Advances in Computer, Signals and Systems (2025)
Clausius Scientific Press, Canada

DOI: 10.23977/acss.2025.090113
ISSN 2371-8838 Vol. 9 Num. 1

Privacy-Preserving Federated Learning Secure Aggregation Strategy Based on Permissioned Blockchain

Wang Chenyi[1], Wang Zhe[1], He Jiekai[1]

[1]*School of Artificial Intelligence, Guangxi Minzu University, Nanning, 530006, China*

Keywords: Blockchain; federated learning; privacy protection; secure aggregation

Abstract: In order to solve the problems of privacy-preserving federated learning in which the poisoning behaviour of client nodes as well as the malicious aggregation behaviour of aggregation server nodes lead to the failure of model training, a secure aggregation privacy-preserving federated learning strategy based on permissioned blockchain is proposed. In order to solve the problem of malicious behavior of aggregation server nodes, a trusted aggregation server node selection algorithm is designed to cancel the right of nodes with malicious aggregation behavior to participate in the aggregation server node election. Each node establishes a reputation value, proposes a reward and punishment algorithm based on the reputation value, and establishes a threshold, and nodes with a reputation value lower than the threshold will be refused to participate in the federated learning process to reduce the threat of client node poisoning attack behaviour on model learning.The experimental results show that the scheme is able to ensure secure model aggregation and achieve high model correctness in the presence of malicious aggregation behaviour at 50% of the nodes and poisoning attacks at 40% of the nodes.

1. Introduction

Privacy Protection Federal learning (Privacy-Preserving Federated Learning, PPFL) is an advanced FL framework that uses homomorphic encryption, differential privacy, secure multi-party computing and other technical means on the basis of Federated Learning (FL) to enhance the privacy protection in the model training process and reduce the risk of user data leakage. It is an advanced FL framework integrating privacy protection mechanism[1~5]. However, during PPFL training, the toxic gradient has higher concealment after processing by privacy protection mechanism, and is more vulnerable to poisoning attacks by client nodes compared with traditional FL[6~7]. Moreover, the traditional FL method used to resist poisoning attacks requires access to the explicit local gradients uploaded by the clients, in accordance with techniques such as Krum[8-9], Trim-mean[10], Bulyan[11], and Auror[12]. However, this practice contradicts the original intention of data privacy security in the training process of the PPFL protection model.

In order to effectively defend against the poisoning attacks of client nodes in PPFL, Liu et al[13]A privacy-enhanced federated learning framework (PEEL) is proposed, with homomorphic encryption as the underlying technology, automatically adjusts the gradient weight of the corresponding nodes by using the median of the coordinate direction, and then measures the similarity between the gradients through the Pearson correlation coefficient. Ma class[14]Design a

double trap door privacy protection strategy of homomorphic encryption, the private key is divided into two respectively sent to two central server, through the communication of two central server, calculate the client node upload local gradient and the previous round of aggregation between the global gradient similarity and then filter out the toxic gradient, can effectively resist the client node poisoning attack, and a separate server cannot access the encrypted data, enhance the privacy of the client upload local gradient security. Zhou class[15]A differential private federal learning model for poisoning attacks is established. Firstly, a weight-based algorithm is designed to detect the parameters of the terminal upload, and then the differential privacy technology is used to protect the privacy of the terminal upload data and the model. Li class[16]A multi-tentacle federated learning framework is proposed, which identifies poisoning attacks through efficient poisoning detection algorithm based on tentacles distribution, and then reduces the influence of poisoning data through random tentacle data exchange protocol. All exchanged data use differential privacy processing to ensure the privacy security of data.

However, the above scheme fails to solve the problem of participant authentication, and the global model is aggregated by the central server, once the central server failure will lead to the failure of the whole learning process[17~18]. Blockchain is characterized by decentralization, immutability and transparency, which provides a new idea for researchers to solve this problem[19~20]. Miao class[21]Designed a privacy protection based on block chain Byzantine robust federal learning scheme, the scheme to clean data set in Solver server, using completely homomorphic encryption to protect data privacy, through the cosine similarity contrast local gradient and clean gradient to eliminate toxic gradient, and then communicate with Verifier server through block chain aggregation. However, if the Solver or Verifier server fails, the entire FL cluster will not work properly, and this method will not completely solve the single point of failure problem. The way of electing new aggregated nodes at each iteration can effectively solve this problem, Shayan et al[22]Proposed a completely decentralized point-to-peer federal learning method, first through the consistency hash function (Consistent Hashing Function, CHF) to randomly select the noise, validation and aggregator of this iteration, each client node using the differential private noise to protect the uploaded data, the verifier using the uploaded protection data, then through the validation data using Shamir secret sharing in the aggregator global model aggregation. However, due to the transparency of the blockchain and the differential private noise provided by the elected noise, if the noise has malicious behavior, it will pose a threat to the privacy security of the client data. Therefore, while protecting the data privacy security and resisting the client poisoning attack in the process of PPFL, it is still a key problem to select the new trusted aggregation server in the aggregation server to ensure the normal operation of the whole model system.

Based on the above analysis, this paper proposes a Privacy-preserving federated learning secure aggregation strategy based on permissioned blockchain, PBSPFL) based on licensing blockchain based on its shortcomings. Major contributions of our work are as follows:

The federal learning model framework of secure aggregation privacy protection based on licensing blockchain is established to realize the selection of trusted aggregation server nodes in the open and transparent transaction environment of trusted blockchain and reduce the impact of toxic gradient on the accuracy of the global model.

To tackle this issue, we propose a node selection algorithm for the trusted aggregation server. This algorithm incorporates the consistency hash function as a key technology. Specifically, we design a trusted aggregation node selection algorithm and establish a consistency hash ring that is exclusively utilized by the trusted aggregation server nodes.

We propose a reward and punishment algorithm based on credit value, which is designed to penalize nodes that exhibit malicious behavior. The nodes with credit value below the threshold will

be refused to participate in the federated learning process.

The experiment show that high model accuracy in FASHION-MNIST and CIFAR-10 data sets

2. The system model and the problem description

In order to achieve more secure and reliable FL model training process, this model introduces the license block chain, consistency hash function and threshold all homomorphic encryption technology [24] build based on license block chain security aggregation privacy protection federal learning model framework, analyzes the threat model under the security assumption, and gives the design goal of this paper. The specific design is shown as follows.

