Hybrid Collaborative Intrusion Detection System Based on Blockchain & Machine Learning

Farah Jemili (jmili_farah@yahoo.fr)  
University of Sousse

Ouajdi Korbaa  
University of Sousse

Research Article

Keywords: Collaborative IDS, Hybrid IDS, Blockchain, Machine Learning

Posted Date: August 21st, 2023

DOI: https://doi.org/10.21203/rs.3.rs-3262119/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License. Read Full License
Abstract

Intrusion Detection Systems (IDS) have traditionally been designed with a centralized structure, where a single device is responsible for monitoring the entire network. However, with the increasing complexity and scale of modern networks, this approach has become less effective. Centralized IDS can suffer from performance issues, limited scalability, and vulnerability to targeted attacks. To address these limitations, there is a growing need to develop collaborative IDS that can distribute the workload across multiple devices and better handle large-scale networks. Collaboration enables IDS to detect intrusions more effectively by combining and analyzing data from multiple sources. The adoption of blockchain technology is essential in achieving a collaborative IDS. Blockchain provides a secure, decentralized way to store and exchange information between different devices, which is critical for building trust and ensuring the integrity of the system. Furthermore, machine learning algorithms can be used to improve the performance of IDS by detecting new and emerging threats. Machine learning can help to identify patterns and anomalies in network traffic, enabling the system to detect and respond to attacks more effectively. By combining these approaches, a reliable and scalable detection system can be developed. The collaborative IDS using blockchain technology and machine learning algorithms can improve the accuracy and efficiency of detecting network intrusions while maintaining the security and integrity of the system.

I. Introduction

The field of intrusion detection systems (IDS) has undergone significant technological evolution due to the complex and vulnerable nature of computer systems on open, distributed networks. With the rise of malicious access and cyber-attacks, centralized and monolithic IDSs are no longer suitable for the contemporary network environment. A more effective approach is a collaborative IDS framework, where multiple components distributed throughout the network collaborate to analyze data and provide global and relevant alerts [1].

In an open, distributed, and heterogeneous context, setting up components that collaborate to perfect the functioning of an IDS is an ongoing challenge because of several reasons. Firstly, the network environment is complex and vast, and the number of components that need to work together to ensure reliable intrusion detection is significant. Secondly, these components may have different architectures, configurations, and operating systems, making it difficult to integrate them into a unified system. Lastly, maintaining the integrity and security of exchanged data is crucial because the system’s reliability and accuracy are based on the quality of data analyzed. Any manipulation or alteration of data can lead to biased results and disrupt the entire system’s functioning [3].

To address these concerns, blockchain technology is a suitable solution. Blockchain provides a decentralized, tamper-proof, and transparent mechanism for securely storing and sharing necessary information. As a distributed ledger technology, it allows for the creation of a shared and trusted network where all nodes have access to the same information and can verify its validity. The use of blockchain...
technology in IDS allows for secure data exchange and establishes collective trust between all nodes. Additionally, it enables the creation of an immutable record of all transactions and data exchanges, ensuring that the integrity of the exchanged data is maintained throughout the network [4].

The effectiveness of an IDS in detecting malicious activities largely depends on its analysis component. Machine learning-based analysis methods refer to the use of algorithms that can learn from data and improve their performance over time. These methods can be used to analyze and extract insights from large and complex datasets. There are various analysis methods, each with its own strengths and limitations. To overcome the limitations of each individual analysis method and take advantage of their strengths, they can be used together in a complementary way. For example, Autoencoders and DNNs can be used together: An autoencoder can be used to pretrain a DNN by learning a compressed representation of the data, which can then be used as input to the DNN for classification or other tasks. This approach can help improve the performance of the DNN, especially when labeled data is limited. Moreover, autoencoders can be used as a form of data augmentation for DNNs. By applying random transformations to the input data, such as scaling, rotation, or cropping, an autoencoder can learn to reconstruct the transformed data, effectively creating new examples that can be used to train the DNN [5].

This approach combines both analysis methods in order to provide more reliable detection and reduce the risk of false positives and false negatives. It also proposes and implements a set of collaborative IDS using blockchain technology to ensure secure data exchange and establish collective trust between all nodes. By combining all these approaches, the resulting IDS can effectively detect and inhibit disruptions on open, large-scale, and infinitely distributed networks.

II. Background

2.1 Collaborative Intrusion Detection Systems

Collaborative Intrusion Detection Systems (CIDS) are a type of intrusion detection system that involves multiple entities working together to detect and respond to security threats. CIDS can be classified into different categories based on the way they collaborate and the type of information they share [27].

