FLCert: Provably Secure Federated Learning Against Poisoning Attacks

Xiaoyu Cao, Zaixi Zhang, Jinyuan Jia, *Member, IEEE*, and Neil Zhenqiang Gong, *Member, IEEE*

***Abstract*— Due to its distributed nature, federated learning is vulnerable to poisoning attacks, in which malicious clients poison the training process via manipulating their local training data and/or local model updates sent to the cloud server, such that the poisoned global model misclassifies many indiscriminate test inputs or attacker-chosen ones. Existing defenses mainly leverage Byzantine-robust federated learning methods or detect malicious clients. However, these defenses do not have provable security guarantees against poisoning attacks and may be vulnerable to more advanced attacks. In this work, we aim to bridge the gap by proposing *FLCert*, an ensemble federated learning framework, that is provably secure against poisoning attacks with a bounded number of malicious clients. Our key idea is to divide the clients into groups, learn a global model for each group of clients using any existing federated learning method, and take a majority vote among the global models to classify a test input. Specifically, we consider two methods to group the clients and propose two variants of FLCert correspondingly, i.e., *FLCert-P* that randomly samples clients in each group, and *FLCert-D* that divides clients to disjoint groups deterministically. Our extensive experiments on multiple datasets show that the label predicted by our FLCert for a test input is provably unaffected by a bounded number of malicious clients, no matter what poisoning attacks they use.**

***Index Terms*— Federated learning, provable security, poisoning attack, ensemble method.**

I. Introduction

FEDERATED learning (FL) [18], [23] is an emerging machine learning paradigm, which enables clients (e.g., smartphones, IoT devices, and organizations) to collaboratively learn a model without sharing their local training data with a cloud server. Due to its promise for protecting privacy of the clients' local training data and the emerging privacy regulations such as General Data Protection Regulation (GDPR), FL has been deployed by industry. For instance, Google has deployed FL for next-word prediction on Android Gboard. Existing FL methods mainly follow a *single-global-model* paradigm. Specifically, a cloud server maintains a *global model* and each client maintains a *local model*. The global model is trained via multiple iterations of communications between the clients and server. In each iteration, three steps are performed: 1) the server sends the current global model to the clients; 2) the clients update their local models based on the global model and their local training data, and send the model updates to the server; and 3) the server aggregates the model updates and uses them to update the global model. The learnt global model is then used to predict labels of test inputs.

However, such single-global-model paradigm is vulnerable to poisoning attacks. In particular, an attacker can inject fake clients to FL or compromise genuine clients, where we call the fake/compromised clients *malicious clients*. For instance, an attacker can use a powerful computer to simulate many fake smartphones. Such malicious clients can corrupt the global model via carefully tampering their local training data or model updates sent to the server. As a result, the corrupted global model has a low accuracy for the normal test inputs [6], [10] (known as *untargeted poisoning attacks*) or certain attacker-chosen test inputs [2], [3] (known as *targeted poisoning attacks*). For instance, in an untargeted poisoning attack, the malicious clients can deviate the global model towards the opposite of the direction along which it would be updated without attacks by manipulating their local model updates [10]. In a targeted poisoning attack, when learning an image classifier, the malicious clients can re-label the cars with certain strips as birds in their local training data and scale up their model updates sent to the server, such that the global model incorrectly predicts a car with the strips as bird [2].

Various Byzantine-robust FL methods have been proposed to defend against poisoning attacks from malicious clients. [4], [5], [32]. The main idea of these methods is to mitigate the impact of statistical outliers among the clients' model updates. They can bound the difference between the global model parameters learnt without malicious clients and the global model parameters learnt when some clients become malicious. However, these methods cannot provably guarantee that the label predicted by the global model for a test input is not affected by malicious clients. Indeed, studies showed that malicious clients can still substantially degrade the test accuracy of a global model learnt by a Byzantine-robust

Manuscript received 3 January 2022; revised 14 August 2022; accepted 30 September 2022. Date of publication 5 October 2022; date of current version 18 October 2022. This work was supported in part by the National Science Foundation under Grant 2131859, Grant 2125977, Grant 2112562, and Grant 1937786; and in part by the Army Research Office under Grant W911NF2110182. An earlier version of this paper was presented in part at the 2021 AAAI Conference on Artificial Intelligence (AAAI) [DOI: 10.1609/aaai.v35i8.16849]. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Pete Burnap. *(Corresponding author: Xiaoyu Cao.)*

Xiaoyu Cao is with Meta Platforms, Menlo Park, CA 94025 USA (e-mail: xiaoyucao@fb.com).

Zaixi Zhang is with the School of Computer Science and Technology, University of Science and Technology of China, Hefei, Anhui 230026, China (e-mail: zaixi@mail.ustc.edu).

Jinyuan Jia is with the Department of Computer Science, University of Illinois Urbana–Champaign, Urbana, IL 27708 USA (e-mail: jinyuan@illinois.edu).

Neil Zhenqiang Gong is with the Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708 USA (e-mail: neil.gong@duke.edu).

Digital Object Identifier 10.1109/TIFS.2022.3212174

method via carefully tampering their model updates sent to the server [3], [6], [10].

A. Our Work

In this work, we propose *FLCert*, the first FL framework that is provably secure against poisoning attacks. Specifically, given n clients, we define N groups, each containing a subset of the clients. In particular, we design two methods to group the clients, which corresponds to two variants of FLCert, i.e., *FLCert-P* and *FLCert-D*, where P and D stand for *probabilistic* and *deterministic*, respectively. In FLCert-P, each of the N groups consists of k clients sampled from the n clients uniformly at random. Note that there are a total of $\binom{n}{k}$ possible groups and thus N could be as large as $\binom{n}{k}$ in FLCert-P. In FLCert-D, we divide the n clients into N disjoint groups deterministically.

For each group, we learn a global model using an arbitrary FL algorithm (called *base FL algorithm*) with the clients in the group. In total, we train N global models. Given a test input $\boldsymbol{x}$, we use each of the N global models to predict its label. We denote n_i as the number of global models that predict label i for $\boldsymbol{x}$ and define $p_i = \frac{n_i}{n}$, where $i = 1, 2, \cdots, L$. We call n_i *label frequency* and p_i *label probability*. Our *ensemble global model* predicts the label with the largest label frequency/probability for $\boldsymbol{x}$. In other words, our ensemble global model takes a majority vote among the N global models to predict label for $\boldsymbol{x}$. Since each global model is learnt using a subset of the clients, a majority of the global models are learnt using benign clients when most clients are benign. Therefore, the majority-vote label among the N global models for a test input is unaffected by a bounded number of malicious clients no matter what poisoning attacks they use.

B. Theory

Our first major theoretical result is that FLCert provably predicts the same label for a test input $\boldsymbol{x}$ when the number of malicious clients is no larger than a threshold, which we call *certified security level*. Our second major theoretical result is that we prove our derived certified security level is tight, i.e., when no assumptions are made on the base FL algorithm, it is impossible to derive a certified security level that is larger than ours. Note that the certified security level may be different for different test inputs.

C. Algorithm

Computing our certified security level for $\boldsymbol{x}$ requires its largest and second largest label frequencies/probabilities. For FLCert-P, when $\binom{n}{k}$ is small (e.g., the n clients are dozens of organizations [17] and k is small), we can compute the largest and second largest label probabilities exactly via training $N = \binom{n}{k}$ global models. However, it is challenging to compute them exactly when $\binom{n}{k}$ is large. To address the computational challenge, we develop a randomized algorithm to estimate them with probabilistic guarantees via training $N \ll \binom{n}{k}$ global models. Due to such randomness, FLCert-P achieves probabilistic security guarantee, i.e., FLCert-P outputs an incorrect certified security level for a test input with some probability that can be set to be arbitrarily small. For FLCert-D, we train N global models and obtain the label frequencies deterministically, making the security guarantee of FLCert-D deterministic.

D. Evaluation

We empirically evaluate our method on five datasets from different domains, including three image classification datasets (MNIST-0.1 [20], MNIST-0.5 [20], and CIFAR-10 [19]), a human activity recognition dataset (HAR) [1], and a next-word prediction dataset (Reddit) [2]. We distribute the training examples in MNIST and CIFAR to clients to simulate FL scenarios, while the HAR and Reddit dataset represent real-world FL scenarios, where each user is a client. We also evaluate five different base FL algorithm, i.e., FedAvg [23], Krum [4], Trimmed-mean [32], Median [32], and FLTrust [5]. Moreover, we use *certified accuracy* as our evaluation metric, which is a lower bound of the test accuracy that a method can provably achieve no matter what poisoning attacks the malicious clients use. For instance, our FLCert-D with FedAvg and $N = 500$ can achieve a certified accuracy of 81% on MNIST when evenly distributing the training examples among 1,000 clients and 100 of them are malicious.

In summary, our key contributions are as follows:

- We propose FLCert, an FL framework with provable security guarantees against poisoning attacks. Specifically, FLCert-P provides probabilistic security guarantee while FLCert-D provides deterministic guarantee. Moreover, we prove that our derived certified security level is tight.
- We propose a randomized algorithm to compute our certified security level for FLCert-P.
- We evaluate FLCert on multiple datasets from different domains and multiple base FL algorithms. Our results show that FLCert is secure against poisoning attacks both theoretically and empirically.

All proofs appear in the Appendix.

II. Related Work

A. Federated Learning (FL)

Suppose we have n clients $\mathbf{C} = \{C_1, C_2, \cdots, C_n\}$ and a server, where each client C_i $(i = 1, 2, \cdots, n)$ holds a local training dataset D_i. Each client also has a unique user ID assigned when the client registers in the FL system. These clients aim to use their local training datasets to collaboratively learn a model that is shared among all clients, with the help of the server. We call such shared model *global model*. For simplicity, we use $\boldsymbol{w}$ to denote the model parameters of the global model. The parameters $\boldsymbol{w}$ are often learnt by solving the following optimization problem $\boldsymbol{w} = \arg\min_{\boldsymbol{w}'} \sum_{i=1}^{n} \ell(D_i; \boldsymbol{w}')$, where ℓ is a loss function, and $\ell(D_i; \boldsymbol{w}')$ is the empirical loss on the local training dataset D_i of the ith client.

