Defense against Local Model Poisoning Attacks to Byzantine-Robust Federated Learning

Shiwei Lu, Ruihu Li (✉), Xuan Chen, Yuena Ma
Department of Basic Sciences, Air Force Engineering University, Xi'an 710051, China

Abstract Federated learning is widely applied to jointly train a machine learning model between different organizations without violating data privacy. Since its training process is vulnerable to Byzantine attack, some Byzantine-robust aggregation rules are proposed to enhance the robustness of learning process, like Krum, Trimmed-mean, and Median. Recently, a new attack method, local model poisoning attack, is proved to be successful in attacking Byzantine-robust federated learning. Although two defense schemes are given simultaneously, high-quality validation dataset and high time cost are necessary for successful defense. In this paper, we analyze the characteristic and vulnerability of local model poisoning attack, and propose a defense scheme with four steps to further enhance the robustness of federated learning under adversary environment. Our defense scheme is proved to successfully defend against local model poisoning attack on Krum, Trimmed-mean, and Median aggregation rules. Experiments and performance analysis show that compared to benchmark defense schemes, our scheme has the following advantages: (1) no need for validation dataset (2) the lowest time cost (3) the strongest detection ability (4) stable detection performance. In some cases, our defense can precisely detect 100% malicious models and remove them from all submitted models. Compared to existing defense schemes, our defense scheme also reduces the detection time from 6.3s to 0.42s.

Keywords Federated learning, Byzantine-robust aggregation rule, Local model poisoning attack, Full knowledge attack, Partial knowledge attack, Defense scheme

1 Introduction

As a new mode of distributed learning, Federated Learning (FL) helps multiple organizations or clients to jointly train an artificial intelligence model without sharing their own dataset. Compared with the model trained by each client alone, a high-accuracy federated model can be obtained after multiple communication rounds in FL. Generally, a master device and many client devices are required to complete FL task, where the master device distributes an initial global model to client devices and clients update their local models with own datasets. After collecting all local models, the master device aggregates them with a certain aggregation rule to form a new global model. By multiple training and aggregations, a desired federated model can be obtained in the master device. Due to the characteristics of privacy protection and distributed learning, FL has been applied in many fields, such as the elementary prognosis of pandemic diseases [1], smart manufacturing systems [2], human-robot collaborative environments [3], quantum machine learning [4], 5G network [5] and etc.

In FL, the mean or Federated-Average (Fed-Avg) aggregation rule [6, 7] are most used under non-adversarial environment to calculate the mean model of local models as the global model. However, these two aggregation rules are vulnerable to adversarial attacks, such as data poisoning attack [8-10] and model poisoning attack [11-14]. Attackers usually make a data poisoning attack by deliberately disturbing the distribution of the original data or injecting poisoning

Received Month dd, yyyy; accepted Month dd, yyyy

E-mail: llzsy2015@163.com

data to indirectly change model parameters and make the accuracy of federated model reduced. However, in model poisoning attack, attackers manipulate some devices to modify model parameters or directly craft new model. For these non-robust federated learnings, an attack initiated by a single client can cause the failure of FL, let alone Byzantine attack (multiple attackers). Since Byzantine attacks can be viewed as worst-case untargeted malicious attacks, researchers usually make efforts to defend Byzantine attack in FL [15]. To make FL robust to malicious attack, several Byzantine-robust aggregation rules were proposed to defend against Byzantine failures, including Krum [16], Bulyan [17], trimmed mean [18], and median [18]. By computing the similarity between local models or model parameters, Byzantine-robust aggregation rules select parameters of benign models as much as possible to form the global model. Although these Byzantine-robust aggregation rules are provably robust against Byzantine failures, recent research in [19] shows that local model poisoning attack can still attack them successfully.

In local model poisoning attack, attacker is assumed to control some client devices and aims to craft malicious models on these controlled devices, which can cause negative impact on aggregation process. The method of crafting malicious models is transferred as an optimization problem and attack can be successfully realized by solving the optimization problem. Specifically, attacker crafts malicious model parameters on controlled devices to make aggregated model parameters deviate from the normal change direction. As a result, FL can not obtain higher accuracy of the global model via frequent training. Like defenses against data poisoning attacks [20-23], two defenses against local model poisoning attack, Error Rate based Rejection (ERR) and Loss Function based Rejection (LFR), are given in [19] to avoid the negative influence of malicious models. However, there are some limitations on performances of ERR and LFR. First, the validation dataset is required to calculate error rate impact or loss impact of each local model. If the validation set in the master device is not effective to verify all local models, then it leads to poor defense effect. Second, the transferability of ERR and LFR is limited between different attacks. For example, LFR can completely defend against local model poisoning attack on median, while it gets 0.58 error rate on Krum attack, which still remains 314% relatively increase to that on no attack (0.14 error rate). Third, the time spent on validation is calculated within the total time of FL and it may seriously affect the efficiency of FL. For example, Google uses a FL platform with 1.5 million clients to predict the next word in Google Board and takes totally 5 days, where parameters of the global model are 1.4 million and the model achieves convergence after 3000 rounds. Assume that we use ERR or LFR to remove malicious models from all local models, and the validation time of each local model in the master device is 10^{-4} seconds. It requires extra 5 days ($1.4\times10^{6}\times10^{-4}\times3000/(60\times60\times24)$) to complete the task, which brings large time cost to FL.

Besides the two defenses abovementioned, Spectral Anomaly Detection (SAD) is another effective method to detect anomalous image data and time series data in learning tasks [24-26]. By pre-training a SAD model with a prepared dataset and using this model for learning task, abnormal data in training dataset can be well detected. As an alternative defense method against model poisoning attack [27], Li et. al used a Variational Autoencoder (VAE) in FL as the SAD model to classify malicious models and benign models in FL. Specially, an auxiliary dataset in the master device is prepared in advance and the VAE model is pre-trained by the auxiliary dataset. When the master device collects all local models, it uses the pre-trained VAE model to detect anomalous model. Lin et. al recently combined a Deep Autoencoding Gaussian Mixture Model (DAGMM) [28] with standard deviation of Free-rider model in FL to detect Free-rider attack, which particularly works well for anomalous model detection [29].

Inspired by researches in [27-29], we attempt to use a VAE and a DAGMM as the SAD model respectively in FL to detect malicious models in local model poisoning attack, but the detection effects are inexpectant. One reason is that the auxiliary dataset may have no effect on training a SAD model with high accuracy. Since the datasets of clients in FL are protected locally, it is

difficult for the master device to collect a high-quality dataset to pre-train SAD model. The other reason is that malicious models in local model poisoning attack are very similar to benign models, which makes it difficult for SAD to get an effective decision boundary.

To make the defense independent on auxiliary dataset and reduce its time cost, in this paper, we propose three different defense schemes against local model poisoning attack on Krum, Trimmed-mean, and Median aggregation rule respectively. Since these three defense schemes are conducted according to the following four steps: Pre-aggregation, Detection, Removal, Re-aggregation (PDRR), they are called defense paradigm *PDRR* for short. Concretely, in a certain attack, the master device requires to pre-aggregate all local models to form a pre-aggregated model and obtain statistical information for detection. Then, the master device distinguishes malicious models from benign models with detection information and removes those malicious models. Further, the remained models are re-aggregated to form a new global model for FL in the next round. Note that since our detection schemes are based on attack characteristics, PDRR can help the master device to detect malicious model without auxiliary dataset and remove them from all received local models.

In summary, our contributions can be summarized as follows:

- We theoretically analyze the characteristics and vulnerabilities of existing local model poisoning attacks in FL. Based on attack characteristics, we further propose an effective defense paradigm against these attacks, which includes three specific defense schemes for Krum attack, Trimmed-mean attack, and Median attack.
- We respectively design different detection method in each defense scheme to detect malicious models crafted by attackers. Compared to existing detection methods, our detection method is the first one to detect malicious models in local model poisoning attack without auxiliary dataset.
- We give experiment results based on LEAF federated framework and real-world dataset [30] to verify the detection performance of our defense schemes. Compare to benchmark defenses, our defense can obtain the lowest false alarm rate (FAR) and the lowest missing alarm rate (MAR). Moreover, the time cost of our defense schemes is also the least.

The paper is organized as follows. In Section II, preliminaries are introduced, including FL, Byzantine-Robust aggregation rules, local model poisoning attack. In Section III, the basic method, characteristics and vulnerabilities of local model poisoning attack based on Krum, Trimmed-mean, and Median are analyzed. Section IV gives defense schemes against local model poisoning attacks. Performance evaluation is discussed in Section V. Finally, Section VI concludes the paper.

2 Preliminaries

In this section, we mainly introduce some concepts related to our work, including Federated learning, Byzantine-Robust aggregation rules, and local model poisoning attack.

2.1 Federated learning

2.1.1 FL Architecture

Federated learning (FL) is a machine learning framework, which can effectively help multiple organizations in data usage and machine learning modeling under the requirements of user privacy protection and data security. There are usually two roles in FL, including a server and many clients, where clients train local models by their owned dataset and the server is responsible for allocating the global model and aggregating local models. Particularly, a Federated Learning process within a communication round is shown in Fig. 1.

Generally, we assume that there are N clients participating in FL, which are denoted by $\{\mathcal{F}_i\}_{i=1}^{N}$. Clients train local models by their own training datasets $\{\mathcal{D}_i\}_{i=1}^{N}$. As shown in Fig. 1, the server allocates the initial model to N clients who train local models and upload model information (weight information or gradient information) to the server (assuming N clients

participating each round of FL). After receiving all local models, the server aggregates them to form a new global model. Within a single communication round of FL, clients need to download the global model from the server and upload their learnt local models to the server once respectively. Through multiple communication rounds, a final global model can be obtained in the server and all clients can share the final global model for tasks.