2.1 Model design

License blockchain is mainly responsible for the registration and authentication of devices, the election of miner nodes and aggregation server nodes, and the execution of rewards and punishments. The publisher of the task, as the authentication server, is responsible for initializing the blockchain network, initializing the FL model and distributing the data set, the mine works (Proof ofWork, PoW) consensus mechanism, and the node selection algorithm based on the consistency hash function, receives the local gradients uploaded by the client nodes and gates the global model. Among them, the miners do not participate in the election of the aggregated server nodes in the current round of FL iteration process, but they can train the local model and upload the local gradient. Moreover, the consistency hash ring generated by the consistency hash function is also maintained by the miners elected each time. The specific process is shown in Figure 1.

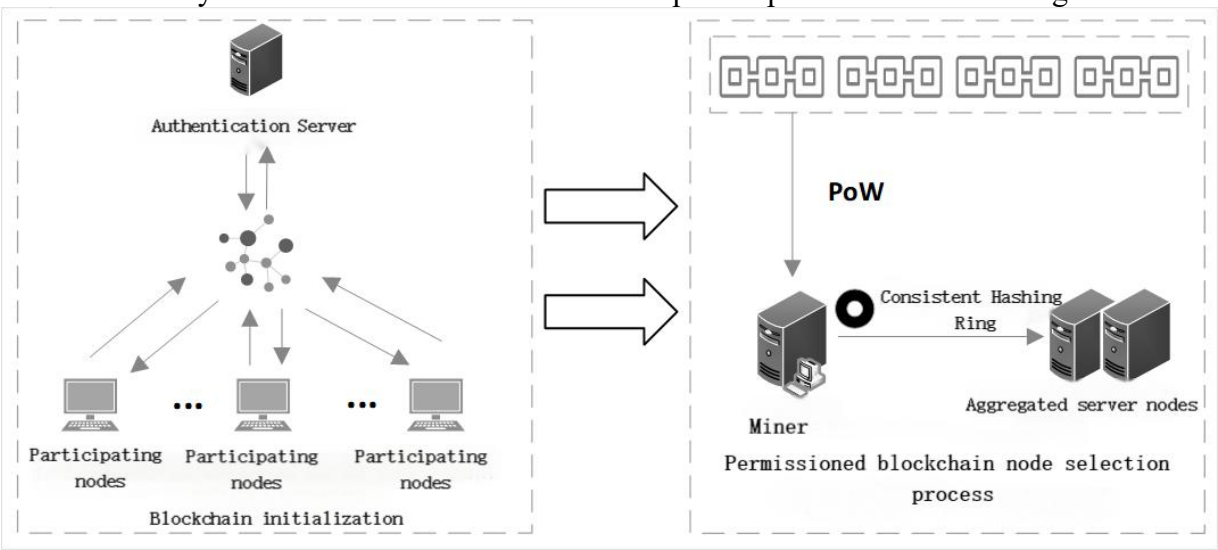

Figure 1 Blockchain initialization and node selection process

The consistency hash ring is the result of the participating nodes mapping to the logical loop structure by CHF. It is initially established by the authentication server node and maintained by the elected miners during each round of FL model iteration, where the number of maps of each node is shown in formula (1). Hn_i

$$Hn_i = \left\lfloor N_{hash} \frac{R_i}{\sum_{j=0}^{m} R_j} \right\rfloor \tag{1}$$

This is the total number of nodes mapping on the consistent hash ring. This paper is the credit value of the first node participating in the aggregation server campaign, representing the sum of the credit value of the nodes participating in the aggregation server campaign.

$$N_{hash}\ N_{hash} = 100\ R_i\ i\ \sum_{j=0}^{m} R_j\ m$$

The higher the credit value of a node in the consistency hash ring, the greater the likelihood of it being chosen as the aggregate server node. The first aggregation server node outputs the hash node as the result of the input value, which then becomes the input value for the consistency hash function, thus determining the second aggregate server node. The specific process is shown in Figure 2.

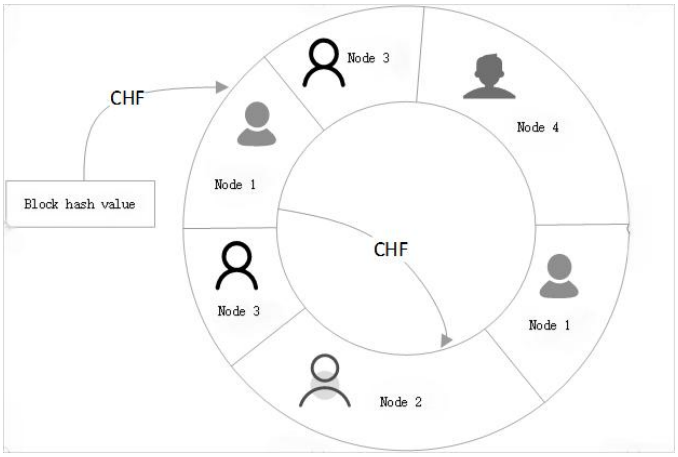

Figure 2 Selection of the aggregate server nodes

The consistency hash function is shown in Equation (2).

$$CHF = H(Key)\ \bmod\ M_{Hash} \tag{2}$$

Where is the hash value of the input value, is a fixed value, indicating that the consistency hash function will map the hash value over the integer of the range, set in this paper.

$$H(Key)\ Key\ M_{Hash}\ H(Key)\ [0, M_{Hash}]\ M_{Hash} = 2^{32}$$

Threshold completely homomorphic encryption (Threshold Fully Homomorphic Encryption, ThFHE) is a combination of completely homomorphism encryption and threshold encryption characteristics of encryption scheme, ciphertext can perform without the decryption of addition and multiplication operation, private key is divided into multiple shares to different parties, only the number of participants reach the threshold to data decryption, with the private key share participants generate part of decryption ciphertext, and then put the part of decryption ciphertext combination together to obtain decryption value. Article [24] constructs a ThFHE scheme that can be generalized to any lattice encryption (Learning With Errors, LWE), as detailed in references [23], [24]. This scheme uses (T, T) distributed decomposition algorithm to decompose polynomial private keys into parts, and nodes with some private keys must cooperate together to decrypt. It is set in this paper. During decryption, the node with a partial private key partially decrypts the ciphertext, and then combines the partially decrypted ciphertext to obtain the clear text, where the encrypted message is shown in formula

$$sk\ T\ T\ T = 2\ ct = (a,b)\ [x]_i\ m\ b\ b = a \bullet sk + m + e \tag{3}$$

Where is the public-key matrix, which is the noise. Partial decryption of the ciphertext is shown in equation (4). $a\ e\ ct$

$$[x]_i = a \bullet sk_i + e^i \tag{4}$$

This is the partial private key owned by the node involved in the decryption, and the noise added for the node. All partially decrypted ciphertexts are combined, as shown in Equation

$$sk_i\ i \in T\ e^i\ \delta = b - \sum_{i=1}^{T} [x]_i \tag{5}$$

The result is essentially a more precise plaintext.