The clients and the server collaborate to solve the optimization problem iteratively. Specifically, in each iteration of the training process, the following three steps are performed:

1) the server broadcasts the current global model to (a subset of) clients; 2) the clients initialize their local models as the received global model, train the local models using stochastic gradient descent, and send the local model updates back to the server; 3) the server aggregates the local model updates and updates the global model.

B. Poisoning Attacks to FL

Recent works [2], [3], [6], [10], [24], [27] showed that FL is vulnerable to poisoning attacks. Based on the attacker's goal, poisoning attacks to FL can be grouped into two categories: untargeted poisoning attacks [6], [10], [24], [27] and targeted poisoning attacks [2], [3].

1) Untargeted Poisoning Attacks: The goal of untargeted poisoning attacks is to decrease the test accuracy of the learnt global model. The attacks aim to increase the indiscriminate test error rate of the global model as much as possible, which can be considered as Denial-of-Service (DoS) attacks. Depending on how the attack is performed, untargeted poisoning attacks have two variants, i.e., *data poisoning attacks* [27] and *local model poisoning attacks* [6], [10]. Data poisoning attacks tamper with the local training data on the malicious clients while assuming the computation process maintains integrity. They inject fake training data points, delete existing training data points, and/or modify existing training data points. Local model poisoning attacks tamper with the computation process on the malicious clients, i.e., they directly tamper with the malicious clients' local models or model updates sent to the server. Note that in FL, any data poisoning attack can be implemented by a local model poisoning attack. This is because on each malicious client, we can always treat the local model trained using the tampered local training dataset as the tampered local model.

2) Targeted Poisoning Attacks: Targeted poisoning attacks aim to force the global model to predict target labels for target test inputs, while its performance on non-target test inputs is unaffected. The target test inputs in a targeted poisoning attack can be a specific test input [3], some test inputs with certain properties [2], or test inputs with a particular trigger embeded [2]. A targeted poisoning attack is also known as a backdoor attack if its target test inputs are trigger-embedded test inputs. A backdoor attack aims to corrupt the global model such that it predicts the attacker-chosen target label for any test input embedded with the predefined trigger.

C. Defenses Against Poisoning Attacks to FL

Many defenses [8], [14], [15], [16], [21], [22], [26], [28], [29], [30] have been proposed against data poisoning attacks in centralized learning scenarios. However, they are insufficient to defend against poisoning attacks to FL. In particular, the empirical defenses [8], [22], [30] require knowledge about the training dataset, which is usually not available to the FL server for privacy concerns. Moreover, the provably secure defenses [14], [15], [16], [21], [26], [28], [29] can guarantee that the label predicted for a test input is unaffected by a bounded number of poisoned training examples. However, in FL, a single malicious client can poison an arbitrary number of its local training examples, breaking their assumption.

Byzantine-robust FL methods [4], [5], [32] leverage Byzantine-robust aggregation rules to resist poisoning attacks. They share the idea of alleviating the impact of statistical outliers caused by poisoning attacks when aggregating the local model updates. For instance, Krum [4] selects a single model update that has the smallest square-Euclidean-distance score as the new global model update; Trimmed-mean [32] computes the coordinate-wise mean of the model update parameters after trim and updates the global model using corresponding mean values; Median [32] updates the global model by computing the coordinate-wise median of the local model updates; FLTrust [5] leverages a small clean dataset to compute a server update and uses it as a baseline to bootstrap trust to local model updates. These methods all suffer from a key limitation: they cannot provide provable security guarantees, i.e., they cannot ensure that the predicted labels of the global model for testing inputs remain unchanged when there exists a poisoning attack.

Another type of defenses focused on detecting malicious clients and removing their local models before the aggregation [10], [25], [33]. Shen et al. [25] proposed to perform clustering on local models to detect malicious clients. Fang et al. [10] proposed two defenses that use a validation dataset to reject potentially malicious local models based on their impact on the error rate or the loss value evaluated on the validation dataset. Zhang et al. [33] proposed to detect malicious clients via checking their model-updates consistency. These defenses showed some empirical effectiveness in detecting malicious clients. However, they cannot provide provable security guarantees, either.

Wang et al. [28] proposed a certified defense against backdoor attacks. A recent work [31] extended this method to FL and called it CRFL. CRFL aims to certify robustness against a particular backdoor attack [2], where all malicious clients train their local models using backdoored local training datasets, and scale their model updates before sending them to the server simultaneously in one iteration of FL. They showed that the accuracy of the learned global model under such specific attack could be certified if *the magnitude of the change in malicious clients' local training data is bounded*. However, a malicious client can arbitrarily change its local training data. Therefore, a single malicious client can break their defense by using poisoning attacks other than the considered particular backdoor attack. On the contrary, our FLCert is provably secure *no matter what attacks the malicious clients perform*.

III. Proposed FLCert

A. Overview

Figure 1 illustrates FLCert, where there are $n = 5$ clients and $N = 3$ groups. In FLCert-P, each group contains $k = 3$ clients randomly sampled from the n clients, while in FLCert-D, the clients are divided into N disjoint groups. We denote the N groups as $\mathbf{G}_1, \mathbf{G}_2, \cdots, \mathbf{G}_N$. We learn a global model for each group of clients using a determinized base FL algorithm. We determinize the base FL algorithm

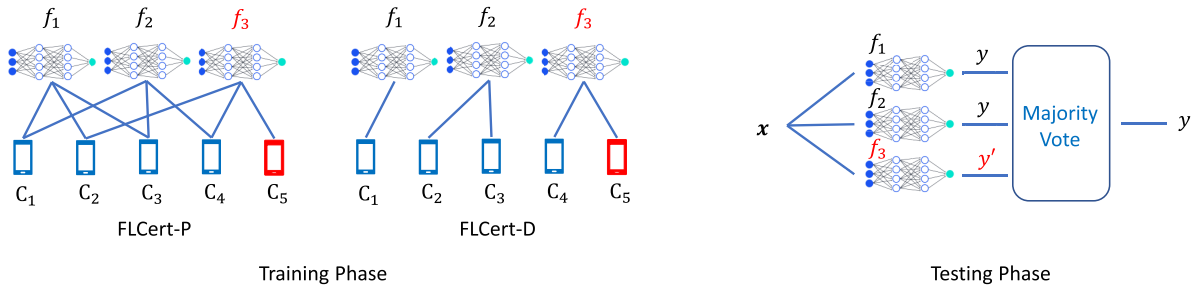


Fig. 1. Illustration of FLCert.

such that we can derive the provable security of FLCert. Since we have N groups, we learn N global models in total. We ensemble the N global models to predict labels for test inputs. Specifically, we take a majority vote among the N global models to predict the label of a test input.

B. Grouping the Clients

In FLCert, the clients are assigned to N groups. Considering the provable security and communication/computation overhead, the grouping should satisfy two constraints: 1) a malicious client should influence a small number of groups, which enables FLCert to be secure against more malicious clients, and 2) a client should belong to a small number of groups, which reduces the communication and computation overhead for clients. To satisfy the two constraints, we propose two ways to divide the clients, which corresponds to two variants of FLCert, i.e., FLCert-P and FLCert-D. Specifically, in FLCert-P, we randomly sample k clients as a group. In FLCert-D, we divide the n clients into N *disjoint* groups using hash values of their user IDs.

1) FLCert-P: In FLCert-P, the server samples k clients uniformly at random as a group, which results in $\binom{n}{k}$ possible groups in total. When $\binom{n}{k}$ is small, we can obtain all possible groups and train one global model for each group. In this case, each client belongs to $\binom{n-1}{k-1}$ groups. However, when $\binom{n}{k}$ is large, we may not be able to train all global models for every possible group. Therefore, in practice, we sample $N \ll \binom{n}{k}$ groups instead, and each client is expected to belong to a small number (i.e., $\frac{kN}{n}$) of groups. A malicious client can only affect the groups it belongs to.

2) FLCert-D: In FLCert-D, the server assigns each client to a group by hashing the client's user ID. Therefore, each client belongs to only one group and a malicious client can only influence the group it belongs to. We use clients' user IDs because they cannot be changed after registration, which means that malicious clients cannot change which groups they belong to. Formally, we use a hash function with output range $[1, N]$ to compute the hash values of the clients' user IDs; and the clients with hash value i form the ith group, where $i = 1, 2, \cdots, N$.

C. Ensemble Global Model

After the server assigns the clients into N groups, each group learns a global model using a determinized base FL algorithm f. In particular, we determinize a base FL algorithm via fixing the seed of the pseudo-random number generator used by the algorithm f. We denote by $f_g(\mathbf{G}_g)$ the global model for the gth group. Note that FLCert allows different groups to use different determinized base FL algorithms, e.g., different groups may use the same base FL algorithm with different fixed seeds for the pseudo-random number generator, and different groups may use different base FL algorithms.

