IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 17 November 2023, accepted 3 January 2024, date of publication 11 January 2024,
date of current version 19 January 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3353131

RESEARCH ARTICLE

MS-FL: A Federated Learning Framework Based on Multiple Security Strategies

WENSHAO YANG, PENGFEI KANG, AND CHAO WEI, (Member, IEEE)
School of Science, Yanshan University, Qinhuangdao 066004, China

Corresponding author: Chao Wei (weichao@ysu.edu.cn)

This work was supported in part by PHD Foundation of Yanshan University under Grant 8190047, in part by the Provincial Natural Science Foundation of Hebei under Grant A 2022203004, and in part by the Science Research Project of Hebei Education Department under Grant QN2020203.

ABSTRACT With the development of data science, AI and data transaction, an increasing number of users are utilizing multi-party data for federated machine learning to obtain their desired models. Therefore, scholars have proposed numerous federated learning frameworks to address practical issues. However, there are still three issues that need to be addressed in current federated learning frameworks: 1) privacy protection, 2) poisoning attack, and 3) protection of the interests of participants. To address these issues, this paper proposes a novel federated learning framework *MS-FL* based on multiple security strategies. The framework's algorithms guarantee that data providers need not worry about data privacy leakage. At the same time, it can defend against poisoning attack from malicious nodes. Finally, to ensure the interests of all parties are protected, a blockchain protocol is utilized. The theoretical derivation proves the effectiveness of this framework. Experimental results show that the algorithm designed in this paper outperforms other algorithms.

INDEX TERMS Federated learning, privacy protection, poisoning attack, multiple security strategies, data transaction.

I. INTRODUCTION

Recent years, due to the presence of sensitive information in the data, enterprises and organizations are regulated by laws and cannot disclose the data. As a result, most of the data are isolated and closed, leading to the formation of numerous data islands and a significant waste of resources. Therefore, the reasonable use of data resources while simultaneously protecting data privacy has become an urgent issue in the current era. Some scholars have proposed the federated learning (FL) [1] method to complete machine learning by utilizing the data of multiple parties involved in machine learning like Fig.1. However, Fredrikson [2] found that part of data information of model trainers could be inferred according to the model after training, which means that even if no data privacy was leaked in the process of federated learning, there is a risk of data privacy leakage from the model after training.

The associate editor coordinating the review of this manuscript and approving it for publication was Sangsoon Lim.

In practice, a fully homomorphic cryptosystem CKKS [3] is widely used to protect data privacy because it supports: (1) addition and multiplication homomorphism of ciphertext; (2) floating-point arithmetic and evaluating an arbitrary polynomial, which is ideal for privacy-protecting machine learning. Therefore, in this study, we exploit CKKS technology to protect data privacy of FL participants. Except for privacy leak, in the process of gradient aggregation, the adversary intends to inject poisoned (biased or misleading) data into a federated learning system, with the objective of modifying the model's parameters in a way that is beneficial to the attacker. If the aggregation server does not have defense mechanism and directly uses average of aggregate gradients, it has been demonstrated that a single poisoner can control entire training process [4], [5].

In this paper, we mainly focus on three types of representative poisoning attacks: (1) Label-flipping attack [6](data poisoning). In a data set, each data sample carries a category label. However, malicious nodes can use wrong labels to train machine learning model. (2) Backdoor attack [7](data poisoning). Malicious nodes aim to seek a set of parameters

to establish strong links between trigger and target label while minimizing the impact on classification of benign inputs. In Fig. 2, the trigger is a white block. (3) Arbitrary model attack [8](model poisoning). If training model locally, malicious nodes can deliver arbitrary wrong gradients to aggregation server to reduce final model accuracy.

In light of the increasing value of data, data trading [9] has experienced substantial growth. In real life, there is a situation where a company needs specific machine learning models, but lacks or has no data, and the data owners do not need the corresponding machine learning models themselves. For example, an educational research institution needs to train a machine learning model using the academic achievements of students from different schools, but these schools themselves do not need this machine learning model. The main concern of this paper is how to utilize multi-party data to train a reliable machine learning model under the condition of protecting privacy information, and allow both the model requesters and data owners to complete the transaction of data value. In this paper, we construct a novel FL framework *MS-FL* based on multiple security strategies to solve the problem of the above real-world scenario. *MS-FL* achieves the following advantages:

- Privacy. As prior works [10], [11] have shown, an adversary may recover data owner's sensitive information such as training samples or memberships by inferring shared gradients. To protect data owners' data privacy, we use multiple security protocols and CKKS techenology to keep each data owner's local gradients confidential. In addition, after executing protocols of *MS-FL*, only model requestor has access to the final model.
- Robustness. The model training process can defend against above three poisoning attacks from malicious nodes.
- Fair transaction. After every round of model training, model parameters will be delivered to model requestor and model requestor must pay to data owners. Smart contract used in *MS-FL* on blockchain makes this transaction atomic and transparent.
- Interest protection. Compared with some existing literature, *MS-FL* guarantees that data owner will not lose all opportunities to benefit from model requestor just because a portion of submitted gradient components deviates from the majority.

The rest of paper is organized as follows. Section II is related work. In Section III, we overview the preliminaries of this paper. Section IV introduces the specific steps and corresponding algorithms to complete federated learning in the application scenario of this paper. Section V and section VI demonstrate security analysis of the system and convergence property of proposed aggregation algorithm in this framework. In section VII, we compare the proposed aggregation algorithm with exiting algorithms in some aspects and give corresponding comparison graphs. Finally, section VIII summarizes this paper.

II. RELATED WORK

A. DEFENSE AGAINST POISONING ATTACKS

The researchers have developed a lot of defenses for poisoning to federal learning. Due to relevance to this paper, we present statistics-based defenses here, whose key idea is to focus the aggregation rules on the benign gradient vectors as much as possible. These methods fall roughly into two categories, one is to extract as good gradient data as possible, such as *krum* [4], *geometric median* [12], *trimmed mean* [13] and *bulyan* [14]. Blanchard [4] designed *krum*, which selects one of the gradients most similar to the other gradients as the global update. The similarity here is measured by calculating the sum of the Euclidean distance of $t - f - 2$ gradients closest to the gradient vector. *Geometric median* [12] takes the median of the gradients as global update. *Trimmed mean* removes some extreme values, and then averages the remaining parameters as the global update. *Bulyan* [14] can be regard as a variant that combines *krum* and *trimmed mean*. *Bulyan* firstly chooses less than $t-2f$ gradients with the same rule as the *krum*, then averages the parameters closest to the median of the gradient vectors as the global update. The other one is to assign different weights to gradients based on some characteristics. Gonzalez et al. [15] firstly calculated cosine similarity between gradients as weight of model update set in each iteration. If the cosine similarity exceeds a certain range, it is judged as malicious party. In addition, *PEFL* [16] provided a method to defend against label-flipping attack and backdoor attack with low computational complexity. They assigned different weights to the gradients of the participants based on the Pearson correlation coefficient between the gradients and medians of gradient components.

In order to preserve the privacy of data owners, our framework incorporates the utilization of a trusted third party (service provider) who adds noise to the model gradients before transmitting them to the model requestor. To obtain usable gradients from this distorted data, we propose a novel aggregation algorithm specifically tailored for the model requestor.