$$\delta\ (m + e - \sum_{i=1}^{T} e^i)\ \delta\ m$$

2.2 Model framework

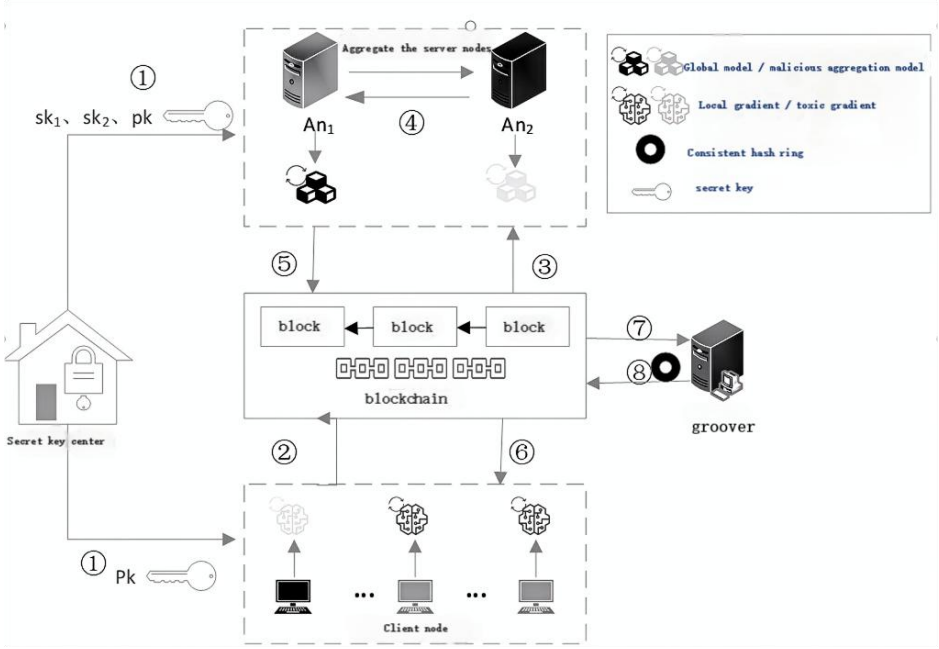

Figure 3 System model framework

As shown in Figure 3, the system model framework is composed of a secret key centerand a blockchain, in which the blockchain consists of aggregated server nodes, client nodes and miners. Before the start of each round of FL training process, the blocks generated after the last round of FL training will be downloaded to the local area, and new miners and server nodes will be elected. The functions of each module of the system model are introduced as follows.

Secret key center. The function of this module is to distribute the secret key to the aggregation server nodes and client nodes during each round of FL training, and divide the private key into two parts and distribute it to two different aggregation server nodes (step ①).

Client nodes. The local model is trained and the local gradient is uploaded. The nodes with toxic attack behavior will transmit their own toxic gradient (step ②), and the accuracy of the global model generated by the aggregate server nodes is verified (step ⑥).

Aggregation of the server nodes. This part is two nodes selected by node selection algorithm as aggregation server nodes, in the block chain to collect client node upload local gradient (step ③),

and in the case of protecting local gradient privacy, against poisoning attack safe aggregation, malicious aggregation behavior nodes will make some behavior hinder the generation of the global model or reduce the accuracy of the global model (step ④), and finally upload their own aggregate global model (step ⑤).

Miners. This part is selected by the PoW consensus protocol, which is responsible for collecting transaction information between various nodes in the blockchain and recording it in the ledger, while executing rewards and punishments for the node reputation value and maintaining a consistent hash ring (Step ⑦). When this round of FL training process is completed, the recorded ledger, the updated node reputation value and the consistency hash ring are uploaded to the blockchain in the form of a block (step ⑧).

3. Security assumptions

Suppose 1 this paper assumes that each node of the blockchain is semi-honest and can follow the FL training rules, but will try to crack the sensitive data encrypted by the other nodes.

Hypothesis 2 This paper assumes that nodes with poisoning attack behavior can obtain the aggregated global model and upload their own malicious gradient, but cannot observe or access the local model updates of other clients.

Hypothesis 3 this paper assumes that nodes with malicious aggregation behavior will not collude with client nodes in the open and transparent transaction environment of the license blockchain, but will give a global model with low accuracy in the model aggregation stage to slow down the convergence rate of the global model.

3.1 Model threat

The key center is a trusted third party, the two aggregate server nodes An1And An2It is selected by the selection algorithm, which may have malicious aggregation behavior, and also tries to infer the sensitive data of the other nodes, but it will not collude with each other in a transparent and open trading environment. Nodes with toxic attacks will carefully upload their own toxic gradient, reducing the accuracy of the global model. The potential threats posed by the appeal entity are as follows:

Popoison attack threat. When a node with poisoning attack behavior acts as the client nodes, it will carefully upload its own toxic gradient, and the goal is to reduce the accuracy of the global model without being detected by the aggregation server node.

Data privacy leakage threat. The local gradient uploaded by the client node is a mapping of the local data of the node. If the client node directly uploads the plaintext gradient, the attacker can infer or obtain the original information of the honest client to some extent, resulting to the client data leakage.

Malicious aggregate threat. When the nodes with malicious aggregation behavior are selected as the aggregation server nodes, a global model with low accuracy will be given in the model aggregation stage, slowing down the convergence rate of the global model.

