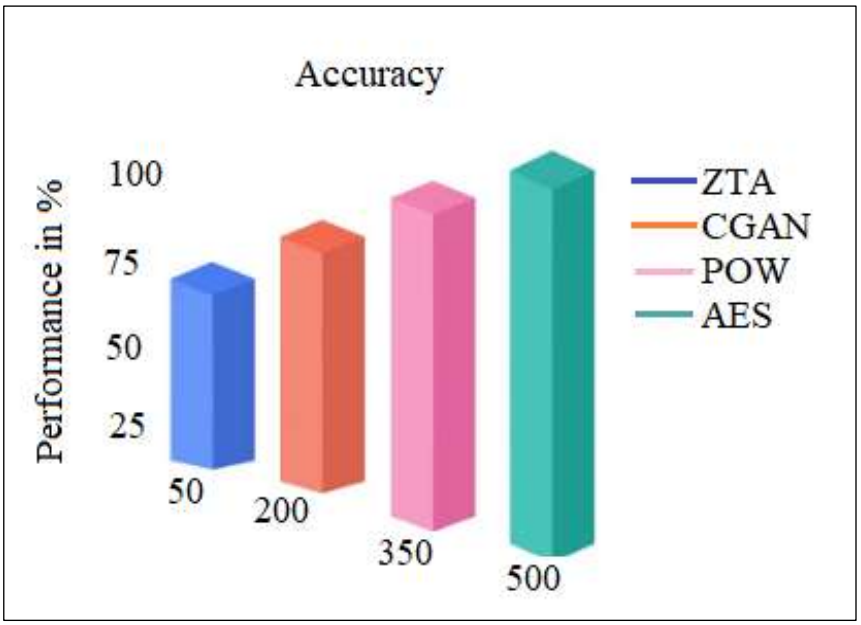

Figure 2: Analysis of Accuracy.
Source: Authors, (2026).

Figure 2 shows AES-enhanced blockchain intrusion detection system for secure networks. The suggested AES approach outperformed well-known methods, such as ZTA, POW and CGAN, with 75%, 86%, and the proposed method 90% prediction accuracy, respectively. key based on smart contract policy, thus ensuring that only authorized and verified nodes can decode sensitive information.

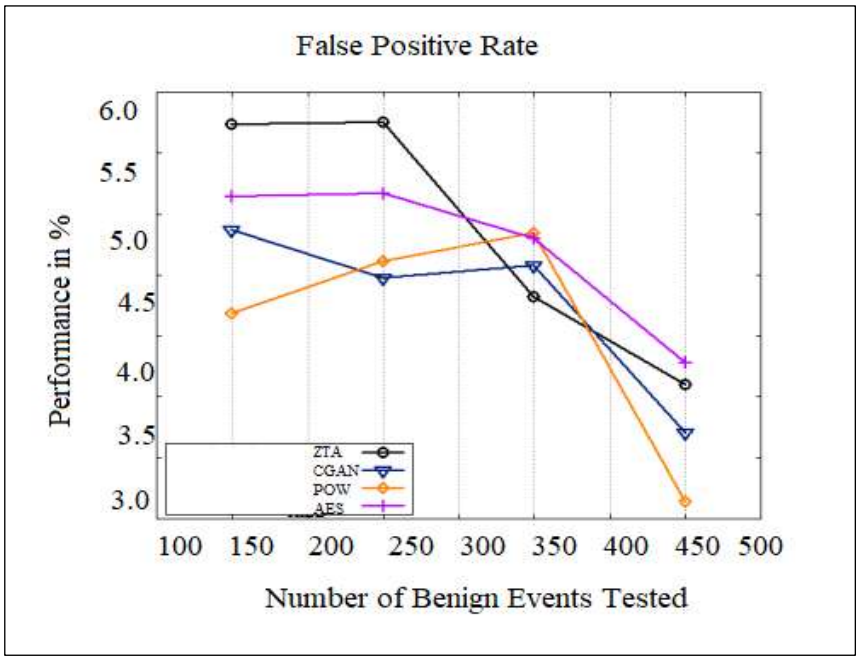

Figure 3: Analysis of False positive rate.
Source: Authors, (2026).