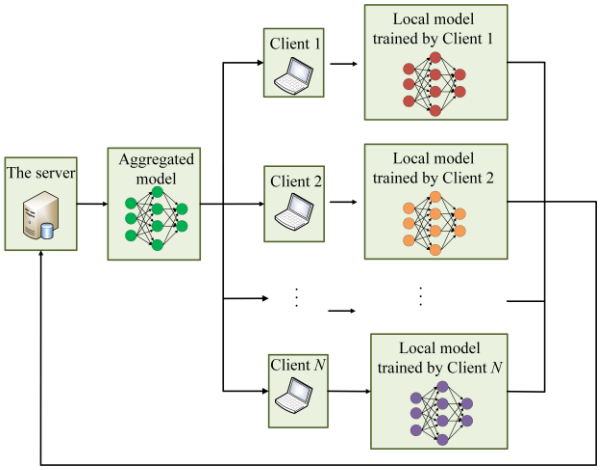


Fig.1 A Federated Learning process within a single communication round

2.1.2 Mean Aggregation Rule

Mean aggregation rule [31] is first proposed to aggregate the received local models in the server. In FL, assume that there are N clients participating entire learning process. Since clients may be able to join or leave at any time, not all clients need to participate in each round of FL. In addition, some clients may be not selected by the server for their bad training models. Without loss of generality, we assume that k $(2 \le k \le N)$ clients are selected to jointly train the federated model $\boldsymbol{w}_G$ in each round. In the rth communication round, the server sends the newest global model $\boldsymbol{w}_G^r$ to k selected clients. Then, client $\mathcal{F}_i$ sets its local model $\boldsymbol{w}_i^r$ as the same as the global model $\boldsymbol{w}_G^r$ and trains the local model with its own dataset $\mathcal{D}_i$. By solving an optimization problem $\min_{\boldsymbol{w}_i^r} f(\boldsymbol{w}_i^r)$, the local model $\boldsymbol{w}_i^r$ can be obtained, where $f(\boldsymbol{w}_i^r) = \mathcal{L}_f\left(\mathcal{D}_i; \boldsymbol{w}_i^r\right)$ is an objective function to measure how well the parameters $\boldsymbol{w}_i^r$ $(i = 1, \ldots k)$ model dataset $\mathcal{D}_i$. After k selected clients upload their local model $\{\boldsymbol{w}_i^r\}_{i=1}^k$, the server aggregates them by mean aggregation rule,

$$\boldsymbol{w}_G^{r+1} \leftarrow \frac{1}{k}\sum_{i=1}^{k} \boldsymbol{w}_i^r, \tag{1}$$

where $\boldsymbol{w}_G^{r+1}$ is the newest global model in the $r+1$ round. After multiple rounds of FL, the loss function or accuracy of the global model can converge. As a variant of the mean aggregation rule, Fed-Avg is proposed by Google to aggregate local models, which are trained by Non-IID and unbalanced dataset [6].

2.2 Byzantine-Robust aggregation rules

The mean aggregation rule is an effective aggregation rule in the non-adversary environment, but it is easy to be attacked in the adversary environment. Even if a single client is manipulated and its model parameters are modified deliberately, the entire FL may fail. As a more advanced attack, Byzantine attack usually controls multiple clients (less than $k/2$) to conduct data poisoning attack or model poisoning attack, which brings the great threat to FL. To defend against malicious attacks, especially Byzantine attack, some Byzantine-Robust aggregation rules are proposed in FL, including Krum, Trimmed-mean, Bulyan, and Median.

2.2.1 Krum

By calculating the similarity between models, Krum finally selects a model the most similar to other models as the global model. In this way, even if the selected model is malicious model, its impact can be constrained to a certain limit. Assume that there are c compromised devices in FL and their local models are crafted by attackers. For each local model $\boldsymbol{w}_i$, the server selects $k-c-2$ other local models which have the smallest Euclidean distance to $\boldsymbol{w}_i$ and then calculates the sum of square distance between $\boldsymbol{w}_i$ and its closest $k-c-2$ local models. Eventually, the local model with the smallest sum of square distance is selected as the global model.

2.2.2 Trimmed-mean and Bulyan

In FL, we assume that there are d parameters in the

global model. Since the structures of all local models are the same as that of the global model, each of local models also has d parameters. In trimmed-mean rule, the server first sorts parameters at the same position of all local models and trims parameters off-centered. The remained parameters are used to calculate the mean as the global model parameter. For instance, the server first sorts the jth parameter of k local models as $w_{1j}, w_{2j}, \ldots w_{kj}$, where w_{ij} is the jth parameter of local model $\boldsymbol{w}_i$. Afterwards, the largest and the smallest β parameters are trimmed and the jth parameter of the global model can be obtained by calculating the mean of $k-2\beta$ remained parameters.

Bulyan is an aggregation rule combining Krum and a variant of Trimmed-mean, which first uses Krum rule iteratively to select θ ($\theta \le k-2c$) local models and then sorts the parameter at the same position of all θ local models. By computing the mean of γ ($\gamma \le k-2c$) parameters the closest to the median, the global model parameter can be obtained. Compared with Krum, Bulyan is less affected by abnormal model parameters.

2.2.3 Median

Just like Trimmed-mean, Median first sorts the jth parameter of k local models as $w_{1j}, w_{2j}, \ldots, w_{kj}$ and selects the median of these parameters as the jth parameter of the global model. Specially, if k is an even number, median needs to calculate the mean of the middle two parameters.

2.3 Local model poisoning attack

Byzantine-Robust aggregation rules abovementioned are effective to avoid negative impacts of malicious models on the aggregated model. However, these malicious models are considered as abnormal values, which either has large Euclidean distances with other benign models or has parameter away from the central model parameter.

In local model poisoning attack, attackers are assumed to have capability to control some client devices and manipulate the local model parameters sent from these devices to the server. Concretely, like Sybil attack in distributed system [32], attackers may inject c faked devices into FL system or compromise c benign client devices. For simplicity, these devices are called *compromised devices*. Afterwards, attackers attempt to craft malicious models on compromised devices and attack aggregation process iteratively. The idea of crafting malicious models is transformed into an optimization problem. First, the change direction of global model is required. In FL, compromised devices can receive the global model at each round. By calculating $\boldsymbol{s}^r = \boldsymbol{w}_G^r - \boldsymbol{w}_G^{r-1}$, attackers can obtain change direction vector $\boldsymbol{s}^r$ at the current round. For simplicity, we abbreviate $\boldsymbol{s}^r$ as $\boldsymbol{s}$. Assume that s_j is the change direction of the jth parameter of $\boldsymbol{s}$, where $s_j = -1$ denotes that the jth parameter of global model relatively decreases at the current round, and in contrast $s_j = 1$ means the jth parameter relatively increases. Notice that $s_j = 0$ is exceptional case (not mentioned in Ref. [19]) and denotes the jth parameter of global model is unchanged within the two communication rounds. To avoid being detected as abnormal parameters, it is better for attacker to keep these parameters unchanged in the crafted models. Then, attacker uses change direction vector $\boldsymbol{s}$ and k benign models $\boldsymbol{w}_1 \ldots, \boldsymbol{w}_c, \boldsymbol{w}_{c+1}, \ldots, \boldsymbol{w}_k$ to craft malicious models $\boldsymbol{w}_1', \boldsymbol{w}_2', \ldots \boldsymbol{w}_c'$ by solving the optimization problem (2), where $\boldsymbol{w}_1 \ldots, \boldsymbol{w}_c$ are c benign models training on compromised device without attack and $\boldsymbol{w}_1', \boldsymbol{w}_2', \ldots \boldsymbol{w}_c'$ are malicious models crafted by attackers.

$$\begin{aligned} &\max_{\boldsymbol{w}_1', \ldots, \boldsymbol{w}_c'} \boldsymbol{s}^{\mathrm{T}}(\boldsymbol{w} - \boldsymbol{w}'), \\ s.t. \quad &\boldsymbol{w} = \mathcal{A}(\boldsymbol{w}_1 \ldots, \boldsymbol{w}_c, \boldsymbol{w}_{c+1}, \ldots, \boldsymbol{w}_k), \\ &\boldsymbol{w}' = \mathcal{A}(\boldsymbol{w}_1', \ldots, \boldsymbol{w}_c', \boldsymbol{w}_{c+1}, \ldots, \boldsymbol{w}_k), \end{aligned} \tag{2}$$

where $\mathcal{A}(\cdot)$ is the selected aggregation rule, and $\boldsymbol{w}$, $\boldsymbol{w}'$ are respectively column vector of the aggregated models. Since attackers can control c compromised devices and manipulate the local models on these compromised devices, $\boldsymbol{w}_1 \ldots, \boldsymbol{w}_c$ are known to attackers and $\boldsymbol{w}_1', \boldsymbol{w}_2', \ldots \boldsymbol{w}_c'$ can be crafted arbitrarily by attackers. In Ref. [19], according to the knowledge whether attacker knows local models on $k-c$ benign devices

($\boldsymbol{w}_{c+1},\ldots,\boldsymbol{w}_k$) or not, attack can be divided into two categories: full-knowledge attack and partial-knowledge attack. In full-knowledge attack, attackers can obtain $\boldsymbol{w}_{c+1},\ldots,\boldsymbol{w}_k$ on every benign device. Although full-knowledge attack has limited applicability in practice of FL, it is used to estimate the upper bound of threats of local model poisoning attack for a given setting of federated learning. In partial-knowledge attack, since local models $\boldsymbol{w}_{c+1},\ldots,\boldsymbol{w}_k$ on benign devices are unknown, attacker needs to use local models $\boldsymbol{w}_1\ldots,\boldsymbol{w}_c$ on compromised devices to craft malicious models. Specific attack methods for each aggregation rule are introduced in Section 3. Intuitively, vector $\boldsymbol{w}-\boldsymbol{w}'$ in (2) represents the influence of attack on the global model. The item $\boldsymbol{s}^T(\boldsymbol{w}-\boldsymbol{w}')$ decides the attack capability of crafted malicious models.

In (2), whether attackers know the aggregation rule $\mathcal{A}(\cdot)$ or not affects their attack capability [14]. If the aggregation rule is known, attack is usually specific and can bring seriously negative impact to FL. Specifically, the organization serving for FL may publicly announce its aggregation rules, such as Fed-Avg in Google, or attackers can obtain the aggregation rule from the internal staff of the organization or by Advanced Persistent Threat (APT) attack [33]. For unknown aggregation rules, attackers conduct a non-specific attack and craft local models on compromised devices based a randomly selected aggregation rule. Since local model poisoning attack has a certain transferability between different aggregation rules, it can also increase error rate of FL model. Generally, the effect of non-specific attack is weaker than specific attack. For instance, Trimmed-mean attack can obtain 0.25 error rate (108% relative increase to no attack scenario) based on Trimmed-mean rule, while Krum attack only gets 0.15 error rate (25% relative increase) on trimmed-mean rule.