One classification of CIDS is based on the level of collaboration between the participating entities. In this classification, there are three main types of CIDS [6]:

- Centralized CIDS: In a centralized CIDS, all participating entities send their security data to a central server, which then analyzes the data and generates alerts if any threats are detected. This approach is easy to implement and manage, but it can be a single point of failure if the central server is compromised.
- Distributed CIDS: In a distributed CIDS, each participating entity has its own local intrusion detection system, and they share information with each other to detect threats. This approach is more resilient to failures than centralized CIDS, but it can be more difficult to manage and coordinate.
• Hybrid CIDS: Hybrid CIDS combines elements of both centralized and distributed CIDS. In this approach, some entities send their data to a central server, while others share information directly with each other. This approach offers the benefits of both centralized and distributed CIDS, but it can be more complex to implement.

Another classification of CIDS is based on the type of information that is shared between the participating entities. In this classification, there are two main types of CIDS [7]:

• Signature-based CIDS: In a signature-based CIDS, participating entities share known attack signatures and use them to detect new attacks. This approach is effective against known attacks, but it is less effective against new and evolving attacks.

• Anomaly-based CIDS: In an anomaly-based CIDS, participating entities share information about normal behavior patterns and use them to detect deviations that may indicate an attack. This approach is effective against new and evolving attacks, but it can generate more false positives than signature based CIDS.

CIDS [8] can provide significant advantages over traditional intrusion detection systems by leveraging the collective knowledge and resources of multiple entities. However, the effectiveness of a CIDS depends on the level of collaboration between the participating entities, the type of information that is shared, and the accuracy of the detection algorithms used.

2.2 Blockchain Technology

Blockchain technology is a distributed ledger technology that allows secure and transparent transactions without the need for a trusted third party. Consensus algorithms are used to maintain the integrity of the blockchain by ensuring that all nodes in the network agree on the state of the ledger [10].

• Proof-of-Work (PoW) is the original consensus algorithm used in blockchain technology. In PoW, nodes in the network compete to solve a mathematical puzzle to add new blocks to the chain. The first node to solve the puzzle broadcasts the new block to the network, and other nodes validate the block by checking the solution. The probability of solving the puzzle is proportional to the computational power of the node, making it a resource-intensive algorithm. The probability of a node finding the solution to the puzzle can be calculated as \( P = \frac{H}{(D \times 2^{32})} \), where \( P \) is the probability, \( H \) is the node's hash rate, and \( D \) is the network difficulty.

• Proof-of-Stake (PoS) is another consensus algorithm that works by allowing nodes to stake their tokens as collateral to participate in the consensus process. In PoS, a node is chosen to create a new block based on its stake in the network. The probability of a node being chosen to create a new block is proportional to its stake in the network, making it a more energy-efficient algorithm than PoW. The probability of a node being chosen to create a new block can be calculated as \( P = \frac{S}{T} \), where \( P \) is the probability, \( S \) is the node's stake, and \( T \) is the total stake in the network.
• **Delegated Proof-of-Stake (DPoS)** is a variation of PoS that introduces a voting system to elect a group of nodes called delegates to create new blocks. In DPoS, token holders vote for delegates who are responsible for creating new blocks on their behalf. The delegates are incentivized to act honestly as their reputation is at stake. The probability of a delegate being chosen to create a new block can be calculated as \( P = \frac{W}{2^{16}} \), where \( P \) is the probability, \( W \) is the weight of the delegate's votes, and \( 2^{16} \) is a constant.

• **Practical Byzantine Fault Tolerance (PBFT)** is a consensus algorithm that is used in permissioned blockchains. In PBFT, nodes in the network communicate with each other to reach a consensus on the state of the ledger. The algorithm works by dividing the nodes into three groups: a primary group, a backup group, and a validation group. The primary group is responsible for proposing a new block, and the backup group is responsible for validating the block. The probability of a node being chosen as a primary or backup node can be calculated as \( P = \frac{1}{3f + 1} \), where \( P \) is the probability, and \( f \) is the number of faulty nodes that can be tolerated by the network.

Consensus algorithms are a crucial component of blockchain technology, and each algorithm has its advantages and disadvantages. The choice of consensus algorithm depends on the specific use case and the requirements of the network [11].

Blockchain has different types. Here are some of the most common ones:

• **Public Blockchain**: This type of blockchain is completely open and public. Anyone can join the network and participate in the consensus process. All transactions are transparent and can be viewed by anyone on the network. Bitcoin and Ethereum are examples of public blockchains.

• **Private Blockchain**: This type of blockchain is only accessible to a select group of people. The network is not open to the public, and the nodes are run by a single organization or group of organizations. Transactions on private blockchains are not transparent and can only be viewed by members of the network. Hyperledger Fabric is an example of a private blockchain.

• **Consortium Blockchain**: This type of blockchain is a hybrid of public and private blockchains. It is controlled by a group of organizations rather than a single entity, but access to the network is restricted. Consortium blockchains are often used for industry-specific applications where multiple organizations need to collaborate on a shared network.

• **Hybrid Blockchain**: This type of blockchain combines elements of both public and private blockchains. It allows for public participation in the consensus process, but also has restrictions on who can access the network. The goal of a hybrid blockchain is to provide the benefits of both public and private blockchains while minimizing their drawbacks.

