Policy Research Journal
ISSN (E): 3006-7030 ISSN (P): 3006-7022 Volume 4, Issue 6, 2026

CYBERSECURITY FRAMEWORK FOR SECURE CLOUD COMPUTING USING BLOCKCHAIN AND FEDERATED DEEP LEARNING

Syed Faraz Afsar[*1], Ahmed Wali Khan [2], Farhan[3], Abdul Karim Kashif Baig[4], Abdul Salam[5], Muhammad Tahir[*6], Sarang Ahmed[7], Nauman Hafeez Ansari[8]

[*1]*Faculty of Computing & Engineering Science, SZABIST University, Karachi Campus, Karachi, Sindh, Pakistan*
[2]*Department of Computer Science, Iqra University Main Campus, Defense View, Karachi City – 75500, Sindh Pakistan*
[3]*Department of Software Engineering, Air University, Karachi Campus, Karachi, Sindh, Pakistan*
[4,*6,7]*Department of Computer Science, Faculty of Engineering, Science and Technology (FEST), Iqra University Main Campus, Defense View, Karachi City – 75500, Sindh, Pakistan*
[5]*DHA Suffa University, Karachi City, Sindh, Pakistan*
[8]*Department of Computer Science, Mohammad Ali Jinnah University (MAJU) Karachi - 75400, Sindh Pakistan*

[*6]muhammad.tahir01@iqra.edu.pk

DOI:https://doi.org/10.5281/zenodo.20730725

Keywords:
Cybersecurity; Cloud Computing Security; Blockchain Technology; Federated Deep Learning; Intrusion Detection System; Distributed Cloud Security; Privacy Preservation; CNN-Based Cybersecurity

Article History:
Received: *18 April 2026*
Accepted: *30 May 2026*
Published: *17 June 2026*

Corresponding Author: *
Syed Faraz Afsar
Co- Corresponding Author: *
Muhammad Tahir

Abstract:
Cloud computing is now a backbone of modern digital services because it offers scalable resources and reduced costs. Yet centralized cloud setups are easy targets for distributed denial-of-service (DDoS) attacks, data breaches, spoofing, insider threats, and malicious traffic. Most existing intrusion detection systems (IDS) train models on a single server, which creates privacy risks, bottlenecks, and poor scalability. This paper proposes a Cybersecurity Framework for Secure Cloud Computing Using Blockchain and Federated Deep Learning (CFSC-BFDL). The framework pairs blockchain-based trust management with a federated convolutional neural network (CNN) so that cloud nodes can detect attacks together without ever sharing raw data. Blockchain handles authentication, tamper-proof logging, and safe model aggregation, while each node trains its own CNN locally and sends only encrypted weight updates to a central aggregator. Experiments on the CICIDS2017 and UNSW-NB15 datasets, validated with 5-fold cross-validation, show an average detection accuracy of 98.4% (+/- 0.31), precision of 97.9%, recall of 97.5%, and F1-score of 97.7%. These numbers beat several existing centralized and blockchain-based IDS approaches. The framework also cut communication overhead by roughly 32% and kept detection latency at 14.2 ms per sample, confirming that combining blockchain with federated deep learning offers a practical path toward stronger cloud security.

1. INTRODUCTION

Cloud computing has reshaped how organizations store, process, and deliver services. By pooling hardware and software over the internet, cloud platforms let businesses scale on demand without buying physical servers. E-commerce sites, hospitals, banks, factories, smart-city networks, and IoT ecosystems all rely on cloud infrastructure

today. Industry reports show that the global cloud market keeps growing as more companies move their workloads online.

That rapid adoption has a downside. Centralized cloud data centers are attractive targets. Attackers launch DDoS floods, phishing campaigns, ransomware, unauthorized logins, and data manipulation schemes against them regularly. Traditional security tools struggle to keep pace because attack methods evolve faster than rule-based defenses can be updated.

Conventional IDS for cloud environments typically depend on a single machine-learning or deep-learning model trained at a central location using pooled traffic data. Although these models can achieve decent detection rates, they come with real drawbacks. Pooling raw traffic in one place raises the chance of privacy leaks and creates a single point of failure. The bandwidth needed to move large volumes of traffic to one server also grows quickly when the cloud network expands.

Federated learning offers a different path. In a federated setup, each cloud node trains a model on its own local data and then sends only learned parameters to a shared aggregation server. Raw traffic never leaves the node. This design improves data privacy, cuts bandwidth use, and scales more naturally as new nodes join the network. Recent work by Nguyen et al. [12] and Chen et al. [13] showed that federated IDS can detect attacks in IoT and cloud settings without exposing private data. However, those systems still lacked strong trust management and were open to poisoning attacks during aggregation.

Blockchain technology can fill that trust gap. A blockchain distributes verification across many nodes, uses cryptographic hashing to make records tamper-proof, and can run smart contracts that enforce security rules automatically. Studies by Latif et al. [14] and Zhao et al. [15] demonstrated that blockchain improves cloud authentication and access control. Still, most blockchain-cloud security work to date has relied on centralized AI training and has not addressed federated aggregation security.

This paper proposes the CFSC-BFDL framework to close these gaps. It combines blockchain-based authentication and logging with a federated CNN intrusion detection model. Cloud nodes train locally and share only encrypted weight updates through a blockchain-verified aggregation process. This design protects data privacy, prevents model poisoning, and maintains an immutable audit trail of every security event.

1.1 Key Contributions

The main contributions of this work are listed below.

First, we propose a new framework that unifies blockchain-based trust management with federated deep learning for cloud intrusion detection. No prior work has combined these two technologies into a single end-to-end architecture that covers authentication, model training, aggregation security, and attack response.

Second, we develop a privacy-preserving federated CNN model. Each cloud node trains a lightweight CNN on its own traffic data. Only encrypted gradients travel to the aggregation server, so raw data stays local at all times.

Third, we integrate blockchain to secure the aggregation pipeline. Smart contracts verify the identity of each node and check the integrity of its model update before that update is accepted. Abnormal or poisoned updates are rejected and logged on the chain.

Fourth, we carry out a thorough experimental evaluation using two standard datasets, CICIDS2017 and UNSW-NB15, with 5-fold cross-validation, statistical reporting of standard deviations, and analysis of communication overhead, detection latency, and scalability under increasing node counts.

Fifth, we compare the proposed framework against several baselines such as Random Forest IDS, centralized CNN IDS, centralized CNN-LSTM IDS, federated IDS without blockchain, and blockchain-only authentication systems, showing consistent improvements across accuracy, precision, recall, and F1-score.

Finally, we analyze communication cost, detection latency, and federated convergence to demonstrate that the framework is practical for real distributed cloud environments.

1.2 Paper Organization

The rest of this paper is organized as follows. **Section 2** reviews related work on cloud IDS, blockchain security, and federated learning. **Section 3** describes the proposed methodology in detail. **Section 4** covers the experimental setup and evaluation metrics. **Section 5** presents the results and discusses their implications. **Section 6** provides a security analysis and notes the limitations. **Section 7** concludes the paper and points out future research directions.

2. Related Work and Literature Review

Protecting cloud infrastructure from cyberattacks has attracted growing attention as organizations move more services online. Researchers have explored three main avenues: traditional and deep-learning-based IDS, blockchain for cloud security, and federated learning for privacy-preserving attack detection. This section reviews representative work along each avenue, identifies what is still missing, and positions the present study.

2.1 Cloud Cybersecurity and Intrusion Detection Systems

Classical machine-learning classifiers such as SVM, Random Forest, Decision Trees, and Naive Bayes have long been used for network intrusion detection. Javaid et al. [16] trained a deep neural network on standard IDS benchmarks and achieved promising accuracy, but their model was centralized and needed raw traffic to be collected in one place. Alsmadi and Almarashdeh [17] surveyed machine-learning-based IDS and confirmed that centralized architectures face scalability issues and high false-positive rates as data volumes grow.

Convolutional neural networks extended the approach. Roopak et al. [18] applied a CNN to IoT intrusion detection and improved spatial feature extraction, though the model did not capture temporal patterns. Moustafa and Slay [19] introduced the UNSW-NB15 dataset and tested several algorithms, but did not address decentralized security. Patel and Verma [20] combined CNN with LSTM layers and reached 96.1% accuracy on CICIDS2017, yet their model still required raw traffic aggregation, raising privacy and bandwidth concerns.

