Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.067521

ARTICLE

Adaptive Multi-Layer Defense Mechanism for Trusted Federated Learning in Network Security Assessment

Lincong Zhao[1], Liandong Chen[1], Peipei Shen[1], Zizhou Liu[1], Chengzhu Li[1] and Fanqin Zhou[2,*]

[1]State Grid Hebei Information and Telecommunication Branch, Shijiazhuang, 050000, China
[2]State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, 100876, China

*Corresponding Author: Fanqin Zhou. Email: fqzhou2012@bupt.edu.cn

Received: 06 May 2025; Accepted: 13 August 2025; Published: 23 October 2025

ABSTRACT: The rapid growth of Internet of things devices and the emergence of rapidly evolving network threats have made traditional security assessment methods inadequate. Federated learning offers a promising solution to expedite the training of security assessment models. However, ensuring the trustworthiness and robustness of federated learning under multi-party collaboration scenarios remains a challenge. To address these issues, this study proposes a shard aggregation network structure and a malicious node detection mechanism, along with improvements to the federated learning training process. First, we extract the data features of the participants by using spectral clustering methods combined with a Gaussian kernel function. Then, we introduce a multi-objective decision-making approach that combines data distribution consistency, consensus communication overhead, and consensus result reliability in order to determine the final network sharing scheme. Finally, by integrating the federated learning aggregation process with the malicious node detection mechanism, we improve the traditional decentralized learning process. Our proposed ShardFed algorithm outperforms conventional classification algorithms and state-of-the-art machine learning methods like FedProx and FedCurv in convergence speed, robustness against data interference, and adaptability across multiple scenarios. Experimental results demonstrate that the proposed approach improves model accuracy by up to 2.33% under non-independent and identically distributed data conditions, maintains higher performance with malicious nodes containing poisoned data ratios of 20%–50%, and significantly enhances model resistance to low-quality data.

KEYWORDS: Trusted federated learning; adaptive defense mechanism; network security assessment; participant trustworthiness scoring; hybrid anomaly detection

1 Introduction

The contemporary digital landscape, marked by the rapid proliferation of Internet of things (IoT) devices, has created a complex network ecosystem that challenges traditional cybersecurity models [1]. With increasing reliance on interconnected infrastructure, network threats have escalated sharply, rendering static rule-based and cryptographic methods inadequate for addressing modern, adaptive cyberattacks [2]. Federated learning (FL) has emerged as a decentralized, privacy-oriented method for collaborative threat detection [3]. It allows secure multi-party model training without sharing raw data, enhancing threat detection capabilities. However, challenges such as data heterogeneity, communication overhead, and vulnerability to adversarial attacks (e.g., data poisoning) limit its adoption [4]. Additionally, the resource constraints of edge devices in federated IoT environments—such as limited memory, bandwidth, and

energy budgets—pose significant implementation and scalability challenges, further hindering practical deployment [5].

These issues are especially critical in network security, where real-time responsiveness and trust are essential [6]. Inconsistent data quality and insufficient mechanisms for identifying malicious behavior degrade model performance and compromise system integrity [7]. Moreover, current methods often overlook complex attacks like model poisoning and Sybil attacks. Addressing these challenges requires a multidimensional approach that integrates adaptive machine learning, dynamic trust evaluation, and intelligent defense mechanisms [8]. Recent studies highlight the promise of combining spectral clustering, blockchain-based validation, and anomaly detection for proactive federated learning frameworks. These systems must dynamically adjust network topology, assess trust, and respond to evolving threats. AzharShokoufeh et al. in [9] further demonstrates the benefits of periodic scheduling in federated IoT data collection, improving bandwidth, energy use, and model accuracy through local processing and update-based communication. The integration of machine learning, cryptography, and adaptive intelligence signals a shift from reactive to proactive network security [10]. Robust multi-layer defense—combining statistical analysis, ML, and cryptographic verification—offers a comprehensive response to the complex and evolving threat landscape in modern networks.

The main contributions can be summarized as follows. 1) A revolutionary multi-layer adaptive defense mechanism is conceptualized and implemented to significantly enhance the resilience and intelligent response capabilities of trusted federated learning in network security assessment scenarios. 2) A sophisticated probabilistic trustworthiness scoring algorithm is developed, supporting dynamic network sharding, intelligent participant selection, and continuous trust re-evaluation throughout the distributed training lifecycle. 3) A comprehensive hybrid detection approach is introduced, combining statistical anomaly analysis, machine learning-based attack pattern recognition, and cryptographic verification to form a multidimensional, adaptive defense strategy.

2 System Model

2.1 Trusted Federated Learning Aggregation Network Sharding Method

In federated learning with multiple trusted parties, performance and security face several key challenges. The heterogeneous nature of participant environments creates substantial data distribution inconsistencies, where non-independent and identically distributed (non-IID) datasets severely compromise both model aggregation efficiency and final performance metrics. Blockchain-based consensus incurs high communication costs due to frequent coordination [11]. Additionally, inconsistent data quality elevates the risk of data/model poisoning, necessitating strong consensus evaluations.

The method applies spectral clustering to derive high-dimensional features from each party's data, utilizing its characteristics and label distributions [12]. A hierarchical clustering algorithm constructs a tree to serve as the candidate for network sharding. A multi-objective decision framework, which assesses data distribution, communication overhead, and reliability, refines the sharding scheme, resulting in a hierarchical model aggregation network for dependable federated learning.