Since there are N groups, we have N global models $f_1(\mathbf{G}_1), f_2(\mathbf{G}_2), \cdots, f_N(\mathbf{G}_N)$ in total. Our FLCert ensembles the N global models to predict labels for test inputs. Specifically, given a test input $\boldsymbol{x}$, we first use each global model to predict its label, i.e., $f_g(\mathbf{G}_g, \boldsymbol{x})$ is the label predicted by the gth global model. Then, we compute *label frequency* $n_j(\boldsymbol{x})$ for each label j, which is the number of global models that predict j for $\boldsymbol{x}$. Formally, $n_j(\boldsymbol{x})$ is defined as follows:

$$n_j(\boldsymbol{x}) = \sum_{g \in [1,N]} \mathbb{1}_{f_g(\mathbf{G}_g, \boldsymbol{x}) = j}, \tag{1}$$

where $\mathbb{1}$ is the indicator function and $\mathbb{1}_{f_g(\mathbf{G}_g, \boldsymbol{x})=j} = 1$ if $f_g(\mathbf{G}_g, \boldsymbol{x}) = j$, otherwise $\mathbb{1}_{f_g(\mathbf{G}_g, \boldsymbol{x})=j} = 0$. The sum of the label frequencies is N. Moreover, we define the *label probability* of label j as $p_j(\boldsymbol{x}) = \frac{n_j(\boldsymbol{x})}{N}$. Our FLCert takes a majority vote among the N global models to predict the label for $\boldsymbol{x}$, i.e., FLCert predicts the label with the largest label frequency/probability for $\boldsymbol{x}$. For convenience, we use label probability for FLCert-P and label frequency for FLCert-D in the rest of this paper. We denote our ensemble learning based FLCert algorithm as F. The ensemble of the N global models is called *ensemble global model*. Moreover, we denote by $F(\mathbf{C}, \boldsymbol{x})$ the label predicted for $\boldsymbol{x}$ by our ensemble global model learnt by F on the clients $\mathbf{C}$. Formally, we have:

$$F(\mathbf{C}, \boldsymbol{x}) = \underset{j}{\text{argmax}}\ n_j(\boldsymbol{x}) = \underset{j}{\text{argmax}}\ p_j(\boldsymbol{x}). \tag{2}$$

When there exist ties, i.e., multiple labels have the same largest label frequency/probability, we use different tie-breaking strategies for FLCert-P and FLCert-D. For FLCert-P, we randomly select a label with the same largest label probability. Such random tie-breaking strategy works for FLCert-P because FLCert-P has probabilistic security guarantees anyway. However, such random tie-breaking strategy invalidates the deterministic security guarantees of FLCert-D due to the randomness. While any deterministic tie-breaking strategy can

address this challenge, we adopt one that selects the label with the smallest class index as the predicted label in FLCert-D. For instance, when $n_1(\boldsymbol{x}) = n_2(\boldsymbol{x}) > n_j(\boldsymbol{x}), \forall j \neq 1 \wedge j \neq 2$, we have $F(\mathbf{C}, \boldsymbol{x}) = 1$, where $\wedge$ means logical AND.

IV. Security Analysis

We prove that the label predicted by FLCert for a test input is unaffected by a bounded number of malicious clients no matter what poisoning attacks they use.

A. Certified Security Level

Recall that $\mathbf{C}$ is the set of n clean clients. For convenience, we denote by $\mathbf{C}'$ the set of clients including the malicious ones. We define the *certified security level* m^* of a test input $\boldsymbol{x}$ as the maximum number of malicious clients that FLCert can tolerate without predicting a different label for $\boldsymbol{x}$. Formally, m^* is the largest integer m that satisfies the following:

$$F(\mathbf{C}', \boldsymbol{x}) = F(\mathbf{C}, \boldsymbol{x}), \forall \mathbf{C}', |\mathbf{C}' - \mathbf{C}| \leq m, \tag{3}$$

where $|\mathbf{C}' - \mathbf{C}|$ is the number of malicious clients in $\mathbf{C}'$, compared to $\mathbf{C}$. Note that the certified security level m^* may be different for different test inputs.

B. Deriving Certified Security Level

Next, we derive the certified security level m^* for a test input $\boldsymbol{x}$. Suppose that when there are no malicious clients, FLCert predicts label y for the test input $\boldsymbol{x}$, i.e., $y = F(\mathbf{C}, \boldsymbol{x}) = \text{argmax}_j\, n_j(\boldsymbol{x})$ and y has the largest label frequency. Moreover, we assume $z = \text{argmax}_{j \neq y}\, n_j(\boldsymbol{x})$ is the label with the second largest label frequency. We denote by p_y and p_z respectively their label probabilities. Moreover, we denote by n'_y and n'_z respectively the label frequency, and p'_y and p'_z respectively the label probabilities for y and z in the ensemble global model when there are malicious clients.

1) FLCert-P: When $\binom{n}{k}$ is small (e.g., several hundred), we can create all possible groups and train $N = \binom{n}{k}$ global models. Suppose m clients become malicious. Then, $1 - \frac{\binom{n-m}{k}}{\binom{n}{k}}$ fraction of groups include at least one malicious client. In the worst-case scenario, for each global model learnt using a group including at least one malicious client, its predicted label for $\boldsymbol{x}$ changes from y to z. Therefore, in the worst-case scenario, the m malicious clients decrease the largest label probability p_y by $1 - \frac{\binom{n-m}{k}}{\binom{n}{k}}$ and increase the second largest label probability p_z by $1 - \frac{\binom{n-m}{k}}{\binom{n}{k}}$, i.e., we have $p'_y = p_y - (1 - \frac{\binom{n-m}{k}}{\binom{n}{k}})$ and $p'_z = p_z + (1 - \frac{\binom{n-m}{k}}{\binom{n}{k}})$. Our ensemble global model still predicts label y for $\boldsymbol{x}$, i.e., $F(\mathbf{C}', \boldsymbol{x}) = F(\mathbf{C}, \boldsymbol{x}) = y$, once m satisfies the following inequality:

$$p'_y > p'_z \Longleftrightarrow p_y - p_z > 2 - 2\frac{\binom{n-m}{k}}{\binom{n}{k}}. \tag{4}$$

In other words, the largest integer m that satisfies the inequality (4) is our certified security level m^* for the test input $\boldsymbol{x}$. The inequality (4) shows that our certified security level is related to the gap $p_y - p_z$ between the largest and second largest label probabilities in the ensemble global model trained on the clients $\mathbf{C}$. For instance, when a test input has a larger gap $p_y - p_z$, the inequality (4) may be satisfied by a larger m, which means that our ensemble global model may have a larger certified security level for the test input.

However, when $\binom{n}{k}$ is large, it is computationally challenging to compute the exact label probabilities via training $\binom{n}{k}$ global models. For instance, when $n = 100$ and $k = 10$, there are already 1.73×10^{13} global models, training all of which is computationally intractable in practice. Therefore, we also derive certified security level using a lower bound $\underline{p_y}$ of p_y (i.e., $\underline{p_y} \leq p_y$) and an upper bound $\overline{p}_z$ of p_z (i.e., $\overline{p}_z \geq p_z$). We use a lower bound $\underline{p_y}$ of p_y and an upper bound $\overline{p}_z$ of p_z because our certified security level is related to the gap $p_y - p_z$ and we aim to estimate a lower bound of the gap.

The lower bound $\underline{p_y}$ and upper bound $\overline{p}_z$ may be estimated by different methods. We propose a Monte Carlo algorithm to estimate a lower bound $\underline{p_y}$ and an upper bound $\overline{p}_z$ via only training N of the $\binom{n}{k}$ global models. Specifically, we sample N groups, each of which includes k clients sampled from the n clients uniformly at random, and we use them to train N global models $f_1(\mathbf{G}_1), f_2(\mathbf{G}_2), \cdots, f_N(\mathbf{G}_N)$. We use the N global models to predict labels for $\boldsymbol{x}$ and count the frequency of each label. We treat the label with the largest frequency as the predicted label y. Recall that, based on the definition of label probability, a global model trained on a random group with k clients predicts label y for $\boldsymbol{x}$ with the label probability p_y. Therefore, the frequency N_y of the label y among the N global models follows a binomial distribution $B(N, p_y)$ with parameters N and p_y. Thus, given N_y and N, we can use the standard one-sided Clopper-Pearson method [9] to estimate a lower bound $\underline{p_y}$ of p_y with a confidence level $1 - \alpha$. Specifically, we have $\underline{p_y} = \mathcal{B}\left(\alpha; N_y, N - N_y + 1\right)$, where $\mathcal{B}(q; \upsilon, w)$ is the qth quantile from a beta distribution with shape parameters υ and w. Moreover, we can estimate $\overline{p}_y = 1 - \underline{p_y} \geq 1 - p_y \geq p_z$ as an upper bound of p_y.

Next, we derive our certified security level based on the probability bounds $\underline{p_y}$ and $\overline{p}_z$. One way is to replace p_y and p_z in inequality (4) as $\underline{p_y}$ and $\overline{p}_z$, respectively. Formally, we have the following inequality:

$$\underline{p_y} - \overline{p}_z > 2 - 2\frac{\binom{n-m}{k}}{\binom{n}{k}}. \tag{5}$$

If an m satisfies inequality (5), then the m also satisfies inequality (4), because $\underline{p_y} - \overline{p}_z \leq p_y - p_z$. Therefore, we can find the largest integer m that satisfies the inequality (5) as the certified security level m^*. However, we found that the certified security level m^* derived based on inequality (5) is not tight, i.e., our ensemble global model may still predict label y for $\boldsymbol{x}$ even if the number of malicious clients is larger than m^* derived based on inequality (5). The key reason is that the label probabilities are integer multiplications of $\frac{1}{\binom{n}{k}}$. Therefore, we normalize $\underline{p_y}$ and $\overline{p}_z$ as integer multiplications of $\frac{1}{\binom{n}{k}}$ to derive a tight certified security level. Specifically, we derive the certified security level as the largest integer m that satisfies the following inequality (formally described in

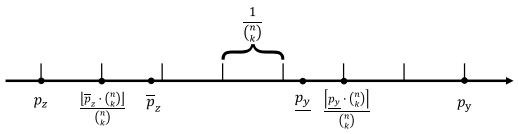


Fig. 2. An example to illustrate the relationships between p_y, $\underline{p_y}$, and $\frac{\lceil \underline{p_y} \cdot \binom{n}{k} \rceil}{\binom{n}{k}}$ as well as p_z, $\overline{p}_z$, and $\frac{\lfloor \overline{p}_z \cdot \binom{n}{k} \rfloor}{\binom{n}{k}}$.