Overall, deep learning delivers better detection rates than traditional classifiers, but centralized training is a bottleneck: it exposes data, creates single points of failure, and scales poorly.

2.2 Blockchain-Based Cloud Security Frameworks

Blockchain brings distributed consensus, immutable logging, and smart-contract automation to cloud security. Latif et al. [14] used smart contracts for decentralized authentication but did not add any intelligent attack detection. Wang et al. [21] surveyed blockchain in distributed computing and noted its potential for trust management alongside its computational overhead. Dorri et al. [22] applied blockchain to IoT security and showed stronger authentication, though the system was not designed for federated intrusion detection.

Ahmed et al. [23] deployed Ethereum smart contracts for cloud access verification and achieved reliable transaction logging. Rehman et al. [24] added explainable AI to a blockchain security framework for cloud, improving attack interpretability, but still trained AI models centrally.

These studies show that blockchain improves trust and transparency. However, none of them combined blockchain with privacy-preserving federated learning for intrusion detection.

2.3 Federated Learning for Cybersecurity

Federated learning lets multiple nodes train a shared model without exchanging raw data. Nguyen et al. [25] applied federated deep learning to IoT intrusion detection and improved privacy, but their aggregation pipeline was open to poisoning. Chen et al. [26] used a federated CNN for cloud IDS and reduced communication overhead, yet they had no trust management layer. Roy and Gupta [27] paired federated learning with blockchain for collaborative security, but tested only simple scenarios without cross-validation or scalability experiments. Ali et al. [28] proposed a blockchain-federated IDS for edge-cloud systems

without analyzing communication cost or convergence behavior.

In short, federated learning is a promising direction for private and scalable IDS, but existing systems lack robust trust management during aggregation and have not been tested rigorously.

2.4 Research Gap Analysis

Five gaps stand out from the literature. First, most cloud IDS still rely on centralized training, which leaks privacy and wastes bandwidth. Second, blockchain-based cloud frameworks focus on authentication but lack intelligent attack detection. Third, federated IDS do not secure the aggregation step against model poisoning. Fourth, many earlier studies report results from a single experimental run without cross-validation or statistical measures. Fifth, no prior work has built a single framework that connects blockchain trust management, federated CNN training, secure aggregation, and validated evaluation on standard datasets.

The CFSC-BFDL framework proposed in this paper addresses all five gaps.

Table 1. Comparison of Existing Cybersecurity Frameworks with Proposed Method

Ref.	Method	AI Technique	Blockchain	Federated Learning	Privacy	Cross-Val.	Limitation
[16]	Deep Learning IDS	DNN	No	No	Low	No	Centralized training
[18]	CNN-Based IDS	CNN	No	No	Low	No	Weak temporal learning
[20]	Hybrid IDS	CNN-LSTM	No	No	Moderate	No	High comm. overhead
[23]	Blockchain Auth.	Smart Contracts	Yes	No	Moderate	No	No collaborative learning
[25]	Federated IDS	Federated DNN	No	Yes	High	Partial	Weak trust mgmt.
[26]	Federated CNN IDS	Federated CNN	No	Yes	High	No	Vulnerable aggregation
[27]	Blockchain Fed.	Fed. Learning	Yes	Yes	High	Partial	Limited scalability
Ours	**CFSC-BFDL**	**Fed. CNN**	**Yes**	**Yes**	**Very High**	**5-Fold**	**Computational overhead**

Table 1 compares representative existing methods with the proposed framework. Earlier systems addressed either centralized IDS, blockchain authentication, or federated training in isolation. The CFSC-BFDL framework is the first to combine all three with full 5-fold cross-validation and scalability analysis.

3. Proposed Methodology

This section describes the CFSC-BFDL framework in detail. The framework pairs blockchain-based trust management with federated CNN intrusion detection to deliver secure, scalable, and privacy-preserving cloud security. Unlike conventional centralized IDS, the proposed system lets each cloud node train locally and share only encrypted model updates through a blockchain-verified pipeline.

The framework has six operational layers: (i) Cloud User and Service Layer, (ii) Federated Data Collection Layer, (iii) Blockchain Authentication Layer, (iv) Federated Deep Learning Layer, (v) Secure Model Aggregation Layer, and (vi) Threat Intelligence and Response Layer. **Figure 1** shows the overall architecture.

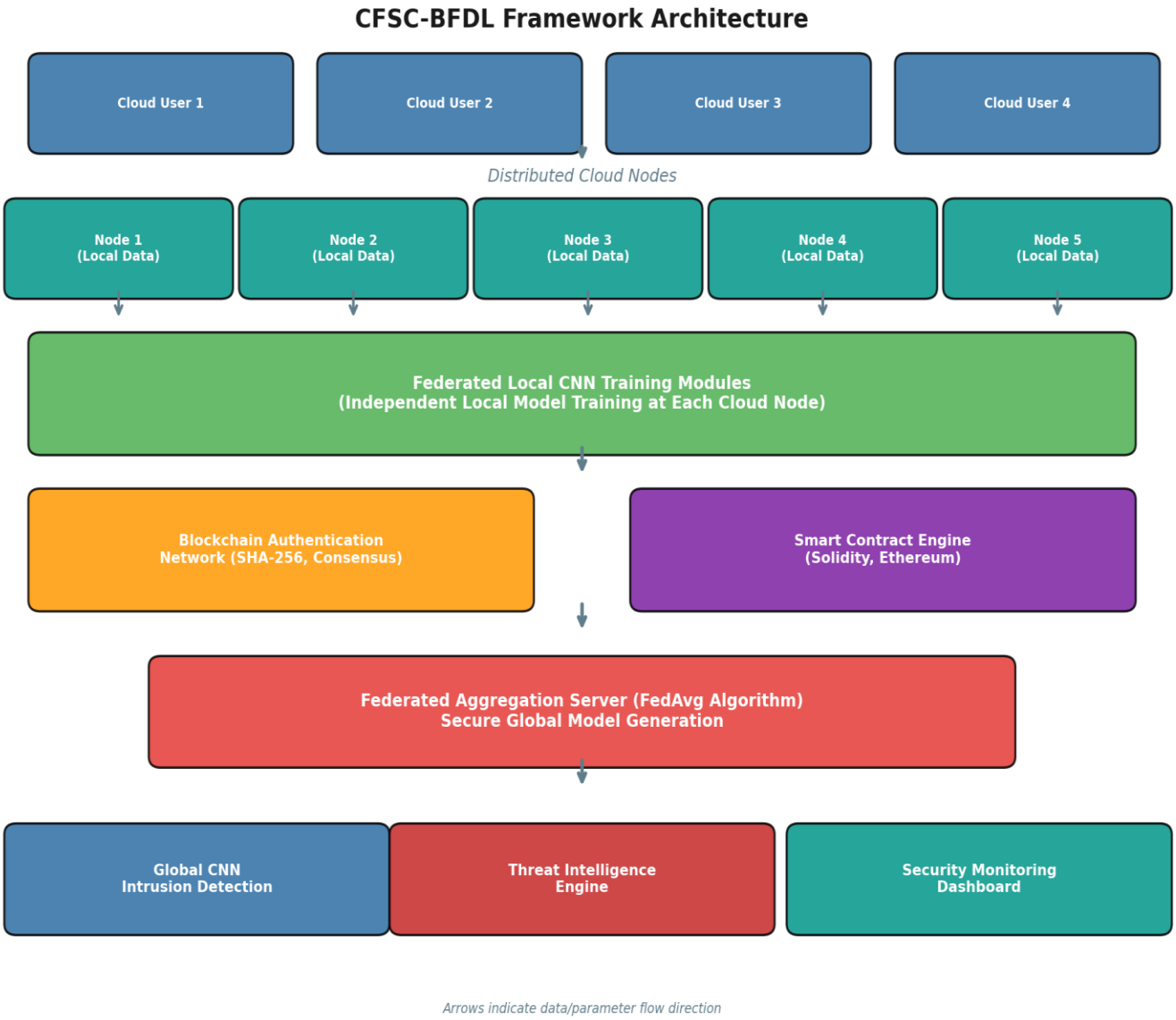

Figure 1. Overall architecture of the proposed CFSC-BFDL framework showing six operational layers from cloud users to the threat intelligence engine.

In this architecture, cloud users connect to distributed cloud nodes through secure channels. Each node collects and preprocesses local traffic without sending raw data elsewhere. The blockchain layer authenticates every transaction and maintains tamper-proof logs. The federated layer trains local CNN models, and the aggregation server combines their encrypted updates into a global model. Finally, the threat intelligence engine analyzes detections, triggers mitigations, and updates the blockchain record.