Each type of blockchain has its own unique features and use cases, and the appropriate type to use depends on the specific application and requirements of the network.

Furthermore, Blockchain technology offers a number of key functionalities that make it a valuable tool for a variety of applications. Some of the main functionalities of Blockchain include [12]:
• Decentralization: The Blockchain is a decentralized system, which means that it operates without a central authority or control. Transactions are verified by a network of nodes rather than a single entity, making the system more secure and resistant to attacks.

• Immutability: Once data is recorded on the Blockchain, it cannot be altered or deleted. This is because the Blockchain uses cryptographic hashes to link each block to the previous one, creating a chain of blocks that is tamper evident.

• Transparency: Transactions on the Blockchain are transparent and can be viewed by anyone on the network. This promotes accountability and trust and helps to prevent fraud and corruption.

• Security: The Blockchain is highly secure, thanks to its decentralized structure and cryptographic algorithms. Transactions are verified using complex mathematical algorithms that make it virtually impossible for hackers to tamper with the data.

• Smart Contracts: Smart contracts are self-executing contracts that are programmed onto the Blockchain. They can automate the transfer of assets and other transactions, making them more efficient and less prone to errors.

The Blockchain’s key functionalities [13] make it a powerful tool for a variety of applications, including Cyber Security applications.

There are many blockchain platforms available today, each with its own unique features and capabilities. Here are some of the most well-known and widely used blockchain platforms:

• Ethereum: Ethereum is a blockchain platform that supports smart contracts, which are self-executing contracts that can automate the transfer of assets and other transactions. Ethereum is widely used for decentralized applications (DApps), including decentralized finance (DeFi) applications.

• Ripple: Ripple is a blockchain platform that is primarily used for cross-border payments and remittances. Ripple's network enables instant, low-cost transfers of funds between different currencies and countries.

• Hyperledger Fabric: Hyperledger Fabric is a blockchain platform that is designed for enterprise use. It offers high levels of scalability, privacy, and security, and it can be customized to meet the specific needs of different industries and organizations.

• Corda: Corda is a blockchain platform that is designed specifically for use in the financial services industry. It is designed to support complex financial agreements and transactions, and it offers high levels of privacy and security.

• Stellar: Stellar is a blockchain platform that is similar to Ripple in that it is primarily used for cross-border payments and remittances. Stellar's network enables fast and low-cost transfers of funds between different currencies and countries.

Intrusion detection systems (IDS) require a blockchain platform that can handle highly sensitive data with high levels of security and privacy. Among the five mentioned platforms, Ethereum and Hyperledger Fabric are the most suitable for IDS in terms of security and privacy.
Ethereum's smart contract functionality can be utilized for implementing a decentralized IDS, where the smart contract can manage the detection of intrusion events and handle the distribution of alerts and responses among network participants. Additionally, Ethereum's advanced cryptography and consensus algorithms provide high levels of security for sensitive data exchange [14].

Hyperledger Fabric's modular architecture, high performance, and flexible consensus mechanisms make it suitable for building highly secure and customizable IDS. Its permissioned network model enables fine-grained access control, which is essential for privacy protection and sensitive data handling.

Both platforms have been used in various research works related to IDS, indicating their suitability for such contexts. However, Hyperledger Fabric would be a better choice than Ethereum. Hyperledger Fabric is specifically designed for enterprise use and offers high levels of privacy and security. It uses a permissioned network, meaning that access to the network and data is restricted to authorized parties, which is crucial in the context of intrusion detection to prevent unauthorized access. Hyperledger Fabric also supports flexible and customizable smart contracts, which can be tailored to the specific needs of an organization or industry. This allows for the development of highly specialized and effective intrusion detection systems. On the other hand, while Ethereum also supports smart contracts and has been widely used in decentralized applications, it is a permissionless network, meaning that anyone can participate and access the data. This makes it less suitable for use in the context of intrusion detection, where privacy and security are of utmost importance [15].

Hyperledger Fabric's focus on security, privacy, and customization make it a more suitable choice for intrusion detection systems than Ethereum. Hyperledger Fabric is a permissioned, modular blockchain platform. It uses a consensus algorithm called Practical Byzantine Fault Tolerance (PBFT) to achieve consensus between nodes in the network. The PBFT algorithm works as follows [19]:

In addition to the PBFT consensus algorithm, Hyperledger Fabric uses smart contracts called chaincode to enable complex business logic and data access control. The chaincode is written in a programming language called Go or other compatible languages. The platform also supports private data collections, allowing for privacy and confidentiality of sensitive data.

The PBFT algorithm has been shown to be both efficient and secure, with a worst-case complexity of $O(n^2)$ message exchanges and a tolerance of up to $(n-1)/3$ Byzantine faults, where $n$ is the total number of nodes in the network.