A key challenge in federated learning is participant data heterogeneity. This heterogeneity is reflected in feature variance and label skewness. To capture these characteristics, we extract statistical moments from each participant's local dataset [13]. For numerical features, we compute four moments: mean (μ), variance (σ^2), skewness (S), and kurtosis (K). The mean describes the central tendency of the data. The variance measures data spread. Skewness reflects data asymmetry. Kurtosis measures peak sharpness and tail weight. Given a numerical feature set $X = \{x_1, x_2, ..., x_n\}$, the formulas for mean, variance, skewness, and kurtosis

are as follows,

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i, \tag{1}$$

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2, \tag{2}$$

$$S = \frac{n}{(n-1)(n-2)}\sum_{i=1}^{n}\left(\frac{x_i - \mu}{\sigma}\right)^3, \tag{3}$$

$$K = \frac{n}{(n-1)(n-2)}\sum_{i=1}^{n}\left(\frac{x_i - \mu}{\sigma}\right)^4. \tag{4}$$

To represent the label distribution in participants' datasets, the probability distribution of different labels is used as feature values. For a dataset with m possible labels, let $L = \{l_1, l_2, ..., l_m\}$ be the label set, N be the total number of samples in the dataset, and n_j be the number of occurrences of label l_j. The probability distribution of label l_j is given by

$$P(l_j) = \frac{n_j}{N}, \quad j \in \{1, 2, \ldots, m\}. \tag{5}$$

By collecting probability distributions for all labels, the label distribution probability vector $P(L) = \{P(l_1), P(l_2), ..., P(l_m)\}$ is obtained. By combining these two parts, a dataset feature set $C = \{X_1, X_2, X_3, ..., X_n\}$ is formed for n participants.

Since the data features of participants usually have high dimensions, complex internal logic, and varying parameter scales, spectral clustering is used to process the extracted data features, obtaining latent feature vectors and reducing dimensionality to decrease computational overhead in subsequent sharding scheme construction. Specifically, given the local feature set C, a similarity matrix is computed using the Gaussian kernel function,

$$S_{n\times n} = \begin{bmatrix} \omega_{11} & \cdots & \omega_{1n} \\ \vdots & \ddots & \vdots \\ \omega_{n1} & \cdots & \omega_{nn} \end{bmatrix}, \tag{6}$$

where $\omega_{ij} = \exp\left(-\|x_i - x_j\|^2/(2\sigma^2)\right)$. Here, σ is the kernel bandwidth parameter that controls the influence range of each data point. A smaller σ creates more localized similarities, while a larger σ creates broader similarities. The indices i and j represent positions in matrix S as well as in dataset C.

Next, we define the degree matrix D as

$$D = \begin{bmatrix} d_1 & 0 & \cdots \\ 0 & d_2 & \vdots \\ \vdots & \vdots & \ddots \\ \cdots & \cdots & d_n \end{bmatrix}, \tag{7}$$

where the diagonal elements are computed as $d_i = \sum_{j=1}^{n} \omega_{ij}$. Using the degree matrix D and similarity matrix S, the Laplacian matrix L is computed as $L = D - S$. By normalizing, the standardized Laplacian matrix L_{std} is obtained by $L_{std} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$. We then compute the eigenvalues λ_i and their corresponding eigenvectors f_i. The smallest k eigenvalues are selected to form the feature matrix $F_{n\times k} = (f_1, f_2, ..., f_k)$. This

reduces the original $n \times n$ feature space to $n \times k$, where k is a constant. We set $k = \min(10, \lfloor n/2 \rfloor)$ to balance dimensionality reduction with information preservation.

Based on the above steps, the feature matrix F of the local data from n participants is computed. Using a hierarchical clustering algorithm, we obtain the optimal partitioning scheme for different classification numbers as a candidate for the final partitioning scheme. Hierarchical clustering is a method that groups nodes with high similarity. It iteratively computes the similarity between different categories and gradually constructs a tree-structured classification model.

In this study, we use the Euclidean distance based on the feature matrix F to measure the similarity between clusters:

$$d(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{p \in C_i} \sum_{q \in C_j} |p - q|, \tag{8}$$

$$|p - q|^2 = (p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_k - q_k)^2. \tag{9}$$

Here, C_i and C_j are two existing clusters. We use the class mean distance to measure their similarity. p_i and q_j represent the i-th component of points p and q in the feature matrix F.

Initially, each participant is treated as a separate cluster. In each iteration, the two closest clusters are merged into a new cluster. This process iterates bottom-up to form a tree structure. In the hierarchical clustering tree, each level represents a new partitioning optimization scheme. Multiple influencing factors must be considered to determine the final partitioning scheme [14,15].

Through our analysis in Section 3.1 on security state detection issues in trusted federated learning [16], we identify that the final selection of the partitioning model aggregation scheme must consider three aspects: similarity of participant data features within a partition [17], communication overhead for achieving distributed data consensus [18], and the resilience of the consensus result against malicious attacks [19]. The data features within a partition determine the performance of local models used by participants. The consensus communication overhead determines the cost of communication per training and data synchronization round [20]. The size of consensus participants determines the partition's ability to resist Byzantine nodes, ultimately affecting the trustworthiness of data records [21].

During machine learning model training, the similarity of the training dataset feature distribution affects model convergence speed and prediction performance. The intra-cluster data similarity metric G_L can measure the degree of similarity in the local datasets of federated learning participants. This study proposes a similarity measure based on intra-cluster average distance $d_{ij} = \sqrt{s_{ij}}$ and $s_{ij} = (p_{i1} - q_{j1})^2 + (p_{i2} - q_{j2})^2 + \cdots + (p_{ik} - q_{jk})^2$. Then the G_L can be expressed as

$$G_L = \frac{1}{m(m-1)} \sum_{i \neq j} d_{ij}, \tag{10}$$

where m is the number of participants in the cluster, and k is the feature dimension of each participant's data. This can be computed using the feature matrix $F_{n \times k}$ obtained in the previous section. Note that the feature matrix includes all participants in the federated learning distributed network. Therefore, the subset of row vectors corresponding to the cluster needs to be extracted for computation.