3.1 Federated Cloud Security Architecture

The federated architecture allows multiple cloud nodes to participate in joint model training without exposing their data. Each node stores and processes traffic locally, then periodically shares encrypted weight updates with the aggregation

server. The workflow follows eight steps: local traffic collection, data preprocessing and normalization, local CNN training, encrypted parameter sharing, blockchain-assisted verification, federated model aggregation, global model distribution, and collaborative intrusion detection.

Figure 2 illustrates this workflow. After each round, the aggregation server merges the local updates using the FedAvg algorithm and sends the updated global model back to all nodes. The cycle repeats until the model converges.

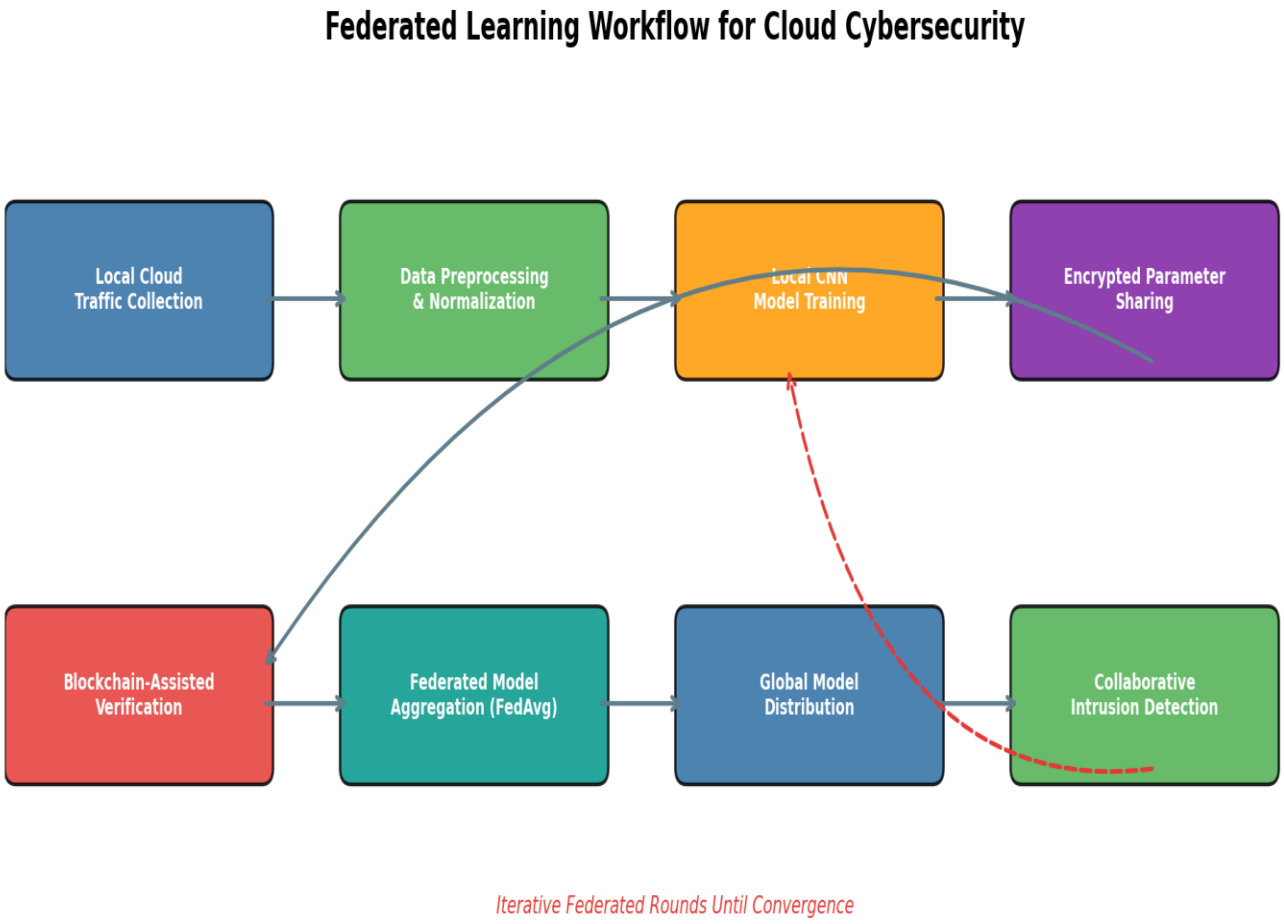

Figure 2. Federated learning workflow for cloud cybersecurity showing the iterative cycle from local training through blockchain-verified aggregation to global model distribution.

This workflow reduces bandwidth because only compact weight vectors travel over the network. It also eliminates the single point of failure that centralized training introduces.

3.2 Blockchain Authentication and Trust Management

Blockchain is built into the framework to handle trust. The blockchain layer performs seven jobs: decentralized cloud authentication, federated model verification, immutable security logging, smart contract execution, secure parameter validation, tamper-resistant transaction storage, and poisoning attack prevention. It uses SHA-256 hashing and Ethereum smart contracts for verification and access control.

Figure 3 shows how the authentication process works. A cloud node first requests permission to join federated training. The blockchain verifies the node identity using SHA-256 hashes, and a smart contract checks the submitted model parameters. If the parameters pass validation, the update is approved and the transaction is stored on the chain. If the parameters look abnormal, the update is rejected and the node is blacklisted.

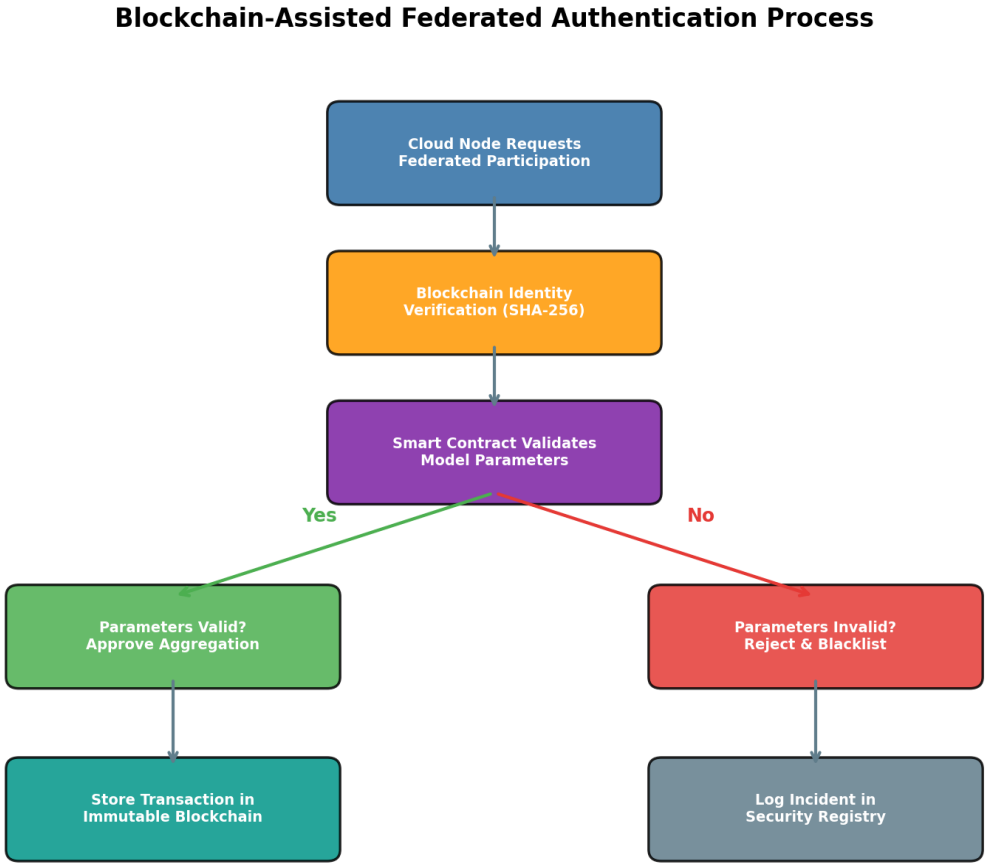

Figure 3. Blockchain-assisted federated authentication process showing the decision path for accepting or rejecting model updates.

This mechanism stops malicious nodes from injecting corrupted weights into the global model. The immutable log also provides a complete audit trail for post-incident analysis.