3.2 Model design

The goal of this scheme is to protect the privacy of data, resist the client poisoning attack, when the aggregation server node failure or malicious aggregation behavior, the new aggregation server can be selected in time so that the model system can run stably and give a high-quality global model. The specific scheme is as follows:

Election of the trusted aggregation server node. The trusted aggregation server node selection algorithm is designed to cancel the power of the node campaign for the aggregation server nodes with malicious aggregation behavior, which effectively alleviates the threat of malicious aggregation.

Safety of the model. The local gradient uploaded by the client and the global gradient at the last iteration are used for cosine similarity verification, and the local model nodes that fail to pass the verification are penalized using the reward and punishment algorithm based on credit value, so as to reduce the impact of poisoning attacks on model learning.

Data privacy protection. The client node encrypts the trained local model based on threshold all-homomorphic encryption, and the key center divides the private key into two parts and sends it to the two aggregation server nodes respectively to ensure the privacy of the data during the aggregation process.

4. PBSPFL policy

This section mainly tells about the construction of PBSPFL strategy, which is divided into two parts: model training process and algorithm implementation. Symbol descriptions are shown in Table 1

Table 1 Symbolic description

symbol	description
g_k^i	Local gradient of the first client node in the first iteration k i
$\tilde{g}_k^i$	g_k^i Results after normalization
w_k^i	Local model of the first client node in the first iteration k i
D_i	Local dataset of the first client node i
$[\![\tilde{g}_k^i]\!]$	$\tilde{g}_k^i$ Results after fully homomorphic encryption
W_k	Global models generated by the first round of iterative aggregation k
g_k	Global gradient generated by iterative aggregation in the first round k
η	Model learning rate
n_{honest}	Number of local gradients was verified by safety
α	The factor to control the intensity of rewards and punishments
N	Number of nodes involved in the model training

4.1 Model training process

PBSPFL Training process is mainly divided into local model training, local gradient upload, security validation, global model aggregation and global model correctness validation five parts, the standardization of safety verification, cosine similarity validation and global model aggregation of global gradient using ciphertext calculation, and use smart contract for ciphertext calculation is open and transparent, specific introduction as follows.

4.2 Local model training.

First, the authentication server node completes the authentication registration of the participating nodes, and issues the data set and the FL initialization model after completing the blockchain

initialization. Secondly, the miner nodes in the FL iteration process are selected, and the aggregation server nodes An in the FL iteration process are selected by using the consistent hash ring maintained by the miners1And An2. Finally, the client node trains the local model by stochastic descent using stochastic gradient descent, and standardizes the trained local gradient and uploads the normalized local gradient. The local gradient of the first client node during the first FL iteration and the standardized local gradient are shown in Equation (6). $k\ i\ g_k^i\ \tilde{g}_k^i$

$$\begin{aligned} g_k^i &= \nabla L(w_k^i, D_i) \\ \tilde{g}_k^i &= \frac{g_k^i}{\left\| g_k^i \right\|} \end{aligned} \tag{6}$$

Where is the local model of the first client node during the first round of FL iteration, which is the local data set of that node, which is the derivative of the empirical loss function. $w_k^i\ k\ i\ D_i$ $\nabla L(w_k^i, D_i)\ L(w_k^i, D_i)$

4.3 Local gradient upload

The standardized local gradient of client nodes can only be uploaded to the blockchain after fully homomorphic encryption and blockchain authentication encryption, aiming to ensure privacy in the process of global model aggregation. It is encrypted by the public key pair in ThFHE, and the client node is signed through the local private key pair, which is used to verify the authenticity and integrity of the uploaded local gradient to ensure that the submitted information is not maliciously tampered with, as shown in formula

$$\tilde{g}_k^i\ \tilde{g}_k^i\ pk\ \tilde{g}_k^i\ [\![\tilde{g}_k^i]\!]\ [\![\tilde{g}_k^i]\!]\ ([\![\tilde{g}_k^i]\!], E_i) = hash_{key_k}([\![\tilde{g}_k^i]\!]) \tag{7}$$

4.4 Safety verification

After receiving the standardized local gradient uploaded by the client node, the aggregation server node will first conduct the standardization confirmation. If the first client node standardizes the verification result, the information uploaded by the node is received, otherwise it will be abandoned. The standardized verification is as shown in formula

$$\begin{aligned} [\![\lambda_i]\!] &= [\![\tilde{g}_k^i]\!] \odot [\![\tilde{g}_k^i]\!] \\ i\ \lambda_i = 1 \quad &= [\![x_k^{i1}]\!]^2 + [\![x_k^{i2}]\!]^2 + \cdots + [\![x_k^{im}]\!]^2 \end{aligned} \tag{8}$$

The aggregation server node determines the local gradient that is the by validation. The cosine similarity was compared by comparing the local gradient with the global model gradient generated during the previous round of FL iteration. First, the standardized processing is encrypted, and then the cosine similarity is calculated. After the two aggregation servers cooperate with the decryption, the local gradient is removed. The cosine similarity is calculated as shown in formula (9).

$$[\![\tilde{g}_k^i]\!]\ g_{k-1}\ g_{k-1}\ [\![\tilde{g}_{k-1}]\!]\ [\![\tilde{g}_{k-1}]\!]\ [\![\tilde{g}_k^i]\!]\ [\![\cos(\tilde{g}_k^i, \tilde{g}_{k-1})]\!]\ \cos(\tilde{g}_k^i, \tilde{g}_{k-1}) \le 0\ [\![\cos(\tilde{g}_k^i, \tilde{g}_{k-1})]\!] = [\![\tilde{g}_k^i]\!] \odot [\![\tilde{g}_{k-1}]\!] \tag{9}$$

4.5 Aggregation of the global models

The aggregate server node performs a federated average aggregation of the verified local gradient, as shown in equation (10).

$$\begin{aligned} [\![g_k]\!] &= \frac{1}{n_{honest}} \sum_{i=1}^{n} [\![\tilde{g}_k^i]\!] \\ W_k &= W_{k-1} - \eta g_k \end{aligned} \tag{10}$$

Where is the number of local gradients verified by safety, which is the learning rate. n_{honest} η