A distributed network communication system can be abstracted as a fully connected weighted graph. The weights between device nodes estimate the internal communication overhead of the cluster and can be

expressed as

$$G_T = \frac{1}{m(m-1)} \sum_{i \neq j} W(N_i, N_j). \tag{11}$$

It calculates the average intra-cluster communication overhead [22]. Here, m is the cluster size, and $W(A, B)$ represents the communication overhead between nodes A and B. According to the principles of the Practical Byzantine Fault Tolerance (PBFT) mechanism [23], achieving consensus on a transaction requires three rounds of broadcast communication within the network. Thus, the average network communication overhead effectively measures the network resource usage for reaching a single transaction consensus.

Since the consensus mechanism follows the principle of majority rule, we assume there are f malicious nodes among the consensus participants. The data recorded in the distributed ledger is trustworthy only if the network size exceeds $3f + 1$ [24]. Based on this principle, assuming the probability of an individual device being attacked follows a binomial distribution [25], the consensus trustworthiness calculation formula is

$$G = \frac{G_C}{G_L G_T}. \tag{12}$$

By substituting the feature matrix and network parameters obtained in the previous section into this formula, we obtain the scores of different candidate partitioning optimization schemes. The scheme with the highest score is selected as the final federated partition aggregation network structure for deployment.

2.2 Improvement of Federated Learning Training Process with Anomaly Detection

In trusted federated learning, the multi-organizational nature introduces significant challenges in ensuring data quality and model credibility. Malicious participants can employ various attack strategies including data poisoning, model poisoning, and coordinated attacks to compromise the global model's integrity. Our approach addresses these threats through a comprehensive anomaly detection mechanism integrated into the model aggregation process.

We define model distance as a quantitative measure of deviation between individual participant models and the expected global consensus. This approach captures anomalous behavior patterns that may indicate malicious activity or data quality issues. The residual model calculation is given by

$$RM_i = \sqrt{L_G - \frac{|d_i|}{|D|} L_i}, \tag{13}$$

where L_G represents the global model parameter vector, D denotes the total data size in federated learning, d_i is the local dataset size of participant i, and L_i is the local model parameter vector of participant i.

Based on the computed residual models, we calculate the distance between the local model and the residual global model excluding the local contribution as an anomaly feature

$$Dis_i = \sum_{j=1}^{m} \sqrt{(L_{ij} - RM_{ij})^2}, \ j \in [1, m], \tag{14}$$

where i represents the participant index, L is the local model vector, RM is the residual global model excluding the local contribution, and m is the number of model parameters. This formula calculates the deviation of the local model from the aggregated model of other participants by computing the geometric distance between participant i and others in parameter space.

A statistical method for anomaly detection is subsequently utilized. Real-world data is often assumed to follow a normal distribution. We identify outliers using the data's mean and standard deviation. In the n-sigma test, a normal node has a probability of 95.45% for $n = 2$ and 99.73% for $n = 3$. The threshold can be adjusted based on different scenarios and requirements. In our implementation, we use $n = 2$ for moderate anomaly detection and $n = 3$ for strict detection, depending on the security requirements.

We calculate the mean and variance of Dis_i as $\overline{Dis_i} = \sum_{i=1}^{n} Dis_i/n, \quad i \in [1, n]$, and $\sigma = \sqrt{\sum_{i=1}^{n}(Dis_i - \overline{Dis_i})^2/(n-1)}$. For each participating model, after computing the above feature values, we determine whether the model distance Dis_i falls within the reasonable range $[r - n\sigma, r + n\sigma]$. Here, r represents the mean value $\overline{Dis_i}$. The value of n can be adjusted as needed. If the model falls within this range, its parameters are considered normal. Otherwise, it is removed and not included in the global model aggregation.

Through this method, we complete the detection and identification of malicious participants. Next, combining the trusted federated learning sharding aggregation structure and malicious node detection mechanism, this study improves the traditional decentralized federated learning process to enhance training efficiency and robustness against malicious nodes.

The following steps illustrate the process of ShardFed.

Step 1 - Data Preparation: Extract the data and label distribution features of participants engaged in model aggregtion. Construct deep learning models for security assessment in network environments and initialize the models.

Step 2 - Shard Calculation: Use a trusted federated learning aggregation network sharding method to determine the optimal sharding aggregation scheme based on participant data distribution characteristics.

Step 3 - Model Deployment: Distribute the initialized global model to participant nodes in each shard via smart contract interfaces. Security assessment servers receive the latest models.

Step 4 - Local Model Training: Each participant trains and updates local security detection models using private data without centralizing security event data.

Step 5 - Malicious Node Detection: Participants share local models via smart contracts. The system calculates model distances and identifies malicious models using the anomaly detection mechanism. Model information and detection results are recorded.

Step 6 - Model Aggregation: Filtered models are aggregated within each shard to generate local shard models. These are then aggregated into a global model using the federated averaging algorithm. The global model enhances the generalization ability of shard models, improving training efficiency. However, each participant still receives its respective shard model for practical deployment.

Step 7 - Shard Model Update: Each shard receives the refined global model, and its performance is evaluated. The shard model is then adjusted using $SM = p \cdot SM + (1 - p) \cdot G$, with SM as the shard model and G as the global model. The parameter p, ranging from 0 to 1, is a performance weight reflecting the shard model's performance relative to the global model. A greater p implies superior shard performance and decreases the global model's impact. Initially, a small p accelerates convergence by incorporating more data from other shards. As convergence progresses, p nears 1, lessening global model influence on shard models.

Steps 3–7 are repeated until all participant models converge to the target performance level.

The overall processes are summarized in Algorithm 1.