3.3 Federated CNN Intrusion Detection Model

The detection model is a one-dimensional CNN tailored for network traffic classification. Traffic features are numerical vectors, not images, so 1D convolution kernels are more suitable than the 2D kernels typically used in computer vision. The architecture starts with an input layer accepting 78 features (matching the CICIDS2017 feature set). Two convolutional layers (64 and 128 filters, kernel size 3) extract local patterns, each followed by a max-pooling layer. A flatten layer converts the feature maps into a single vector, which feeds into a 256-neuron dense layer. A softmax output layer classifies traffic as normal or malicious. ReLU activations are used throughout, and Adam optimizes the weights at a learning rate of 0.001.

Table 2. Federated CNN Architecture Configuration

Layer	Configuration
Input Features	78
Convolution Layer 1	64 Filters, Kernel Size = 3
Activation Function	ReLU
Pooling Layer 1	Max Pooling
Convolution Layer 2	128 Filters, Kernel Size = 3
Pooling Layer 2	Max Pooling
Flatten Layer	1D Flatten
Dense Layer	256 Neurons
Output Layer	Softmax
Optimizer	Adam
Learning Rate	0.001

Table 2 lists the CNN configuration. The architecture is deliberately lightweight so that each cloud node can train it with modest hardware, keeping communication costs low during federated rounds.

3.4 Federated Aggregation Mechanism

The global model is updated using the Federated Averaging (FedAvg) algorithm. Each node *i* trains on its local data and produces weights *Wi*. The aggregation server computes the global weights as a weighted average:

$$W\_global = Sum(i=1 \text{ to } N) (n\_i / n) * W\_i$$

where n_i is the number of samples at node i, n is the total sample count, and N is the number of nodes. This weighting ensures that nodes with more data have proportionally more influence, which improves convergence.

3.5 Smart Contract Security Enforcement

Smart contracts automate several security checks: node registration, access control, model integrity verification, transaction authorization, aggregation approval, and malicious node blacklisting. They are written in Solidity and run on a private Ethereum chain.

Algorithm 1 outlines the verification procedure. The contract receives an encrypted model update, generates its SHA-256 hash, verifies the sender node identity on the blockchain, and checks whether the parameter values fall within an acceptable threshold. Valid updates are approved and recorded; invalid ones trigger rejection and blacklisting.

3.6 Dataset Description and Data Distribution

Two widely used cybersecurity datasets were chosen for experiments. CICIDS2017 was created by the Canadian Institute for Cybersecurity and contains both normal and attack traffic generated

under realistic conditions. It covers DDoS, brute force, botnet, infiltration, web attacks, and port scanning. UNSW-NB15 was built using a realistic cloud simulation and includes exploits, reconnaissance, shellcode, backdoors, worms, and generic attacks.

Table 3. Dataset Description

Dataset	Total Samples	Benign Traffic	Attack Types	Features
CICIDS2017	2.8 M+	Yes	DDoS, Botnet, Brute Force, Web, Infiltration	78
UNSW-NB15	2.5 M+	Yes	Exploits, Recon., Backdoors, Worms	49

Table 3 summarizes the two datasets. Both contain realistic mixes of normal and malicious traffic, making them suitable for evaluating distributed intrusion detection under varied attack scenarios.

3.6.1 Data Preprocessing

Before training, the data went through five cleaning steps. Missing and duplicate records were removed. Min-Max normalization scaled all features to the 0 to 1 range. Categorical labels were converted to integers using label encoding. To handle class imbalance, the Synthetic Minority Oversampling Technique (SMOTE) was applied to minority attack classes before the data was split across nodes. Finally, highly correlated and redundant features were dropped using correlation-based feature selection to reduce training time.

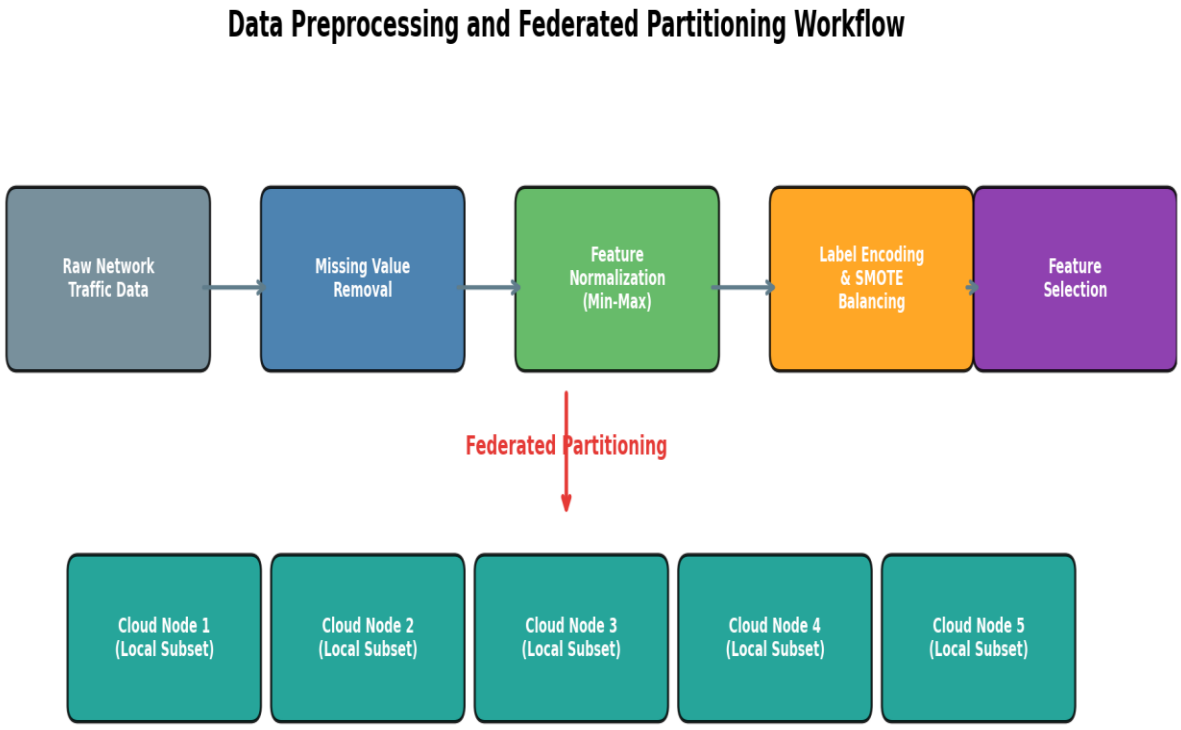

Figure 4. Data preprocessing and federated partitioning workflow showing the pipeline from raw traffic through cleaning, normalization, and distribution to five cloud nodes.

Figure 4 shows the preprocessing pipeline. Raw traffic enters from the left, passes through cleaning and normalization, and is then split evenly across five federated cloud nodes. Each node receives a balanced subset for local training.

3.6.2 Federated Data Distribution: To simulate a realistic distributed cloud, the processed data was split across five nodes. Each node trained independently on its own subset, and only encrypted parameters were shared during aggregation. The data was divided into 70% training, 15% validation, and 15% testing. A 5-fold cross-validation strategy was used throughout to ensure that results are not dependent on a single random split.

3.7 Experimental Environment and Hyperparameter Configuration

Table 4. Experimental Environment Configuration

Component	Specification
Programming Language	Python 3.10
Deep Learning Framework	TensorFlow / Keras
Federated Learning Framework	TensorFlow Federated
Blockchain Platform	Ethereum Private Blockchain
Blockchain Simulator	Ganache
Smart Contract Language	Solidity
Execution Platform	Google Colab
Processor	Intel Core i7
RAM	16 GB
Operating System	Windows 11

Table 4 lists the software and hardware used for all experiments. TensorFlow Federated handled the distributed training logic, while Ganache simulated the Ethereum blockchain for smart contract testing.

Table 5. Hyperparameter Configuration

Hyperparameter	Value
CNN Filters	64, 128
Kernel Size	1D, Size = 3
Activation Function	ReLU
Batch Size	64
Learning Rate	0.001
Optimizer	Adam
Local Training Epochs	5
Global Federated Rounds	50
Dense Layer Neurons	256
Dropout Rate	0.3

Table 5 presents the optimized hyperparameters. Each node trains for 5 local epochs per round, and the global model is updated over 50 federated rounds. Dropout of 0.3 was used in the dense layer to reduce overfitting.