3 Analysis of attack methods

In this section, we analyze model poisoning attacks based on Krum, Trimmed-mean, and Median aggregation method. Since Bulyan is the combination of Krum and a variant of Trimmed mean, attack on Krum can transfer to Bulyan. Analysis consists of basic method, characteristic and vulnerability of each attack. Importantly, each of attack method is from [19], we give the basic method just to convenient for further analysis.

3.1 Attack on Krum

3.1.1 Basic method

Krum selects one of local models as the global model, where attack aims to craft local models on compromised device and make one of crafted models selected by Krum rule. The attack goal is to make the global model deviate correct change direction. Specifically, assume that there are c malicious models $\boldsymbol{w}_1',\ldots,\boldsymbol{w}_c'$ under attack and k benign models $\boldsymbol{w}_1\ldots,\boldsymbol{w}_c,\boldsymbol{w}_{c+1},\ldots,\boldsymbol{w}_k$ when there is no attack.

Full knowledge: In this scenario, attacker knows the information of benign models (i.e., $\boldsymbol{w}_1\ldots,\boldsymbol{w}_c,\boldsymbol{w}_{c+1},\ldots,\boldsymbol{w}_k$). To ensure optimization problem (2) solvable, malicious model $\boldsymbol{w}_1'$ is restricted as $\boldsymbol{w}_1'=\boldsymbol{w}_{\mathrm{Re}}-\lambda\boldsymbol{s}$ ($\lambda>0$), where $\boldsymbol{w}_{\mathrm{Re}}$ is the global model received by clients at the current round. The other $c-1$ malicious models are required to be very close to $\boldsymbol{w}_1'$. To implement the attack, the other $c-1$ malicious models are assumed as the same as $\boldsymbol{w}_1'$. By solving problem (2) and obtaining $\boldsymbol{w}_1'$, $c-1$ models are randomly sampled within the distance ε to $\boldsymbol{w}_1'$. In addition, to make $\boldsymbol{w}_1'$ selected by Krum, it just needs to ensure $\boldsymbol{w}_1'$ is similar to $k-c-2$ benign local models. With these approximations, the optimization problem (2) is transferred as follows,

$$\begin{aligned} &\max_{\lambda}\ \lambda,\\ s.t.\ \ &\boldsymbol{w}_1'=krum(\boldsymbol{w}_1'\ldots,\boldsymbol{w}_c',\boldsymbol{w}_{c+1},\ldots,\boldsymbol{w}_k),\\ &\boldsymbol{w}_1'=\boldsymbol{w}_{\mathrm{Re}}-\lambda\boldsymbol{s},\\ &\boldsymbol{w}_i'=\boldsymbol{w}_1',\quad i=2,3\ldots c. \end{aligned} \tag{3}$$

In Eq. (2), $\boldsymbol{s}^{\mathrm{T}}(\boldsymbol{w}-\boldsymbol{w}')=\boldsymbol{s}^{\mathrm{T}}[\boldsymbol{w}-(\boldsymbol{w}_{\mathrm{Re}}-\lambda\boldsymbol{s})]=\boldsymbol{s}^{\mathrm{T}}(\boldsymbol{w}-\boldsymbol{w}_{\mathrm{Re}})+\lambda\boldsymbol{s}^{\mathrm{T}}\boldsymbol{s}$, where $\boldsymbol{s},\boldsymbol{w}$ and $\boldsymbol{w}_{\mathrm{Re}}$ are already known and thus $\boldsymbol{s}^{\mathrm{T}}(\boldsymbol{w}-\boldsymbol{w}_{\mathrm{Re}})$ is a constant. In addition, $\boldsymbol{s}^{\mathrm{T}}\boldsymbol{s}=d$ is also a constant, where d is the number of global model parameters. In the Ref. [19], the upper bound of λ is given and proved as follows,

$$\lambda \le \sqrt{\frac{1}{(k-2c-1)d} \cdot \min_{c+1 \le i \le k} \sum_{l \in \tilde{\Gamma}_{w_i}^{k-c-2}} D^2(\boldsymbol{w}_l, \boldsymbol{w}_i)} + \frac{1}{\sqrt{d}} \cdot \max_{c+1 \le i \le k} D(\boldsymbol{w}_i, \boldsymbol{w}_{\mathrm{Re}}), \tag{4}$$

where $D(\boldsymbol{w}_l, \boldsymbol{w}_i)$ is the Euclidean distance between $\boldsymbol{w}_l$ and $\boldsymbol{w}_i$, $\tilde{\Gamma}_{w_i}^{k-c-2}$ is the set of $k-c-2$ benign models that have the smallest Euclidean distance to $\boldsymbol{w}_i$.

Binary search is used to solve the optimization problem until a suitable value of λ is obtained. If the search interval is small enough and λ is unsolved, attack fails.

Partial knowledge: In reality, since benign devices are not manipulated, attacker usually does not know local models on benign devices. Thus, the vector of change direction $\boldsymbol{s}$ can not be obtained. In partial knowledge attack, attacker uses before-attack models on compromised devices to calculate the mean $\boldsymbol{w} = \frac{1}{c}\sum_{i=1}^{c} \boldsymbol{w}_i$ as an estimation of global model obtained at the current round. Then, the estimation of change direction can be computed by $\tilde{\boldsymbol{s}} = \boldsymbol{w} - \boldsymbol{w}_{\mathrm{Re}}$ and the estimation is used to solve the optimization problem (2).

Different from full knowledge attack, partial knowledge attack only adds a single malicious model at the beginning to solve the optimization problem, which is shown as follows,

$$\begin{aligned} &\max_{\lambda} \lambda, \\ s.t. \quad &\boldsymbol{w}_1' = krum(\boldsymbol{w}_1', \boldsymbol{w}_1, \ldots, \boldsymbol{w}_c), \\ &\boldsymbol{w}_1' = \boldsymbol{w}_{\mathrm{Re}} - \lambda \tilde{\boldsymbol{s}}. \end{aligned} \tag{5}$$

Binary search is also applied in partial knowledge scenario to solve λ. If the search interval is small enough and λ is unsolved, an additional malicious model $\boldsymbol{w}_2' = \boldsymbol{w}_1'$ is added to resolve the optimization problem. The process is repeated until finding a solution of λ. After solving λ, malicious model $\boldsymbol{w}_1'$ can be obtained and the other $c-1$ malicious models are randomly sample within the distance ε to $\boldsymbol{w}_1'$.

3.1.2 Attacking characteristics

In Krum attack, malicious model $\boldsymbol{w}_1'$ is selected by Krum aggregation rule with the assistance of the other $c-1$ malicious models. Since $\boldsymbol{w}_1' = \boldsymbol{w}_{\mathrm{Re}} - \lambda \boldsymbol{s}$ and $Euc(\boldsymbol{w}_1', \boldsymbol{w}_i'), \quad i = 2, \ldots, c$, (Euclidean distance) is less than ε, the parameters at the same position of malicious models are very close. Figure 2 gives the first and the second parameters of each model.

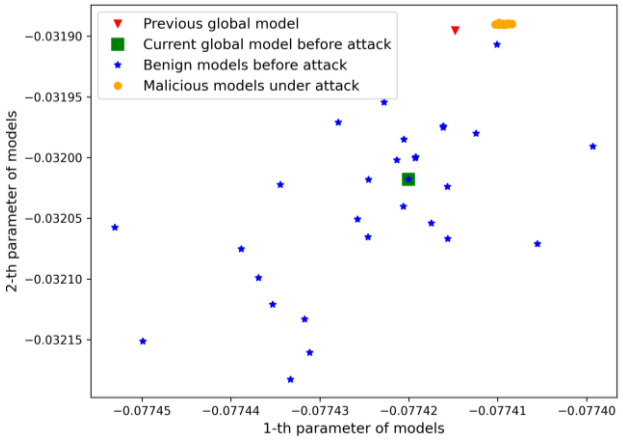


Fig.2 The first and the second parameters of each model on FEMNIST dataset: (1) 50 clients participating FL, 30 benign clients and 20 compromised clients (2) Red point is the global model aggregated by Krum in the round 1(previous round), green point is the global model aggregated in round 2 (current round) when there is no attack, blue points are 30 benign models, and yellow points are 20 malicious models. (3) $\boldsymbol{w}_1'$ is one of yellow point as the current global model under full-knowledge attack (4) ε is set as 10^{-6}.

As shown in Fig. 2, malicious model parameters are distributed at the right top of the previous global model, while benign model parameters are mostly distributed at the left bottom. Although some benign model parameters are at the right bottom, there is no benign model parameter distributes at the area (right top) where malicious model parameters are.

For this phenomenon, it mainly because benign models are iteratively obtained by the optimizer and training dataset, which aims to get the optimal aggregated parameter set, while malicious models are crafted by attacker and set up to prevent global model from reaching optimal aggregated parameter set. Thus, benign model parameters iteratively shift in the same direction relative to the previous global model (target direction of optimization), while malicious model parameters are set in opposite position to block

optimization.

Although a relatively large ε can bring large disturbance to malicious models and changes the parameter position of $\boldsymbol{w}_2', \ldots, \boldsymbol{w}_c'$, it makes the sum square distance of $\boldsymbol{w}_1'$ increase and reduces success rate of attack. Thus, a relatively small ε is a popular selection for attacker under Krum attack.

3.1.3 Attacking vulnerability

Suppose that the first c local models among the k models are malicious models. We set the previous global model (red point in Fig. 2) as the base model and calculate the change direction $\boldsymbol{s}_{bi}, i = 1, 2, ..., k$ between local models and the base model. $\boldsymbol{s}_{bi}$ is a vector with d elements and jth element of $\boldsymbol{s}_{bi}$ represents the change direction of jth parameter in local model $\boldsymbol{w}_i$ relative to that in the base model. Since the other $c-1$ malicious models are randomly sampled within the distance ε to $\boldsymbol{w}_1'$, the change direction $\boldsymbol{s}_{bi}, i = 2, ..., k$ and $\boldsymbol{s}_{b1}$ are usually very similar or highly correlated. In other words, the change directions between $\boldsymbol{s}_{bi}, i = 2, ..., c$ and $\boldsymbol{s}_{b1}$ are more similar than those between $\boldsymbol{s}_{bi}, i = c+1, c+2, ..., k$ and $\boldsymbol{s}_{b1}$. Therefore, it is an effective method to distinguish malicious model from benign model by change direction $\boldsymbol{s}_{bi}, i = 1, 2, ..., k$.