Algorithm 1: Improved federated learning model training process

Input: Number of rounds for slice algorithm, security assessment model, and training parameters, local dataset for each participant local_data

Output: global_model

Initialize: Initialize global_model and shard_model = model;
shard_plan = get_shard_plan(local_metadata);
for *round in range(n)* **do**
    for *shard in shard_plan* **do**
        p ← test(shard_model, test_data);
        shard_model ← p * shard_model + (1-p) * global_model // Update shard model
        for *participant in shard* **do**
            local_model ← shard_model
            local_model ← train(local_model, local_data) // Local model training
        dis_list ← compute(local_model_list) // Calculate model distance
        bzt_set ← filterMalicious(dis_list) // Byzantine node identification
        for local_model not in bzt_set **do**
            shard_model ← aggregate(local_model) // Slice model aggregation
        global_model ← aggregate(shard_model_list) // Global model aggregation

2.3 Computational Complexity Analysis

The computational complexity of our sharding method consists of several components. Feature extraction has a complexity of $O(nd)$, where n is the number of participants and d is the feature dimensionality. Constructing the similarity matrix requires $O(n^2 d)$ operations to compute pairwise Gaussian kernel similarities. The eigen decomposition step, used for computing the spectral embedding, has a complexity of $O(n^3)$; however, this can be reduced to $O(nk^2)$ when using sparse eigen solvers and $k \ll n$. Hierarchical clustering has a worst-case complexity of $O(n^3)$, but more efficient implementations can achieve $O(n^2 \log n)$. Finally, the multi-objective evaluation step incurs a cost of $O(n^2)$ for each candidate scheme, resulting in a total evaluation complexity of $O(n^3)$. The time complexity, quantified as $O(n^3)$, suffices for federated learning models with several hundred participants. In instances of larger scale, approximation methods like Nyström sampling offer complexity reduction to $O(nk^2)$, where k remains small. Though the present study targets medium-scale FL systems, the architecture's scalability to high-speed or expansive deployments via distributed shard coordination, asynchronous updates, and efficient clustering techniques is acknowledged for future exploration.

3 Simulation Scheme and Result Analysis

3.1 Threat Models

To provide a comprehensive security analysis framework, we define two primary threat models that the proposed trustworthy federated learning system addresses. These threat models establish the foundation for evaluating the robustness and security guarantees of our approach.

3.1.1 Data Poisoning Attacks

In data poisoning attacks, malicious participants deliberately corrupt their local training datasets to degrade the performance of the global model. Specifically, let $\mathcal{D}_i = \{(x_j^{(i)}, y_j^{(i)})\}_{j=1}^{n_i}$ represent the local dataset of participant i, where n_i is the number of samples. A malicious participant i modifies a fraction

$\rho \in (0,1]$ of their labels, creating a poisoned dataset $\mathcal{D}_i^{poison}$ where

$$y_j^{(i),poison} = \begin{cases} \text{random\_label}() & \text{with probability } \rho \\ y_j^{(i)} & \text{with probability } 1-\rho \end{cases} \tag{15}$$

The objective of data poisoning is to maximize the degradation of global model accuracy while remaining undetected by the aggregation mechanism.

3.1.2 Model Poisoning Attacks

In model poisoning attacks, malicious participants submit crafted model updates that deviate significantly from the expected gradient directions. Let θ_t represent the global model parameters at round t, and Δ_i^t represent the local update from participant i. A malicious participant generates poisoned updates $\Delta_i^{poison,t} = \alpha \cdot \text{sign}(\nabla_\theta \mathcal{L}(\theta_t, \mathcal{D}_{val})) + \beta \cdot \varepsilon_i$ where α controls the attack magnitude, $\mathcal{L}(\theta_t, \mathcal{D}_{val})$ is the loss on a validation set, and ϵ_i represents random noise to avoid detection. The parameter β adds stochasticity to the attack vector.

3.1.3 Theoretical Security Guarantees

For the proposed anomaly detection mechanism with threshold τ, we provide the following theoretical guarantee.

Theorem 1 (Detection Accuracy Bound): *Given m malicious participants out of n total participants, where each malicious participant has deviation score $d_i > \tau$, the probability of correctly identifying all malicious participants is bounded by*

$$P(\text{correct detection}) \geq 1 - m \cdot \exp\left(-\frac{(\tau - \mu_d)^2}{2\sigma_d^2}\right), \tag{16}$$

where μ_d and σ_d represent the mean and standard deviation of deviation scores for honest participants, respectively.

This bound demonstrates that the detection accuracy improves exponentially with the separation between honest and malicious participant behaviors, providing theoretical justification for the proposed approach.

3.2 Experiment Setup

This experiment aims to verify the effectiveness of the trusted federated learning sharding aggregation scheme in improving local model performance and reducing communication overhead under non-IID data distributions among participants. Additionally, it evaluates the robustness of the global model when malicious nodes launch mainly data poisoning attacks, using an anomaly detection-based method for identifying malicious participants.

The main parameters used in the experiment are depicted as follows. During dataset partitioning, non-IID data employ the Dirichlet distribution with a heterogeneity parameter α = 1.0. In local training of the convolutional neural network (CNN) model, the learning rate is set to 0.01. For model sharding, the Gaussian kernel function hyperparameter σ for feature extraction is set to 1, and the number of eigenvalues used for spectral clustering is set to 5. The communication overhead matrix is a symmetric matrix with randomly generated elements from 10 to 20 ms, and its diagonal elements are zero. During federated aggregation,

the global model executes 20 rounds, with each participant completing 2 local training rounds. Finally, the threshold for statistical anomaly detection in malicious node identification is set to 1.5.

This experiment adopts a one-dimensional CNN-based model for IoT network traffic security assessment, as shown in Fig. 1. The architecture comprises three convolutional layers, two max pooling layers, and three fully connected layers. Input data is processed through convolution and pooling to extract features and reduce dimensionality, followed by flattening and fully connected layers for nonlinear learning. ReLU activations add nonlinearity, and the output layer produces probabilities for normal and abnormal classes. The model is implemented in PyTorch, trained with cross-entropy loss, and optimized using stochastic gradient descent. The trusted federated learning sharding aggregation structure via direct model aggregation enhances model performance and communication efficiency under non-IID data distributions.