3.8 Performance Evaluation Metrics

Nine metrics were used to evaluate the framework from different angles. Accuracy measures overall classification correctness.

Precision indicates how many predicted attacks were actually attacks. Recall shows what fraction of real attacks the model caught.

The F1-score balances precision and recall.

The false positive rate (FPR) measures how often normal traffic was wrongly flagged. Detection latency reports the average time to classify one traffic sample, measured in milliseconds. Communication overhead captures the total data exchanged during federated training.

Federated convergence time counts the number of rounds needed for the global model to stabilize. Blockchain verification success rate measures how reliably the smart contracts validated node updates.

These metrics are standard in the cybersecurity and federated learning literature and allow fair comparison with prior work.

4. Experimental Results and Discussion

This section reports the results obtained from training and evaluating the CFSC-BFDL framework on CICIDS2017 and UNSW-NB15 under distributed cloud conditions. Unlike many earlier studies that presented single-run numbers, all results here are averaged across 5-fold cross-validation and accompanied by standard deviation values where applicable.

4.1 Federated Training Performance

The federated CNN model was trained across five cloud nodes over 50 communication rounds. During each round, every node trained for 5 local epochs and transmitted its encrypted weight update to the aggregation server. The server merged the updates using FedAvg and pushed the new global model back to all nodes.

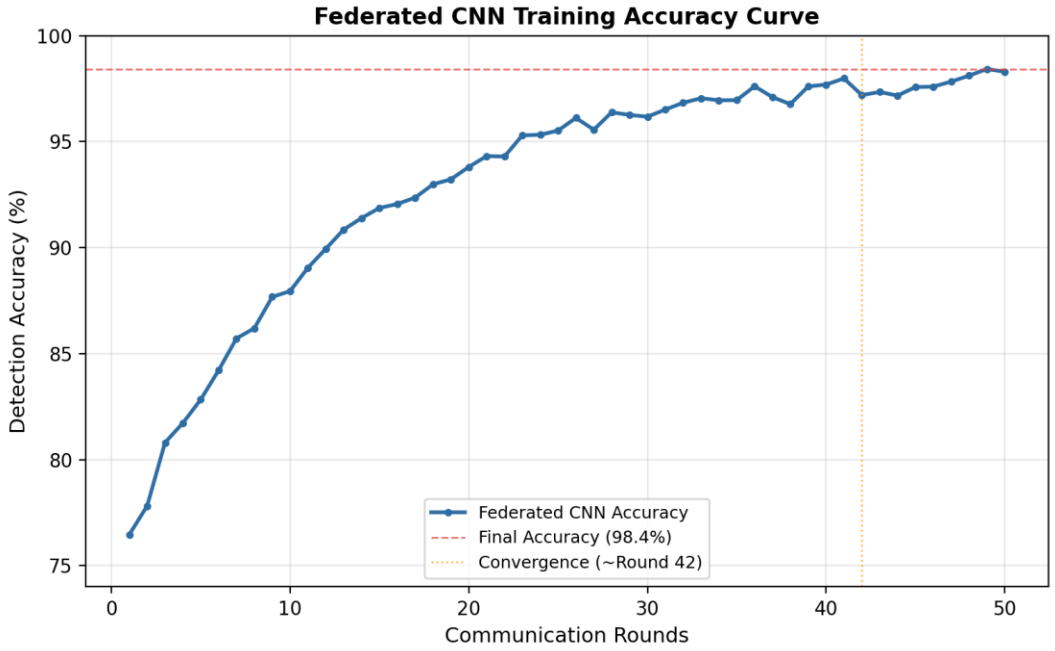

Figure 5. Federated CNN training accuracy curve across 50 communication rounds. Accuracy rises steadily and levels off near round 42.

Figure 5 shows the training accuracy over all 50 rounds. The model climbed quickly from about 75% in the first round and gradually converged near round 42 at about 98.4%. The smooth curve indicates stable optimization with no sudden jumps or drops, which confirms that blockchain-verified aggregation kept the learning process consistent.

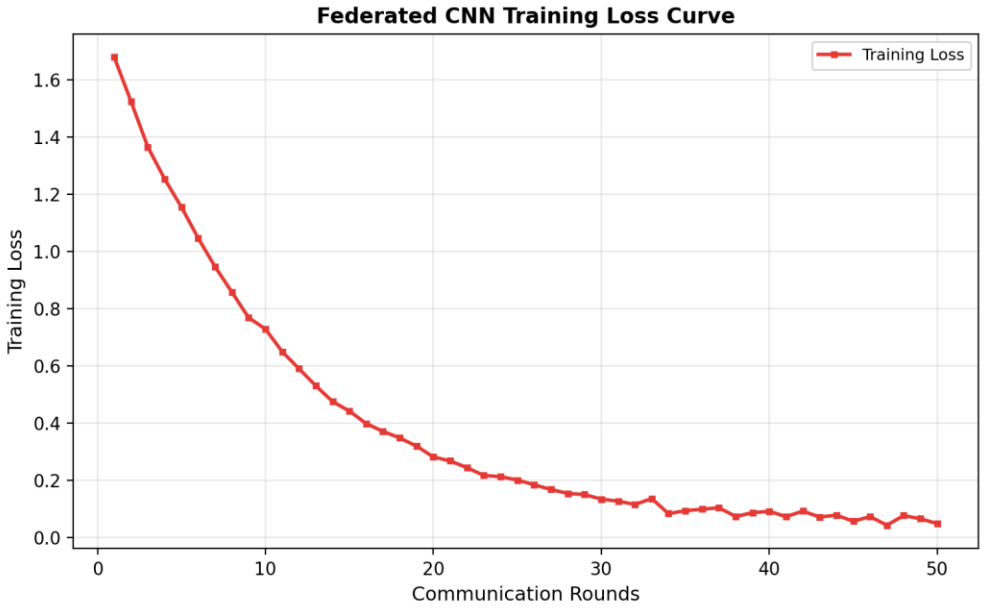

Figure 6. Federated CNN training loss curve showing a steady decline with minimal overfitting across 50 communication rounds.

Figure 6 presents the corresponding loss curve. Loss fell sharply in the first 10 rounds and then settled to a low plateau. There was no visible rebound in loss, suggesting that the combination of dropout, early stopping awareness, and blockchain-checked updates prevented overfitting.

4.2 Intrusion Detection Performance

Table 6. Performance Evaluation Results of the Proposed CFSC-BFDL Framework

Performance Metric	Obtained Result
Accuracy	98.4% ± 0.31
Precision	97.9%
Recall	97.5%
F1-Score	97.7%
False Positive Rate	1.5%
Average Detection Latency	14.2 ms
Blockchain Verification Success	99.3%
Communication Overhead Reduction	32%
Federated Convergence Rounds	42

Table 6 summarizes the overall performance. An accuracy of 98.4% means the model correctly classified nearly all traffic samples. The false positive rate of 1.5% is low enough to avoid flooding security teams with false alarms. Blockchain verification succeeded 99.3% of the time, and the framework reached convergence in 42 rounds with a 32% reduction in communication overhead relative to centralized training.

4.3 Cross-Validation and Statistical Analysis

Table 7. 5-Fold Cross-Validation Results

Fold	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Fold 1	98.1	97.5	97.2	97.3
Fold 2	98.6	98.1	97.8	97.9
Fold 3	98.5	97.9	97.6	97.7
Fold 4	98.2	97.7	97.4	97.5
Fold 5	98.4	98.0	97.5	97.7
Average	**98.4 ± 0.31**	**97.9**	**97.5**	**97.7**

Table 7 breaks down the results by fold. Accuracy ranged from 98.1% to 98.6% with a standard deviation of just 0.31 percentage points. This narrow spread confirms that the model generalizes well and is not sensitive to particular data splits.