Theorem 1):

$$\frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil}{\binom{n}{k}} - \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} > 2 - 2 \cdot \frac{\binom{n-m}{k}}{\binom{n}{k}}. \tag{6}$$

Figure 2 illustrates the relationships between p_y, $\underline{p_y}$, and $\frac{\lceil \underline{p_y} \cdot \binom{n}{k} \rceil}{\binom{n}{k}}$ as well as p_z, $\overline{p}_z$, and $\frac{\lfloor \overline{p}_z \cdot \binom{n}{k} \rfloor}{\binom{n}{k}}$. When an m satisfies inequality (5), the m also satisfies inequality (6), because $\underline{p_y} - \overline{p}_z \leq \frac{\lceil \underline{p_y} \cdot \binom{n}{k} \rceil}{\binom{n}{k}} - \frac{\lfloor \overline{p}_z \cdot \binom{n}{k} \rfloor}{\binom{n}{k}}$. Therefore, the certified security level derived based on inequality (5) is smaller than or equals the certified security level derived based on inequality (6). Note that when $\underline{p_y} = p_y$ and $\overline{p}_z = p_z$, both (5) and (6) reduce to (4) as the label probabilities are integer multiplications of $\frac{1}{\binom{n}{k}}$. The following theorem formally summarizes our certified security level.

Theorem 1: Given n clients $\mathbf{C}$*, an arbitrary base federated learning algorithm* f*, a group size* k*, and a test input* $\boldsymbol{x}$*, we define an ensemble global model* F *as Equation (2).* y *and* z *are the labels that have the largest and second largest label probabilities for* $\boldsymbol{x}$ *in the ensemble global model.* $\underline{p_y}$ *is a lower bound of* p_y *and* $\overline{p}_z$ *is an upper bound of* p_z*. Formally,* $\underline{p_y}$ *and* $\overline{p}_z$ *satisfy the following conditions*:

$$\max_{i \neq y} p_i = p_z \leq \overline{p}_z \leq \underline{p_y} \leq p_y. \tag{7}$$

Then, F *provably predicts* y *for* $\boldsymbol{x}$ *when at most* m^* *clients in* $\mathbf{C}$ *become malicious, i.e., we have*:

$$F(\mathbf{C}', \boldsymbol{x}) = F(\mathbf{C}, \boldsymbol{x}) = y, \forall \mathbf{C}', |\mathbf{C}' - \mathbf{C}| \leq m^*, \tag{8}$$

where m^* *is the largest integer* m *(*$0 \leq m \leq n - k$*) that satisfies inequality (6).*

Our Theorem 1 is applicable to any base federated learning algorithm, any lower bound $\underline{p_y}$ of p_y and any upper bound $\overline{p}_z$ of p_z that satisfy (7). When the lower bound $\underline{p_y}$ and upper bound $\overline{p}_z$ are estimated more accurately, i.e., $\underline{p_y}$ and $\overline{p}_z$ are respectively closer to p_y and p_z, our certified security level may be larger. The following theorem shows that our derived certified security level is tight, i.e., when no assumptions on the base federated learning algorithm are made, it is impossible to derive a certified security level that is larger than ours for the given probability bounds $\underline{p_y}$ and $\overline{p}_z$.

Theorem 2: Suppose $\underline{p_y} + \overline{p}_z \leq 1$*. For any* $\mathbf{C}'$ *satisfying* $|\mathbf{C}' - \mathbf{C}| > m^*$*, i.e., at least* m^*+1 *clients are malicious, there exists a base federated learning algorithm* f^* *that satisfies (7) but* $F(\mathbf{C}', \boldsymbol{x}) \neq y$ *or there exist ties.*

Given a test set $\mathcal{D}$, Algorithm 1 shows our algorithm to compute the predicted labels and certified security levels for all test inputs in $\mathcal{D}$. The function SAMPLE&TRAIN randomly samples N groups with k clients and trains N global models using the base federated learning algorithm f. We use $1 - \underline{p_{\hat{y}_t}}$ as an upper bound for $p_{\hat{z}_t}$ because we want to limit the probability of incorrect bound pairs $(\overline{p}_{\hat{z}_t}, \underline{p_{\hat{y}_t}})$ within $\alpha/|\mathcal{D}|$. Given the probability bounds $\underline{p_{\hat{y}_t}}$ and $\overline{p}_{\hat{z}_t}$ for a test input $\boldsymbol{x}_t$, the function SEARCHLEVEL finds the certified security level $\hat{m}_t^*$ via finding the largest integer m that satisfies (6). For example, SEARCHLEVEL can simply start m from 0 and iteratively increase it by one until finding $\hat{m}_t^*$.

Algorithm 1 Computing Predicted Label and Certified Security Level in FLCert-P

Input: $\mathbf{C}$, f, k, N, $\mathcal{D}$, α.
Output: Predicted label and certified security level for each test input in $\mathcal{D}$.
$f_1(\mathbf{G}_1), \cdots, f_N(\mathbf{G}_N) \leftarrow$ SAMPLE&TRAIN($\mathbf{C}$, f, k, N)
for $\boldsymbol{x}_t$ **in** $\mathcal{D}$ **do**
  counts$[i] \leftarrow \sum_{l=1}^{N} \mathbb{I}(f_l(\mathbf{G}_l, \boldsymbol{x}_t) = i), i \in \{1, 2, \cdots, L\}$
  /* $\mathbb{I}$ is the indicator function */
  $\hat{y}_t \leftarrow$ index of the largest entry in counts
  $\underline{p_{\hat{y}_t}} \leftarrow \mathcal{B}\left(\frac{\alpha}{|\mathcal{D}|}; N_{\hat{y}_t}, N - N_{\hat{y}_t} + 1\right)$
  $\overline{p}_{\hat{z}_t} \leftarrow 1 - \underline{p_{\hat{y}_t}}$
  if $\underline{p_{\hat{y}_t}} > \overline{p}_{\hat{z}_t}$ **then**
    $\hat{m}_t^* \leftarrow$ SEARCHLEVEL($\underline{p_{\hat{y}_t}}, \overline{p}_{\hat{z}_t}, k, |\mathbf{C}|$)
  else
    $\hat{y}_t \leftarrow$ ABSTAIN, $\hat{m}_t^* \leftarrow$ ABSTAIN
  end if
end for
return $\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_d$ and $\hat{m}_1^*, \hat{m}_2^*, \cdots, \hat{m}_d^*$

In Algorithm 1, since we estimate the lower bound $\underline{p_{\hat{y}_t}}$ using the Clopper-Pearson method, there is a probability that the estimated lower bound is incorrect, i.e., $\underline{p_{\hat{y}_t}} > p_{\hat{y}_t}$. When the lower bound is estimated incorrectly for a test input $\boldsymbol{x}_t$, the certified security level $\hat{m}_t^*$ output by Algorithm 1 for $\boldsymbol{x}_t$ may also be incorrect, i.e., there may exist an $\mathbf{C}'$ such that $|\mathbf{C}' - \mathbf{C}| \leq \hat{m}_t^*$ but $F(\mathbf{C}', \boldsymbol{x}_t) \neq \hat{y}_t$. In other words, our Algorithm 1 has probabilistic guarantees for its output certified security levels. However, in the following theorem, we prove the probability that Algorithm 1 returns an incorrect certified security level for at least one test input is at most α.

Theorem 3: The probability that Algorithm 1 returns an incorrect certified security level for at least one testing example in $\mathcal{D}$ *is bounded by* α*, which is equivalent to*:

$$Pr(\cap_{\boldsymbol{x}_t \in \mathcal{D}}(h(\mathbf{C}', \boldsymbol{x}_t) = \hat{y}_t, \forall \mathbf{C}', M(\mathbf{C}') \leq \hat{m}_t^* | \hat{y}_t \neq ABSTAIN)) \geq 1 - \alpha. \tag{9}$$

Note that when the probability bounds are estimated deterministically, e.g., when $\binom{n}{k}$ is small and the exact label probabilities can be computed via training $N = \binom{n}{k}$ global models, the certified security level obtained from our Theorem 1 is also deterministic. When there are many clients or N is too small, it is possible that not all clients participate in training the N global models. However, our probabilistic guarantee still holds in this case.

2) FLCert-D: FLCert-D still predicts label y for $\boldsymbol{x}$ after attacks if the following condition is satisfied:

$$n'_y(\boldsymbol{x}) > n'_z(\boldsymbol{x}) \text{ or } (n'_y(\boldsymbol{x}) = n'_z(\boldsymbol{x}) \wedge y < z), \tag{10}$$

where FLCert-D still predicts label y when $(n'_y(\boldsymbol{x}) = n'_z(\boldsymbol{x}) \wedge y < z)$ holds because of our tie-breaking strategy. Next, we derive a lower bound of $n'_y(\boldsymbol{x})$ and an upper bound of $n'_z(\boldsymbol{x})$, which depend on m, the number of malicious clients in $\mathbf{C}'$. The largest m, for which the lower bound of $n'_y(\boldsymbol{x})$ and the upper bound of $n'_z(\boldsymbol{x})$ satisfy Equation (10), is our certified security level m^* for $\boldsymbol{x}$.

If a group $\mathbf{G}'_g$ $(g = 1, 2, \cdots, N)$ after attack does not include malicious clients, then we know that $\mathbf{G}'_g$ and $\mathbf{G}_g$ include the same clients, i.e., $\mathbf{G}'_g = \mathbf{G}_g$. Moreover, since our base FL algorithm for each group is determinized, the global models learnt for $\mathbf{G}'_g$ and $\mathbf{G}_g$ predict the same label for $\boldsymbol{x}$, i.e., we have $f_g(\mathbf{G}'_g, \boldsymbol{x}) = f_g(\mathbf{G}_g, \boldsymbol{x})$. If a group $\mathbf{G}'_g$ includes at least one malicious client, the global model learnt for this group changes its predicted label for $\boldsymbol{x}$ from y to z after attack in the worst case. In other words, when a group includes malicious clients, the label frequency $n'_y(\boldsymbol{x})$ decreases by 1 and the label frequency $n'_z(\boldsymbol{x})$ increases by 1. According to our grouping strategy, m malicious clients influence at most m groups. Therefore, we have the following bounds in the worst case:

$$n'_y(\boldsymbol{x}) \geq n_y(\boldsymbol{x}) - m, \; n'_z(\boldsymbol{x}) \leq n_z(\boldsymbol{x}) + m. \tag{11}$$

The condition in Equation (10) is satisfied, i.e., $F(\mathbf{C}', \boldsymbol{x}) = F(\mathbf{C}, \boldsymbol{x}) = y$, when m is bounded as follows:

$$m \leq \frac{n_y(\boldsymbol{x}) - [n_z(\boldsymbol{x}) + \mathbb{1}_{z<y}]}{2}. \tag{12}$$

Therefore, we have our certified security level $m^* = \left\lfloor \frac{n_y(\boldsymbol{x}) - [n_z(\boldsymbol{x}) + \mathbb{1}_{z<y}]}{2} \right\rfloor$ for a test input $\boldsymbol{x}$. We summarize our provable security of FLCert in the following theorem.