Figure 3 shows AES-enhanced blockchain intrusion detection system for secure networks. The suggested AES approach outperformed well-known methods, such as ZTA, POW and CGAN, with 85%, 76%, and 60% , the proposed method 55% prediction false positive rate , respectively. key based on smart contract policy, thus ensuring that only authorized and verified nodes can decode sensitive information.

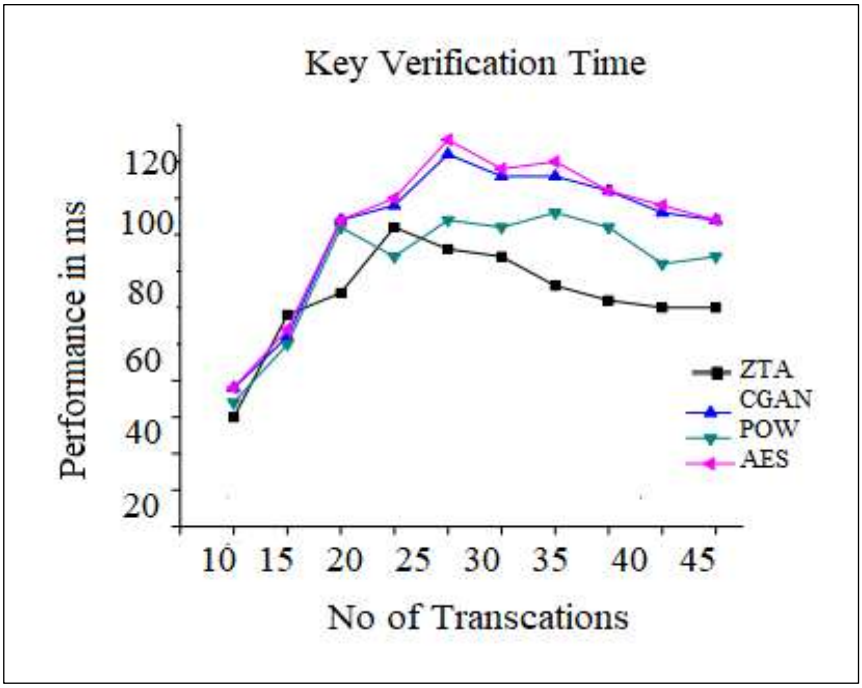

Figure 4: Analysis of Key Verification Time.
Source: Authors, (2026).

Figure 4 shows AES-enhanced blockchain intrusion detection system for secure networks. The suggested AES approach outperformed well-known methods, such as ZTA, POW and CGAN, with 75%, 86%, and 88% the proposed method 91% prediction key verification time, respectively. key based on smart contract policy, thus ensuring that only authorized and verified nodes can decode sensitive information.