B. PRIVACY-PRESERVING MACHINE LEARNING

Recently, privacy-preserving ML mainly bases on the following three underlying technologies: Differential Privacy (DP) [17], Secure Multi-Party Computation (SMC) [18], and Homomorphic Encryption (HE) [19]. Many scholars combine homomorphic encryption and machine learning to achieve privacy protection, which is used in our frame.

In literature [20], Han et al. presented an efficient algorithm for logistic regression on homomorphic encrypted data. Chen et al. [21] combined HE and secret sharing to build secure largescale sparse logistic regression model and achieved both efficiency and security. In literatures [22] and [23], scholars also used HE to train machine learning models. Nonetheless, the aforementioned works solely address the protection of participants' data privacy, without

accounting for the model privacy of the initiator of the model training.

C. PROTECTION OF THE INTERESTS OF PARTICIPANTS

In order to motivate participants in FL, many scholars have designed FL mechanisms with rewards. However, how to distribute the rewards fairly to protect the interests of the participants has become a new difficult problem.

Li et al. [24] proposed an incentive method of contribution and benefit sharing, and Zhao et al. [25] proposed an incentive mechanism based on reputation. Both methods use trusted managers to calculate the contributions of each FL participant and distribute the benefits. However, the methods devised in literatures [26] and [27] do not require the involvement of trusted managers. Zhang et al. [26] designed a smart contract-based incentive mechanism according to the size and centroid distance of customer data used in local model training. The model in literature [27] requires that the reward of each worker is determined by the voting results of the next round of workers. Nevertheless, the scenario of separation of model requestors and data owners is not considered in the above researches.

III. PRELIMINARIES

A. FEDERATED LEARNING

Unlike traditional machine learning that centralizes all training data, FL (Federated learning) is a promising distributed setting that complete machine learing while allowing all data owners to keep data local like Fig. 1. In FL, the server orchestrates whole lifecycle of training until model accuracy reaches desired level, or the number of iterations reaches the preset value. The goal of learning is to find optimal model parameters so that the output of model is infinitely close to the true label.

In this paper, we focus on horizontal federated learning [28], which means the data from data owners have different IDs and same features. For example, suppose we have a central server and t clients $\{C_1, C_2, \ldots, C_t\}$ like Fig. 1, each client has a local dataset $D_j, j = 1, 2, \ldots, t$. We use $D = \{D_1, D_2, \ldots, D_t\}$ to denote the joint dataset. The following is the objective function with optimal parameters G.

$$F(x, G, y) = \min_{G} \mathbb{E}_{(x,y)\sim\tilde{D}} L(x, G, y), \tag{1}$$

where x is the training data, y is the label, $\mathbb{E}$ is expectation, $L(x, w, y)$ is the empirical loss function and $\tilde{D}$ is the distribution of the clients' data. After local model training, the client C_j sends local model G_i^j in the i-th iteration to aggregation server. When there are no malicious clients, The server can directly take the average value of clients' models as the global model like following formula,

$$G_i = \frac{1}{t} \sum_{j=1}^{t} G_i^j. \tag{2}$$

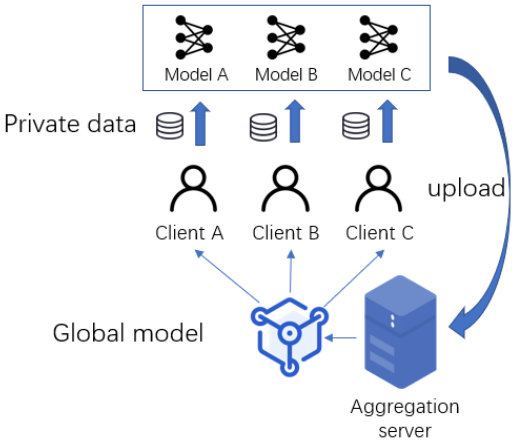


FIGURE 1. Federated learning.

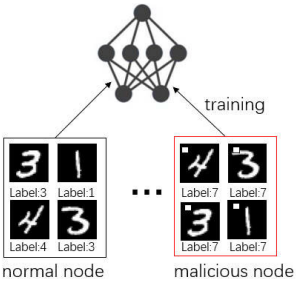


FIGURE 2. Backdoor attack.

The next step is that aggregation server sends the global model G_i to all clients and finally each client updates local model. This process will not stop until the update of parameter is tiny enough.

B. LOGISTIC REGRESSION

The proposed framework in this paper can accommodate various machine learning models, such as logistic regression, linear regression, neural networks, etc. Here, we demonstrate the operational process of this framework using the logistic regression model as an example.

Logistic regression is a classification model which is often used to solve classification problems. In a data set $\{(x_i, y_i)\}_{i=1}^{N}$, x_i is a d-dimensional feature vector, $y_i \in \{0, 1\}$. σ is Sigmoid function. The target is to find a model $G \in R^{d+1}$, which satisfies $y_i = \sigma\,(Gx_i)\,(i = 1, \ldots, N)$.

$\widetilde{y}$ represents the output of sigmoid function, the cost function L is:

$$L\,(G) = -\frac{1}{N} \sum_{i=1}^{N} \left(y_i ln\widetilde{y}_i + (1 - y_i)\ln(1 - \widetilde{y}_i)\right). \tag{3}$$

We use gradient descent to update model parameters (α is learning rate, G_j is the j-th component of G here):

$$G_j = G_j - \alpha \frac{\partial L}{\partial G_j} = G_j - \frac{\alpha}{N} \sum_{i=1}^{N} (\widetilde{y} - y_i)\, x_{ij}. \tag{4}$$

C. CIPHERTEXT GRADIENT UPDATE

Now $[\cdot]$ is used to denote the result of homomorphic encryption of plaintext data. So far, the fully homomorphic encryption cannot be used to directly calculate Sigmoid function. Literature [29] presented the fitting polynomial of Sigmoid function. After the model parameter G is encrypted by CKKS, the fitting polynomial in ciphertext can be used as:

$$[\sigma\ (Gx_i)] = a_0 + a_1\left(\frac{[G]\,x_i}{8}\right) + a_3\left(\frac{[G]\,x_i}{8}\right)^3.$$
$$(a_0, a_1, a_3) \approx (0.5, 1.20096, -0.81562)\,. \tag{5}$$

According to equ. (4) and equ. (5), we can obtain:

$$\left[\frac{\partial L}{\partial G_j}\right] = \frac{1}{N}\sum_{i=1}^{N}\left([\sigma\ (Gx_i)] - y_i\right)x_{ij}. \tag{6}$$

The model parameter update formula in ciphertext is:

$$[G] = [G] - \alpha\left[\frac{\partial L}{\partial G}\right]. \tag{7}$$

IV. PROPOSED FRAMEWORK

In this section, we preset system model and basic assumption of our proposed framework *MS-FL*, then present *MS-FL* on how to accomplish model transaction and mitigate the poisoning attacks in detail. For ease of reference, the symbols that appeared in below and corresponding descriptions are listed in TABLE 1.

A. SYSTEM MODEL

There are three basic entities in our system (Fig. 3):

- Model requestor: model requestors do not have data but need model, they are also the initiator of the machine learning. It is honest but curious.
- Data owner: data owners have data and they are curious about others' privacy. Some of them are malicious, who will poison in the model training.
- Service provider (SP): Service provider is responsible to receive all gradients submitted by data owners and aggregate them. It will add noise in the gradients to protect data owner's privacy. It is also honest but curious.