### 2.3 Machine Learning

Machine learning is a subfield of artificial intelligence that deals with the design and development of algorithms that can learn from data and make predictions or decisions without being explicitly programmed. The goal of machine learning is to develop models that can automatically discover patterns in data and generalize to new, unseen data. One popular type of machine learning algorithm is the neural network, which is a computational model inspired by the structure and function of the brain. A
neural network is composed of multiple interconnected nodes, called neurons, that can receive and process input signals, and produce output signals. The connections between neurons are weighted, and the weights are learned from data through a process called training [16].

Autoencoder Networks with Deep Neural Networks (DNNs) are a type of neural network that combines two different architectures: the autoencoder network and the deep neural network. An autoencoder network is a neural network that is trained to reconstruct its own inputs. It consists of an encoder network that maps the input data to a lower-dimensional representation, and a decoder network that maps the lower-dimensional representation back to the original input. The encoder network learns a compressed representation of the input, while the decoder network learns to reconstruct the input from the compressed representation.

A DNN, on the other hand, is a neural network that has multiple hidden layers between the input and output layers. DNNs can learn complex functions that map inputs to outputs by combining simple nonlinear transformations in each layer.

The idea behind using an autoencoder network to pretrain a DNN is to use the encoder network to learn a compressed representation of the input data, and then use the weights of the encoder network to initialize the weights of the first layer of the DNN. By doing so, the DNN can learn a better representation of the input, leading to improved accuracy.

An autoencoder network with DNNs can be written as follows [21]:

Let $X$ be the input data, and $Y$ be the reconstructed output data. Let $f(X)$ be the output of the encoder network, and $g(f(X))$ be the output of the decoder network.

The loss function of the autoencoder network can be defined as the mean squared error between the input data and the reconstructed output data:

$$L(X, Y) = \frac{1}{N} \sum_{i=1}^{N} (X_i - Y_i)^2$$

where $N$ is the number of samples in the training dataset, $X_i$ is the $i$-th input sample, and $Y_i$ is the corresponding reconstructed output sample.

The loss function of the DNN can be defined as the cross-entropy loss between the predicted output and the true output:

$$L(Y_{\text{pred}}, Y_{\text{true}}) = -\sum_{i=1}^{N} (Y_{\text{true}_i} \log(Y_{\text{pred}_i}) + (1 - Y_{\text{true}_i}) \log(1 - Y_{\text{pred}_i}))$$

where $Y_{\text{true}_i}$ is the true output label of the $i$-th sample, and $Y_{\text{pred}_i}$ is the predicted output label of the $i$-th sample.

The total loss function of the autoencoder network with DNNs can be defined as the sum of the loss functions of the autoencoder network and the DNN:
\[ L_{\text{total}} = L(X, Y) + \lambda \cdot L(Y_{\text{pred}}, Y_{\text{true}}) \]

where \( \lambda \) is a hyperparameter that controls the relative importance of the autoencoder loss and the DNN loss.

During training, the autoencoder network [23] is first trained to minimize the reconstruction loss \( L(X, Y) \). Once the autoencoder network has converged, the weights of the encoder network are used to initialize the weights of the first layer of the DNN. The DNN is then fine-tuned by minimizing the total loss \( L_{\text{total}} \) using backpropagation and stochastic gradient descent.

### III. Related Work

In recent studies, researchers have proposed using blockchain technology in collaborative intrusion detection systems to enhance security and trust management among multiple intrusion detection systems. Sajjad et al. [25] presented a collaborative detection and mitigation system that targets the IIoT industrial domain. The system is designed to identify anomalous device behavior by comparing it against pre-defined policies. The evaluation of the system indicated that it effectively detected all aggressive movements made by attackers, achieving a high detection rate of 97%. Although the researchers demonstrated the benefits of using a cooperative mitigation system based on Ethereum, the study also identified some limitations. Specifically, the implementation of the system on Ethereum was found to be more scalable than on Hyperledger. However, it was also noted that Ethereum's proof of work complexity is higher compared to that of Hyperledger. This complexity could potentially impact the overall performance of the system and limit its effectiveness in certain scenarios.