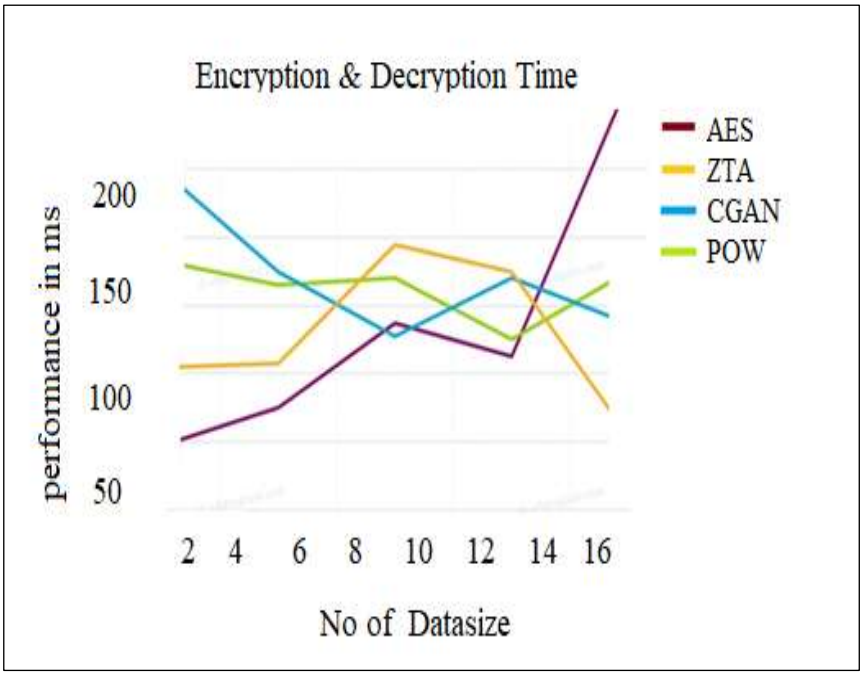

Figure 5: Analysis of Encryption & decryption Time.
Source: Authors, (2026).

Figure 5 shows AES-enhanced blockchain intrusion detection system for secure networks. The suggested AES approach outperformed well-known methods, such as ZTA, POW and CGAN, with 65%, 76%, 86%, and the proposed method 90% prediction encryption and decryption time, respectively. key based on smart contract policy, thus ensuring that only authorized and verified nodes can decode sensitive information.

V.CONCLUSION

In summary, intrusion detection and well-designed data protection techniques prove an integrated and multidimensional solution to ensure blockchain network security. The framework uses the decentralized and immutable architecture of blockchain to create a secure environment for monitoring transactions and node activity. Intrusion detection systems (IDS) can be used to detect suspicious or unusual patterns in real-time, overcoming the inherent weaknesses of traditional blockchain processes, including high computational costs and vulnerability to attacks. The data can be pre-processed by normalizing the Z-score to maintain a uniform distribution of transaction features, thereby improving the stability and performance of the specified model. Support vector machines (SVM) and distributed identities (DIDs) backbone or consensus-based key verification can be used to effectively distinguish between normal and malicious activities and ensure effective node authentication and secure verification of transactions. Multilayer perceptron neural networks (MLPNNs) can further improve prediction accuracy by capturing problems that standard models overlook. Finally, the Advanced Encryption Standard (AES) helps protect the confidentiality and integrity of sensitive information in storage and transmission. Together, this workflow addresses both security and performance concerns, providing a scalable, efficient, and reliable system for blockchain intrusion detection. By combining pre-processing, machine learning, entity management, verified consensus and encryption, it increases the reliability of the computer network while reducing the limitations of traditional blockchain implementations.

VI. AUTHOR'S CONTRIBUTION

Conceptualization: Hemanth Uppala, R. Renuga Devi.
Methodology: Hemanth Uppala, R. Renuga Devi.
Investigation: Hemanth Uppala, R. Renuga Devi.
Discussion of results: Hemanth Uppala, R. Renuga Devi.
Writing – Original Draft: Hemanth Uppala, R. Renuga Devi.
Writing – Review and Editing: Hemanth Uppala, R. Renuga Devi.
Resources: Hemanth Uppala, R. Renuga Devi.
Supervision: Hemanth Uppala, R. Renuga Devi.
Approval of the final text: Hemanth Uppala, R. Renuga Devi.

VII. REFERENCE

[1] Liang, Wei, et al. "Data fusion approach for collaborative anomaly intrusion detection in blockchain-based systems." IEEE Internet of Things Journal 9.16 (2021): 14741-14751.

[2] Abdel-Basset, Mohamed, et al. "Federated intrusion detection in blockchain-based smart transportation systems." IEEE Transactions on Intelligent Transportation Systems 23.3 (2021): 2523-2537.

[3] Liu, Hong, et al. "Blockchain and federated learning for collaborative intrusion detection in vehicular edge computing." IEEE Transactions on Vehicular Technology 70.6 (2021): 6073-6084.