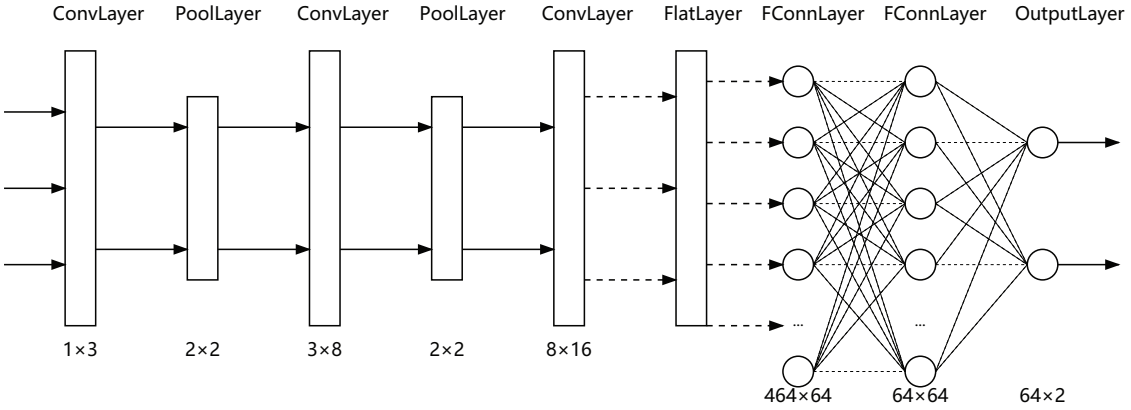

Figure 1: Neural network architecture

To validate the proposed method in network security assessment scenarios, the NSL-KDD dataset [26], a classic benchmark in network security, is mainly used. This dataset includes connection states, packet content features, and time-based traffic statistical features, providing a comprehensive representation of network connections. The performance on the dataset BoT-IoT [27] is also tested.

To ensure non-IID data allocation for each participant, a Dirichlet distribution-based method is introduced. The Dirichlet distribution, a classical high-dimensional continuous probability distribution, is denoted as

$$Dir\left(X|\alpha\right) = \frac{1}{B(\alpha)} \prod_{i=1}^{K} x_i^{\alpha_i - 1} = \frac{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)}{\prod_{i=1}^{K} \Gamma\left(\alpha_i\right)} \prod_{i=1}^{K} x_i^{\alpha_i - 1}. \tag{17}$$

For a random vector $X = (X_1, ..., X_N) \sim Dir(\alpha)$, it satisfies $X_i > 0$ and $\sum_{i=0}^{N} X_i = 1$. The labeled data distribution can be expressed as $p(x, y) = p(x|y)p(y)$. Since $p(x|y)$ is difficult to compute, sample allocation is controlled by adjusting $p(y)$ [28]. By generating a probability vector $q \sim Dir(\alpha)$, where q represents the label distribution and α controls the shape of the distribution, different levels of data heterogeneity can be achieved. When $\alpha \to 0$, samples are highly concentrated in a few categories per participant, leading to extreme data skewness. When $\alpha \to \infty$, the data distribution among participants becomes uniform.

To simulate data quality issues from malicious participants, some labels in the local datasets are randomly modified to implement data poisoning attacks, while model poisoning attacks are simulated by injecting adversarial perturbations into model parameters during the aggregation phase. The effectiveness of

integrating anomaly detection into the federated learning training process is then evaluated to demonstrate the robustness improvement.

To comprehensively assess the advantages of the proposed trusted federated learning method over conventional decentralized federated learning in terms of accuracy, communication efficiency, and robustness against attacks, multiple metrics are defined.

Model accuracy is evaluated using $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$ and F1score = $\frac{2\times(Precision\times Recall)}{Precision+Recall}$, where TP, TN, FP, and FN represent true positives, true negatives, false positives, and false negatives, respectively.

Communication efficiency is measured as $E = z/d$, where z represents the amount of uploaded model data, and d denotes the communication delay for the full consensus process. The metric E reflects the speed of data transmission, where a larger E indicates higher communication efficiency.

3.3 Experimental Results Analysis

This section employs the trusted federated learning shard optimization method, incorporating a mechanism for detecting malicious nodes into the FL model aggregation framework. This integration aims to demonstrate its efficacy in enhancing model training performance, particularly when multiple participants have non-IID data, within the scenario of detecting network anomaly traffic. Based on feature similarity and communication cost, the final aggregation strategy consists of four clusters: participants {0, 2, 5, 9}, {1, 3}, {6, 7, 8}, and participant 4 as an individual shard.

For model training effectiveness, Fig. 2a,b compares the performance of the federated learning sharding aggregation method against direct global aggregation. Under a 10-participant scenario, the sharding approach achieves 90% accuracy in 6 training rounds, whereas direct aggregation requires 9 rounds (50% slower). After 20 training rounds, the average model accuracy of sharded participants reaches 96.47%, compared to 95.63% in the non-sharded approach, demonstrating a 0.84% improvement.

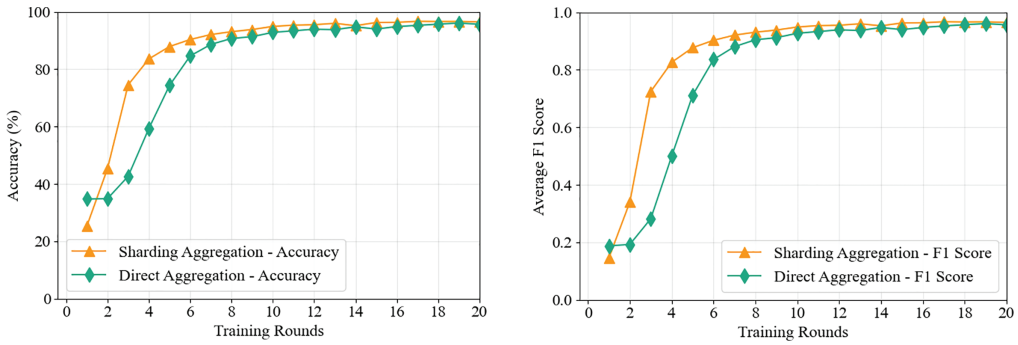

Figure 2: Comparison of average accuracy among participants (**a**) and Comparison of Average F1 score among participants (**b**)

Table 1 presents the model training performance of the proposed trustworthy federated learning sharding aggregation scheme under different participant scales. The data indicates that as the number of contributors in federated learning increases, the overall heterogeneity among datasets also increases. This leads to a decline in both accuracy and F1-score.