B. BASIC ASSUMPTION

1) Data samples of each data owner are IID (independent and identically distributed).
2) There are three types of attacks that malicious nodes can launch: label-flipping attack, backdoor attack and arbitrary model poisoning.
3) The communication between any two members of this system is secure and reliable.

C. SPECIFIC PROCESS

The process of the *MS-FL* and corresponding algorithm are demonstrated in below.

step 1 The model requestor uses Algorithm 1 to encrypt the model parameters with the public key and sends them to the service provider ($Enc(z, pk) \rightarrow v$ denotes that plaintext z is encrypted to ciphertext v by CKKS public key pk).

step 2 The service provider receives the encrypted model parameters and sends them to data owners.

step 3 The data owners will deliver the updated model gradients to service provider after executing Algorithm 2.

step 4 The service provider adds noise into model gradients by Algorithm 3 to protect the privacy of data owners, then send them to model requestor.

step 5 Upon receiving encrypted gradients containing noise, the model requestor initiates the process of decryption by utilizing its private key. Once the

TABLE 1. Symbols.

Symbols	Meaning
n	The dimension of model parameters
t	The number of data owners
D	Data set
y	Label
Y	Label matrix$(m \times 1)$
X	Data matrix$(n \times m)$
α	Learning rate
g_j	The j-th component of model parameter
G	Model parameter vector $G = (g_1, \ldots, g_n)$
f	The number of malicious nodes
m	Batch size
γ	Global model learning rate
$\beta^k = (b_1^k, b_2^k, \ldots, b_n^k)$	Model gradient for the k-th update
β_i^k	The i-th data owner's gradient
β_{ij}^k	The j-th component of β_i^k
$\mathcal{G}$	Vector space
$\|\|.\|\|$	2-norm
$\langle\ \rangle$	Inner product of vectors
x	The input data
N	The number of data samples

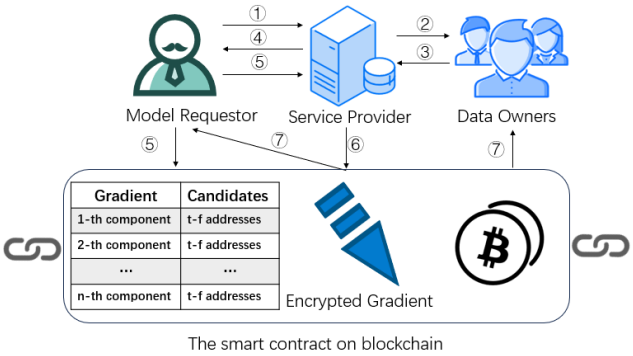


FIGURE 3. Proposed framework.

gradients have been decrypted, the model requestor employs Algorithm 4 ($Dec(v, sk) \rightarrow z$ denotes that ciphertext v is decrypted to plaintext z by CKKS private key sk) to facilitate the selection of $t - f$ data owners' parameters for aggregation with respect to each gradient component containing noise.

Following the successful completion of selection, the model requestor proceeds to create a table like Fig. 3 that contains the addresses of the data owners who participated in aggregating each gradient component. This table serves as a record of the contributions made by each data owner, and enables the model requestor to keep track of the overall progress of the aggregation process.

Finally, the model requestor initiates the creation of a smart contract that comprises the aforementioned table and some cryptocurrency. This smart contract serves as a mechanism for ensuring that all participants in the aggregation process are fairly compensated for their contributions. Once the smart contract has been created, the model requestor transmits the address of the contract to service provider, thereby completing the overall process of gradient aggregation.

step 6 Upon receipt of the smart contract, the service provider assumes a pivotal role in facilitating the successful execution of the gradient aggregation process. To this end, the service provider commences a series of rigorous checks and balances, designed to ensure the accuracy, reliability, and fairness of the overall process.

The first step in this process involves a thorough examination of the amount of cryptocurrency contained within the smart contract. If the amount of cryptocurrency is found to be insufficient to meet the required amount stipulated by the data owners, the service provider refrains from transmitting the no-noise encrypted gradient to the smart contract. This step is crucial in ensuring that the interests of all parties involved in the gradient aggregation process are adequately protected, and that the process proceeds in a transparent and equitable manner.

Once the cryptocurrency contained within the smart contract has been verified, the service provider moves on to the next stage of the process, which involves the selection of gradient components that are devoid of noise for aggregation. This selection process is guided by the table provided by the model requestor, which contains relevant information regarding the contributions made by the participating data owners.

Having identified the most suitable gradient components for aggregation, the service provider initiates the process of aggregation, utilizing Algorithm 5 to obtain the encryped aggregation result. Upon completion of the aggregation process, the service provider transmits the result to the account of the model requestor via the smart contract.

Algorithm 1 Model Encryption

Input: pk, n, G;
Output: $[G]$;
1: **for** int i=1 to n **do**
2: $\quad \left[g_j\right] \leftarrow Enc(g_j; pk)$ (In the first round of training, g_j is generated by model requestor randomly);
3: **end for**
4: Send $[G] = ([g_1], \ldots, [g_n])$ to service provider;

Algorithm 2 Local Training

Input: $[G], D, m, \alpha$;
Output: $\left[\beta_i^k\right]$;
1: Randomly select m data from data set D to constitute data matrix $X \in R^{n \times m}$ and label matrix $Y \in R^{1 \times m}$.
2: $(\widetilde{y}_1, \ldots, \widetilde{y}_m) \leftarrow [G] \times X$.
3: **for** int i=1 to m **do**
4: $\quad [\widetilde{y}_i] \leftarrow [\sigma(\widetilde{y}_i)]$. (equation 5)
5: **end for**
6: **for** int j=1 to n **do**
7: $\quad \left[b_j^k\right] \leftarrow \frac{1}{m} \sum_{i=1}^{m} ([\widetilde{y}_i] - y_i)\, x_{ji}$.
8: **end for**
9: $\left[\beta_i^k\right] \leftarrow \left(b_1^k, b_2^k, \ldots, b_n^k\right)$.
10: The i-th data owner send $\left[\beta_i^k\right]$ to service provider.

Algorithm 3 Privacy Protection

Input: $\left\{\left[\beta_i^k\right]\right\}_{i=1}^{t}$;
Output: $\left\{\widehat{\left[\beta_i^k\right]}\right\}_{i=1}^{t}$ ($\left\{\left[\beta_i^k\right]\right\}_{i=1}^{t}$ with noise);
1: Generate n pairs of non-zero random numbers $\left(c_j, d_j\right), j = 1, 2, \ldots, n$.
2: **for** int j=1 to n **do**
3: $\quad$ **for** int i=1 to t **do**
4: $\quad\quad \widehat{\left[\beta_{ij}^k\right]} \leftarrow c_j \left[\beta_{ij}^k\right] + d_j$. (add noise)
5: $\quad$ **end for**
6: **end for**
7: Send $\left\{\widehat{\left[\beta_i^k\right]}\right\}_{i=1}^{t}$ to model requestor.

step 7 In the last step, the cryptocurrency contained within the smart contract is equally distributed among the corresponding account addresses of the participating data owners on the table, thereby ensuring that all parties involved in the gradient aggregation process are fairly compensated for their contributions. This step is crucial in maintaining the trust and goodwill of the data owners, and in ensuring that they remain committed to the process of gradient aggregation.