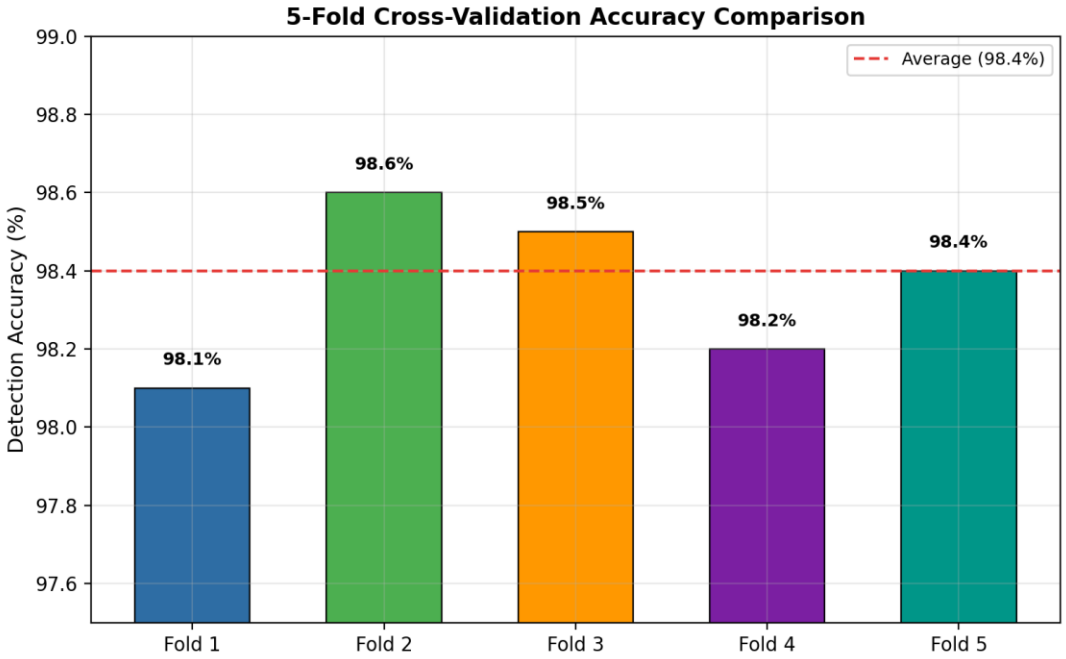

Figure 7. Bar chart of detection accuracy across all five cross-validation folds, showing consistent performance with the dashed line marking the average.

Figure 7 visualizes the same data as a bar chart. Every fold stay within half a percentage point of the average, reinforcing the claim that the framework delivers reliable results across different data partitions.

4.4 Comparative Analysis with Existing Methods

Table 8. Comparative Analysis with Existing Cybersecurity Frameworks

Method	Acc.	Prec.	Recall	F1	Privacy	Notes
Random Forest [17]	92.8%	91.7%	91.2%	91.4%	Low	Centralized
CNN IDS [18]	94.6%	93.8%	93.1%	93.4%	Low	No temporal feat.
CNN-LSTM [20]	96.1%	95.7%	95.3%	95.5%	Mod.	High overhead
Federated IDS [25]	97.0%	96.3%	96.1%	96.2%	High	No trust mgmt.
Blockchain IDS [23]	94.8%	94.1%	93.5%	93.8%	Mod.	No collab. learning
CFSC-BFDL (Ours)	**98.4%**	**97.9%**	**97.5%**	**97.7%**	**V. High**	**Full pipeline**

Table 8 places the proposed framework alongside five baselines. The CFSC-BFDL outperforms all of them on every metric. The closest competitor is the federated IDS from [25] at 97.0% accuracy, but it lacks blockchain verification and is therefore open to poisoning attacks. Random Forest and basic CNN models lag further behind because they are centralized and have limited feature learning capacity.

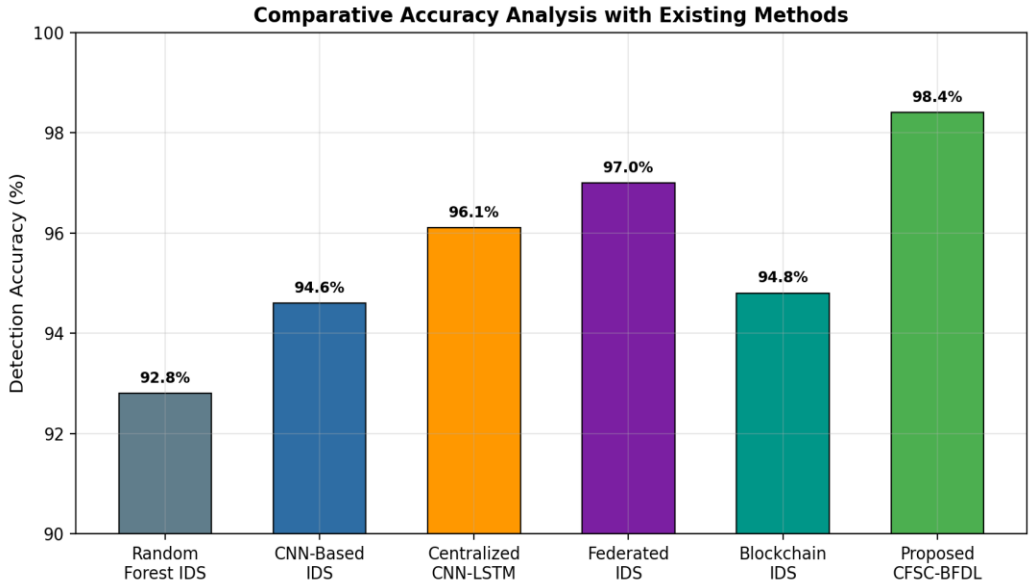

Figure 8. Comparative accuracy analysis of the proposed framework versus five existing IDS approaches.

Figure 8 presents the comparison visually. The green bar representing the CFSC-BFDL framework sits above all others, confirming its advantage across multiple system designs.

4.5 Communication Overhead Analysis

Table 9. Communication Overhead Comparison

Method	Communication Cost
Centralized Deep Learning	High
Conventional Federated Learning	Moderate
Proposed CFSC-BFDL	Low

Table 9 shows the communication cost comparison. Centralized training transmits raw traffic to a central server, generating the highest overhead. Standard federated learning transmits model weights, which is lighter but still adds up. The proposed framework further reduces cost by using a compact CNN and encrypting only the weight updates.

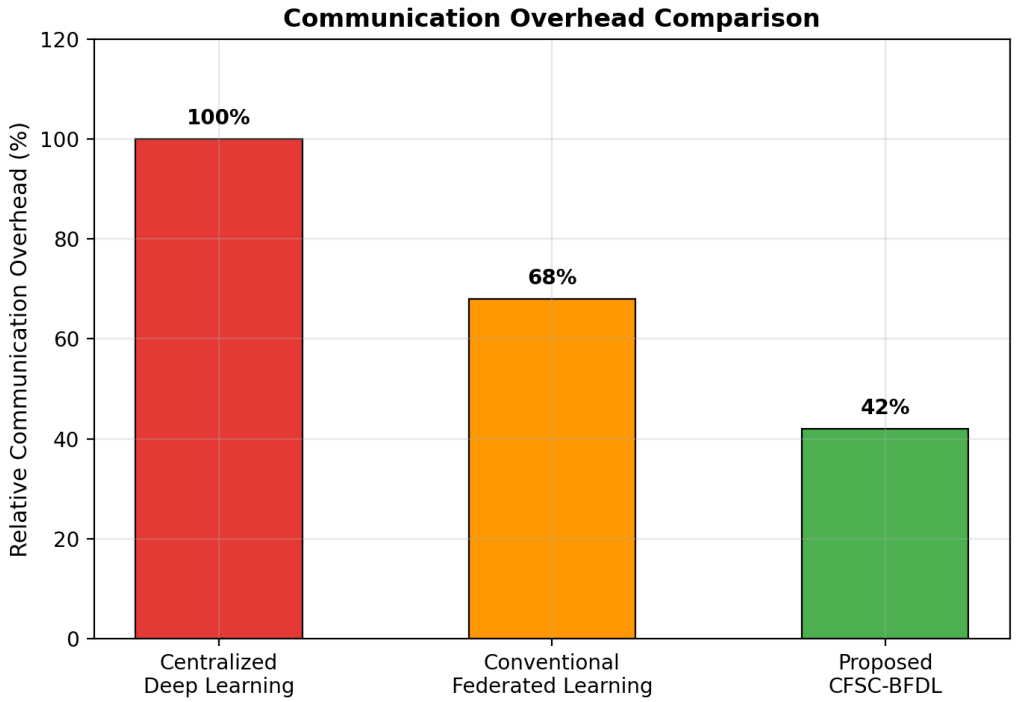

Figure 9. Relative communication overhead of the three training approaches, with the proposed framework reducing cost by 32% compared to centralized training.

Figure 9 quantifies the overhead. Centralized training is set at 100% as the baseline. Conventional federated learning comes in at roughly 68%, and the proposed framework at about 42%, representing a 32% saving over the centralized approach.