3.2 Attack on Trimmed-mean

3.2.1 Basic method

Trimmed-mean sorts the jth parameter of all local models and trims off-centered parameters. Then, the aggregated model is calculated by averaging remained parameters. Attack on trimmed-mean aims to solve the optimization problem by crafting malicious model parameters that can reverse the change direction of global model parameters. Let $w_{max,j}$ and $w_{min,j}$ respectively denote the maximum and minimum of the jth parameter in benign models. Thus, $w_{max,j} = \max\{w_{(c+1)j}, w_{(c+1)j}, \ldots, w_{kj}\}$ and $w_{min,j} = \min\{w_{(c+1)j}, w_{(c+1)j}, \ldots, w_{kj}\}$.

Full knowledge: Full-knowledge attack is proposed to obtain an optimal solution of Eq. (2) by the following way: if $s_j = -1$, then c parameters of malicious model are crafted by randomly sampling in the interval $[w_{max,j}, b \cdot w_{max,j}]$ (when $w_{max,j} > 0$) or $[w_{max,j}, w_{max,j}/b]$ (when $w_{max,j} \le 0$), where b is a constant and $b > 1$. If $s_j = 1$, parameters of malicious model are randomly sampled in the interval $[w_{min,j}/b, w_{min,j}]$ (when $w_{min,j} > 0$) or $[b \cdot w_{min,j}, w_{min,j}]$ (when $w_{min,j} \le 0$). Specially, if $s_j = 0$, the jth parameters in malicious models are set as the same as that in the previous global model.

Partial knowledge: In partial knowledge scenario, malicious models are crafted based on before-attack local models on compromised devices. First, the mean model is computed by $\tilde{\boldsymbol{w}} = \left(\sum_{i=1}^{c} \boldsymbol{w}_i\right) / c$. Then, the vector of change directions $\tilde{\boldsymbol{w}}$ are estimated by the mean model. The maximum $w_{max,j}$ and minimum $w_{min,j}$ of benign model parameters are also estimated by before-attack local models on compromised devices. Especially, attacker calculates the mean μ_j and standard deviation σ_j of jth parameter on compromised devices. On the assumption that jth parameters of all benign models are sampled from a Gaussian distribution, $w_{max,j}$ is estimated as smaller than $\mu_j + 3\sigma_j$ or $\mu_j + 4\sigma_j$ and $w_{min,j}$ is estimated as larger than $\mu_j - 4\sigma_j$ or $\mu_j - 3\sigma_j$ with a large probability. Thus, when $s_j = -1$, jth parameters of c malicious models are randomly sampled in the interval $[\mu_j + 3\sigma_j, \mu_j + 4\sigma_j]$ to make the crafted parameters larger than the maximum of benign model parameters with a large probability; otherwise, malicious model parameters are randomly sampled in the interval $[\mu_j - 4\sigma_j, \mu_j - 3\sigma_j]$ to make the crafted parameters smaller than the minimum of benign model parameters.

3.2.2 Attacking characteristics

Take the jth parameter of all local models as an example, Fig. 3 shows the attack on trimmed-mean to explain why attack mentioned above is effective. In Fig. 3, jth parameters of all k models are sorted as a vector $\boldsymbol{w}_S$ with ascending order and w_S^i is the ith element of $\boldsymbol{w}_S$. Assume the number of trimmed parameters is 2β ($2\beta \ge c$) and w_S^M is the median in the sorted vector

$\boldsymbol{w}_S$. Before attack, there are β minimum parameter and β maximum parameters of benign models removed and the remained parameters near the median are averaged to obtain the aggregated model parameter, which is larger than the jth parameter of the previous global model (i.e. $s_j = 1$).

In local model poisoning attack, attacker attempts to craft malicious model parameter smaller than the minimum of benign model parameter. As a result, some minimum parameters of benign models that should have been removed are replaced by malicious model parameters. Malicious model parameters are all removed as minimum parameters in the trimmed-mean rule, which makes more maximum parameters of benign models removed and less minimum parameters of benign models remained. Therefore, the value of jth aggregated parameter under attack decreases

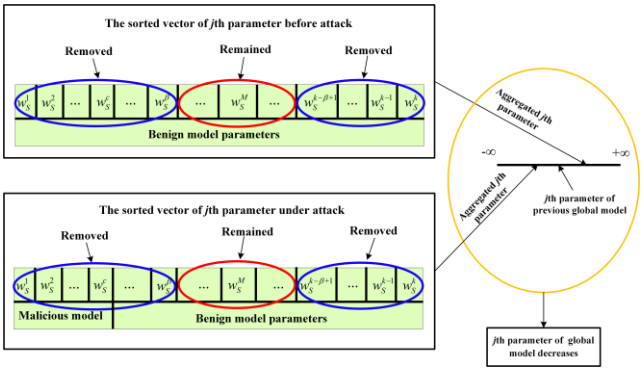


Fig.3 Attack on trimmed-mean

3.2.3 Attacking vulnerability

When $s_j = 1$ or $s_j = -1$, jth parameters of malicious models are crafted as c minimum or maximum among all local model parameters. Thus, if we ignore the parameter position where $s_j = 0$, malicious model parameters corresponding to $s_j = 1$ or $s_j = -1$ are likely to be removed with a large probability by trimmed-mean aggregation rule. In other words, malicious model has more removed parameters compared with those in benign models.

3.3 Attack on Median

Since median aggregation only selects the median as the parameter of the global model, it can be viewed as a special case of trimmed-mean aggregation by trimming all non-median parameters. Attack on Median is the same as that on Trimmed-mean.

Attacking characteristics and vulnerability: Attack on median crafts malicious model parameter as c minimum or maximum among all local model parameters. Thus, jth malicious model parameter ($j = 1,\ldots,d$ and $s_j \neq 0$) must have an extremely low probability (close to 0) to be selected by Median aggregation rule. By calculating the remained parameter number, we can distinguish malicious models from benign models.

4 Defense Schemes

The idea of defense schemes is based on the following two assumptions: (1) the differences of datasets on benign devices lead to the diversity of their local model parameters (2) benign model parameters are normally distributed around the corresponding aggregated model parameters, but malicious model parameters are crafted on the same side of aggregate model parameters. Based on the two assumptions, we propose a defense scheme with four steps against local model poisoning attacks, including pre-aggregation, detection, removal, and re-aggregation (PDRR). Since the aggregation rule and vulnerability of each attack are different, the details of pre-aggregation, detection and re-aggregation have some differences.

4.1 Defense against Krum attack

Pre-aggregation: The server saves the global model sent to selected clients as the previous global model $\boldsymbol{w}_p$. After all local models $\boldsymbol{w}_i$ ($i = 1,\ldots,k$) are trained and uploaded, the server aggregates all local models by Krum aggregation rule and obtains a pre-aggregated global model. We denote the pre-aggregated global model by $\boldsymbol{w}_{pag}$.

Detection: After getting $\boldsymbol{w}_{pag}$, the server calculates the vector of change direction $\boldsymbol{s}_{pag}^{p}$ between $\boldsymbol{w}_{pag}$ and

$\boldsymbol{w}_p$. If $\boldsymbol{w}_{pag}$ and $\boldsymbol{w}_p$ are all flattened vectors, then $\boldsymbol{s}_{pag}^p = \mathrm{Sgn}(\boldsymbol{w}_{pag} - \boldsymbol{w}_p)$ where Sgn(·) is the sign function. Meanwhile, the server computes change directions vectors $\boldsymbol{s}_i^p$ ($i = 1,...,k$) between local models $\boldsymbol{w}_i$ ($i = 1,...,k$) and $\boldsymbol{w}_p$. If Krum attack is successful, malicious model must be selected as the pre-aggregated global model. Since malicious models have the similar change directions, we can distinguish malicious models and benign models by calculating the similarities between vector $\boldsymbol{s}_i^p$ ($i = 1,...,k$) and the vector $\boldsymbol{s}_{pag}^p$. Here, the similarity is calculated by Pearson correlation coefficient and denoted by ρ_i^{pag} ($i = 1,...,k$), but other similarity measurement methods are also adopted. According to analysis of attack characteristics in section 3.1.2, we know change direction vectors between malicious models are highly similar. Thus, when malicious model is selected as pre-aggregated global model, other malicious models must have large correlation coefficients (almost close to 1). As shown in Fig. 4, *Pearson correlation coefficients* between $\boldsymbol{s}_i^p$ ($i = 1,...,k$) and $\boldsymbol{s}_{pag}^p$ is given. Correlation coefficient of malicious models are all close to 1, but those of benign models are less than 0.1. We can select a threshold $\lambda_D = q \cdot \max(\rho_i^{pag})$ to detect malicious model, where $q \in [0,1]$ is a threshold parameter. If $\rho_i^{pag} \geq \lambda_D$, then local model $\boldsymbol{w}_i$ is detected as the malicious model. In Fig. 4, any q in the range [0.1, 0.99] is a suitable threshold parameter to detect all malicious models.

Removal: Having detected malicious models, we can remove these malicious models from local models or directly reject to receive models from malicious clients.

Re-aggregation: After removing malicious models from all local models received, a new global is obtained by reaggregated remained models and used for FL at the next round.