Algorithm 4 Aggregation Algorithm

Input: $\left\{\left[\widehat{\beta_i^k}\right]\right\}_{i=1}^{t}, sk$;
Output: selected gradient components;
1: $\left\{\widehat{\beta_i^k}\right\}_{i=1}^{t} = Dec(\left\{\left[\widehat{\beta_i^k}\right]\right\}_{i=1}^{t}; sk)$.
2: **for** int j=1 to n **do**
3: assign $\left\{\widehat{\beta_{ij}^k}\right\}_{i=1}^{t}$ to the list $\{b_1, b_2, \ldots, b_t\}$ in order.
4: **for** int i=1 to t **do**
5: Pick $t-f-1$ points that have the smallest distance from b_i in the set $\{b_1, b_2, \ldots, b_t\}$, which constitute $\{b_{1^*}, b_{2^*}, \ldots, b_{(t-f-1)^*}\}$, suppose the distance set between these points and b_i is $\left\{h^i_{1^*}, h^i_{2^*}, \ldots, h^i_{(t-f-1)^*}\right\}$.
6: **for** int j=1 to t-f-1 **do**
7: **if** $b_{j^*} < b_i$ **then**
8: $h^i_{j^*} \leftarrow -h^i_{j^*}$.
9: **end if**
10: **end for**
11: $b'_i \leftarrow \sum_{j=1}^{t-f-1} h^i_{j^*}$.
12: **end for**
13: Select the smallest b'_i in the set $\{b'_i\}_{i=1}^{t}$ and then choose $t-f$ data owners as candidates whose gradient component is or is closest to b_i in the set $\{b_1, b_2, \ldots, b_t\}$.
14: **end for**
15: Take the account addresses of candidates for each component of gradient to establish the smart contract on blockchain.
16: Put cryptocurrency into smart contract.

Algorithm 5 Model Transaction

Input: $\{[\beta_i^k]\}_{i=1}^{t}$;
Output: $[\beta^{k+1}]$;
1: **for** int j=1 to n **do**
2: Add $t-f$ values in the set $\left\{\left[\beta_{ij}^k\right]\right\}_{i=1}^{t}$ corresponding to the accounts in the newly created smart contract to get the result $[\beta^{k+1}]$
3: **end for**
4: Send $[\beta^{k+1}]$ to smart contract.

Finally, the model requestor processes the received aggregation outcomes in preparation for the next iteration, utilizing Algorithm 6 to complete model updation.

V. SECURITY ANALYSIS

The function privacy and semantic security of the CKKS against chosen-plaintext attacks (or IND-CPA security for short) [3] guarantee the strong security of the protocol. Because the plaintext of model will not appear in the joint view of data owners and service provider as well as the IND-CPA security of CKKS, it is impossible that data owners or service provider have opportunity to obtain any information about the model. Therefore, in the rest of this section,

Algorithm 6 Model Updating

Input: $[\beta^{k+1}], sk, \gamma, t-f$;
Output: G;
1: $\beta^{k+1} \leftarrow Dec([\beta^{k+1}]; sk)$.
2: $G \leftarrow G - \gamma/(t-f) \times \beta^{k+1}$.

we provide hybrid argument and simulator [16] to further demonstrate that during the execution of *MS-FL* protocols, model requestor and service provider cannot learn nothing about data owners' privacy.

Theorem 1: Given a homomorphic encryption security parameter p, any subset of data owners U and $C = \{model\ requestor, service\ provider\}$, *let* $REAL_C^{U,p}$ *be a random variable representing the joint view of parties in C in the real execution of above protocols. There exists a probabilistic polynomial-time (PPT) simulator SIM such that the output of SIM is computationally indistinguishable from* $REAL_C^{U,p}$.

Proof: According to the definition of $REAL_C^{U,p}$, it consists of all internal state and messages received by the parties in C during the execution of the protocols. We adopt the standard hybrid argument used in literatures [30] and [31] to prove this proposition. Given the security parameter p, we define a PPT simulator *SIM* through a series of subsequent hybrid operations to the random variables in $REAL_C^{U,p}$. *SIM* can be regarded as a "stand-in" for set C when performing the following hybrid operations. We perform hybrid operation at steps of *MS-FL* where the set C have possible chance to speculate about the privacy of the data owners, and verify that the output of the *SIM* is computationally indistinguishable from $REAL_C^{U,p}$ like random variable in Hyb 1, which means that set C has no ability to detect changes in the data owner's private information and the privacy of data owners is safeguarded. The detailed proof is described below.

Hyb 1 Initialize a random variable whose distribution is indistinguishable from $REAL_C^{U,p}$.

Hyb 2 In this hybrid, we change the behavior of an honest data owner in step 3, so that this data owner encrypts a randomly selected vector $\eta = (\eta_1, \eta_2, \ldots, \eta_n)$ instead of the gradient $\left[\beta_x^k\right]$ send to service provider. Since only the contents of the ciphertexts are changed, the IND-CPA security property of CKKS as well as the two non-collusive model requestor and service provider setting guarantees that this hybrid is indistinguishable from the previous one.

Hyb 3 In this hybrid, we simulate the service provider to perturb the η_i instead of $\left[\beta_{xi}^k\right]$ by the noise c'_i and d'_i, which is sampled uniformly at random, in step 4, model requestor can only get $\widehat{\eta_i} = c'_i\eta_i + d'_i$. It is well known that the parameters added by uniformly random numbers are also uniformly random. Besides, the IND-CPA security property of CKKS, as well as the two non-collusive model

requestor and service provider setting guarantees that this hybrid is indistinguishable from the previous one.

Hyb 4 In this hybrid, we change the input of algorithm 4 with $\hat{\eta}$ instead of $\left[\widehat{\beta_x^k}\right]$, and simulate model requestor to take $\hat{\eta}$ into aggregation. The IND-CPA security property of CKKS, as well as the two non-collusive model requestor and service provider setting guarantees that this hybrid is indistinguishable from the previous one.

Hyb 5 In this hybrid, we simulate service provider delivers aggregation result $[\beta^k]$ including η instead of $\widehat{\beta_x^k}$ to model requestor. The information contained in the final view of the model requestor are: $\widehat{\beta_{ij}^k}$, $\sum_{p=1}^{t-f}\widehat{\beta_{i_pj}^k}$ and $\sum_{p=1}^{t-f}\beta_{i_pj}^k$. This hybrid is indistinguishable from the previous one because model requestor can only get equations

$$\begin{cases} c_j\beta_{ij}^k + d_j = \widehat{\beta_{ij}^k} (j = 1, 2, \ldots, n; i = 1, 2, \ldots, t) \\ c_j \sum_{p=1}^{t-f} \beta_{i_pj}^k + (t-f)\, d_j = \sum_{p=1}^{t-f} \widehat{\beta_{i_pj}^k}, \end{cases} \tag{8}$$

which is unable to figure out $\left\{\beta_i^k\right\}_{i=1}^t$, besides the IND-CPA security property of the CKKS.

□

The argument proves that there is a simulator *SIM* sampled from the distribution described above so that its output is computationally indistinguishable from the output of *REAL*. Hence, the frame holds the security property that model requestor and service provider will learn nothing about the data owners' private data and other participants cannot obtain the real model except model requestor.