4.6 Global model correctness verification.

Both the two nodes of the aggregation server selected in this paper will collect the local gradient uploaded by the client and go to step 3) ~4), which will generate the corresponding standardized verification set, cosine similarity set and global model, and sign the results with their own private key and upload them to the blockchain in the way of transaction. The uploaded transaction is shown in formula

$$\begin{aligned} \lambda &= (\lambda_1, \lambda_2 \cdots \lambda_n) \cos W_k \\ (C_k^{An_i}, E_{An_i}) &= hash_{key_{An_i}} (C_k^{An_i}) \\ C_k^{An_i} &= (\lambda_k^{An_i} = (\lambda_1, \lambda_2, \cdots, \lambda_n), \\ \cos_k^{An_i} &= (\cos(\tilde{g}_k^1, \tilde{g}_{k-1}), \cdots, \cos(\tilde{g}_k^n, \tilde{g}_{k-1})), \\ W_k^{An_i}) & \end{aligned} \tag{11}$$

After receiving the transaction uploaded by the server node, the miner node first compares the standardized validation set and the cosine similarity set in the two information and the global model. If both are trusted aggregation server nodes, the values of the corresponding part should be equal. If there are unequal values, the miner node will verify the accuracy of the global model generated by the two nodes by comparing the test set and compare the test results with the accuracy of the global model in the previous FL iteration. If the difference is greater than the threshold, that is to say, the aggregation server node generated the global model accuracy is far lower than the global model in the process of FL iteration, will be judged as malicious aggregation behavior, cancel the power of the aggregation server node, if both nodes are judged to be malicious aggregation behavior, the global model for the global model in the process of FL iteration, continue to a new round of FL training. If the difference between the global model accuracy generated by the two nodes and the global model accuracy in the previous FL iteration is less than the threshold value, the final global model will be voted on by the client node.

$$\lambda = (\lambda_1, \lambda_2 \cdots \lambda_n) \cos W_k \; P(W_k^{An_i}) \; P(W_{k-1}) \; S_{\mathrm{A}} \; S_{\mathrm{A}}$$

5. Algorithm implementation

To address the potential for malicious aggregation behavior among selected aggregation server nodes and to ensure rewards and penalties for node reputation values, this paper proposes a trusted aggregation server node selection algorithm. It also introduces a credit-based rewards and penalties algorithm to guarantee that the nodes involved in the aggregation server campaign are highly

credible and to minimize the impact of poisoning attacks on the accuracy of the global aggregation model.

Based on the trusted block chain and consistency hash function idea, design a high confidence aggregation server node election algorithm, the algorithm of aggregation server node aggregation global model validation, when the aggregation server node detected malicious aggregation behavior, will cancel the node election the aggregation of server node power. We put the nodes with trusted aggregation behavior into the set, and obtain the number of nodes we can map to the consistent hash ring through our own credit value and consistency hash algorithm. The specific process is shown in Algorithm 1. M

Algorithm 1: a trusted aggregation server node selection algorithm
nput: The sum of messages sent after the two aggregate server nodes during the first FL iteration k $(C_k^{An_1}, E_{An1})$ $(C_k^{An_2}, E_{An_2})$ Output: The new aggregation server node, An1And An2
1. The miner node extracts, and of two transactions $\lambda_k^{An_i}$ $\cos_k^{An_i}$ $W_k^{An_i}$ 2. Judge whether the corresponding and of two nodes are equal λ cos W_k 3, if are all equal 4. The transaction uploaded by the aggregation server will be submitted to the client node for voting 5. end if 6, the unequal presence of if 7. Miner nodes verify the accuracy of the two nodes in generating the global model $P(W_k^{An_i})$ 8. for i=1 to 2 do 9. if $P(W_{k-1}) - P(W_k^{An_i}) > S_A$ 10, is the node with malicious aggregation behavior, removed from the collection An_i M 11. end for i 12. end if 13. If The global model accuracy of two nodes is verified by miners 14. The miner node uploads the transactions of both nodes to the client node for voting 15, else if Only one node is verified by the global model accuracy of several miners 16. The miner node only transmits the transactions that verify through the aggregation server node 17. The global model in the last round of FL iteration of se is the final global model of this round, and the miner node punishes the two aggregation servers through the reward and punishment algorithm 18. end if 19, if has an aggregate server node verified by the miner node 20. The client node votes on the.,, and generated by the aggregate server node λ cos W_k 21. If the node with the largest number of votes exceeds half of the nodes participating in the training client, the aggregation result of the node is taken as the final result of this round of FL iteration 22. end if 23. If, there is a trusted aggregation server node after the client node verification 24. Store the final results of the verified aggregate server nodes and the nodes with

malicious behavior in the collection, and store the honest nodes in the collection. λ cos *false* *true*

25. Miner nodes punish the nodes in the set through the reward and punishment algorithm based on the credit value, and reward the nodes in the set *false* *true*

26. The global model in the last FL iteration of ose is the final global model of this round, and the miner node punishes the two aggregation servers through the reward and punishment algorithm

27. Miner nodes update the consistency hash ring and upload the ledger as blocks to the blockchain

28. Select the new miners based on the PoW consensus, and select the new aggregation server node An based on the updated consistency hash ring1And An2

29. return An 1， An2

6. Theoretical analysis

This section will theoretically prove that the PBSPFL strategy designed in this paper is privacy.

6.1 Privacy analysis

To protect client node privacy during interactions, we reference the Threshold Fully Homomorphic Encryption (ThFHE) method [24]. We demonstrate that the PBSPFL model ensures the privacy of client node data, even when data is aggregated by server nodes.

Citation extension 1 In the ThFHE scheme, the private key is always distributed throughout the whole process. Each private key owner uses the share of the private key distributed to it to partially decrypt it, and sends the decryption results to the decryption user, without revealing its part of the private key.[25]

Theorem 1 In the case of malicious aggregation behavior, the communication between two aggregation server nodes does not reveal the privacy of the client node.