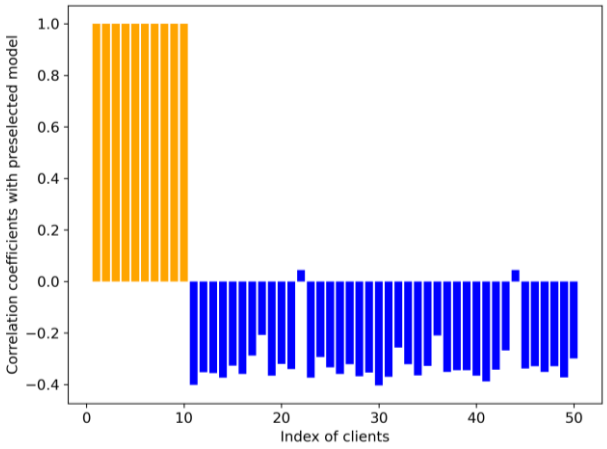


Fig.4 Pearson correlation coefficient between $\boldsymbol{s}_i^p$ ($i = 1,...,k$) *and* $\boldsymbol{s}_{pag}^p$: (a) FEMNIST dataset and CNN model are used in FL, where 1-10 clients are malicious clients controlled by attacker and 11-50 are benign clients (b) Detection scheme is deployed in the 1th round before aggregation (c) Full knowledge attack (d) Orange bars and blue bars are for malicious models and benign model respectively

4.2 Defense against Trimmed-mean attack

Pre-aggregation: The server first sorts each parameter at the same position of all local models and removes 2β parameters off-centered ($\beta \geq c$). Then, parameters of the pre-aggregated global model $\boldsymbol{w}_{pag}$ are obtained by averaging remained parameters. While sorting and trimming parameters, the server computes *the number of trimmed parameters* in each model. For example, if the number of trimmed parameters is denoted by n_i^c ($i = 1,2...,k$) and the trimmed parameter belongs to local model $\boldsymbol{w}_i (i = 1,2...,k)$, then $n_i^c = n_i^c + 1$.

Detection: We attempt to directly distinguish malicious models from benign models by the number of trimmed parameters in each model (i.e. classify the two classes by n_i). However, the result shown in Fig. 5 is not suitable for detection, because some benign models also have large number of trimmed parameters.

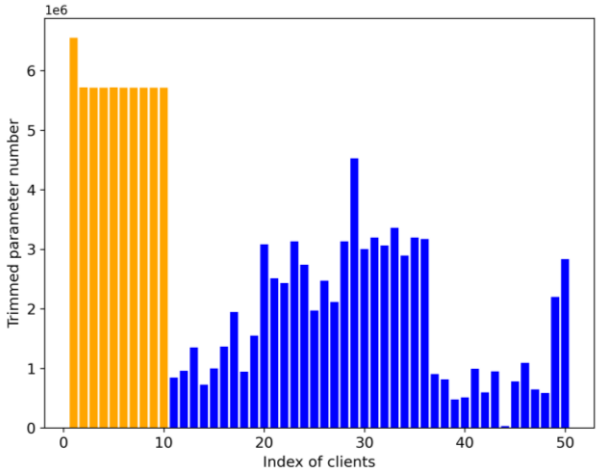


Fig. 5 The number of trimmed parameters in each model: (a) FEMNIST dataset and CNN model are used in FL, where 1-10 clients are malicious clients controlled by attacker and 11-50 are benign clients (b) Detection scheme is deployed in the 1th round before aggregation and trimmed parameter $\beta = 5$ (c) Full knowledge attack (d) Orange bars and blue bars are for malicious models and benign model respectively.

To further improve the detection, we recalculate the number of trimmed parameters in each model by ignoring the parameter in the *j*th position where $s^p_{pag,j} = 0$ ($s^p_{pag,j}$ is the *j*th element of $\boldsymbol{s}^p_{pag}$). When $s^p_{pag,j} = 0$, it means the *j*th parameter in malicious models are crafted near the center and thus there is no need to compute n_i^c containing the parameters at these positions. For simplicity, we call the parameters at these positions *zero change direction parameters*. Fig 6 shows the number of trimmed parameters in each model without zero change direction parameters. Obviously, the numbers of trimmed parameters without zero change direction parameter in malicious models are subequal and substantially differentiate from those in benign models. Therefore, we can detect malicious models by their high similarity and exception on trimmed parameter number. We can set a threshold $\lambda_D = q \cdot \max(n_i^c)$ to detect malicious model, where $q \in [0,1]$. If $n_i^c \geq \lambda_D$, local model $\boldsymbol{w}_i$ is regarded as the malicious model.

Removal: The server removes the malicious models detected and saves the remained models.

Re-aggregation: The server aggregates remained local models to form a new aggregated model as the global model.

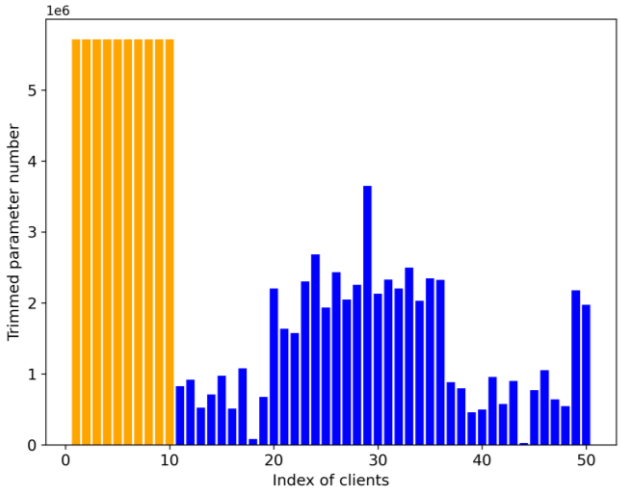


Fig. 6 The numbers of trimmed parameters in each model without zero change direction parameters: parameter settings are the same as those in Fig. 5

4.3 Defense against Median attack

Pre-aggregation: If the number of global model parameters d is odd, the server sorts each parameter of all local model and selects $\lceil d/2 \rceil$th parameter as the parameter of the pre-aggregated global model $\boldsymbol{w}_{pag}$; otherwise, $(d/2)$ th and $((d/2)+1)$ th parameters are selected to obtain the parameter of the pre-aggregated global model by averaging. While selecting median parameters, the server counts *the number of remained parameters* in each model (i.e. parameters selected by the server to calculate median). The number of remained parameters in each model is denoted by n_i^r ($i = 1, 2..., k$).

Detection: Since the parameters in malicious models with zero change direction are probably distributed near the median, we count the number of remained parameters without zero change direction parameters in each local model and the result is shown in Fig. 7.

Significantly, the numbers of remained parameters in malicious models are almost close to 0 and have obvious differences from those in benign models. Thus, we can use a detection threshold $\lambda_D = (1-q)\max(n_i^r)$ to detect malicious models, where $q \in [0,1]$. If $n_i^r \leq \lambda_D$,

local model w_i is regarded as the malicious model.

Removal: The server removes the malicious models detected and saves the remained models.

Re-aggregation: The server aggregates remained local models to form a new aggregated model as the global model.

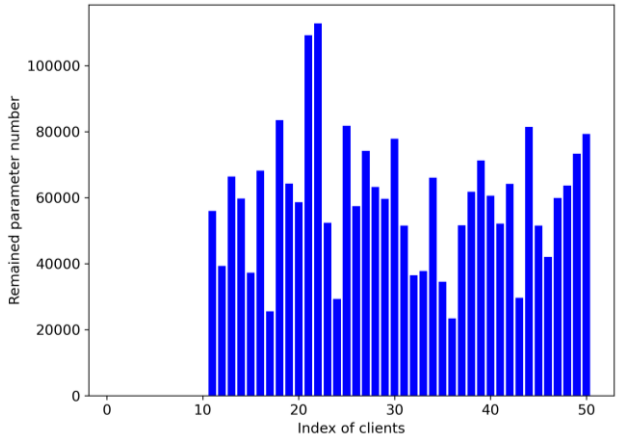


Fig. 7 The numbers of remained parameters in each model without zero change direction parameters: (a) FEMNIST dataset and CNN model are used in FL, where 1-10 clients are malicious clients controlled by attacker and 11-50 are benign clients (b) Detection scheme is deployed in the 1th round before aggregation (c) Full knowledge attack

5 Performance Evaluation

In this section, we apply our defense scheme (PDRR) on Krum attack, Trimmed-mean attack, and Median attack respectively. FL task is to train a CNN model on FEMNIST dataset [30] to classify the picture 0-9, A-Z and a-z. We add attack at the initial stage of FL and all attacks are set as partial knowledge attack. The benchmark defense schemes are set as VAE [27], DAGMM [28], ERR [19], and LFR [19]. Since our schemes and benchmark schemes are the same in removal and re-aggregation steps, we mainly compare the detection performance of these schemes. Finally, we analyze the performance of each scheme on local model poisoning attack, including (1) Dependency on auxiliary dataset (2) Time cost (3) Detection ability (4) Detection Stability. Our experiments are implemented with Tensorflow 1.4, RTX 2080Ti GPU, 24 Intel(R) Xeon(R) Gold-5118 2.30GHZ CPU, 1TB SSD and Ubuntu 18.04.

5.1 Experiment Setup

We consider a FL to complement classification task on FEMNIST dataset, which contains 801,074 data (picture with 28×28 pixels) sampling from 3,500 writers [30]. These data are classified as 62 categories including digits 0-9, characters a-z and A-Z. In LEAF federated framework [30], FEMNIST dataset is allocated to 192 clients with the heterogeneous setting, where each client has different training dataset and testing dataset. A CNN model is used for FL task, which contains two convolutional layers ($5\times5\times32$ and $5\times5\times64$) and two pooling layers with 2×2 filters and 2 strides, followed by a dense layer with 2048 units. The number of model parameters $d=6603710$.

Since parameters in malicious models are different from those in benign models, malicious model parameters can be viewed as anomaly points and detected by anomaly detection schemes, including VAE [27], DAGMM [28]. In addition, researchers in original article [19] gave two defense schemes, ERR and LFR, to reject the local model bringing the large error rate or large loss to global model. Since ERR and LFR are also two schemes first detecting malicious models and then remove malicious model from local models, we can compare all defense schemes with their detection performance.

Experiments are set on three aggregation rules (Krum, Trimmed-mean, and Median) and partial knowledge attack. The number of devices participating FL and compromised devices are set as $k=50$, $c=20$ (large c brings a high rate of success of Krum attack). The indexes of compromised devices are assumed as 1-20. In Krum attack, if search interval of λ is smaller than 10^{-9}, attack fails. When attack is successful and w_1' is obtained, the other malicious models are randomly sampled within $\varepsilon=10^{-7}$. Trimmed parameters number in Trimmed mean aggregation rule is assumed as $2\beta=20$. To ensure that crafted malicious models are as similar as benign models, constant b in Trimmed-mean attack and Median attack are set as 1.05.