As the number of participants increases, the advantage of the sharding aggregation scheme over the direct aggregation scheme becomes more significant. Specifically, when the number of participants increases to 20 and 40, the sharding aggregation scheme improves performance by 1.77% and 2.33%, respectively.

This aligns with the expectation that the sharding scheme is more suitable for scenarios with a larger number of participants. The experimental results verify that the proposed trustworthy federated learning sharding aggregation scheme enhances model training performance in multi-participant heterogeneous data scenarios. Moreover, as the network scale expands, the advantages of this scheme become more pronounced.

Table 1: Performance comparison for different numbers of participants

Numbers of Participants	10 Participants		20 Participants		40 Participants	
Scheme	Accuracy	F1 Score	Accuracy	F1 Score	Accuracy	F1 Score
Segmentation aggregation	96.47%	0.9645	95.18%	0.9522	87.82%	0.8671
Direct aggregation	95.63%	0.9559	93.85%	0.9371	85.05%	0.8467

To verify the ability of the trustworthy federated learning model aggregation process to resist malicious participants, we conducted experiments under a scenario with 20 participants. We tested different numbers of malicious nodes and various poisoning data proportions per node. The improved model aggregation process was evaluated for its effectiveness in maintaining model performance.

Fig. 3a,b illustrates a scenario with four malicious nodes, where each node contains 50% poisoned data. Compared to the direct aggregation model, the trustworthy federated learning aggregation process incorporating a malicious node detection mechanism exhibits significant advantages in both convergence speed and model performance. Specifically, using accuracy and F1 score as evaluation metrics on the test set, the direct aggregation scheme suffers from slower convergence and lower accuracy due to the influence of low-quality malicious participant models. However, after applying the malicious node detection mechanism, the model performance and convergence speed are greatly improved, approaching the original accuracy observed in the non-poisoned scenario (Fig. 2a,b).

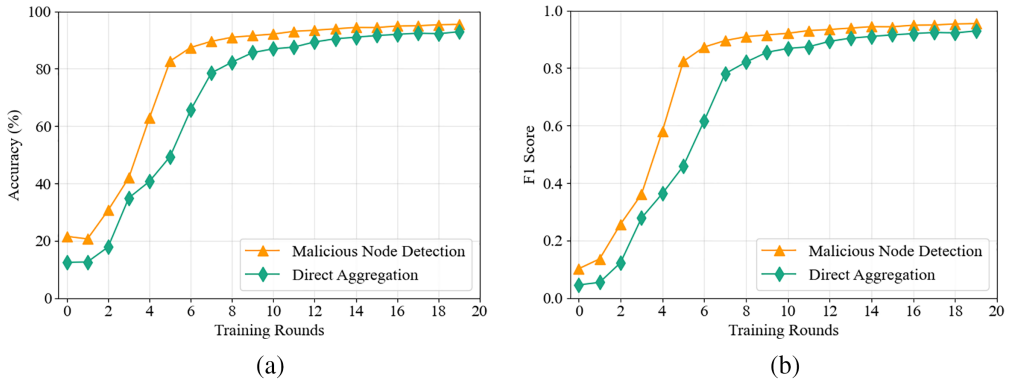

Figure 3: Model accuracy comparison (**a**) and F1 score comparison (**b**)

Table 2 displays the model accuracy on the test set under different numbers of malicious nodes and varying poisoning proportions. When the proportion of malicious nodes is moderate, the detection mechanism is more effective. When the proportion is low, the impact of data quality on the global model is limited, making accuracy improvements less noticeable. However, when the proportion of malicious nodes reaches nearly 50%, the detection mechanism struggles to filter out all malicious participants, leading to poor model performance.

To demonstrate the effectiveness of the proposed ShardFed algorithm under non-IID data and interference conditions, we compared it with two state-of-the-art federated learning algorithms: FedProx [29] and FedCurv [30]. Accuracy was used as the evaluation metric to assess the performance of different training methods under non-IID data distributions and with 20% noisy training data.

Table 2: Performance comparison for different numbers of malicious nodes

Number of malicious nodes	2		4		8	
Poisoning data ratio	**20%**	**50%**	**20%**	**50%**	**20%**	**50%**
Improved scheme	94.94%	94.41%	94.50%	95.47%	86.91%	83.19%
Unimproved scheme	94.21%	93.51%	93.14%	92.88%	86.24%	73.70%

Fig. 4a,b presents a comparison of accuracy across different federated learning algorithms. The results indicate that compared to FedProx and FedCurv, both of which optimize for non-IID data distributions, ShardFed significantly improves convergence speed by leveraging sharded global model information. The experiment also confirms that introducing noisy training data simulates low-quality datasets, and the proposed malicious node detection mechanism greatly enhances resistance against low-quality model contributors, improving adaptability across multiple scenarios.