Makhdoom et al. [22] introduced a collaborative intrusion detection system that utilizes blockchain technology to enable communication and information sharing among multiple intrusion detection systems. The proposed system offers a promising solution to enhance the detection of emerging cyber-attack incidents. However, the study also identified limitations associated with managing trust among system peers. The authors noted that the presence of an internal attacker could introduce false attack signatures into the network, leading to performance degradation. This issue highlights the need for additional measures to ensure the integrity of the system and mitigate the impact of potential attacks. Li et al. [20] proposed a novel approach to enhance the robustness of intrusion detection systems (IDS) by combining blockchain technology. The study showed that the integration of blockchain with IDS can lead to improved trust management and detection of malicious nodes, ultimately enhancing the overall robustness of the system. Furthermore, the results demonstrated that the proposed solution could effectively aggregate alarms and identify false entries, leading to a reduction in error rates. While the study presents a promising approach to improving IDS robustness, it is important to note that there may be additional limitations that need to be addressed. For example, the performance impact of adding blockchain to IDS systems, as well as the potential for increased complexity, could pose challenges in certain scenarios.
Khan et al. [17] conducted a study to evaluate the effectiveness of unmanned aerial vehicles (UAVs) or drones against cyber-attacks that target data transmission and storage within drones. The study utilized a case study of collaborative intrusion detection to demonstrate the feasibility and efficiency of a decentralized machine learning approach based on blockchain. The results showed that the proposed solution could improve data privacy and security for drone-based systems. However, the study also identified scalability as a limitation, particularly in cases with larger datasets. Benaddi et al. [9] highlight the importance of ensuring high levels of security for IoT devices, particularly when exchanging sensitive data securely among nodes on a network. However, the vast amount of data generated daily presents a challenge that requires effective management to guarantee security and confidentiality. To address these challenges, the authors propose using Intrusion Detection Systems (IDS) with blockchain as the main registry for secure data storage. This approach can provide enhanced security and confidentiality for IoT devices by leveraging the immutability and decentralized nature of blockchain technology. While the use of IDS with blockchain shows promise for improving IoT security, further research is needed to evaluate its effectiveness and practicality in real-world applications.

Kumar et al. [19] proposed a solution to address the requirements and challenges of Intrusion Detection Systems (IDS). To share the logs and alerts generated by IDS, they proposed implementing a secure distributed ledger that stores these alerts as transactions on the blockchain. By doing so, the alerts become unfalsifiable and accessible to all participating IDS. However, the proposed solution may face scalability issues when dealing with a large volume of data, which can impact the performance of the IDS. Alkadi et al. [3] proposed a collaborative intrusion detection system that facilitates data exchange between clouds while reducing overhead costs. The system can also be used as a decision-making tool to aid in secure data migration for cloud users and providers. The intrusion detection method uses a Bidirectional Long Short-Term Memory (BiLSTM) learning algorithm to process network sequence data, and the performance was evaluated using UNSW-NB15 and BoT-IoT datasets. However, the scalability of the proposed solution to large-scale datasets remains an open issue.

Liang et al. [21] developed an innovative intrusion detection system that integrates multi-agents, blockchain, and deep learning to enhance security. The proposed system uses the NSL-KDD database, and all communication agent activities are recorded on the blockchain to ensure protection against potential threats. The system's effectiveness was tested in various settings, including networks of varying complexity and different types of attacks, resulting in high performance. However, the scalability of the proposed system remains to be investigated, as it may encounter difficulties in handling large amounts of data in real-world environments. Kolokotronis et al. [18] proposed a solution to mitigate attacks on intrusion detection systems (IDS) using new technologies. The proposed system includes a trust chain, a blockchain where sensitive data exchanged between collaborative IDS (CIDS) nodes is stored to prevent falsification by malicious nodes. Additionally, each IDS shares trust-related information on IDS nodes to improve the system's security. A protocol that combines proof of work (PoW) and proof of stake (PoS) was also proposed. This protocol prioritizes trustworthy IDS nodes with higher computing power and stakes to generate the next block. While the proposed solution addresses some security concerns, it may face scalability issues when dealing with a larger number of nodes.
Ide [14] introduced the CollabDict blockchain protocol for collaborative anomaly detection in IoT networks. The protocol utilizes statistical generalization and voting proof mechanisms to establish consensus and ensure data confidentiality by only sharing client-aggregated statistics. CollabDict also addresses the technical challenges of validation, consensus finding, and data confidentiality through a statistical machine learning algorithm. However, the scalability of the protocol for large-scale IoT networks remains a challenge. Alexopoulos et al. [2] proposed a blockchain-based solution using CIDS to improve malicious detection. Each alert message is considered a transaction produced by an IDS node, and all collaborating nodes use consensus mechanisms to validate the alert. While the solution offers better protection, the reliability of the system remains a challenge due to the potential for false positives and false negatives.

While these studies have demonstrated promising results, they have also identified some limitations. These limitations include scalability issues when dealing with large amounts of data, potential impacts on the performance of the system, and managing trust among system peers.

Our proposed collaborative IDS based on blockchain and machine learning can provide a scalable, efficient, and trustworthy solution for detecting and preventing cyber-attacks. The use of hybrid architecture provides a distributed ledger that allows for secure and transparent sharing of data among system peers without the need for a central authority. This ensures that the system can handle large amounts of data without scalability issues as all nodes in the network can contribute to the computation of the IDS. Besides, the use of machine learning algorithms combined in a complementary way can improve the performance of the system by combining the strengths of both supervised and unsupervised learning. This allows for the IDS to detect both known and unknown attacks and can adapt to new types of attacks over time. This can lead to more accurate and efficient detection of attacks, which can help reduce the false positive and false negative rates. Lastly, managing trust among system peers can be achieved through the use of smart contracts on the blockchain. Smart contracts are self-executing contracts with the terms of the agreement between the parties being directly written into lines of code. This allows for automated trust verification and can prevent malicious actors from tampering with the data or the machine learning models used in the IDS.