VI. CONVERGENCE ANALYSIS

In this section, we provide the convergence analysis of *MS-FL*. For a specific FL task, the optimal global model G^* can be obtained by equation (1). Based on article [12], we can derive that, the difference between the global model learnt by algorithm 4 under attacks and the optimal global model G^* is bounded. To prove Theorem 2, we list corresponding lemma and assumptions in article [12].

Assumption 1: The population risk function $F : \mathcal{G} \to \mathbb{R}$ is L-strongly convex, and differentiable over G with M-Lipschitz gradient. That is, for all G, G′ ∈ 𝒢,

$$F\left(G'\right) \geq F(G) + \left\langle \nabla F(G), G' - G \right\rangle + \frac{L}{2}\left\|G' - G\right\|^2,$$

and

$$\left\|\nabla F(G) - \nabla F\left(G'\right)\right\| \leq M\left\|G - G'\right\|.$$

Under Assumption 1, according to article [32], using the standard gradient descent update

$$G_j = G_{j-1} - \eta \times \nabla F\left(G_{j-1}\right),$$

where η is some fixed stepsize, G_j approaches G^* exponentially fast. In particular, choosing $\eta = \frac{L}{2M^2}$, it holds that

$$\left\|G_j - G^*\right\| \leq \left(1 - \frac{L^2}{4M^2}\right)^{j/2} \left\|G_0 - G^*\right\|.$$

Assumption 2: There exist positive constants σ_1 and α_1 such that for any unit vector $v \in B$, $\langle \nabla f\left(x, G^*\right), v\rangle$ is sub-exponential with scaling parameters σ_1 and α_1, $\forall|\lambda| \leq \frac{1}{\alpha_1}$,

$$\sup_{v\in B} \mathbb{E}\left[\exp\left(\lambda\left\langle \nabla f\left(x, G^*\right), v\right\rangle\right)\right] \leq e^{\sigma_1^2\lambda^2/2},$$

where B denotes the unit sphere $\{G : \|G\|_2 = 1\}$ and $f(x, G)$ is local loss function.

Assumption 3: There exist positive constants σ_2 and α_2 such that for any $G \in \mathcal{G}$ with $G \neq G^*$ and unit vector $v \in B$, $\langle h(x, G) - \mathbb{E}[h(x, G)], v\rangle / \|G - G^*\|$ is sub-exponential with scaling parameters (σ_2, α_2), i.e., for all $|\lambda| \leq \frac{1}{\alpha_2}$,

$$\sup_{G\in\mathcal{G}, v\in B} \mathbb{E}\left[\exp\left(\frac{\lambda\langle h(x, G) - \mathbb{E}[h(x, G)], v\rangle}{\|G - G^*\|}\right)\right] \leq e^{\sigma_2^2\lambda^2/2},$$

where

$$h(x, G) = \nabla f(x, G) - \nabla f\left(x, G^*\right).$$

Assumption 4: For any $\delta \in (0, 1)$, there exists a real number $M' = M'(n, \delta)$ that is non-increasing in n such that

$$P\left\{\sup \frac{\left\|\frac{1}{n}\sum_{i=1}^n \left(\nabla f\left(x_i, G\right) - \nabla f\left(x_i, G'\right)\right)\right\|}{\|G - G'\|} \leq M'\right\} \geq 1 - \frac{\delta}{3}.$$

Lemma 1: Suppose that Assumption 1 - Assumption 4 hold such that L, M, σ_1, σ_2, α_1, α_2, are all of $\mathcal{G}(1)$ and $\log M' = O(\log n)$. q is Byzantine number. Assume that $G \subset \left\{G : \|G - G^*\| \leq r\sqrt{n}\right\}$ for some positive parameter r such that $\log(r) = O(n\log(N/k))$, and $2(1+\epsilon)q \leq k \leq t$. Fix any $\alpha \in (q/k, 1/2)$ and any $\delta > 0$ such that $\delta \leq \alpha - q/k$ and $\log(1/\delta) = O(n)$. There exist universal positive constants c_1, c_2 such that if

$$\frac{N}{k} \geq c_1 C_\alpha^2 n \log(N/k),$$

where

$$C_\alpha = \frac{2(1-\alpha)}{1-2\alpha},$$

then with probability at least

$$1 - \exp(-kD(\alpha - q/k\|\delta)),$$

where $D\left(\delta'\|\delta\right) = \delta'\log\frac{\delta'}{\delta} + \left(1-\delta'\right)\log\frac{1-\delta'}{1-\delta}$ denotes the binary divergence, the iterates $\left\{G_j\right\}$ with $\eta = L/\left(2M^2\right)$ satisfy

$$\left\|G_j - G^*\right\| \leq \left(\frac{1}{2} + \frac{1}{2}\sqrt{1 - \frac{L^2}{4M^2}}\right)^j \left\|G_0 - G^*\right\| + c_2\sqrt{\frac{nk}{N}}.$$

In the above lemma, based on the description of article [12], t participants are divided into k parties. In this

paper, for robustness of FL, we set $k = t$. In addition, δ can be viewed as the expected fraction of batches that are "statistically bad"; the larger the batch sample size N/k, the smaller δ. In this paper, we temporarily assume that there is no "statistically bad" data batch except Byzantine nodes. So the probability $1 - \exp(-kD(\alpha - q/k\|\delta))$ can be regarded as 1.

Based on above lemma and assumptions, we can obtain the following theorem.

Theorem 2: Suppose v is the probability that the component of global model G_j of MS-FL does not fall within the suitable interval of the component of benign nodes. With probability at least $1 - v$, the difference between G_j and G^* satisfies:

$$\begin{aligned}&\left\|G_j - G^*\right\| \\ &\leq \left[\left(\frac{1}{2} + \frac{1}{2}\sqrt{1 - \frac{L^2}{4M^2}}\right)^j + 2\left(1 - \frac{L^2}{4M^2}\right)^{j/2}\right] \\ &\quad \times \left\|G_0 - G^*\right\| + c_2\sqrt{\frac{nt}{N}}.\end{aligned}$$

Proof: Because the data set is IID, the gradient vectors of bengin nodes present Gaussian distribution. We denote the $b^j_{i\_min}$ and $b^j_{i\_max}$ are the minimum value and max value of i-th component of gradient of all bengin nodes in the j-th iteration. obviously, the i-th component of bengin gradients must be in the interval $\left[b^j_{i\_min}, b^j_{i\_max}\right]$ but malicious nodes will not.