Theorem 4: Given n clients $\mathbf{C}$ that are divided into N groups by hashing their user IDs, a determinized training algorithm for each group, and a test input $\boldsymbol{x}$. y and z are the labels that have the largest and second largest label frequencies for $\boldsymbol{x}$, where ties are broken by selecting the label with the smallest class index. Then, F provably predicts label y for $\boldsymbol{x}$ when at most m^* clients become malicious, i.e., we have:

$$F(\mathbf{C}', \boldsymbol{x}) = F(\mathbf{C}, \boldsymbol{x}) = y, \forall \mathbf{C}', |\mathbf{C}' - \mathbf{C}| \leq m^*, \tag{13}$$

where $m^* = \left\lfloor \frac{n_y(\boldsymbol{x}) - [n_z(\boldsymbol{x}) + \mathbb{1}_{z<y}]}{2} \right\rfloor$.

V. Communication/Computation Cost Analysis

We compare the communication and computation cost of our FLCert with the conventional single-global-model setting in which the base FL algorithm is used to learn a single global model for all the clients. Note that, for each client, its communication and computation cost is linear to the number of global iterations the client involves in. Therefore, in the following analysis, we focus on the average number of global iterations a client involves in for our FLCert and the single-global-model setting.

Suppose we are given a base FL algorithm f. In the single-global-model setting, f is used to learn a single global model with all the clients. Assume f performs T global iterations between the clients and server to learn the global model. In each global iteration, the server randomly selects β fraction of the clients and sends the current global model to them; the selected clients update their local models and send the updated local models to the server; and the server aggregates the local models as a new global model. β is often set to be smaller than 1 to save communication cost per global iteration. Therefore, the communication and computation cost for a client is $O(\beta T)$ on average in the single-global-model setting.

In FLCert, we use a base FL algorithm to train a global model for each group of clients. When learning a global model for a group of clients, the server randomly selects β_e fraction of the clients in the group in each global iteration. Assume each global model is learnt via T_e global iterations between the corresponding clients and the server. Each client is involved in $\frac{kN}{n}$ global models on average in FLCert-P, while each client is involved in only one global model in FLCert-D. Therefore, the communication and computation cost for a client is $O(\frac{kN\beta_e T_e}{n})$ for FLCert-P and $O(\beta_e T_e)$ for FLCert-D.

We note that each global model in our FLCert is learnt to fit local training data on a group of clients, while the global model in the single-global-model setting is learnt to fit local training data on all the clients. Therefore, learning each global model in our FLCert may require fewer global iterations, i.e., $T_e < T$. Moreover, when setting $kN = n$ and $T_e = \beta T / \beta_e$, our FLCert and the single-global-model setting have the same communication and computation cost for the clients. In other words, compared to the single-global-model setting, our FLCert can provide provable security against malicious clients without incurring additional communication and computation cost for the clients.

VI. Evaluation

A. Experimental Setup

1) Datasets: We use multiple datasets from different domains for evaluation, including three image classification datasets (MNIST-0.1, MNIST-0.5, and CIFAR-10), a human activity recognition dataset (HAR), and a next-word prediction dataset (Reddit). By default, we use MNIST-0.5 unless otherwise mentioned.

a) MNIST-0.1: We follow [10] to distribute the training examples to the clients. In their work, a parameter called degree of Non-iid is proposed to control the distribution of data among clients, where a larger value indicates the data are further from independently and identically distributed (IID). We set the degree of Non-iid to 0.1 in MNIST-0.1, which indicates IID data among clients.

b) MNIST-0.5: Clients in FL often have non-IID local training data. Therefore, in MNIST-0.5, we set the degree of Non-iid to 0.5 when distributing the training examples to clients to simulate non-IID local training data.

c) CIFAR-10: Like MNIST-0.5, we set the degree of Non-iid to 0.5 when distributing training examples to clients to simulate non-IID local training data.

d) Human activity recognition (HAR): HAR [1] is a real-world dataset consisting of human activity data collected from 30 users. The task is to predict a user's activity among six possible activities (e.g., WALKING, STANDING, and LAYING), given the sensor signal data collected from the user's smartphone. There are in total 10,299 examples in HAR dataset. We randomly select 75% of each user's examples as the training dataset and use the rest of examples as the test dataset. In HAR, each user is considered as a client naturally.

e) Reddit: Reddit [2] is a next-word prediction dataset consisting of posts collected from Reddit in a randomly chosen month (November 2017). Given a sequence of words, the task is to predict which word will appear next. For Reddit dataset, each Reddit user is a client. There are 80,000 users in total and each user has 247 posts on average.

2) Evaluated Base FL Algorithms: FLCert can use any base FL algorithm. To show such generality, we evaluate multiple popular base FL algorithms, including FedAvg [23], Krum [4], Trimmed-mean [32], Median [32], and FLTrust [5]. These base FL algorithms essentially use different aggregation rules to aggregate the clients' local model updates and update the global model in each global iteration. For FLTrust, we assume the server has a small clean dataset with 100 training examples randomly sampled from the training dataset, as suggested by the authors [5]. By default, we use FedAvg in our experiments unless otherwise mentioned.

3) Global Model Architectures: To show the generality of our FLCert, we use different neural network architectures as the global models on different datasets. Specifically, we borrow the CNN from [7] for MNIST-0.1 and MNIST-0.5. For CIFAR-10, we use the popular ResNet20 [12] architecture. For HAR, we use a fully connected neural network with two hidden layers, each of which includes 256 neurons and uses ReLU activation. For Reddit, we follow [13] to use a 2-layer LSTM as global model.

4) Evaluation Metrics: We use *certified accuracy (CA)* to evaluate the provable security of FLCert against poisoning attacks. Given a test dataset and a number of malicious clients m, we define the certified accuracy *CA@m* as the fraction of test inputs that 1) are correctly classified by the FL algorithm and 2) have certified security levels no smaller than m. For untargeted attacks, the test dataset includes the normal test examples. For backdoor attacks, the test dataset includes the test examples embedded with a pattern trigger. CA@m is a lower bound of test accuracy that a method can achieve no matter what poisoning attacks the malicious clients use once there are at most m of them.

Moreover, for both FLCert and the single-global-model setting, we also use the standard *test accuracy* and *attack success rate* to evaluate the *empirical* performance against an existing untargeted poisoning attack [10] and backdoor attack [2], respectively. Attack success rate is the fraction of trigger-embedded test inputs that are classified as the attacker-chosen target label. We note that CA@0 reduces to the standard test accuracy when there are no attacks in deterministic scenarios, i.e., for FLCert-D and if we can train all $\binom{n}{k}$ global models for FLCert-P. However, when $\binom{n}{k}$ is too large and we sample N groups of clients in FLCert-P, the empirical test accuracy may be different from CA@0.

5) Existing Poisoning Attacks: To calculate the standard test accuracy and attack success rate, we need existing poisoning attacks. For untargeted attacks, we use the full-knowledge attacks proposed by Fang et al. [10], i.e., Krum attack for Krum, and Trim attack for Trimmed-mean, Median, and FLTrust. For FedAvg, since a single malicious client can arbitrarily manipulate the global model [32], we use an attack that forces the aggregated model update to be 0. In other words, a global model will learn nothing from the training process if there is any malicious client in its training process. For backdoor attacks, we consider the same trigger as proposed by Gu et al. [11] and set 0 as the target label for MNIST-0.1 and MNIST-0.5. For CIFAR-10, we use the same pixel-pattern trigger and the target label "bird" in [2]. For HAR, we design our trigger by setting a feature value to 0 for every 20 features and we choose "WALKING_UPSTAIRS" as the target label. For Reddit, we consider the text trigger "pasta from Astoria is" and the target label "delicious" as proposed in [2].

6) Parameter Settings: We consider different numbers of clients for different datasets to show generality of FLCert. Specifically, we assume 1,000 clients for MNIST-0.1 and MNIST-0.5 datasets. For CIFAR-10 dataset, we assume 100 clients. For HAR and Reddit, each user is considered as a client naturally, which results in 30 and 80,000 clients, respectively. In FLCert-D, we randomly generate a 64-bit user ID for each client and use the Python built-in *hash()* function to divide clients into N groups based on user IDs for each dataset. In FLCert-P, we choose $k = \frac{n}{N}$ such that the expected communication/computation cost for each client is the same as FLCert-D; and we set $\alpha = 0.001$, i.e., given a test dataset, FLCert-P outputs an incorrect certified security level for at least one test input in 1 out of 1,000 runs on average. Moreover, we use the same fixed seed for a base FL algorithm when learning the global models for different groups.

In FLCert, unless otherwise mentioned, we set $\beta_e = 1$ (i.e., the server selects all the clients in a group in each global iteration when learning a global model for the group) in all the datasets except Reddit as the groups are small in these datasets. For Reddit, we set $\beta_e = 0.025$, i.e., 2.5% of clients in a group are chosen in each global iteration when learning a global model for the group. A base FL algorithm uses 200, 200, 200, 200, and 1,000 global iterations when learning a global model in either FLCert or the single-global-model setting for MNIST-0.1, MNIST-0.5, CIFAR-10, HAR, and Reddit, respectively. In each global iteration, each client trains its local model for 5, 5, 40, 5, and 12 local iterations using stochastic gradient descent with learning rates 0.001, 0.001, 0.01, 0.001, and 20 as well as batch sizes 32, 32, 64, 32, and 64 for the five datasets, respectively. We set $N = 500, 500, 20, 15$ and 500 for the five datasets, respectively. We consider different parameters for different datasets because of their different data characteristics.