Since similarities between local models are different in different communication rounds of FL, we execute defense schemes at the communication round 1, 20, 50, 100 respectively. Our defense scheme is executed as the introduction in section IV and benchmark detections are

executed with the following settings.

VAE [27]: In spectral anomaly detection, the server needs a dataset to pre-train the federated model (CNN model) and collects weight information of federated model. Then, weigh information is fitted into VAE model for training and pre-trained VAE is used to detect malicious models in local model poisoning attack. Thus, to train VAE model in the server, a task dataset should be prepared in advance. Considering that the task dataset is hard to obtain, we randomly sample half of the test dataset (4004 pictures and labels) of all clients to build the prepared dataset (2002 pictures and labels). Both the encoder and decoder in VAE model are set as two dense layers with 500 units, and the dimension of latent vector is 100. After inputting local models under attack, the pre-trained VAE model calculates the *reconstruction probability* of each model. Local models with high reconstruction probability are classified as malicious models.

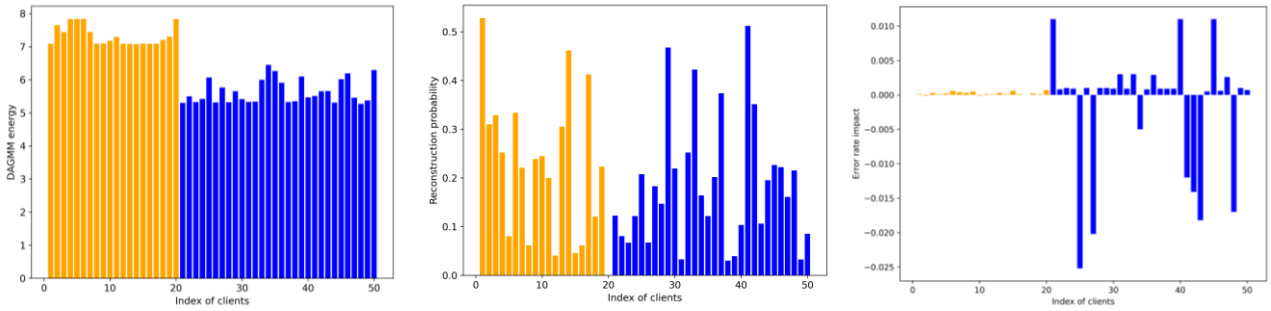
DAGMM [28]: DAGMM combines a deep autoencoder with a Gaussian mixture model to detect abnormal data by calculating low dimensional embedding, reconstruction error and density estimation. In our experiment, Deep autoencoder and Gaussian mixture model are both made by fully connected network with two hidden layers. Deep autoencoder has 128 units and Gaussian mixture model has 32 units. The training strategy and dataset of DAGMM are the same as those of VAE. Since DAGMM model is trained by normal model, it is difficult to reconstruct the malicious model through the pre-trained DAGMM and low dimensional embedding of the malicious model is also different from normal model. Thus, malicious models can be detected by pre-trained DAGMM with high *DAGMM energy*.

ERR and LFR [19]: Both of the two methods need a validation dataset in the server to verify the *error rate impact* or *loss function impact* of each local model. ERR labels local models as malicious models that have the largest impact on error rate. LFR labels local models that have the largest impact on loss function as malicious models. Here, we select c local models with the largest error rate impact or loss impact as malicious models. Validation dataset is built just like the pre-trained dataset for VAE model. For simplicity, we call datasets in the server for VAE, DAGMM, ERR and LFR *auxiliary dataset.*

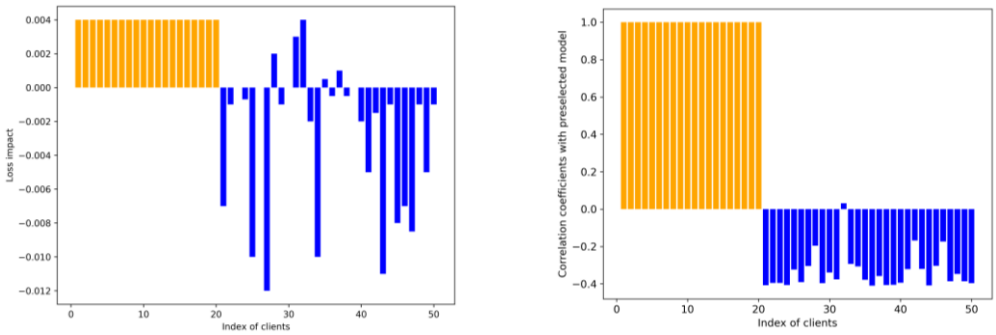
5.2 Detection performance analysis

After the server first receiving all local model at the round 1, it detects local models with all defense schemes. We first give the detection result of all schemes on three attacks respectively (at the round 1). Then, detection indictors of each scheme at different rounds are shown for comparison.

When all clients first train and upload their local models at the round 1, the server collects all local model and uses different detection schemes to detect malicious models. Fig.8-Fig.10 respectively shows the detection results on Krum attack, Trimmed-mean attack, and Median attack, where horizontal axes are clients index (1-20 are set as malicious clients and 21-50 are benign clients) and vertical axes of are index values used in different detection methods. The detection index values of benchmark methods are introduced in section 5.1 and those of our schemes are in section 4. In addition, all attacks in Fig.8-Fig.10 are partial-knowledge attacks.

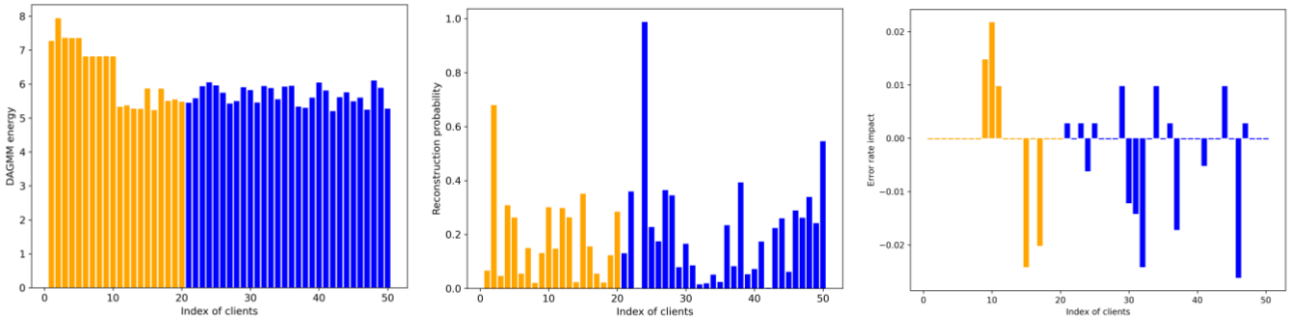


(a) (b) (c)

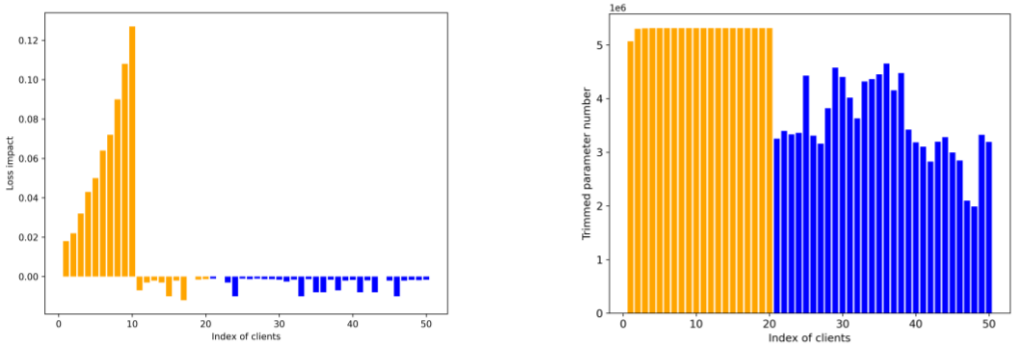


(d) (e)

Fig. 8 Detection results of defense schemes on Krum attack: (a) DAGMM (b)VAE (c) ERR (d) LFR (e) PDRR(our scheme)

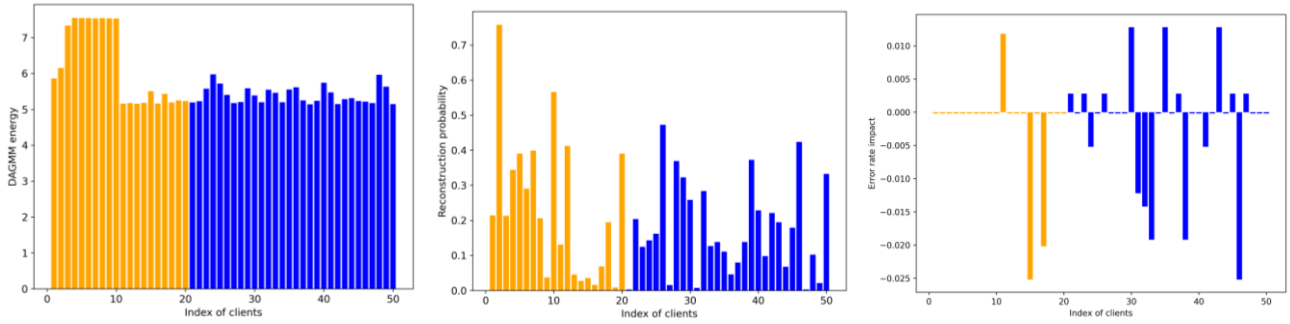


(a) (b) (c)

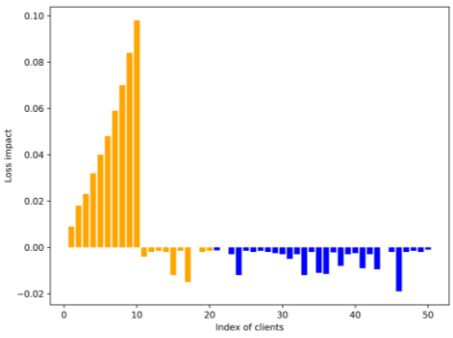


(d) (e)

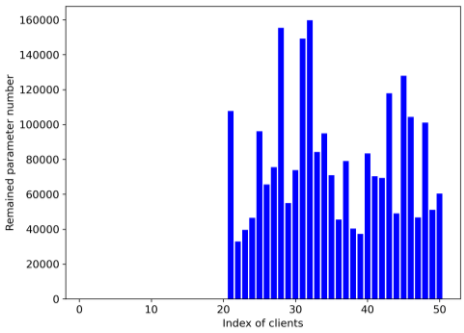
Fig. 9 Detection results of defense schemes on Trimmed-mean attack: (a) DAGMM (b)VAE (c) ERR (d) LFR (e) PDRR(our scheme)



(a) (b) (c)



(d)



(e)

Fig. 10 Detection results of defenses scheme on Median attack: (a) DAGMM (b)VAE (c) ERR (d) LFR (e) PDRR(our scheme)

As shown in Fig 8- Fig. 10, DAGMM, LFR and PDRR have better detection effects on Krum attack than ERR and VAE, while DAGMM and LFR have poor performance on Trimmed-mean attack and Median attack. Only our defense scheme (PDRR) can detect all malicious models in three attack scenarios. In other words, PDRR has the best detection ability and stability among all defense schemes.