When the number of malicious nodes in the network is less than the number of benign nodes, the malicious nodes exercise the worst attack over the aggregation process, causing the resulting model to be positioned at the edge of the interval $\left[b^j_{i\_min}, b^j_{i\_max}\right]$ for each component of the gradient if we select the method of *geometric median*. We use G_{j_geo} as model aggregation result that is most affected by malicious nodes, which is obtained by *geometric median* method. However, according to Lemma 1, the difference between the global model G_{j_geo} and the optimal global model G^* can still be bounded.

we suppose that the result of our aggregation algorithm τ^j_i is the i-th component of gradient in the j-th iteration. Because all users can not obtain specific distribution and value of their gradients in our framework, under the three poisoning attacks proposed in this paper, we can obtain the following probabilistic relationship

$$P\left(b^j_{i\_min} \leq \tau^j_i \leq b^j_{i\_max}\right) \gg P\left(\tau^j_i \notin \left[b^j_{i\_min}, b^j_{i\_max}\right]\right),$$

and

$$v = \sum_{i=1}^{n} P\left(\tau^j_i \notin \left[b^j_{i\_min}, b^j_{i\_max}\right]\right).$$

$\bar{G}_{j\_b}$ denotes the average parameter of benign nodes in j-th iteration. Based on the aforementioned lemma and assumptions, when the number of malicious nodes is less than that of honest nodes, we have the following equations for the j-th global iteration with probability at least $1 - v$,

$$\begin{aligned}&\left\|G_j - G^*\right\| \leq \left\|G_j - \bar{G}_{j\_b} + \bar{G}_{j\_b} - G^*\right\| \leq \left\|G_j - \bar{G}_{j\_b}\right\| \\ &\quad + \left\|\bar{G}_{j\_b} - G^*\right\| \leq \left\|G_{j\_geo} - \bar{G}_{j\_b}\right\| + \left\|\bar{G}_{j\_b} - G^*\right\| \\ &\leq \left\|G_{j\_geo} - G^* + G^* - \bar{G}_{j\_b}\right\| + \left\|\bar{G}_{j\_b} - G^*\right\| \\ &\leq \left\|G_{j\_geo} - G^*\right\| + 2\left\|\bar{G}_{j\_b} - G^*\right\| \\ &= \left[\left(\frac{1}{2} + \frac{1}{2}\sqrt{1 - \frac{L^2}{4M^2}}\right)^j + 2\left(1 - \frac{L^2}{4M^2}\right)^{j/2}\right] \\ &\quad \times \left\|G_0 - G^*\right\| + c_2\sqrt{\frac{nt}{N}}.\end{aligned}$$

□

VII. PERFORMANCE EVALUATION

In this section, we conduct experiments with real-world datasets to evaluate the performance of the aggregation algorithm proposed in this paper. All experiment run in a Lenovo server with i5-6900hq CPU and 8G RAM.

A. DATASET

We implemented experiments under two typical real datasets, which are MNIST and Fashion-MNIST. The MNIST dataset is used for handwritten digit recognition including 60000 training images and 10000 test images. The grayscale of these images has been normalized to 28 × 28 pixels. Just like MNIST, the Fashion-MNIST dataset also consists of ten classes of images, including fashion items such as trousers, sandals, bags, et al.

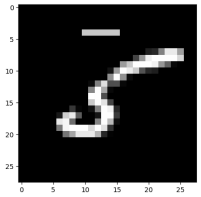
B. EXPERIMENT SETUP

The dataset is partitioned into ten subsets, with each subset assigned to a different data owner. Each data owner performs logistic regression algorithm to complete machine learning locally. To simulate malicious nodes, a portion of these subsets is selected and designated as such. In order to expedite the testing of the aggregation algorithm, the experiment is conducted using plain text, and is divided into four distinct stages:

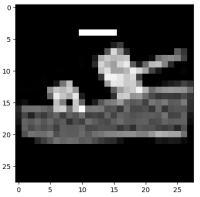
1) Iteration/Training: Each data owner executes a round of gradient descent using their local data.
2) Selection/Weight Assignment: The aggregation algorithm is utilized to either select subsets of each gradient component from data owners for aggregation or to assign different weights to their gradients.
3) Aggregation: The gradient data processed by previous step is averaged to arrive at the aggregated gradient.
4) Testing: After every ten rounds of training, the accuracy of the updated parameters is tested using the test set.

This process is repeated in a loop to compare the accuracy of *MS-FL* with other algorithms.

In the experiment compared with *PEFL*, we mainly focus on three poisoning attacks: label-flipping attack, backdoor attack and model poisoning attack. To simulate the label-flipping attack, we change labels in malicious data owners'



(a) MNIST Backdoor



(b) Fashion-MNIST Backdoor

FIGURE 4. Backdoor attack simulated in this paper.

dataset to $\hat{y} = (y+1) mod 10$. To reproduce the backdoor attack, we put a trigger into the images of malicious data owners (like Fig. 4(a) and 4(b)) as well as images of test set. To verify the vulnerability of *PEFL* for model poisoning attack, we only select one node as malicious participant, whose each gradient component adds a same arbitrary value in the interval $[-6000, 6000]$ after each round of iteration.

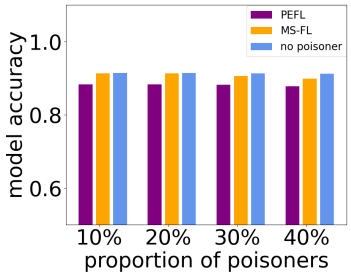
In the experiment compared with *trimmed mean* and *geometric median*, to test the robustness of our aggregation algorithm for arbitrary model attack, malicious nodes (40%) generate random value in [0, 10000] as upgraded gradients. When the *trimmed mean* used to aggregate gradients, we have to cut off 40% values on both sides of the median of each gradient component to ensure the accuracy of model. Because the test set of MNIST and Fashion-MNIST is not enough to observe distinction between *MS-FL* and the other two algorithms, we use training set as test set in this experiment.

Hyper-parameters: The total number of data owners we used in the experiment is 10, and the maximum number of malicious nodes is 4. When we use *MS-FL* to aggregate gradients, we set $f = 4$ regardless of the real number of malicious data owners. For each round of parameters updating, we set the batch size of 1000 and the global learning rate of $\gamma = 10^{-5}$ in MNIST scenario and $\gamma = 10^{-6}$ in Fashion-MNIST scenario.

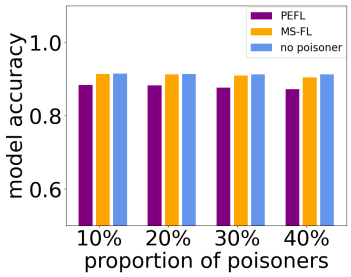
C. DISCUSSION OF EXPERIMENT RESULT

1) EFFECT OF DIFFERENT PROPORTIONS OF POISONERS

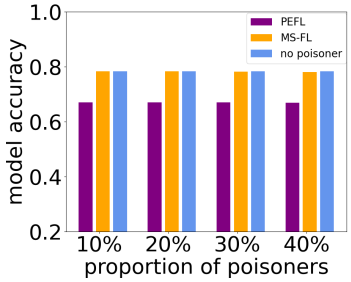
Fig. 5 outlines the comparison of test accuracy at the 300th iteration with different proportion of poisoners. It is well known that less benign data owners in training brings about lower model accuracy because as the proportion of poisoners increases, the amount of valuable data in the training process decreases. However it is not apparent in Fig. 5 and the reason is that no less than 60% of honest participants is sufficient for the model to have good accuracy in MNIST scenario and Fashion-MNIST scenario as well as the robustness of our algorithm and *PEFL*. In addition, no matter in which application scenario or attack, *MS-FL* can still maintain a comparative advantage compared with *PEFL*. We attribute it to the characteristic of *MS-FL*, which aggregates gradient components closer to the median of benign data as much as possible but not the median of all data.



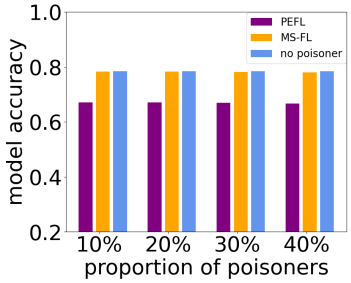
(a) MNIST label-flipping attack.