**IV Proposed Approach**

A Collaborative Intrusion Detection System (CIDS) is an advanced process that enhances security by simultaneously monitoring and analyzing information from multiple networks. The need for CIDS arises from the fact that a single IDS node may not effectively detect complex and advanced attacks, as it may lack knowledge of previous attacks and other malicious events. CIDS allows all IDS nodes to communicate with each other and share information on emerging network issues, thereby immunizing and preventing attacks from worsening. However, organizations are hesitant to share their data due to privacy concerns. Nevertheless, integrating blockchain with CIDS ensures a satisfactory level of trust through consensus procedures. Blockchain secures information by guaranteeing that it has not been modified during circulation [24].
The proposed solution involves two key components (Fig. 2):

**Intrusion Detection Module**

This module is responsible for detecting and analyzing potential intrusions within a network. It captures network traffic and extracts relevant data for further analysis. It employs machine learning algorithms to identify and classify potential security threats.

**Blockchain-based Communication Module**

The communication system utilizes blockchain technology to facilitate secure and transparent information exchange among different components of the system. Blockchain ensures the integrity and immutability of data by providing a decentralized and distributed ledger. It enables IDS nodes to share information about emerging network issues, collaborate in real-time, and prevent attacks from escalating.

The Communication Blockchain module connects to the Hyperledger Fabric network, which consists of multiple IDS nodes represented as IDS Node (Peer) entities. The Hyperledger Fabric Chaincode (smart contract) provides the logic for communication and information exchange among the IDS nodes. Each IDS node acts as a peer within the Hyperledger Fabric network, interacting with other peers and accessing the shared ledger for secure and transparent communication (See Fig. 3).

**4.1 Intrusion Detection Module**

Preprocessing, the initial implementation step, involves transforming the raw data extracted from the CICIDS2017 dataset. It includes processes such as removing redundant records and normalizing the data to conform to the required learning format. The dataset spans five days, with approximately 80.3% representing normal traffic and the remaining 19.7% comprising fourteen types of attacks. Due to insufficient data points related to Heartbleed, SQL Injection, or Infiltration, these entries were removed from the dataset. Additionally, besides the provided attack labels, each record was assigned a binary label indicating benign or attack status [27].

Datasets often contain missing values, which can be attributed to recording errors or issues with feature extraction. To address this, we decided to eliminate rows that contained NaN, Null, or Inf values. Given the dataset's size, this has a minimal impact on the results.

After performing the preprocessing step on the CICIDS2017 dataset and splitting it into training, validation, and testing sets, we used an Autoencoder and a Deep Neural Network (DNN) for learning from this preprocessed dataset.

- **Autoencoder Architecture:** We employed an Autoencoder, which consists of an encoder network (E) and a decoder network (D). The encoder E compresses the input data into a lower-dimensional latent space representation, while the decoder D aims to reconstruct the original input from this
compressed representation. The Autoencoder was trained end-to-end to minimize the reconstruction error, encouraging it to learn meaningful and compact representations.

- **Autoencoder Training**: We trained the Autoencoder on the training set (X_train) to learn the optimal latent representations. The training process involves minimizing the reconstruction loss, which measures the discrepancy between the original input and its reconstruction. By backpropagating the error through the Autoencoder's layers and updating the learnable parameters, we optimized the encoder E and decoder D to capture the most important features of the input data.

- **Encoder Output**: Once the Autoencoder was trained, we utilized the encoder E to obtain the encoded representations of the preprocessed data from the training, validation, or testing sets. The encoded representations of the training set are denoted as Z_train, the validation set as Z_val, and the testing set as Z_test.

- **DNN Architecture**: We designed a Deep Neural Network (DNN) architecture for the classification task, taking the encoded representations as input. The DNN consists of multiple layers of interconnected neurons, allowing it to learn complex patterns and make predictions. The parameters of the DNN are denoted as θ_dnn.

- **DNN Training**: We trained the DNN on the encoded representations Z_train along with their corresponding labels y_train. The training process involves feeding the encoded representations through the DNN layers and optimizing the DNN parameters θ_dnn to minimize a suitable loss function. This optimization process is typically performed using gradient descent or its variants, with backpropagation used to compute the gradients.

- **Model Evaluation**: After training the DNN, we evaluated its performance on the validation set (Z_val) to assess its generalization ability. This evaluation helps us tune hyperparameters, such as the number of layers, neurons, and activation functions, to improve the model's performance. We iterated this step multiple times to fine-tune the DNN architecture.