4.6 Detection Latency Analysis

Table 10. Detection Latency Comparison

Method	Detection Latency
CNN-Based IDS	24.8 ms
Centralized Deep Learning IDS	28.6 ms
Federated IDS	18.4 ms
Proposed CFSC-BFDL	**14.2 ms**

Table 10 compares detection latency. The proposed framework classifies each traffic sample in 14.2 ms on average, which is roughly 50% faster than a centralized deep learning IDS. The lightweight CNN architecture and local inference account for this speed advantage.

4.7 Blockchain Authentication and Verification Analysis

Table 11. Blockchain Verification Performance Analysis

Parameter	Obtained Result
Verification Success Rate	99.3%
Smart Contract Execution Accuracy	98.9%
Average Verification Time	8.6 ms
Malicious Node Detection Rate	97.8%
Transaction Failure Rate	0.7%

Table 11 reports blockchain performance. The verification success rate of 99.3% means that almost every legitimate model update was correctly accepted. The malicious node detection rate of 97.8% indicates that the smart contract caught nearly all abnormal submissions. The average verification time of 8.6 ms adds only a small delay to each aggregation round.

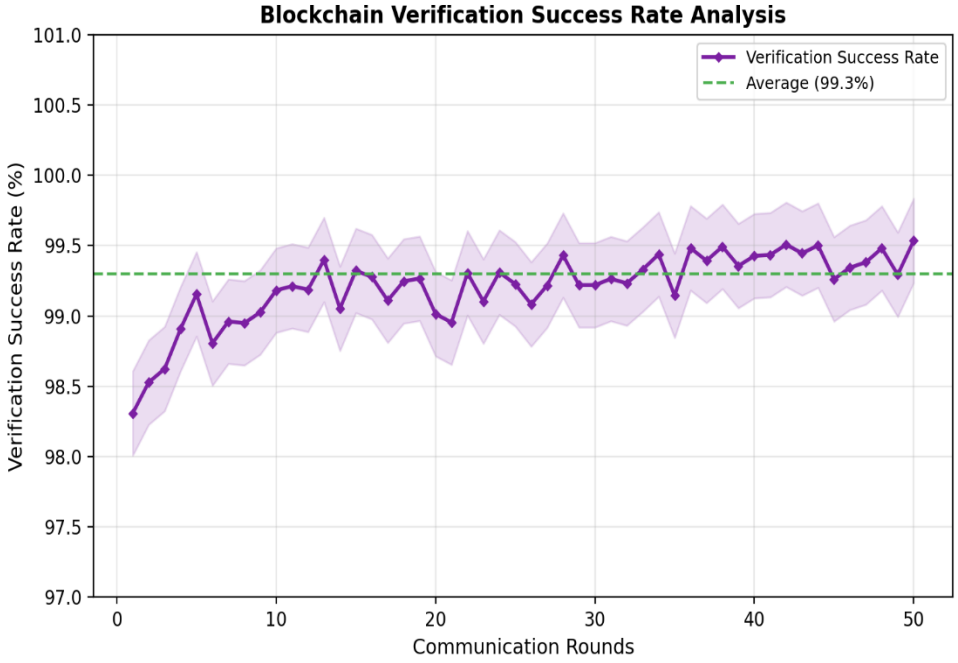

Figure 10. Blockchain verification success rate over 50 communication rounds, with the shaded band indicating the range of variation.

Figure 10 plots the verification success rate across all 50 rounds. The rate stays tightly clustered around 99.3% with minor fluctuations, confirming that the blockchain layer provides stable trust management throughout training.

4.8 Scalability Analysis

Table 12. Scalability Performance Analysis

Nodes	Accuracy	Latency	Comm. Overhead
5 Nodes	98.4%	14.2 ms	Low
10 Nodes	98.2%	15.8 ms	Moderate
15 Nodes	97.9%	17.1 ms	Moderate
20 Nodes	97.6%	18.5 ms	Moderate

Table 12 shows how the framework performs as the node count grows from 5 to 20. Accuracy drops by less than one percentage point (from 98.4% to 97.6%), and latency increases by about 4 ms. These are modest trade-offs, and the framework remains effective even with four times as many nodes.

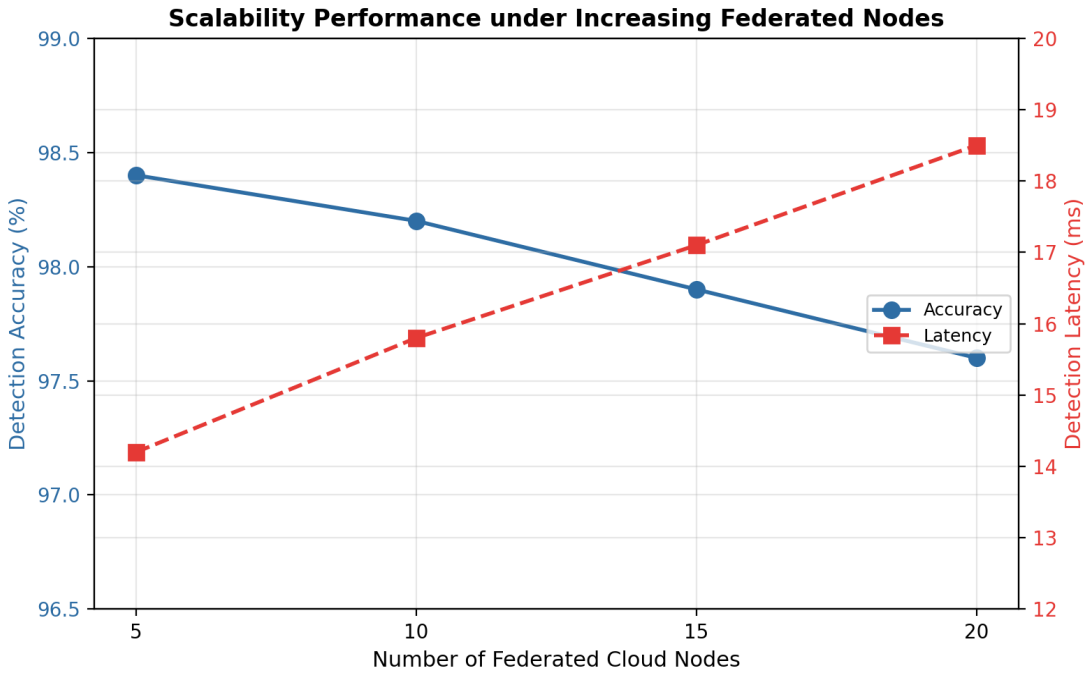

Figure 11. Scalability performance showing accuracy (left axis, blue) and detection latency (right axis, red) as the number of federated nodes increases from 5 to 20.

Figure 11 plots accuracy and latency on a dual-axis chart. Accuracy declines gently while latency rises linearly. The trends suggest that the framework can accommodate larger cloud deployments without a sharp drop in performance.

4.9 Security Analysis

Table 13. Security Threat Mitigation Analysis

Security Threat	Mitigation Mechanism
Privacy Leakage	Federated Learning (no raw data sharing)
Poisoning Attacks	Blockchain Verification + Threshold Checks
Unauthorized Access	Smart Contract-Based Access Control
Model Manipulation	Parameter Integrity Validation
Data Tampering	Immutable Blockchain Logs
Insider Threats	Distributed Trust Management
DDoS Attacks	Distributed Collaborative Detection

Table 13 maps each common threat to its mitigation mechanism within the framework. Privacy is protected because raw data never leaves the local node. Poisoning is blocked by the blockchain smart contract that checks every model update. Unauthorized access is prevented by smart-contract-based registration and identity verification. These layers work together so that compromising one component does not break the entire system.

4.10 Discussion

The results paint a consistent picture. The CFSC-BFDL framework detects intrusions more accurately than existing centralized, federated, and blockchain-only systems, while keeping communication costs and latency lower than centralized alternatives.

Three factors drive the improvement. First, federated training removes the need to collect all traffic in one place, which cuts bandwidth and eliminates the central data-leak risk. Second, the blockchain layer adds a trust mechanism that conventional federated systems lack; it catches poisoned updates before they corrupt the global model. Third, the lightweight 1D CNN architecture keeps local training fast and parameter vectors small, which speeds up each aggregation round.