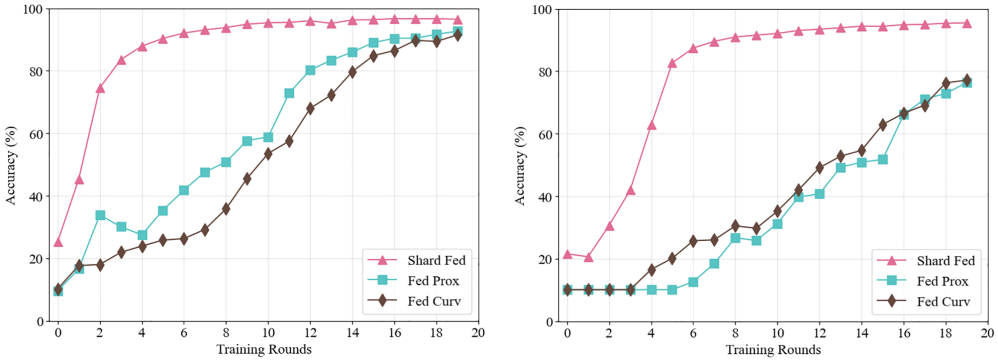

Figure 4: Comparison of accuracy (**a**) without poisoned data, and (**b**) with poisoned data

To validate the advantages of our proposed trustworthy federated learning sharding aggregation scheme with malicious node detection, we assessed its performance against standard machine learning models like K-Nearest Neighbors (KNN), Support Vector Machines (SVM), and Decision Trees. As shown in Table 3, it performs well in various scenarios, including NSL-KDD and BoT-IoT. Initially designed for data poisoning, the method is also effective for model poisoning, using global model bias to detect malicious participants, as detailed in Table 4.

Table 3: Multi-dataset performance evaluation

Dataset	Clean Data		10% Poisoned		20% Poisoned	
	Accuracy	F1 Score	Accuracy	F1 Score	Accuracy	F1 Score
NSL-KDD	96.47%	0.9645	95.47%	0.9539	94.12%	0.9401
BoT-IoT	91.85%	0.9182	90.34%	0.9029	88.76%	0.8871

Table 4: Performance comparison for data poisoning and model poisoning attacks

Attack Type	2 Malicious Nodes		4 Malicious Nodes		8 Malicious Nodes	
	Accuracy	F1 Score	Accuracy	F1 Score	Accuracy	F1 Score
Model poisoning (20%)	0.9387	0.9170	0.9312	0.8940	0.8433	0.8210
Model poisoning (50%)	0.9214	0.9080	0.9168	0.8820	0.7987	0.8170

The results demonstrate that federated learning models, due to their higher complexity, perform better on complex problems compared to Decision Trees and SVM. Compared to KNN, the proposed scheme performs slightly better in non-interference conditions. However, when interference data is introduced, the proposed malicious node detection mechanism enhances robustness significantly, making it more suitable for complex and dynamic federated learning scenarios.

4 Conclusion and Future Work

This paper proposes a shard-based federated learning framework for network security assessment, integrating anomaly-aware malicious node detection and multi-objective shard formation via statistical feature extraction and spectral clustering. The ShardFed algorithm combines sharding with residual-model anomaly detection to effectively filter poisoned models. Experiments on the NSL-KDD dataset demonstrate improved accuracy, convergence, and robustness over mainstream approaches such as FedProx and FedCurv, especially under data poisoning attacks, and show good scalability to larger participant numbers.

Despite the promising results, the limitations include testing on limited datasets and focusing on mainly data poisoning attacks. Future research will validate using modern datasets, enhance defenses against advanced threats like model poisoning and Sybil attacks, and explore scalability, especially focusing on reducing the computational cost of the initial sharding process. Additionally, the trust-scoring system might unfairly exclude participants, suggesting a rejoin mechanism could be added.

Acknowledgement: This work received the support from cooperative project with Beijing University of Posts and Telecommunications.

Funding Statement: This work is supported by State Grid Hebei Electric Power Co., Ltd. Science and Technology Project, Research on Security Protection of Power Services Carried by 4G/5G Networks (Grant No. KJ2024-127).

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Lincong Zhao and Liandong Chen; Methodology, Peipei Shen and Fanqin Zhou; Simulation design and implementation, Peipei Shen and Zizhou Liu; Validation and analysis, Zizhou Liu and Chengzhu Li; Writing—original draft preparation, Liandong Chen, Peipei Shen, and Zizhou Liu; Writing—review and editing, Lincong Zhao and Fanqin Zhou; Project

administration and funding acquisition, Lincong Zhao. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The datasets analyzed during the current study are available in the NSL-KDD dataset repository (https://www.unb.ca/cic/datasets/nsl.html) and the BoT-IoT dataset repository (https://research.unsw.edu.au/projects/bot-iot-dataset) (accessed on 4 August 2025).