[4] Heidari, Arash, Nima Jafari Navimipour, and Mehmet Unal. "A secure intrusion detection platform using blockchain and radial basis function neural networks for internet of drones." IEEE Internet of Things Journal 10.10 (2023): 8445-8454.

[5] Li, Wenjuan, et al. "Toward a blockchain-based framework for challenge-based collaborative intrusion detection." International Journal of Information Security 20.2 (2021): 127-139.

[6] Aliyu, Ibrahim, et al. "A blockchain-based federated forest for SDN-enabled in-vehicle network intrusion detection system." IEEE Access 9 (2021): 102593-102608.

[7] He, Xiaoqiang, et al. "CGAN-based collaborative intrusion detection for UAV networks: A blockchain-empowered distributed federated learning approach." IEEE Internet of Things Journal 10.1 (2022): 120-132.

[8] Alevizos, Lampis, et al. "Blockchain-enabled intrusion detection and prevention system of APTs within zero trust architecture." Ieee Access 10 (2022): 89270-89288.

[9] Kumar, Randhir, et al. "A privacy-preserving-based secure framework using blockchain-enabled deep-learning in cooperative intelligent transport system." IEEE Transactions on Intelligent Transportation Systems 23.9 (2021): 16492-16503.

[10] Abou El Houda, Zakaria, et al. "Blockchain-enabled federated learning for enhanced collaborative intrusion detection in vehicular edge computing." IEEE Transactions on Intelligent Transportation Systems 25.7 (2024): 7661-7672.

[11] Moulahi, Tarek, et al. "Privacy-preserving federated learning cyber-threat detection for intelligent transport systems with blockchain-based security." Expert Systems 40.5 (2023): e13103.

[12] Pranto, Tahmid Hasan, et al. "Blockchain and machine learning for fraud detection: A privacy-preserving and adaptive incentive based approach." Ieee Access 10 (2022): 87115-87134.

[13] Truong, Vu Tuan, and Long Bao Le. "MetaCIDS: Privacy-preserving collaborative intrusion detection for metaverse based on blockchain and online federated learning." IEEE Open Journal of the Computer Society 4 (2023): 253-266.

[14] Cui, Jie, et al. "Collaborative intrusion detection system for SDVN: A fairness federated deep learning approach." IEEE transactions on parallel and distributed systems 34.9 (2023): 2512-2528.

[15] Said, Rachid Ben, Zakaria Sabir, and Iman Askerzade. "CNN-BiLSTM: A hybrid deep learning approach for network intrusion detection system in software-defined networking with hybrid feature selection." IEEE Access 11 (2023): 138732-138747.

[16] Ferrag, Mohamed Amine, et al. "Cyber security intrusion detection for agriculture 4.0: Machine learning-based solutions, datasets, and future directions." IEEE/CAA Journal of Automatica Sinica 9.3 (2021): 407-436.

[17] Abou El Houda, Zakaria, Abdelhakim Senhaji Hafid, and Lyes Khoukhi. "MiTFed: A privacy preserving collaborative network attack mitigation framework based on federated learning using SDN and blockchain." IEEE Transactions on Network Science and Engineering 10.4 (2023): 1985-2001.

[18] Nguyen, Gia Nhu, et al. "Secure blockchain enabled Cyber–physical systems in healthcare using deep belief network with ResNet model." Journal of parallel and distributed computing 153 (2021): 150-160.

[19] Ramanan, Paritosh, Dan Li, and Nagi Gebraeel. "Blockchain-based decentralized replay attack detection for large-scale power systems." IEEE Transactions on Systems, Man, and Cybernetics: Systems 52.8 (2021): 4727-4739.

[20] S. Alharbi, D. Alghazzawi, A. Hakeem, L. Mohaisen, L. Cheng and A. Attiah, "A Blockchain-Based Collaborative Intrusion Detection Systems Framework," in IEEE Internet of Things Journal, vol. 11, no. 15, pp. 25481-25493, 1 Aug.1, 2024, doi: 10.1109/JIOT.2023.3347492.