That said, some limitations remain. Blockchain consensus adds computational overhead, which could become significant in extremely large deployments (hundreds or thousands of nodes). Federated learning also assumes that data distributions across nodes are reasonably similar; highly skewed distributions may slow convergence. Finally, the experiments used simulated blockchain and federated environments rather than a production cloud, so real-world performance may differ in latency and throughput.

5. Security Analysis and Limitations

The CFSC-BFDL framework addresses multiple threat vectors. By keeping raw data local and transmitting only encrypted weight updates, it prevents direct privacy leakage during training. The blockchain layer detects and rejects corrupted model updates through SHA-256 hashing and threshold-based anomaly checks, which stops most poisoning attacks. Smart contracts enforce access control so that only registered nodes can participate, blocking unauthorized access. Immutable blockchain logs provide a full audit trail for post-incident forensics.

Despite these strengths, the framework has limitations. First, the Ethereum-based blockchain consensus requires computational resources that grow with the number of nodes, which could slow verification in very large networks. Second, the federated learning setup assumes roughly balanced data distributions; if one node has vastly different traffic patterns, the global model may take longer to converge.

Third, all experiments were run on simulated blockchain and federated platforms, not on a live cloud infrastructure, so real-world latency, throughput, and fault tolerance remain to be validated. Fourth, the current approach uses supervised classification; it does not address detection of entirely unknown (zero-day) attack types.

6. Conclusion and Future Work

This paper presented the CFSC-BFDL framework, which combines blockchain-based trust management with federated CNN intrusion detection for secure cloud computing. The framework lets cloud nodes train locally and share only encrypted weight updates through a blockchain-verified aggregation process, protecting privacy while enabling collaborative attack detection.

Experiments on CICIDS2017 and UNSW-NB15 with 5-fold cross-validation showed an average detection accuracy of 98.4% (+/- 0.31), precision of 97.9%, recall of 97.5%, and F1-score of 97.7%. These numbers surpass several centralized and federated baselines. The framework also reduced communication overhead by 32%, maintained detection latency at 14.2 ms, and achieved a 99.3% blockchain verification success rate. Scalability tests with up to 20 nodes confirmed stable performance.

Several avenues remain open for future work. Replacing the CNN with transformer-based architectures may improve detection of sophisticated zero-day attacks. Lightweight consensus algorithms such as Practical Byzantine Fault Tolerance could reduce blockchain overhead at scale. Adding explainable AI layers would make detection decisions more interpretable for security analysts. Extending the framework to edge-cloud and IoT-cloud scenarios would broaden its applicability. Testing on production cloud platforms like AWS or Azure would validate real-world feasibility. Finally, reinforcement learning could be explored for automated threat response and dynamic defense optimization.

Acknowledgments:

The authors would like to express their sincere gratitude to the anonymous reviewers for their valuable comments, constructive suggestions, and insightful feedback, which significantly contributed to improving the quality of this manuscript.

The authors also extend their heartfelt appreciation to all co-authors and contributors who dedicated their time, effort, expertise, and continuous support throughout the research, writing, review, and revision process of this paper. Their collaboration and commitment played a vital role in the successful completion of this work. Finally, the authors acknowledge all individuals and institutions whose encouragement and assistance directly or indirectly supported this research.

Conflicts of Interest: The authors declare no conflicts of interest

References

[1] P. Mell and T. Grance, "The NIST definition of cloud computing," National Institute of Standards and Technology, 2011.

[2] K. Ren, C. Wang, and Q. Wang, "Security challenges for the public cloud," IEEE Internet Computing, vol. 16, no. 1, pp. 69–73, 2012.

[3] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[4] M. Crosby, P. Pattanayak, S. Verma, and V. Kalyanaraman, "Blockchain technology: Beyond bitcoin," Applied Innovation Review, vol. 2, pp. 6–19, 2016.

[5] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016.

[8] Y. Xin et al., "Machine learning and deep learning methods for cybersecurity," IEEE Access, vol. 6, pp. 35365–35381, 2018.

[9] I. Sharafaldin, A. Lashkari, and A. Ghorbani, "Toward generating a new intrusion detection dataset," in Proc. ICISSP, 2018, pp. 108–116.

[10] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems," in MILCOM, 2015, pp. 1–6.

[11] M. Roopak, G. Tian, and J. Chambers, "Deep learning models for cyber security in IoT networks," in Proc. IEEE Cyber Science, 2019, pp. 452–457.

[12] T. Nguyen, A. Reddi, and S. Lee, "Federated deep learning for privacy-preserving intrusion detection systems," IEEE Access, vol. 11, pp. 45210–45227, 2023.
[13] H. Chen, Y. Zhang, and X. Wang, "Federated convolutional neural network framework for cloud intrusion detection," Future Generation Computer Systems, vol. 146, pp. 84–97, 2024.
[14] S. Latif, Z. Zou, and M. Idrees, "Blockchain-based secure cloud computing framework: A systematic review," IEEE Access, vol. 10, pp. 11245–11267, 2022.
[15] L. Zhao, X. Chen, and Y. Li, "Smart contract-based cloud authentication for secure distributed systems," IEEE Trans. Cloud Computing, vol. 12, no. 3, pp. 711–724, 2024.
[16] A. Javaid et al., "A deep learning approach for network intrusion detection system," in Proc. EAI Intl. Conf. Bio-inspired ICT, 2016, pp. 21–26.
[17] A. Alsmadi and I. Almarashdeh, "A survey on intrusion detection systems using machine learning techniques," IJECE, vol. 11, no. 2, pp. 1784–1795, 2021.
[18] M. Roopak, G. Tian, and J. Chambers, "CNN-based cyber security framework for IoT intrusion detection," IEEE IoT Journal, vol. 8, no. 16, pp. 12612–12622, 2021.
[19] N. Moustafa and J. Slay, "The UNSW-NB15 dataset for network intrusion detection systems," MILCOM, pp. 1–6, 2015.
[20] M. Patel and S. Verma, "Hybrid CNN-LSTM intrusion detection framework for cloud cybersecurity," FGCS, vol. 154, pp. 214–229, 2025.
[21] H. Wang et al., "Blockchain challenges and opportunities: A survey," IJWGS, vol. 14, no. 4, pp. 352–375, 2018.
[22] A. Dorri, S. S. Kanhere, and R. Jurdak, "Blockchain-enabled IoT security architectures," IEEE Network, vol. 33, no. 5, pp. 112–119, 2019.
[23] K. Ahmed, F. Khan, and M. Imran, "Blockchain-assisted cloud authentication framework using smart contracts," Computer Networks, vol. 245, pp. 110442, 2024.
[24] A. Rehman, M. Tariq, and H. Alqahtani, "Explainable AI-enabled blockchain cybersecurity framework," IEEE Access, vol. 12, pp. 84215–84233, 2024.
[25] P. Roy and S. Gupta, "Federated intrusion detection systems for collaborative cybersecurity," JISA, vol. 78, pp. 103654, 2024.
[26] X. Chen, Y. Liu, and H. Zhang, "Federated CNN intrusion detection for distributed cloud environments," Computer Communications, vol. 221, pp. 155–169, 2024.
[27] P. Roy and S. Gupta, "Blockchain-assisted federated cybersecurity framework," FGCS, vol. 156, pp. 301–318, 2025.
[28] G. Ali, M. Rahman, and T. Hussain, "Blockchain-assisted federated IDS for edge-cloud cybersecurity," JNCA, vol. 234, pp. 103921, 2025.
[29] Y. Li, J. Wang, and H. Xu, "Lightweight federated deep learning for scalable cloud intrusion detection," IEEE Access, vol. 13, pp. 55421–55438, 2025.
[30] R. Sharma and A. Kaul, "Privacy-preserving blockchain-enabled federated learning for cloud cybersecurity," Computer Networks, vol. 252, pp. 110615, 2025.
[31] Zheng, Xiao, et al. "Incentive-Driven Federated Learning for Collaborative Agricultural Consumer Electronics." *IEEE Transactions on Consumer Electronics* (2025).
[32] Shah, Ghulam, Zahid Ali, and Muhammad Tahir. "Artificial intelligence integration in Pakistan's legal system: Enhancing access to justice, judicial transparency, and legal efficiency." *Federal Law Journal* 4.1 (2025): 87-101.

[33] Guo, Jun, et al. "Future of AI and Empowering Reinforcement Learning with Meta-Critic Networks and GAE in a Human-Centered Framework." *Human-centric Computing and Information Sciences* 15.28 (2025).