- **Model Testing**: Finally, we assessed the DNN's performance on the testing set (Z_test) to obtain an unbiased estimate of its accuracy and effectiveness. This evaluation helped us determine how well the model generalizes to unseen data and provides insights into its real-world performance.

By utilizing the Autoencoder to learn compact representations and training the DNN on these representations, we aim to build a model that can effectively classify and detect intrusions in network traffic. The Autoencoder helps in capturing important features and reducing noise in the data, while the DNN leverages these representations to make accurate predictions.

### 4.2 Blockchain-based Communication Module

To use Blockchain Communication based on Hyperledger Fabric for communication between nodes in the context of intrusion detection, we followed the following steps (Fig. 4):

- **Install Hyperledger Fabric**: We installed Hyperledger Fabric on each node involved in the communication: IDS nodes and monitoring nodes. This step typically involves downloading the
necessary binaries and setting up the required dependencies.

- **Set up the Network:** We defined the network by creating a configuration file that specifies the number of nodes and their roles. As we have three IDS nodes and two monitoring nodes, each IDS node represents a different security component, and the monitoring nodes are responsible for collecting and analyzing intrusion data: `ids_nodes.append(node_id), monitoring_nodes.append(node_id)`

- **Define Smart Contracts:** Smart contracts define the communication protocols and data structures for exchanging intrusion detection information. For instance, the smart contract may specify functions for reporting detected intrusions or requesting additional data from other nodes:
  ```python
context.setContract(intrusionContract)
```

- **Establish Communication Channels:** We created communication channels within the Hyperledger Fabric network to facilitate selective sharing of data between IDS nodes and monitoring nodes. We created a channel named "IntrusionDetectionChannel" and add the three IDS nodes and two monitoring nodes to this channel:
  ```python
  contract.createChannel ("IntrusionDetectionChannel", [ids1, ids2, ids3]. Concat ([ monitoring 1, monitoring 2]) )
  ```

- **Data Exchange and Validation:** We implemented the logic within the smart contracts to enable data exchange and validation. The IDS node can report a detected intrusion by invoking a function in the smart contract and passing relevant details such as the source IP, destination IP, and intrusion type. The smart contract can then validate the data against predefined rules, ensuring the accuracy and integrity of the reported intrusion:
  ```python
this.validateIntrusionData(sourceIP, destinationIP, intrusionType)
```

- **Transaction Execution:** We executed transactions between the IDS nodes and monitoring nodes using the defined communication channels. The monitoring node can request additional information about a reported intrusion by invoking a function in the smart contract and specifying the intrusion ID. The IDS node that detected the intrusion can respond to the request by providing more details through the smart contract:
  ```python
  intrusionContract.reportIntrusion(context, sourceIP, destinationIP, intrusionType)
  ```

- **Data Persistence and Consensus:** Hyperledger Fabric ensures data persistence and consensus by utilizing a distributed ledger. Each valid transaction executed between the nodes is recorded in the ledger, creating an immutable and transparent history. Consensus among the participating nodes is achieved using a consensus algorithm, the one used in our contribution is Practical Byzantine Fault Tolerance (PBFT), ensuring that all nodes agree on the validity of transactions:
  ```javascript
  console.log('Stored Intrusion Data:', storedData.toString())
  ```

By utilizing Blockchain Communication based on Hyperledger Fabric, nodes involved in intrusion detection can securely exchange information, maintain data integrity, and establish a trust framework for collaboration and analysis. The decentralized and transparent nature of the blockchain technology enhances the overall effectiveness and reliability of intrusion detection systems.

**V. Experimentation**
We set up three IDS nodes distributed across the network. Each IDS node run the Autoencoder and DNN based detection module and connected to the blockchain network as a participant. We also set up dedicated monitoring nodes that will receive and analyze the intrusion data from the IDS nodes. These nodes will connect to the blockchain network and access the data recorded by the IDS nodes. The IDS nodes will continuously monitor the network traffic and detect potential intrusions. Transactions between the IDS nodes and monitoring nodes through the blockchain network involve reporting detected intrusions.

We define performance metrics to evaluate the effectiveness of the IDS system, such as accuracy, precision, recall and F1_Score. And we compare the performance of the IDS system with and without the integration of Blockchain Communication in order to assess the impact of blockchain on the detection accuracy (see Table 1).