7) Equipment and Environment: We run our experiments on a server with Intel Xeon Gold 6230R Processor and

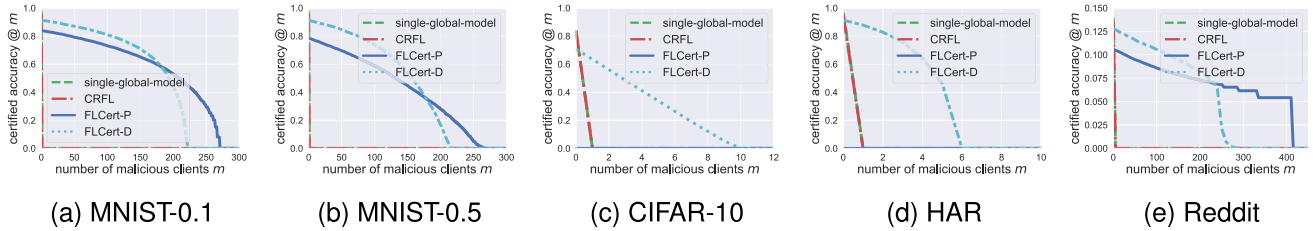


Fig. 3. Comparing FLCert with single-global-model and CRFL on different datasets when FedAvg is the base FL algorithm.

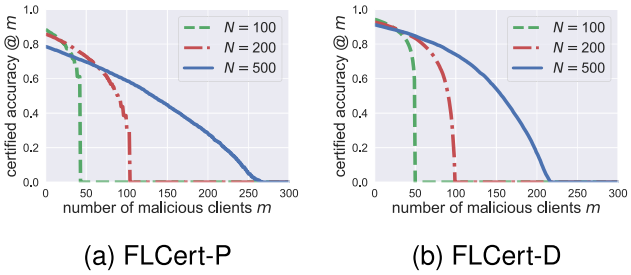


Fig. 4. Impact of N on certified accuracy of FLCert.

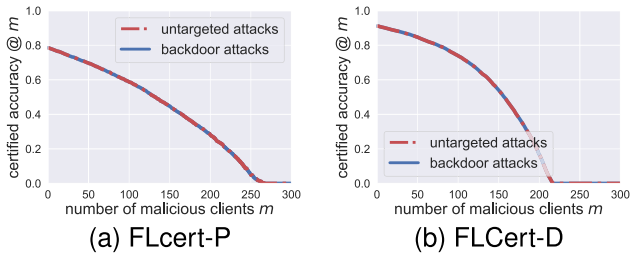


Fig. 5. Certified accuracy of FLCert against untargeted attacks and backdoor attacks.

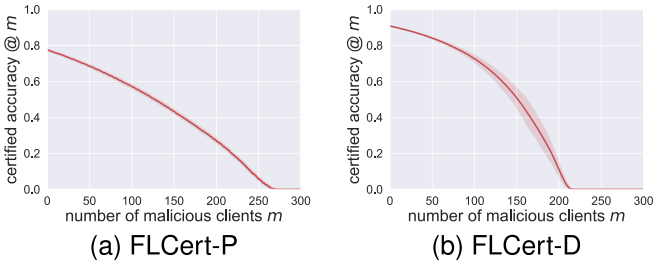


Fig. 6. Impact of seeds in the hash function used to group clients and the base FL algorithm on the certified accuracy of FLCert, where dataset is MNIST-0.5, $N = 500$, and FedAvg is the base FL algorithm. We repeat the experiments for 50 times, each of which uses a distinct seed for the hash function and a distinct seed for FedAvg. The solid line shows the mean certified accuracy of the 50 trials and the shade represents the standard deviation.

10 NVIDIA Quadro RTX 6000 GPUs. We train our models using the MXNet GPU framework.

B. Experimental Results

1) Certified Accuracy: We first show results on certified accuracy.

a) Comparing FLCert with single-global-model and CRFL: We compare the two variants of FLCert with two baselines, i.e., single-global-model FL and CRFL [31]. For CRFL, we use their default parameter settings, i.e., the clipping threshold $\rho = 15$, standard deviation $\sigma = 0.01$, and sample size $M = 1,000$. Figure 3 shows the results on all five datasets, where FedAvg is the base FL algorithm. We observe that when there are no malicious clients (i.e., $m = 0$), single-global-model FedAvg and CRFL have larger certified accuracy than FLCert. However, single-global-model FedAvg and CRFL have 0 certified accuracy when just one client is malicious. This is because a single malicious client can arbitrarily manipulate the global model learnt by FedAvg [32], reducing the certified accuracy of single-global-model FedAvg to 0, and its local training data, reducing the certified accuracy of CRFL to 0.

Moreover, we notice that there is a trade-off between the two variants of FLCert for MNIST-0.1, MNIST-0.5, and Reddit, where N is large. Specifically, when m is small, FLCert-D has higher certified accuracy. This is because in FLCert-P, we estimate the label probabilities, which reduces the gap between the largest and the second largest label probabilities. On the contrary, when m is large, FLCert-P achieves higher certified accuracy, which is because FLCert-P can naturally certify more malicious clients than FLCert-D in the extreme cases if we compare Equation 6 with Equation 12. However, FLCert-P cannot certify any malicious client on CIFAR-10 and HAR. This is because the number of groups N is small for these two datasets, e.g., $N = 15$ for HAR, which results in inaccurate estimation of label probabilities. A possible solution is to design probability estimation methods that are more accurate with a small number of samples. We also found that if we increase N to 500 while keeping $k = 2$, the certified accuracy of FLCert-P is still larger than 0 when up to 8 out of 30 clients are malicious for HAR. However, FLCert-P incurs a much larger communication/computation overhead than FLCert-D in such scenario.

b) N achieves a trade-off between accuracy and provable security: We explore the impact of the number of groups N on FLCert. Figure 4 shows the results. We observe that N controls a trade-off between *accuracy under no attacks* (i.e., $m = 0$) and provable security. Specifically, FLCert with a larger N has a lower accuracy under no attacks but can tolerate more malicious clients. This is because when N is larger, the average number of clients in each group becomes smaller. Therefore, the accuracy of each individual global model becomes lower, leading to a lower accuracy for the ensemble global model

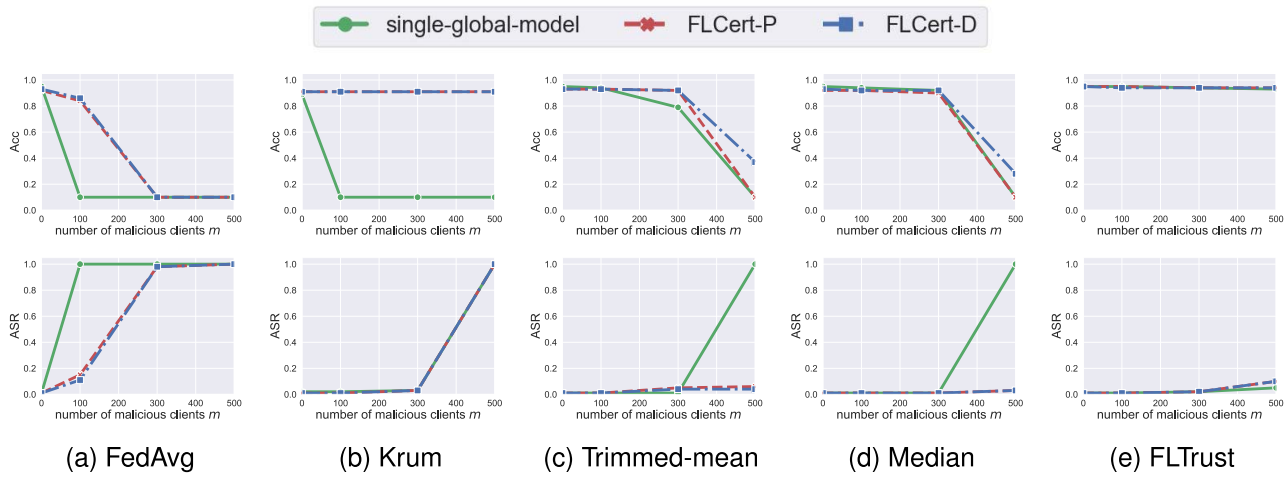


Fig. 7. Test accuracy (Acc) against untargeted attacks and attack success rate (ASR) against backdoor attacks for the single-global-model settings and FLCert when different base FL algorithms are used for MNIST-0.5 and $N = 200$. A higher test accuracy and a lower ASR mean better empirical performance.

under no attacks. Meanwhile, since the number of groups is larger, our FLCert can tolerate more malicious clients.

c) Untargeted attacks vs. backdoor attacks: We evaluate the certified accuracy for both untargeted attacks and targeted attacks. Figure 5 shows the results. We observe that the certified accuracy of both FLCert-P and FLCert-D against untargeted attacks is similar to that against backdoor attacks. The reason is that the trigger in a backdoor attack is often small to be stealthy, and thus a clean global model's predicted label for a test input is unaffected by the trigger.

d) Impact of seeds in hashing and base FL algorithms: Recall that we use the built-in Python *hash()* function with seeds to divide clients into groups for FLCert-D and we determinize a base FL algorithm via fixing the seed for both FLCert-P and FLCert-D. We study the impact of the seeds on the certified accuracy of FLCert. Specifically, we generate 50 pairs of seeds for the hash() function and base FL algorithm. Then, we run our FLCert for 50 times on a dataset, each of which uses a distinct pair of seeds. In each run, we use the same seed for different groups. Figure 6 shows the certified accuracy of FLCert with $N = 500$ on MNIST-0.5 when the base FL algorithm is FedAvg. We observe that the standard deviation is relatively small compared to the average certified accuracy, which means that our FLCert is insensitive to the seed in the *hash()* function used to group clients and the seed in the base FL algorithm.

e) FLCert with different base FL algorithms: We explore FLCert with different base FL algorithms, where the dataset is MNIST-0.5 and $N = 200$. We do not use the default $N = 500$ because when $N = 500$, the (expected) number of clients in each group is 2, for which Krum, Trimmed-mean, and Median are not defined. Figure 8 shows the results. We notice that the certified accuracy of FLCert is similar when FedAvg, Trimmed-mean, or Median is used as the base FL algorithm, and is better than the certified accuracy when the base FL algorithm is Krum. This is because Krum selects a single local model as the new global model while the other base FL algorithms consider all the received local models to update the global model. As a result, the global models learnt by Krum are less accurate than the global models learnt by the other base FL algorithms. Therefore, the ensemble global model of our FLCert has a lower certified accuracy when the base FL algorithm is Krum. Moreover, we observe that FLTrust achieves the highest certified accuracy when the number of malicious clients m is small. This is because the server is assumed to hold a small clean dataset in FLTrust, which helps learn more accurate global models with small groups of clients.