We prove that the exchange between aggregate server nodes only standardized verification results, cosine similarity verification results and global gradient aggregation results need to be decrypted. We perform the three encrypted operations in a smart contract way, making the three computing processes open and transparent, and ensuring that the results of the aggregation server decryption can only be one of the three results[26]. If the aggregate server node An1 has results to decrypt, An$[\![x]\!]$1 sends the ciphertext to another aggregation server node, An$[\![x]\!]$2. An2 uses its private key share of the ciphertext and sends the partially decrypted ciphertext to An$[\![x]\!]_2$1. An1 then uses its own private key share to partially decrypt the ciphertext, and the two partially decrypted ciphertexts are combined to obtain the plaintext.$[x]_1$

In this process, due to the open and transparent nature of blockchain and smart contracts, the aggregation server node can only decrypt the standardized verification results, the cosine similarity verification results and the global gradient aggregation results, so the aggregation server node will not obtain any other information of the client node. At the same time, the aggregation server node uploads the ciphertext that need to be decrypted, and finally the final result can be decrypted locally, so the aggregation server nodes cannot be colluded. Therefore, even if the aggregation server node has malicious aggregation behavior, the privacy of client data can be guaranteed[27].

7. Simulation experiment and performance analysis

This section first introduces the simulation experimental environment of model training, and

analyzes and evaluates the performance of the global model accuracy. Finally, it analyzes the node selection algorithm and the reward and punishment algorithm based on credibility value.

7.1 Experimental environment

The server environment configuration for all the experiments conducted in this paper is as follows: Intel (R) Core (TM) i7-10700 CPU 2.90 GHz, 1 6 GB of memory, Windows 10 operating system. In the experiment, VS Code was used to build Python development environment, Python programming language was used to realize blockchain functional modules, Pytorch 2.0.1 in Python 3.7 was used to build FL Framework, and convolutional neural network was used as the training model, and the hyperparameters such as the convolution, the size of the nucleus and the number of neurons in each module needed to be determined according to the experimental effect[28].

7.2 Experimental dataset

This paper evaluates the performance of the proposed PBSPFL model based on the FASHION-MNIST and CIFAR-10 datasets, respectively. This paper uses 2 0 devices for training the model, of which two devices will be selected as aggregate server nodes. In this paper, the entire FASHION-MNIST and CIFAR-10 training sets were randomly divided into 20 subsets of equal size and no sample overlap, and randomly distributed to the device to train the local model.

7.3 Model comparison scheme

In order to show the high availability of the aggregated global model of the proposed scheme, the traditional federated average algorithm (federated gradient average, FedAvg) is used as the benchmark scheme, and the correct rate of the global model is compared with the ShieldFL and Biscotti model schemes[29].

First, we compared the accuracy rate of the benchmark scheme FedAvg, the scheme PBSPFL and the global model on the FASHION-MNIST and CIFAR-10 datasets without the FL model training. The accuracy rate in this experiment was defined as the percentage of the number of correctly classified samples to the total number of samples. The experiment updates with 100 FL iterations, and the accuracy of the global model generated by the verified aggregation server is recorded. The 2 0 devices participating in each round of FL training will be divided into two aggregation server nodes and 1 7 client nodes, among which the miner node also performs the task of client nodes. The model accuracy pairs of the different schemes on the FASHION-MNIST and CIFAR-10 datasets are shown in Figure 4.

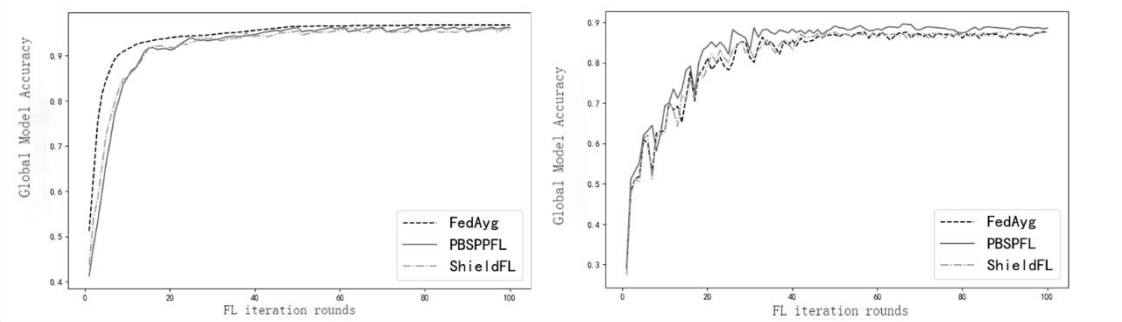

Figure 4 Nodes without malicious behavior participate in the FL model training

On the FASHION-MNIST dataset, after 100 rounds of FL iteration, FedAvg has the highest accuracy and similar accuracy between the PBSPFL and ShieldFL schemes, slightly inferior to the

benchmark model FedAvg. After 50 rounds of FL iteration, the accuracy of the PBSPFL scheme has stabilized at around 95.8%. It can be seen from the experimental results that the PBSPFL scheme can have the accuracy rate similar to the FedAvg while protecting the privacy and security performance of the client side. On the Fashion-MNIST, the accuracy of the PBSPFL scheme dataset. Therefore, the PBSPFL scheme can guarantee the privacy security of the client nodes with the high accuracy of the model.

In order to verify the PBSPFL scheme against the poisoning attack, the global model of the aggregation is still high availability, this section experiment set up 20% and 40% of the malicious poisoning attack nodes involved in FL model training, and compared with FedAvg, ShieldFL, Biscotti model test, of which the default poisoning attack for the label flip attack. The experimental aggregation server nodes in this section are all trusted nodes.

The global model accuracy rates of the different model schemes on the FASHION-MNIST and CIFAR-10 datasets are shown in Figure 5.

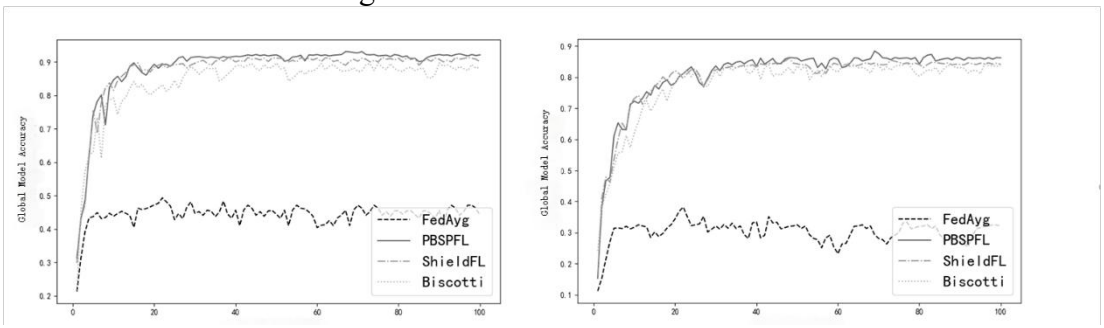

Figure 5 20% of the nodes with malicious poisoning attacks participate in the FL model training

Since FedAvg has no security mechanism against toxic attacks, the toxic gradient has a huge impact on the accuracy of its global model, so the accuracy of FedAvg fluctuates around 40%. In the FASHION-MNIST dataset, the Biscotti accuracy was maintained at 88.4%, and the PBSPFL scheme accuracy was stabilized at 92.2%, slightly higher than the ShieldFL stable accuracy at 90.8%. The accuracy of the present scheme was maintained at 85.4% on the CIFAR-10 dataset,

The global model accuracy rates of the different model schemes on the FASHION-MNIST and CIFAR-10 datasets are shown in Figure 6.