<table>
<thead>
<tr>
<th>Proposed IDS Performance</th>
<th>accuracy</th>
<th>precision</th>
<th>recall</th>
<th>f1-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNN based IDS [5]</td>
<td>0.91</td>
<td>0.92</td>
<td>0.91</td>
<td>0.91</td>
</tr>
<tr>
<td>Autoencoder based ids [28]</td>
<td>0.93</td>
<td>0.92</td>
<td>0.92</td>
<td>0.92</td>
</tr>
<tr>
<td>Proposed IDS without Blockchain</td>
<td>0.94</td>
<td>0.95</td>
<td>0.92</td>
<td>0.94</td>
</tr>
<tr>
<td>Proposed IDS with blockchain</td>
<td>0.97</td>
<td>0.97</td>
<td>0.94</td>
<td>0.96</td>
</tr>
</tbody>
</table>

An IDS based on autoencoder and DNN outperforms a DNN-only IDS due to its ability to learn meaningful representations of normal behavior, detect anomalies, reduce false positives, generalize to unseen data, and handle high-dimensional traffic efficiently. The combination of unsupervised feature learning with supervised classification enables the IDS to accurately identify network intrusions and adapt to new attack patterns. By leveraging the strengths of both models, the IDS achieves improved performance in intrusion detection and offers a more effective defense against malicious activities.

An IDS based on autoencoder and DNN outperforms also an IDS solely based on an autoencoder due to several reasons. First, the combination of the autoencoder and DNN allows for enhanced feature representation. While autoencoders can capture certain patterns and anomalies, they may have limitations in capturing complex and abstract features. By incorporating a DNN into the IDS, the DNN can further extract and learn high-level representations from the encoded data. This enhances the ability of the IDS to detect and classify different types of intrusions accurately.

Second, the DNN component of the IDS brings the advantage of supervised learning. It can be trained using labeled intrusion data, enabling it to learn from both normal and malicious network patterns. This helps the system to distinguish between benign and malicious activities more effectively, leading to improved accuracy in intrusion detection.
Additionally, an IDS with blockchain technology outperforms an IDS without blockchain due to its enhanced security, immutability, and decentralized nature. By leveraging blockchain, the IDS can provide a tamper-proof and transparent record of network activities, ensuring the integrity and authenticity of the collected data. The decentralized nature of blockchain eliminates the reliance on a central authority, making it more resilient against single points of failure and reducing the risk of data manipulation or tampering. Additionally, the use of smart contracts in blockchain-based IDS allows for automated and reliable enforcement of security policies and consensus mechanisms among network nodes. This combination of enhanced security, immutability, decentralization, and automated policy enforcement significantly strengthens the effectiveness and reliability of the IDS, making it a more robust solution for intrusion detection.

VI Conclusion

In conclusion, the integration of blockchain technology and machine learning-based analysis methods has shown great potential in enhancing the performance and reliability of intrusion detection systems (IDS). Blockchain provides a decentralized and secure platform for data exchange, ensuring the integrity and transparency of the information shared among IDS components. By leveraging machine learning techniques such as autoencoders and deep neural networks, IDSs can benefit from improved detection accuracy and the ability to analyze complex and large-scale datasets. The collaborative IDS framework, empowered by blockchain, enables nodes to work together, share information, and establish collective trust, leading to more effective detection and mitigation of malicious activities. As the field of cybersecurity continues to evolve, the combination of blockchain and machine learning-based IDS approaches holds promise for enhancing network security in open, distributed environments.

There are several promising perspectives for the integration of blockchain technology and machine learning-based analysis methods in intrusion detection systems (IDS). Firstly, the application of federated learning approaches, where models are trained collaboratively across multiple IDS nodes while preserving data privacy, can provide a valuable avenue for improving the overall detection capabilities of the system. Additionally, integrating other emerging technologies such as edge computing and Internet of Things (IoT) devices into the IDS ecosystem can expand its scope and enhance its ability to detect and mitigate attacks at the network edge. By leveraging these perspectives, future IDS solutions can be more resilient, adaptive, and capable of addressing the evolving challenges in securing distributed networks.

Declarations

Ethical Approval

The authors declare that this manuscript is original, has not been published before and is not currently being considered for publication elsewhere.

Competing Interests
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

**Funding**

The authors declare that they have no funding.

**Availability of Data and Materials**

The datasets generated during and/or analyzed during the current study are available from the authors.

**References**


Figures

Let $N$ be the total number of nodes in the network, and let $f$ be the maximum number of faulty nodes that can be tolerated. The PBFT algorithm requires a minimum of $3f + 1$ nodes in the network to work effectively.

1. The client sends a transaction request to the primary node.
2. The primary node broadcasts the request to all the backup nodes.
3. The backup nodes validate the request and send back a response to the primary node.
4. The primary node collects the responses from the backup nodes and broadcasts a pre-commit message to all the nodes in the network.
5. If more than $2f$ nodes in the network receive the pre-commit message and validate it, they send back a pre-commit message to the primary node.
6. Once the primary node receives pre-commit messages from more than $2f$ nodes, it broadcasts a commit message to all the nodes in the network.
7. If more than $2f$ nodes in the network receive the commit message and validate it, they commit the transaction to the blockchain.

Figure 1

The PBFT algorithm
Figure 2

Blockchain for Collaborative Intrusion Detection Systems
Figure 3

Hyperledger Fabric Blockchain Communication
Figure 4

Hyperledger Fabric based Communication

Figure 5

Models Performances Comparison
Figure 6

Proposed Approach Evaluation