2) Test Accuracy and Attack Success Rate: Figure 7 shows the test accuracy against existing untargeted attacks and attack success rate against existing backdoor attacks for the conventional single-global-model settings and FLCert when different base FL algorithms are used, for MNIST-0.5 dataset. We use $N = 200$ as Krum, Trimmed-mean, and Median are not defined when $N = 500$. The numbers for the untargeted poisoning attacks represent test accuracy and a larger test accuracy indicates a better FL method. The numbers for the backdoor attacks represent attack success rate and a smaller number indicates a better FL method.

We notice that in general, the single-global-model setting achieves comparable or higher test accuracy and lower attack success rate than FLCert when there are no malicious clients or the number of malicious clients is small. However, our FLCert becomes better when the number of malicious clients is larger. An exception is when FLTrust is the aggregation rule, where both single-global-model and FLCert are robust against different attacks. Our results indicate that FLCert can tolerate more malicious clients against empirical attacks than single-global-model.

3) Communication/Computation-Security Trade-off: Suppose the base FL algorithm uses T_e global iterations when learning each global model. Figure 9a shows the certified accuracy of FLCert for MNIST-0.5 as T_e increases, where FedAvg is the base FL algorithm, $\beta_e = 1$, $N = 500$, and $m = 100$. Our certified accuracy increases as T_e increases and converges after T_e is large enough. Suppose the single-global-model

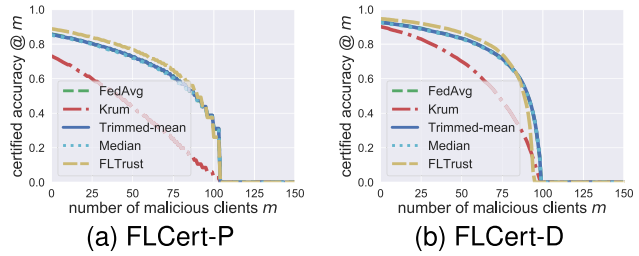


Fig. 8. FLCert with different base FL algorithms.

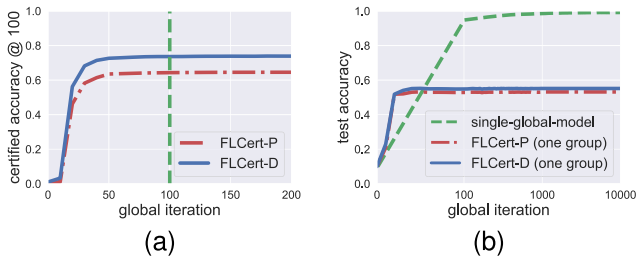


Fig. 9. (a) The communication/computation-security trade-off of FLCert. The vertical line shows the setting of the number of global iterations, where our FLCert has the same communication/computation cost for the clients as the single-global-model setting. (b) Convergence of learning the global model in the single-global-model setting and learning a global model for a certain group in FLCert. The dataset is MNIST-0.5 and base FL algorithm is FedAvg.

setting uses the default parameter setting $T = 1,000$ and $\beta = 0.1$. The vertical line in Figure 9a corresponds to the setting of T_e with which our FLCert has the same communication/computation cost for a client as the single-global-model setting, i.e., $T_e = \beta T/\beta_e = 0.1 \times 1000/1 = 100$. We notice that with such setting of T_e, our FLCert-P and FLCert-D can already achieve a high certified accuracy. For instance, FLCert-D achieves a certified accuracy of 0.73, which is 99% of 0.74, the largest certified accuracy FLCert-D can achieve by using a large T_e. Our results show that, compared to the single-global-model setting, our FLCert can achieve provable security against a bounded number of malicious clients without additional communication and computation cost for the clients.

One reason is that learning a global model for a group in FLCert converges faster than learning a global model for all clients in the single-global-model setting. Figure 9b shows the test accuracy under no attacks in the single-global-model setting and a global model for a certain group in FLCert-P and FLCert-D as the number of global iterations increases. We observe that the global model in the single-global-model setting converges with roughly 1,000 global iterations, while a global model for a group in our FLCert converges with less than 100 global iterations. The reason is that the global model in the single-global-model setting aims to fit the local training data on all clients, while a global model in our FLCert aims to fit the local training data on only a group of clients.

We note that FLCert incurs some cost at inference time. Specifically, we need to store the parameters of N global models. Moreover, we need to query the N global models to make a prediction for a given input. However, we note that such cost is affordable. First, we have shown that a moderate N (e.g., $N = 500$) is enough to achieve a high certified accuracy. Second, we can leverage model compression techniques to further reduce the model size. Third, the N global models can make predictions in parallel.

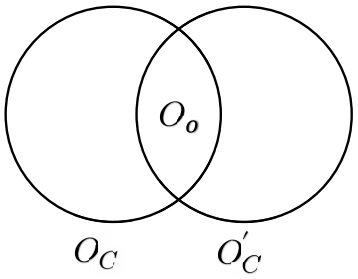


Fig. 10. Illustration of $\mathbf{O}_C$, $\mathbf{O}'_C$, and $\mathbf{O}_o$.

VII. Conclusion, Limitations, and Future Work

In this work, we propose FLCert, an ensemble FL framework that provides provable security guarantees against poisoning attacks from malicious clients. We propose two variants FLCert-P and FLCert-D based on how the clients are grouped and derive their theoretical security guarantees. Moreover, we design a randomized algorithm to compute the certified security level for FLCert-P in practice. Our empirical results on multiple datasets show that our FLCert can effectively defend against poisoning attacks with provable security guarantees.

One limitation of our work is that we do not leverage any prior knowledge on the learning task or the base FL algorithm when deriving our certified security levels. It is an interesting future work to involve such prior knowledge when deriving certified security guarantees.

Proof of Theorem 1

We first define a *subsample* as a set of k clients sampled from the n clients uniformly at random without replacement. We further define the space of all subsamples of k clients from $\mathbf{C}$ as $\mathbf{O}_C = \{\mathcal{S}(\mathbf{C}, k)\}$ and the space of all possible subsamples from $\mathbf{C}'$ as $\mathbf{O}'_C = \{\mathcal{S}(\mathbf{C}', k)\}$. Let $\mathbf{O}_o = \{\mathcal{S}(\mathbf{C} \cap \mathbf{C}', k)\} = \mathbf{O}_C \cap \mathbf{O}'_C$ denote the space of all possible subsamples from the set of normal clients $\mathbf{C} \cap \mathbf{C}'$, and $\mathbf{O} = \{\mathcal{S}(\mathbf{C} \cup \mathbf{C}', k)\} = \mathbf{O}_C \cup \mathbf{O}'_C$ denote the space of all possible subsamples from either $\mathbf{C}$ or $\mathbf{C}'$. Figure 10 illustrates $\mathbf{O}_C$, $\mathbf{O}'_C$, and $\mathbf{O}_o$. We use a random variable $\mathbf{X}$ to denote a subsample $\mathcal{S}(\mathbf{C}, k)$ and $\mathbf{Y}$ to denote a subsample $\mathcal{S}(\mathbf{C}', k)$ in $\mathbf{O}$. We know that $\mathbf{X}$ and $\mathbf{Y}$ have the following probability distributions:

$$\Pr(\mathbf{X} = a) = \begin{cases} \frac{1}{\binom{n}{k}}, & \text{if } a \in \mathbf{O}_C \\ 0, & \text{otherwise,} \end{cases} \tag{14}$$

$$\Pr(\mathbf{Y} = a) = \begin{cases} \frac{1}{\binom{n}{k}}, & \text{if } a \in \mathbf{O}'_C \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

Assume that a federated learning algorithm f takes a subsample a and a test input $\mathbf{x}$ as its input and outputs a label $f(a, \mathbf{x})$.

We have the following equations:

$$\begin{aligned} p_y &= \Pr(f(\mathbf{X},\mathbf{x})=y) && (16)\\ &= \Pr(f(\mathbf{X},\mathbf{x})=y|\mathbf{X}\in\mathbf{O}_o)\cdot\Pr(\mathbf{X}\in\mathbf{O}_o)\\ &\quad+\Pr(f(\mathbf{X},\mathbf{x})=y|\mathbf{X}\in(\mathbf{O}_C-\mathbf{O}_o))\\ &\quad\cdot\Pr(\mathbf{X}\in(\mathbf{O}_C-\mathbf{O}_o)), && (17)\\ p'_y &= \Pr(f(\mathbf{Y},\mathbf{x})=y) && (18)\\ &= \Pr(f(\mathbf{Y},\mathbf{x})=y|\mathbf{Y}\in\mathbf{O}_o)\cdot\Pr(\mathbf{Y}\in\mathbf{O}_o)\\ &\quad+\Pr(f(\mathbf{Y},\mathbf{x})=y|\mathbf{Y}\in(\mathbf{O}'_C-\mathbf{O}_o))\\ &\quad\cdot\Pr(\mathbf{Y}\in(\mathbf{O}'_C-\mathbf{O}_o)). && (19)\end{aligned}$$