(b) MNIST backdoor attack.



(c) Fashion-MNIST label-flipping attack.



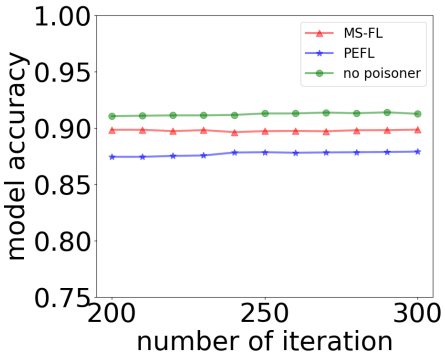
(d) Fashion-MNIST backdoor attack.

FIGURE 5. Comparison of accuracy with different number of poisoners.

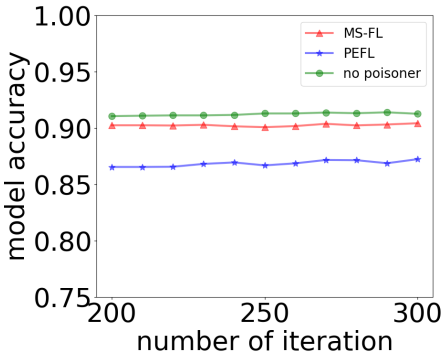
2) EFFECT OF DIFFERENT ITERATIONS (LABEL-FLIPPING ATTACK AND BACKDOOR ATTACK)

In Fig. 6, we illustrate the comparison of accuracy executed by *PEFL* and *MS-FL* with 40% malicious nodes from the 200th iteration to 300th iteration.

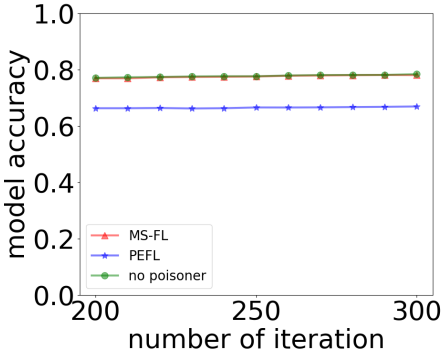
With an increase in the number of iterations, the model becomes progressively more adept at extracting effective data features. Consequently, the test accuracy of the model also improves, until the model ultimately achieves convergence. Here, we ignore the problems of under-fitting and over-fitting caused by too small or large model capacity. According to



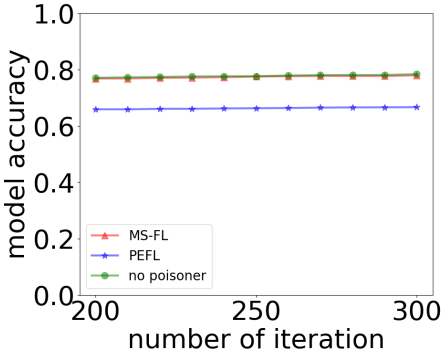
(a) MNIST label-flipping attack.



(b) MNIST backdoor attack.

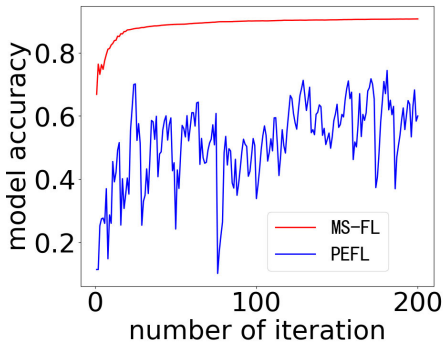


(c) Fashion-MNIST label-flipping attack.

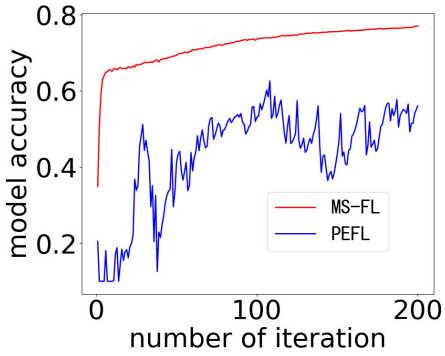


(d) Fashion-MNIST backdoor attack.

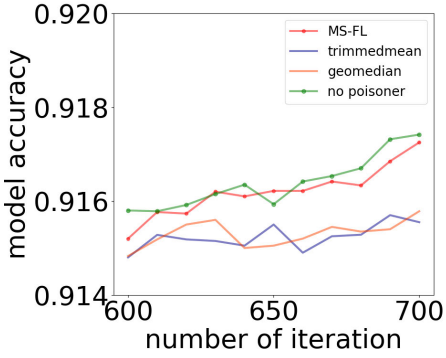
FIGURE 6. Comparison of accuracy with different iterations (data poisoning attack).



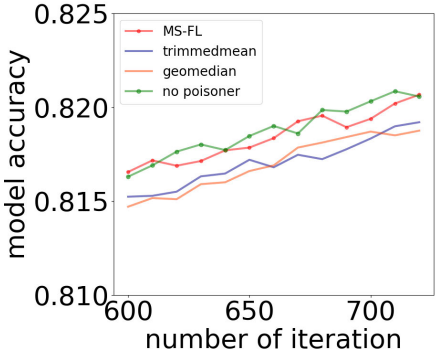
(a) MNIST.



(b) Fashion-MNIST.



(c) MNIST.



(d) Fashion-MNIST.

FIGURE 7. Comparison of accuracy with different iterations (model poisoning attack).

Fig. 6, we notice that model accuracy of *PEFL* is lower than *MS-FL* and the accuracy variation of the model has been nearly stable. Besides, in Figs. 6(c) and 6(d), the accuracy of *MS-FL* is nearly equivalent to that of the model without

poisoner, which means that malicious data has less effect on *MS-FL*.

3) EFFECT OF DIFFERENT ITERATIONS (MODEL POISONING ATTACK)

In the experiment of Figs. 7(a) and 7(b), there is only one malicious node mentioned in subsection VII-B but the *PEFL*'s model accuracy fails to stabilize, which means that it is vulnerable for *PEFL* to respond to model poisoning attack. In the meanwhile, the malicious node with extreme value has almost no effect on *MS-FL*.

Figs. 7(c) and 7(d) demonstrates the difference of model accuracy among *trimmed mean*, *geometric median* and *MS-FL* when malicious nodes occupy 40% in all data owners. No matter in MNIST scenario or Fashion-MNIST scenario, *MS-FL* provides better accuracy than the other two algorithms. This is because *trimmed mean* and *geometric median* eliminate too much valuable data away from the median of data owners' gradient component to resist the effect of malicious nodes and *MS-FL* does not rely on the median of data owners.

VIII. CONCLUSION

In this paper, a novel federated learning framework *MS-FL* based on multiple security strategies has been proposed for data application scenario, in which the model requestor and the data owner are separated. This framework has been proved secure for privacy of model requestor and data owners. Moreover, it is capable of protecting interests of participants of FL and the aggregation algorithm of *MS-FL* has ability to defend typical byzantine poisoning attacks. At the end of this paper, the experimental results demonstrate comparable performance of aggregation algorithm of *MS-FL* in terms of accuracy and robustness. In future work, we will continue to explore more algorithms to safeguard data privacy and defend poisoning attacks, aiming to enhance the accuracy of federated learning models.