Considering that local models vary in different rounds of FL, here we assume the benign model as a positive class (P) and the malicious model as a negative class (N) to further compare the detection performance of different defense schemes. Table. 1 gives false alarm rate (FAR) and missing alarm rate (MAR) of different defense schemes for Krum attack. The two indictors are calculated at round 1, 20, 50 and 100 of FL.

Table. 1 FAR and MAR of defense schemes for Krum attack

	FAR				MAR			
	1	**20**	**50**	**100**	**1**	**20**	**50**	**100**
DAGMM	0	0.33	0.5	0.5	0	0.5	0.75	0.75
VAE	0.27	0.4	0.43	0.3	0.4	0.6	0.65	0.45
ERR	0.43	0.4	0.43	0.5	0.65	0.6	0.65	0.75
LFR	0	0.1	0.13	0.2	0	0.15	0.2	0.3
PDRR	0	0	0	0	0	0	0	0

Table. 2 gives the two indictors of different defense schemes for Trimmed-mean attack. Since Median attack is a variant of Trimmed-mean attack, detection performances of defense schemes for Median attack are similar to those for Trimmed-mean attack.

Table. 2 FAR and MAR of different defense schemes for Trimmed-mean attack

	FAR				MAR			
	1	**20**	**50**	**100**	**1**	**20**	**50**	**100**
DAGMM	0.27	0.3	0.2	0.37	0.4	0.45	0.3	0.55
VAE	0.4	0.33	0.4	0.43	0.6	0.5	0.6	0.65
ERR	0.43	0.5	0.47	0.5	0.65	0.75	0.7	0.75
LFR	0.27	0.2	0.3	0.27	0.4	0.3	0.45	0.4
PDRR	0	0	0	0	0	0	0	0

Compared to benchmark schemes, our defense scheme can detect malicious model with 0 FAR and 0 MAR. In other words, malicious model can be accurately detected and removed with PDRR. We regard there are two reasons causing high FAR and MAR in benchmark schemes. One is that all benchmark methods need a pre-trained dataset or a validation dataset in the server. Thus, the selection of the pre-trained dataset or validation dataset affects the detection effect. The other is that all malicious models are crafted closely to benign models. Therefore, anomaly detections by DAGMM or VAE are usually difficult to directly distinguish malicious models from benign models. For ERR and LFR, although the validation dataset is perfect, the benign model may have a large impact on error rate or loss of the

aggregated model. Since PDRR uses the characteristics and vulnerability of local model poisoning attack to defend the attack, its performance does not depend on the selection of validation dataset. Thus, the detection of PDRR is the most stable in all defense schemes.

In PDRR, we can detect malicious models by calculating Pearson correlation coefficient (Krum attack), removed parameter number (Trimmed-mean attack), and remained parameter number (Median attack) respectively. The detection parameter q is necessary in each defense scheme. To compare the performance of PDRR between full-knowledge attack and partial-knowledge attack, we respectively give the q range in Krum attack, Trimmed-mean attack and Median attack in Fig 11. When we use these q value, malicious models can be completely detected.

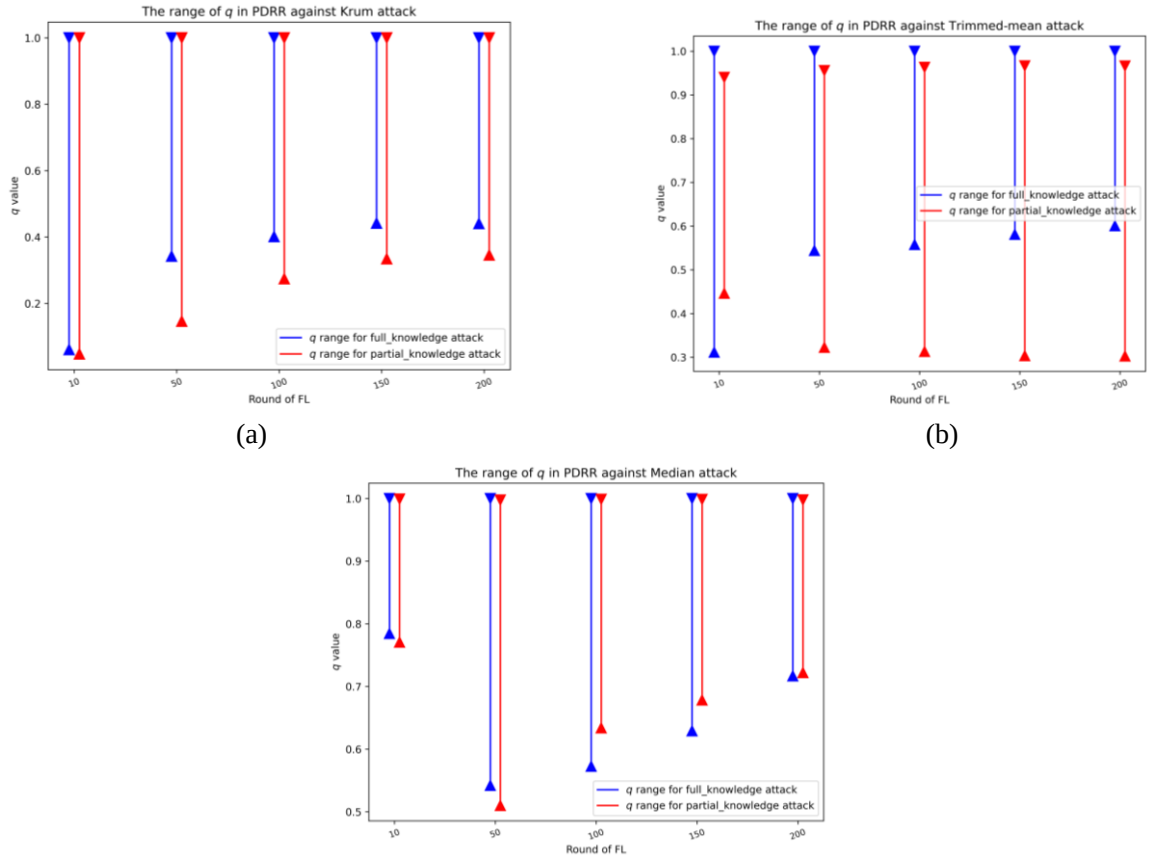


Fig. 11 q range of PDRR against full-knowledge attack and partial-knowledge attack: (a) Krum attack (b)Trimmed-mean attack (c) Median attack.

The length of q range represents the difference between malicious model and benign model. For example, in Fig. 11(a), the longer the q range, the larger the difference of correlation coefficient between malicious model and benign model. Since change direction vector is estimated by $\tilde{\boldsymbol{s}} = \boldsymbol{w} - \boldsymbol{w}_{\mathrm{Re}}$ ($\boldsymbol{w} = \frac{1}{c}\sum_{i=1}^{c} \boldsymbol{w}_i$) in partial knowledge attack, it usually makes $\tilde{\boldsymbol{s}}$ denser than actual change direction $\boldsymbol{s}$ (used in full-knowledge attack). Thus, PDRR against Krum attack has a longer q range in partial-knowledge attack than that in full-knowledge attack. In Fig. 11(b) and Fig. 11(c), dense $\tilde{\boldsymbol{s}}$ in partial-knowledge attack makes more malicious model parameters removed in Trimmed-mean attack and less malicious model parameters remained in Median attack, which increases q range in partial-knowledge attack. However, it is not absolute. In 10th round (Fig. 11(b)), 10th round (Fig. 11(c)) and 50th round (Fig. 11(c)), the length of q range in full-knowledge attack are longer than that in partial-knowledge attack. In other words, $\tilde{\boldsymbol{s}}$ may become sparser than $\boldsymbol{s}$ in these cases.

In Fig. 11(a), since all change direction vectors of

malicious models s_i^p ($i=1,...,k$) have the same direction with s_{pag}^p, the maximum of q in each round is always 1 no matter in full-knowledge attack or partial-knowledge attack. However, in Fig. 11(b) and Fig. 11(c), PDRR only gets maximum of q in full-knowledge attack. Thus, malicious models are more similarly in full-knowledge attack.

5.3 Time cost

When the scales of local models are too large, ERR and LFR needs long time to verify the impact of local models on the aggregated model. Although VAE and DAGMM are not required to verify local models with validation dataset, they also need to fit all local models to the pre-trained model and obtain their abnormal score. As an alternative scheme, PDRR only requires a simple process with pre-aggregation and count, and then it can rapidly detect malicious models.

Here, we calculate the time cost of defense schemes for different attacks. All settings of experiment are the same with those in section B. All schemes are executed in the GPU, where DAGMM, VAE and ERR uses Tensorflow accelerated with CUDA, while PDRR completes pre-aggregation and detection with CUPY (NumPy-like API accelerated with CUDA). Table. 3 records the time cost on a single detection of all local models trained by clients at the round 1 in FL. In addition, since LFR scheme has the almost same time cost as ERR, here we only show the time cost of ERR in Table. 3.