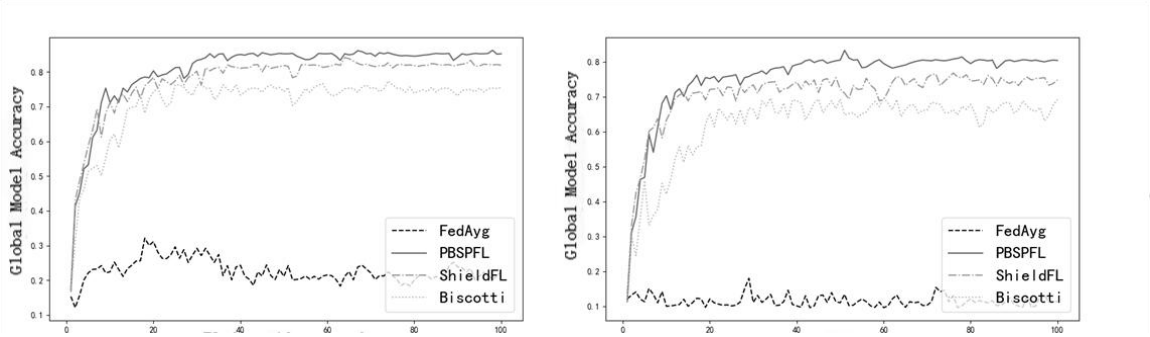

Figure 6 40% of the nodes with malicious poisoning attacks participate in the FL model training

Under the FASHION-MNIST dataset, as the number of nodes with malicious poisoning attacks in the model increases to 4 0%, the model accuracy of FedAvg fluctuates around 20%, and after 4 0 rounds of iteration, the accuracy gradually drops below 20%. In this paper, the model is still well available with the other two comparison models. The PBSPFL scheme stabilized after 4 0 rounds of iterations, and the accuracy rate stabilized at around 84.8%. Biscotti The most affected in the presence of malicious poisoning attacks in 40% of the nodes, and its accuracy has stabilized at around 75%. The model accuracy rate of ShieldFL is maintained at 80.2%, which is lower than

4.6%. On the CIFAR-10 dataset, 40% of the PBSPFL scheme nodes maintained 79.6% of the model accuracy.

The experimental results show that the PBSPFL scheme can still guarantee the high availability of the global model with malicious poisoning attacks in 20% and 40% of the client nodes.

7.4 Algorithmic effect analysis

In order to evaluate the effect of the node selection algorithm of the trusted aggregation server to identify malicious aggregation behavior, 40% of the nodes are set to have malicious poisoning behavior. We need to increase the number of nodes exhibiting malicious aggregation behavior to 20%, 40%, and 50%, respectively, with half of them also engaging in malicious poisoning attacks. The influence of different numbers of nodes with malicious aggregation behavior after 1 00 rounds of FL iterations on the global model accuracy of the PBSPFL scheme is shown in Figure 7.

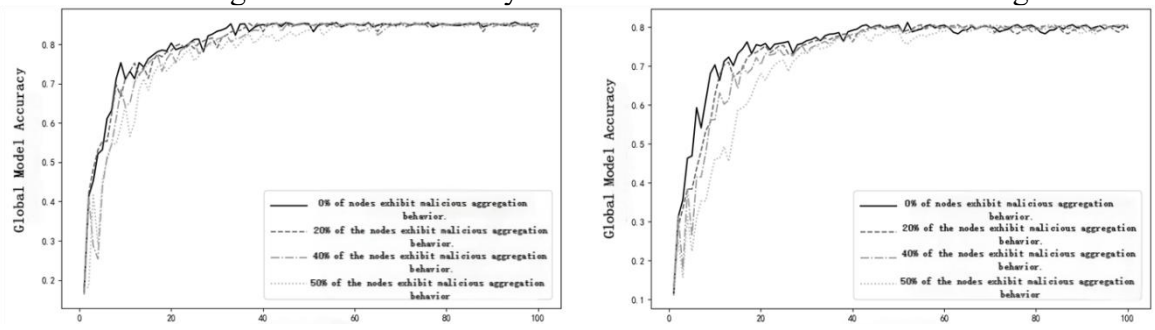

Figure 7 The nodes with different numbers of malicious aggregation behavior participate in the FL model training

As the nodes with malicious aggregation behavior increase, the FL model converges more slowly. In the case of no malicious aggregation behavior, the global model on the FASHION-MNIST dataset reached the convergence level in the 4 0th iteration, while in 50% of the nodes had the malicious aggregation behavior, the FASHION-MNIST dataset gradually converged until the 60th FL iteration, and gradually converged on the CIFAR-10 dataset. When the model reaches the convergence level, under the condition of nodes with different numbers of malicious aggregation behavior, the correct rate of the global model reaches 84% in the FASHION-MNIST data set, and the accuracy rate reaches 79% in the CIFAR-10 data set. Therefore, the trusted aggregation server node selection algorithm can guarantee the quality of malicious aggregation attacks on the server.

8. Conclusion

The current PPFL model safeguards data privacy and resists client poisoning attacks. However, the entire model system can be compromised due to the malicious behavior or failure of the aggregation server. This paper introduces a framework for secure aggregation privacy protection, which is based on a licensed blockchain. In the open and transparent trading environment of the licensed blockchain, the trusted aggregation server node selection algorithm selects two aggregation nodes. The privacy security of data is ensured through a threshold-based fully homomorphic encryption protection model. The cosine similarity and a reward-punishment algorithm based on credit value are used to penalize nodes exhibiting malicious behavior. The impact of toxic gradient reduction on the global model accuracy is examined. Experimental results indicate that even with 50% of the nodes engaging in malicious aggregation behavior and 40% conducting poisoning attacks, high model accuracy can still be achieved on the FASHION-MNIST and CIFAR-10 datasets. Future work will further optimize the communication consumption of the model and

improve the efficiency of the model in the work.