REFERENCES

[1] S. Bai, G. Yang, G. Liu, H. Dai, and C. Rong, "NttpFL: Privacy-preserving oriented no trusted third party federated learning system based on blockchain," *IEEE Trans. Netw. Service Manage.*, vol. 19, no. 4, pp. 3750–3763, Dec. 2022.

[2] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Security*, Oct. 2015, pp. 1322–1333.

[3] D. D, A. K. K., and R. M, "Research on homomorphic encryption for arithmetic of approximate numbers," in *Proc. Int. Conf. Intell. Syst. Commun., IoT Security (ICISCoIS)*, Hong Kong, Feb. 2023, pp. 409–437.

[4] P. Blanchard, E. Mahdi E. Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.

[5] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *Proc. IEEE Symp. Security Privacy (SP)*, May 2019, pp. 739–753.

[6] Y. Jiang, W. Zhang, and Y. Chen, "Data quality detection mechanism against label flipping attacks in federated learning," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 1625–1637, 2023.

[7] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. Int. Conf. Artif. Intell. Statist.*, Jun. 2020, pp. 2938–2948.

[8] C. Xu, Y. Jia, L. Zhu, C. Zhang, G. Jin, and K. Sharif, "TDFL: Truth discovery based Byzantine robust federated learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 12, pp. 4835–4848, Dec. 2022.

[9] Z. Liu, C. Hu, H. Xia, T. Xiang, B. Wang, and J. Chen, "SPDTS: A differential privacy-based blockchain scheme for secure power data trading," *IEEE Trans. Netw. Service Manage.*, vol. 19, no. 4, pp. 5196–5207, Dec. 2022.

[10] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the GAN: Information leakage from collaborative deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, Oct. 2017, pp. 603–618.

[11] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients-how easy is it to break privacy in federated learning?" in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, Dec. 2020, pp. 16937–16947.

[12] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," in *Proc. Abstr. ACM Int. Conf. Meas. Model. Comput. Syst.*, Jun. 2018, pp. 1–25.

[13] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proc. Int. Conf. Mach. Learn.*, Jul. 2018, pp. 5650–5659.

[14] R. Guerraoui and S. Rouault, "The hidden vulnerability of distributed learning in byzantium," in *Proc. Int. Conf. Mach. Learn.*, Jul. 2018, pp. 3521–3530.

[15] L. Muñoz-González, K. T. Co, and E. Lupu, "Byzantine-robust federated machine learning through adaptive model averaging," 2019, *arXiv:1909.05125*.

[16] X. Liu, H. Li, G. Xu, Z. Chen, X. Huang, and R. Lu, "Privacy-enhanced federated learning against poisoning adversaries," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 4574–4588, 2021.

[17] X. Ma, X. Sun, Y. Wu, Z. Liu, X. Chen, and C. Dong, "Differentially private Byzantine-robust federated learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 12, pp. 3690–3701, Dec. 2022.

[18] G. He, W. Su, S. Gao, N. Liu, and S. K. Das, "NetChain: A blockchain-enabled privacy-preserving multi-domain network slice orchestration architecture," *IEEE Trans. Netw. Service Manage.*, vol. 19, no. 1, pp. 188–202, Mar. 2022.

[19] Y. Song, F. Wei, K. Zhu, and Y. Zhu, "Anomaly detection as a service: An outsourced anomaly detection scheme for blockchain in a privacy-preserving manner," *IEEE Trans. Netw. Service Manage.*, vol. 19, no. 4, pp. 3794–3809, Dec. 2022.

[20] K. Han, S. Hong, J. H. Cheon, and D. Park, "Logistic regression on homomorphic encrypted data at scale," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019, pp. 9466–9471.

[21] C. Chen, J. Zhou, L. Wang, X. Wu, W. Fang, J. Tan, L. Wang, A. X. Liu, H. Wang, and C. Hong, "When homomorphic encryption marries secret sharing: Secure large-scale sparse logistic regression and applications in risk control," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2021, pp. 2652–2662.

[22] H. Chen, R. Gilad-Bachrach, K. Han, Z. Huang, A. Jalali, K. Laine, and K. Lauter, "Logistic regression over encrypted data from fully homomorphic encryption," *BMC Med. Genomics*, vol. 11, no. S4, pp. 3–12, Oct. 2018.

[23] J. H. Cheon, D. Kim, Y. Kim, and Y. Song, "Ensemble method for privacy-preserving logistic regression based on homomorphic encryption," *IEEE Access*, vol. 6, pp. 46938–46948, 2018.

[24] Y. Li, C. Chen, N. Liu, H. Huang, Z. Zheng, and Q. Yan, "A blockchain-based decentralized federated learning framework with committee consensus," *IEEE Netw.*, vol. 35, no. 1, pp. 234–241, Jan. 2021.

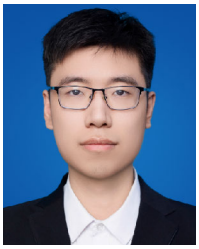
[25] Y. Zhao, J. Zhao, L. Jiang, R. Tan, D. Niyato, Z. Li, L. Lyu, and Y. Liu, "Privacy-preserving blockchain-based federated learning for IoT devices," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1817–1829, Feb. 2021.

[26] W. Zhang, Q. Lu, Q. Yu, Z. Li, Y. Liu, S. K. Lo, S. Chen, X. Xu, and L. Zhu, "Blockchain-based federated learning for device failure detection in industrial IoT," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5926–5937, Apr. 2021.

[27] K. Toyoda and A. N. Zhang, "Mechanism design for an incentive-aware blockchain-enabled federated learning platform," in *Proc. IEEE Int. Conf. Big Data*, Dec. 2019, pp. 395–403.

[28] A. Hammoud, H. Otrok, A. Mourad, and Z. Dziong, "On demand fog federations for horizontal federated learning in IoV," *IEEE Trans. Netw. Service Manage.*, vol. 19, no. 3, pp. 3062–3075, Sep. 2022.

[29] M. Kim, Y. Song, S. Wang, Y. Xia, and X. Jiang, "Secure logistic regression based on homomorphic encryption: Design and evaluation," *JMIR Med. Informat.*, vol. 6, no. 2, Apr. 2018, Art. no. e8805.
[30] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. Mcmahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, Oct. 2017, pp. 1175–1191.
[31] G. Xu, H. Li, Y. Zhang, S. Xu, J. Ning, and R. H. Deng, "Privacy-preserving federated deep learning with irregular users," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 2, pp. 1364–1381, Mar. 2022.
[32] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.



WENSHAO YANG received the Bachelor of Science degree from Nanchang University, Nanchang, China, in 2020. He is currently pursuing the master's degree with the School of Science, Yanshan University. His research interests include the fields of information security, federated learning, and blockchain.

PENGFEI KANG received the Bachelor of Science degree from Yanshan University, Qinhuangdao, China, in 2021, where he is currently pursuing the master's degree with the School of Science. His research interests include the fields of blockchain and game theory, with a current focus on blockchain and its applications.

CHAO WEI (Member, IEEE) received the Ph.D. degree from the School of Computer Science, Tohoku University, Japan, in 2015. He is currently an Associate professor with the School of Science, Yanshan University. His research interests include blockchain, network security, and privacy protection.