Table. 3 The time cost on a single detection of all local models

Defense scheme	Krum attack	Trimmed-mean attack	Median attack
DAGMM	6.3s	7.8s	6.8s
VAE	54.5s	56.7s	56.8s
ERR	32.2s	32.7s	35.8s
PDRR	0.42s	0.84s	0.63s

Since VAE uses Monte Carlo estimation to sample from hidden variables and calculate reconstruction probability, it brings the highest time cost on detection. ERR and LFR need to verify each local model and brings relatively high time cost. PDRR has the lowest time cost on detection and thus it is more suitable to defend against local model poisoning attack in FL.

Finally, we compare functionalities of benchmark defense scheme and our defense scheme in Table. 4, where detection stability refers to whether the defense effect of the defense scheme is stable in different aggregation rules.

Table.4 Functionalities of defense schemes

Defense scheme	Auxiliary dataset	Time cost	Detection ability	Detection Stability
DAGMM	essential	relatively low	relatively weak	unstable
VAE	essential	highest	weak	unstable
ERR	essential	relatively high	weak	relatively stable
LFR	essential	relatively high	relatively strong	relatively stable
PDRR	undesired	lowest	strongest	stable

Obviously, PDRR is the only defense scheme without requiring auxiliary dataset, but detection effect is perfect. In addition, since PDRR just needs an extra pre-training process to distinguish malicious model from benign model, its time cost is the lowest. Thus, we regard PDRR is more suitable to detect malicious models in local model poisoning attack.

In local model poisoning attack, attacker usually selects specific attack model to attack specific aggregation rule. Although different attack models have transferability, attack success rate may be reduced dramatically when the attack model is not suitable for aggregation rule (e.g. Krum attack + Trimmed-mean aggregation rule in Ref. [19]，error rate is only 0.15 (error rate without attack is 0.14), but Krum attack + Krum aggregation rule, error rate is 0.70). Therefore, we focus on Krum attack on Krum aggregation rule. Since attack characteristics are similar between trimmed-mean attack and Median attack, the transferability is also strong between these two attack models. Therefore, our defense scheme (PDRR) is also transferable in these two attack models.

6 Conclusion, Limitations, and Future work

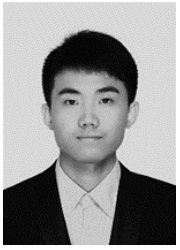
In this paper, we analyze the characteristics and vulnerability of the existing local model poisoning attacks in FL. In addition, a defense scheme PDRR is proposed to defend against these attacks according to their vulnerability. It is the first defense scheme to defend against local model poisoning attack without auxiliary dataset in the server. Performance analysis shows that our defense scheme is very effective for detecting malicious models in local model poisoning attack. Compared with benchmark defense schemes, PDRR has the lowest false alarm rate and missing alarm rate for detection and brings the lowest time cost to FL. Since our scheme removes restrictions on auxiliary datasets, its detection ability is the most stable in all defense schemes. Furthermore, we should note that PDRR is based on the assumption that all client datasets are heterogeneous to defend against local model poisoning attacks. In this assumption, the similarity between benign models is low, but that between malicious models is extremely high. If any two benign devices have similar datasets, their local models may be highly relevant, which may cause PDRR failure to defend against Krum attack. Besides, PDRR has its limitations in Trimmed-mean attack, where pre-aggregation and detection steps must ensure that trimmed parameter β is larger than malicious model number c (i.e. $\beta \geq c$). If $\beta < c$, some benign model parameters may be remained, which is harmful for detection result. In the future work, we hope to study how to defend Krum attack when clients have similar datasets and to defend Trimmed-mean attack if $\beta < c$

Acknowledgement The work was supported by the National Natural Science Foundation of China (Grand Nos. 11901579, 11801564).

References

1. Aman P, Rakshit N. FedPandemic: A cross-device federated learning approach towards elementary prognosis of diseases during a pandemic. arXiv preprint arXiv: 2104.01864
2. Qu Y, Pokhrel S R, Garg S, et al. A blockchained federated learning framework for cognitive computing in Industry 4.0 networks. IEEE Trans. Ind. Informat., 2020.
3. Papadopoulos G T, Leonidis A, Antona M, et al. User profile-driven large-scale multi-agent learning from demonstration in federated human-robot collaborative environments. arXiv preprint arXiv:2103.16434, 2021.
4. Chen S Y C, Yoo S. Federated Quantum machine learning. arXiv preprint arXiv:2103.12010, 2021.
5. Liu Y, Peng J, Kang J, et al. A secure federated learning framework for 5G networks. IEEE Wireless Communications, 2020, 27(4): 24-31.
6. McMahan H B, Eider M, Daniel R, et al. Communication-efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, 2017, 1273-1282.
7. McMahan H B, Eider M, Daniel R et al. Federated learning of deep networks using model averaging. arXiv preprint arXiv:1602.05629, 2016.
8. Battista B, Blaine N, and Pavel L. Poisoning attacks against support vector machines. In International Conference on Machine Learning, 2012.
9. Matthew J, Alina O, Battista B, et al. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In Conference of IEEE Symposium on Security & Privacy, 2018, 19-35.
10. Li B, Wang Y, Singh A, et al. Data poisoning attacks on factorization-based collaborative filtering. arXiv preprint arXiv:1608.08182, 2016.
11. Bagdasaryan E, Veit A, Hua Y, et al. How to backdoor federated learning. International Conference on Artificial Intelligence and Statistics. PMLR, 2020: 2938-2948.
12. Wu Z X, Ling Q, Chen T Y, et al. Federated variance-reduced stochastic gradient descent with robustness to byzantine attacks. arXiv preprint arXiv:1912.12716v1, 2019.
13. Kairouz P, McMahan H B, Avent B, et al. Advances and open problems in federated learning. arXiv preprint arXiv: 1912.04977, 2019.
14. Sun Z, Kairouz P, Suresh A T, et al. Can you really backdoor federated learning? arXiv preprint arXiv:1911.07963, 2019.

15. Nitin A, Chakraborty S, Mittal P, et al. Analyzing federated learning through an adversarial lens. In: Proceedings of the 36th International Conference on Machine Learning, PMLR 2019, 634-643.
16. Blanchard P, EL M, Guerraoui R, et al. Machine learning with adversaries: Byzantine tolerant gradient descent. In: Proceedings of the 31st Conference on Neural Information Processing System Advances in Neural Information Processing Systems, 2017, 118-128.
17. El M, Guerraoui R, and Rouault S. The hidden vulnerability of distributed learning in Byzantium. In: Proceedings of the 35th International Conference on Machine Learning, PMLR 2018, 3521-3530.
18. Yin D, Chen Y D, Ramchandran K, and Peter B. Byzantine-robust distributed learning: Towards optimal statistical rates. In: Proceedings of the 35th International Conference on Machine Learning, PMLR 2018, 5650-5659.
19. Fang M H, Cao X Y, Jia J Y, and Gong Z Q. Local model poisoning attacks to Byzantine-robust federated learning. In: Proceedings of the 29th Usenix Security Symposium, 2020.
20. Barreno M, Nelson B, Joseph A D, et al. The security of machine learning. Machine Learning, 2010, 81(2): 121-148.
21. Jiashi Feng, Huan Xu, Shie Mannor, and Shuicheng Yan. Robust logistic regression and classification. In: Proceedings of the 27th International Conference on Neural Information Processing Systems, 2014, 253–261.
22. Jacob Steinhardt, Pang Wei Koh, and Percy Liang. Certified defenses for data poisoning attacks. arXiv:1706.03691, 2017.
23. Suciu O, Marginean R, Kaya Y, et al. When does machine learning fail? generalized transferability for evasion and poisoning attacks. In: Proceedings of the 27th USENIX Security Symposium, 2018, 1299-1316.
24. An J, Cho S. Variational autoencoder based anomaly detection using reconstruction probability. Special Lecture on IE, 2015, 2(1), 1-18.
25. Xu H, Chen W, Zhao N, et al. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In: Proceedings of the 2018 World Wide Web Conference, 2018, 187-196.
26. Kieu T, Yang B, Guo C, et al. Outlier detection for time series with recurrent autoencoder ensembles. IJCAI. 2019: 2725-2732.
27. Li S, Cheng Y, Wang W, et al. Learning to detect malicious clients for robust federated learning. arXiv preprint arXiv:2002.00211, 2020.
28. Zong B, Song Q, Min M R, et al. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: Proceedings of the 27th International Conference on Learning Representations. 2018.
29. Lin J, Du M, Liu J. Free-riders in federated learning: Attacks and defenses. arXiv preprint arXiv:1911.12560, 2019.
30. Caldas S, Duddu S M K, Wu P, et al. Leaf: A benchmark for federated settings. arXiv preprint arXiv:1812.01097, 2018.
31. Konečný J, McMahan H B, Yu F X, et al. Federated learning: Strategies for improving communication efficiency. arXiv preprint arXiv:1610.05492, 2016.
32. Fung C, Yoon C J M, Beschastnikh I. Mitigating sybils in federated learning poisoning. arXiv preprint arXiv:1808.04866, 2018.
33. Min M, Xiao L, Xie C, et al. Defense against advanced persistent threats in dynamic cloud storage: A colonel blotto game approach. IEEE Internet of Things Journal, 2018, 5(6): 4250-4261.



Shiwei Lu is a Doctor at Department of Basic Sciences, Air Force Engineering University, China. He received his MS from Air Force Engineering University and achieved his B.S. degree in computer science and technology from Zhejiang University, in 2017. His major interests include cyberspace security and machine learning.

Ruihu Li received his Ph.D degree from North-western Polytechnical University of Technology in 2004. He is currently a professor in Department of Basic Sciences, Air Force Engineering University. His research interests include group theory, coding theory and cryptography.

Xuan Chen is a postgraduate at Department of Basic Sciences, Air Force Engineering University, China. He received his BS from Air Force Engineering University. His major interests include Machine learning and Adversarial attacks.

Yuena Ma is currently a professor in Air Force Engineering University. She received her PH. D in Electronic Science and technology from Northwestern Polytechnical University. Her research interests include Self-Orthogonal (or Dual Containing) Code, Quantum Error-Correcting Code, Communication security, Neural network deep learning.