References

[1] Xiao X, Tang Z, Xiao B. A survey on privacy and security issues in federated learning[J]. Chin. J. Comput, 2023, 46(5): 1019-1044.

[2] Xiao Xiong, Tang Zhuo, Xiao Bin, et al. Review of Privacy Protection and Security Defense Research in Federal Learning [J]. Journal of Computer Science, 2023,46 (05): 1019-1044.

[3] Kairouz P, McMahan H B, Avent B, et al. Advances and open problems in federated learning[J]. Foundations and Trends® in Machine Learning, 2021, 14(1–2): 1-210.

[4] Liu YX, Chen H, Liu YH, Li CP. Privacy-preserving Techniques in Federated Learning[J]. Journal of Software, 2022, 33(3): 1057-1092.

[5] Liu Yixuan, Chen Hong, Liu Yuhan, Li Cuiping. Privacy Technology in federated Learning [J]. Journal of Software, 2022,33 (3): 1057-1092.

[6] Li T, Sahu A K, Talwalkar A, et al. Federated learning: Challenges, methods, and future directions[J]. IEEE signal processing magazine, 2020, 37(3): 50-60.

[7] Zang XL ,Luo WH.Implementing Federated Learning Intrusion Detection Using Dynamic Clipping Differential Privacy[J]. Journal of Chinese Computer Systems. 2024, 45(6): 1474-1481.

[8] Zhang Xiaolong, Luo Wenhua. Implement federated learning intrusion detection using dynamic trimmed differential privacy [J]. Small microcomputer system. 2024,45(6):1474-1481.

[9] Fung C, Yoon C J M, Beschastnikh I. Mitigating sybils in federated learning poisoning[J]. arXiv preprint arXiv:1808.04866, 2018.

[10] Tolpegin V, Truex S, Gursoy M E, et al. Data poisoning attacks against federated learning systems[C]//Computer security–ESORICs 2020: 25th European symposium on research in computer security, ESORICs 2020, guildford, UK, September 14–18, 2020, proceedings, part i 25. Springer International Publishing, 2020: 480-501.

[11] Dong C, Weng J, Li M, et al. Privacy-preserving and byzantine-robust federated learning[J]. IEEETransactions on Dependable and Secure Computing, 2023, 21(2): 889-904.

[12] Blanchard P, El Mhamdi E M, Guerraoui R, et al. Machine learning with adversaries: Byzantine tolerant gradient descent[J]. Advances in neural information processing systems, 2017, 30.

[13] Yin D, Chen Y, Kannan R, et al. Byzantine-robust distributed learning: Towards optimal statistical rates[C]//International conference on machine learning. Pmlr, 2018: 5650-5659.

[14] Guerraoui R, Rouault S. The hidden vulnerability of distributed learning in byzantium[C]//International Conference on Machine Learning. PMLR, 2018: 3521-3530.

[15] Shen S, Tople S, Saxena P. Auror: Defending against poisoning attacks in collaborative deep learning systems[C]//Proceedings of the 32nd annual conference on computer security applications. 2016: 508-519.

[16] Liu X, Li H, Xu G, et al. Privacy-enhanced federated learning against poisoning adversaries[J]. IEEE Transactions on Information Forensics and Security, 2021, 16: 4574-4588.

[17] Ma Z, Ma J, Miao Y, et al. ShieldFL: Mitigating model poisoning attacks in privacy-preserving federated learning[J]. IEEE Transactions on Information Forensics and Security, 2022, 17: 1639-1654.

[18] Zhou J, Wu N, Wang Y, et al. A differentially private federated learning model against poisoning attacks in edge computing[J]. IEEE Transactions on Dependable and Secure Computing, 2022, 20(3): 1941-1958.

[19] Li G, Wu J, Li S, et al. Multitentacle federated learning over software-defined industrial internet of things against adaptive poisoning attacks[J]. IEEE Transactions on Industrial Informatics, 2022, 19(2): 1260-1269.

[20] Majeed U, Hong C S. FLchain: Federated learning via MEC-enabled blockchain network[C]//2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS). IEEE, 2019: 1-4.

[21] Kim Y J, Hong C S. Blockchain-based node-aware dynamic weighting methods for improving federated learning performance[C]//2019 20th Asia-pacific network operations and management symposium (APNOMS). IEEE, 2019: 1-4.

[22] Qammar A, Karim A, Ning H, et al. Securing federated learning with blockchain: a systematic literature review[J]. Artificial Intelligence Review, 2023, 56(5): 3951-3985.

[23] SUN R, LI C, WANG W, et al. Research progress of blockchainbased federated learning[J]. Journal of Computer Applications, 2022, 42(11): 3413.

[24] Sun Rui, Li Chao, Wang Wei, et al. Progress in blockchain-based federal learning research [J]. Computer Applications, 2022,42 (11): 3413-3420.

[25] Miao Y, Liu Z, Li H, et al. Privacy-preserving Byzantine-robust federated learning via blockchain systems[J]. IEEE Transactions on Information Forensics and Security, 2022, 17: 2848-2861.

[26] Shayan M, Fung C, Yoon C J M, et al. Biscotti: A blockchain system for private and secure federated learning[J].

IEEE Transactions on Parallel and Distributed Systems, 2020, 32(7): 1513-1525.
[27] Chillotti I, Gama N, Georgieva M, et al. TFHE: fast fully homomorphic encryption over the torus[J]. Journal of Cryptology, 2020, 33(1): 34-91.
[28] Chowdhury S, Sinha S, Singh A, et al. Efficient threshold FHE with application to real-time systems[J]. Cryptology ePrint Archive, 2022.
[29] Zu J, Xu F, Jin T, et al. Reward and Punishment Mechanism with weighting enhances cooperation in evolutionary games[J]. Physica A: Statistical Mechanics and its Applications, 2022, 607: 128165 .