Ethics Approval: Not applicable. The study did not include human or animal subjects.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Hu K, Gong S, Zhang Q, Seng C, Xia M, Jiang S. An overview of implementing security and privacy in federated learning. Artif Intell Rev. 2024;57(8):204. doi:10.1007/s10462-024-10846-8.
2. Mohy-eddine M, Guezzaz A, Benkirane S, Azrour M. Malicious detection model with artificial neural network in IoT-based smart farming security. Cluster Comput. 2024;27(6):7307–22. doi:10.1007/s10586-024-04334-5.
3. Wang X, Shankar A, Li K, Parameshachari B, Lv J. Blockchain-enabled decentralized edge intelligence for trustworthy 6G consumer electronics. IEEE Transact Cons Elect. 2024;70(1):1214–25. doi:10.1109/tce.2024.3371501.
4. Lewis C, Varadharajan V, Noman N. Attacks against federated learning defense systems and their mitigation. J Mach Learn Res. 2023;24(30):1–50.
5. Imteaj A, Mamun Ahmed K, Thakker U, Wang S, Li J. Amini MH. In: Razavi-Far R, Wang B, Taylor ME, Yang Q, editors. Federated learning for resource-constrained IoT devices: panoramas and state of the art. Cham, Switzerland: Springer International Publishing; 2023. p. 7–27 doi:10.1007/978-3-031-11748-0_2.
6. Quy VK, Nguyen DC, Van Anh D, Quy NM. Federated learning for green and sustainable 6G IIoT applications. Inter Things. 2024;25(2):101061. doi:10.1016/j.iot.2024.101061.
7. Tao Y, Cui S, Xu W, Yin H, Yu D, Liang W, et al. Byzantine-resilient federated learning at edge. IEEE Transact Comput. 2023;72(9):2600–14. doi:10.1109/tc.2023.3257510.
8. Wang H, Xu Z, Zhang Y, Wang Y. Adaptive layered-trust robust defense mechanism for personalized federated learning. In: ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Hyderabad, India: IEEE; 2025. p. 1–5.
9. AzharShokoufeh D, DerakhshanFard N, RashidJafari F, Ghaffari A. Optimizing IoT data collection through federated learning and periodic scheduling. Knowl Based Syst. 2025;317(3):113526. doi:10.1016/j.knosys.2025.113526.
10. Yaacoub JPA, Noura HN, Salman O. Security of federated learning with IoT systems: issues, limitations, challenges, and solutions. Int Things Cyber-Phys Syst. 2023;3(3):155–79. doi:10.1016/j.iotcps.2023.04.001.
11. Kalapaaking AP, Khalil I, Rahman MS, Atiquzzaman M, Yi X, Almashor M. Blockchain-based federated learning with secure aggregation in trusted execution environment for internet-of-things. IEEE Transact Indust Inform. 2022;19(2):1703–14. doi:10.1109/tii.2022.3170348.
12. Zhou Y, Pang X, Wang Z, Hu J, Sun P, Ren K. Towards efficient asynchronous federated learning in heterogeneous edge environments. In: IEEE INFOCOM 2024-IEEE Conference on Computer Communications. Vancouver, BC, Canada: IEEE; 2024. p. 2448–57.
13. Ribero M, Vikalo H. Reducing communication in federated learning via efficient client sampling. Pattern Recognit. 2024;148:110122. doi:10.1016/j.patcog.2023.110122.
14. Oyewole GJ, Thopil GA. Data clustering: application and trends. Artif Intell Rev. 2023;56(7):6439–75. doi:10.1007/s10462-022-10325-y.
15. Pitafi S, Anwar T, Sharif Z. A taxonomy of machine learning clustering algorithms, challenges, and future realms. Appl Sci. 2023;13(6):3529. doi:10.3390/app13063529.
16. Yurdem B, Kuzlu M, Gullu MK, Catak FO, Tabassum M. Federated learning: overview, strategies, applications, tools and future directions. Heliyon. 2024;10(19):e38137. doi:10.1016/j.heliyon.2024.e38137.

17. Zeng Y, Mu Y, Yuan J, Teng S, Zhang J, Wan J, et al. Adaptive federated learning with non-IID data. Comput J. 2023;66(11):2758–72. doi:10.1093/comjnl/bxac118.
18. Zhang W, Zhou T, Lu Q, Yuan Y, Tolba A, Said W. FedSL: a communication-efficient federated learning with split layer aggregation. IEEE Inter Things J. 2024;11(9):15587–601. doi:10.1109/jiot.2024.3350241.
19. Guerraoui R, Gupta N, Pinot R. Byzantine machine learning: a primer. ACM Comput Surv. 2024;56(7):1–39. doi:10.1145/3616537.
20. Song R, Liu D, Chen DZ, Festag A, Trinitis C, Schulz M, et al. Federated learning via decentralized dataset distillation in resource-constrained edge environments. In: 2023 International Joint Conference on Neural Networks (IJCNN). Gold Coast, Australia: IEEE; 2023. p. 1–10.
21. Lewis-Pye A, Roughgarden T. Byzantine generals in the permissionless setting. In: International Conference on Financial Cryptography and Data Security. Cham, Switzerland: Springer; 2023. p. 21–37.
22. Abbas S, Ahmad B, Benchohra M, Salim A. Fractional difference, differential equations, and inclusions: analysis and stability. San Francisco, CA, USA: Morgan Kaufmann Publishers, Elsevier; 2024. [cited 2025 Aug 4]. Avilable from: https://abdelkrim-salim.owlstown.net/publications/22287-fractional-difference-differential-equations-and-inclusions-analysis-and-stability.
23. Qi J, Guan Y. Practical Byzantine fault tolerance consensus based on comprehensive reputation. Peer-to-Peer Networ Applicat. 2023;16(1):420–30. doi:10.1007/s12083-022-01408-2.
24. Civit P, Gilbert S, Guerraoui R, Komatovic J, Paramonov A, Vidigueira M. All byzantine agreement problems are expensive. In: Proceedings of the 43rd ACM Symposium on Principles of Distributed Computing; Nantes, France; 2024. p. 157–69.
25. Linde W. Probability theory: a first course in probability theory and statistics. Berlin, Germany: Walter de Gruyter GmbH & Co KG; 2024. [cited 2025 Aug 4]. Avilable from: https://speciation.net/Database/Companies/Walter-de-Gruyter-GmbH-amp-Co-KG/-;i703a-1.
26. Tavallaee M, Bagheri E, Lu W, Ghorbani AA. A detailed analysis of the KDD CUP 99 data set. In: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications. Ottawa, ON, Canada; 2009. p. 1–6.
27. Buiya MR, Laskar A, Islam MR, Sawalmeh S, Roy M, Roy R, et al. Detecting IoT cyberattacks: advanced machine learning models for enhanced security in network traffic. J Comput Sci Technol Stud. 2024;6(4):142–52. doi:10.32996/jcsts.2024.6.4.16.
28. Chen D, Hu J, Tan VJ, Wei X, Wu E. Elastic aggregation for federated optimization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; Vancouver, BC, Canada; 2023. p. 12187–97.
29. Li X, Huang K, Yang W, Wang S, Zhang Z. On the convergence of fedavg on non-iid data. arXiv:1907.02189. 2019.
30. Shoham N, Avidor T, Keren A, Israel N, Benditkis D, Mor-Yosef L, et al. Overcoming forgetting in federated learning on non-iid data. arXiv:1910.07796. 2019.