Note that we have:

$$\Pr(f(\mathbf{X},\mathbf{x})=y|\mathbf{X}\in\mathbf{O}_o)=\Pr(f(\mathbf{Y},\mathbf{x})=y|\mathbf{Y}\in\mathbf{O}_o), \quad (20)$$

$$\Pr(\mathbf{X}\in\mathbf{O}_o)=\Pr(\mathbf{Y}\in\mathbf{O}_o)=\frac{\binom{n-m}{k}}{\binom{n}{k}}, \quad (21)$$

where m is the number of malicious clients. Therefore, we know:

$$\begin{aligned}&\Pr(f(\mathbf{X},\mathbf{x})=y|\mathbf{X}\in\mathbf{O}_o)\cdot\Pr(\mathbf{X}\in\mathbf{O}_o)\\ &\quad=\Pr(f(\mathbf{Y},\mathbf{x})=y|\mathbf{Y}\in\mathbf{O}_o)\cdot\Pr(\mathbf{Y}\in\mathbf{O}_o). \quad (22)\end{aligned}$$

By subtracting (17) from (19), we obtain:

$$\begin{aligned}&p'_y-p_y\\ &=\Pr(f(\mathbf{Y},\mathbf{x})=y|\mathbf{Y}\in(\mathbf{O}'_C-\mathbf{O}_o))\cdot\Pr(\mathbf{Y}\in(\mathbf{O}'_C-\mathbf{O}_o))\\ &\quad-\Pr(f(\mathbf{X},\mathbf{x})=y|\mathbf{X}\in(\mathbf{O}_C-\mathbf{O}_o))\cdot\Pr(\mathbf{X}\in(\mathbf{O}_C-\mathbf{O}_o)).\end{aligned} \quad (23)$$

Similarly, we have the following equation for any $i\neq y$:

$$\begin{aligned}&p'_i-p_i\\ &=\Pr(f(\mathbf{Y},\mathbf{x})=i|\mathbf{Y}\in(\mathbf{O}'_C-\mathbf{O}_o))\cdot\Pr(\mathbf{Y}\in(\mathbf{O}'_C-\mathbf{O}_o))\\ &\quad-\Pr(f(\mathbf{X},\mathbf{x})=i|\mathbf{X}\in(\mathbf{O}_C-\mathbf{O}_o))\cdot\Pr(\mathbf{X}\in(\mathbf{O}_C-\mathbf{O}_o)).\end{aligned} \quad (24)$$

Therefore, we can show:

$$\begin{aligned} p'_y-p'_i &= p_y-p_i+(p'_y-p_y)-(p'_i-p_i) && (25)\\ &= p_y-p_i\\ &\quad+\big[\Pr(f(\mathbf{Y},\mathbf{x})=y|\mathbf{Y}\in(\mathbf{O}'_C-\mathbf{O}_o))\\ &\quad-\Pr(f(\mathbf{Y},\mathbf{x})=i|\mathbf{Y}\in(\mathbf{O}'_C-\mathbf{O}_o))\big]\\ &\quad\cdot\Pr(\mathbf{Y}\in(\mathbf{O}'_C-\mathbf{O}_o))\\ &\quad-[\Pr(f(\mathbf{X},\mathbf{x})=y|\mathbf{X}\in(\mathbf{O}_C-\mathbf{O}_o))\\ &\quad-\Pr(f(\mathbf{X},\mathbf{x})=i|\mathbf{X}\in(\mathbf{O}_C-\mathbf{O}_o))]\\ &\quad\cdot\Pr(\mathbf{X}\in(\mathbf{O}_C-\mathbf{O}_o)). && (26)\end{aligned}$$

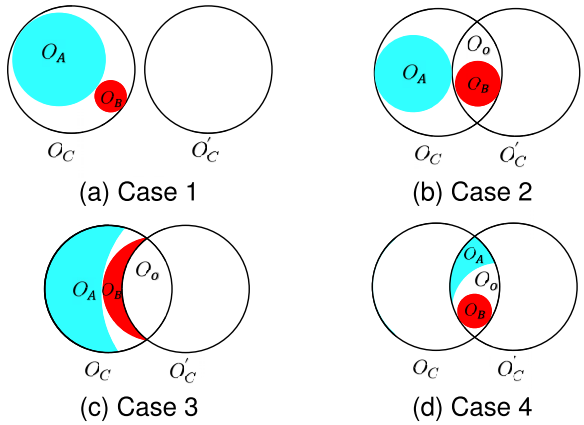


Fig. 11. Illustration of $\mathbf{O}_C, \mathbf{O}'_C, \mathbf{O}_o, \mathbf{O}_A$, and $\mathbf{O}_B$ in the four cases.

Note that we have:

$$\begin{aligned}&\Pr(f(\mathbf{Y},\mathbf{x})=y|\mathbf{Y}\in(\mathbf{O}'_C-\mathbf{O}_o))\\ &\quad-\Pr(f(\mathbf{Y},\mathbf{x})=i|\mathbf{Y}\in(\mathbf{O}'_C-\mathbf{O}_o))\geq-1, && (27)\\ &\Pr(f(\mathbf{X},\mathbf{x})=y|\mathbf{X}\in(\mathbf{O}_C-\mathbf{O}_o))\\ &\quad-\Pr(f(\mathbf{X},\mathbf{x})=i|\mathbf{X}\in(\mathbf{O}_C-\mathbf{O}_o))\leq 1, && (28)\\ &\Pr(\mathbf{Y}\in(\mathbf{O}'_C-\mathbf{O}_o))\\ &\quad=\Pr(\mathbf{X}\in(\mathbf{O}_C-\mathbf{O}_o))\\ &\quad=1-\frac{\binom{n-m}{k}}{\binom{n}{k}}. && (29)\end{aligned}$$

Therefore, (26) gives:

$$\begin{aligned} p'_y-p'_i &\geq p_y-p_i+(-1)\cdot\left[1-\frac{\binom{n-m}{k}}{\binom{n}{k}}\right]-\left[1-\frac{\binom{n-m}{k}}{\binom{n}{k}}\right] && (30)\\ &= p_y-p_i-\left[2-2\cdot\frac{\binom{n-m}{k}}{\binom{n}{k}}\right] && (31)\\ &= \frac{\lceil p_y\cdot\binom{n}{k}\rceil}{\binom{n}{k}}-\frac{\lfloor p_z\cdot\binom{n}{k}\rfloor}{\binom{n}{k}}-2\left[1-\frac{\binom{n-m}{k}}{\binom{n}{k}}\right] && (32)\\ &\geq \frac{\lceil \underline{p}_y\cdot\binom{n}{k}\rceil}{\binom{n}{k}}-\frac{\lfloor \overline{p}_z\cdot\binom{n}{k}\rfloor}{\binom{n}{k}}-2\left[1-\frac{\binom{n-m^*}{k}}{\binom{n}{k}}\right] && (33)\\ &> 0, && (34)\end{aligned}$$

which indicates $F(\mathbf{C}',\mathbf{x})=y$.

Proof of Theorem 2

We prove Theorem 2 by constructing a federated learning algorithm f^* such that $F(\mathbf{C}',\mathbf{x})\neq y$ or there exist ties.

We follow the definitions of $\mathbf{O}, \mathbf{O}_C, \mathbf{O}'_C, \mathbf{O}_o, \mathbf{X}$, and $\mathbf{Y}$ in the previous proof. Next, we consider four cases (Figure 11 illustrates them).

Case 1: $m\geq n-k$.

In this case, we know $\mathbf{O}_o=\emptyset$. Let $\mathbf{O}_A\subseteq\mathbf{O}_C$ and $\mathbf{O}_B\subseteq\mathbf{O}_C$ such that $|\mathbf{O}_A|=\lceil \underline{p}_y\cdot\binom{n}{k}\rceil$, $|\mathbf{O}_B|=\lfloor \overline{p}_z\cdot\binom{n}{k}\rfloor$, and $\mathbf{O}_A\cap$

$\mathbf{O}_B = \emptyset$. Since $\underline{p_y} + \overline{p}_z \leq 1$, we have:

$$|\mathbf{O}_A| + |\mathbf{O}_B| = \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor \tag{35}$$

$$\leq \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \left\lfloor (1 - \underline{p_y}) \cdot \binom{n}{k} \right\rfloor \tag{36}$$

$$= \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \binom{n}{k} - \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil \tag{37}$$

$$= \binom{n}{k} = |\mathbf{O}_C|. \tag{38}$$

Therefore, we can always find such a pair of disjoint sets $(\mathbf{O}_A, \mathbf{O}_B)$. Figure 11a illustrates $\mathbf{O}_A, \mathbf{O}_B, \mathbf{O}_C$, and $\mathbf{O}'_C$. We can construct f^* as follows:

$$f^*(a, \mathbf{x}) = \begin{cases} y, & \text{if } a \in \mathbf{O}_A \\ z, & \text{if } a \in \mathbf{O}_B \cup \mathbf{O}'_C \\ i, i \neq y \text{ and } i \neq z, & \text{otherwise.} \end{cases} \tag{39}$$

We can show that such f^* satisfies the following probability properties:

$$p_y = \Pr(f^*(\mathbf{X}, \mathbf{x}) = y) = \frac{|\mathbf{O}_A|}{|\mathbf{O}_C|} = \frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil}{\binom{n}{k}} \geq \underline{p_y}, \tag{40}$$

$$p_z = \Pr(f^*(\mathbf{X}, \mathbf{x}) = z) = \frac{|\mathbf{O}_B|}{|\mathbf{O}_C|} = \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} \leq \overline{p}_z. \tag{41}$$

Therefore, f^* satisfies the probability condition (7). However, we have:

$$p'_z = \Pr(f^*(\mathbf{Y}, \mathbf{x}) = z) = 1, \tag{42}$$

which indicates $F(\mathbf{C}', \mathbf{x}) = z \neq y$.

Case 2: $m^* < m < n - k$, $0 \leq \underline{p_y} \leq 1 - \frac{\binom{n-m}{k}}{\binom{n}{k}}$, and $0 \leq \overline{p}_z \leq \frac{\binom{n-m}{k}}{\binom{n}{k}}$.

Let $\mathbf{O}_A \subseteq \mathbf{O}_C - \mathbf{O}_o$ such that $|\mathbf{O}_A| = \lceil \underline{p_y} \cdot \binom{n}{k} \rceil$. Let $\mathbf{O}_B \subseteq \mathbf{O}_o$ such that $|\mathbf{O}_B| = \lfloor \overline{p}_z \cdot \binom{n}{k} \rfloor$. Figure 11b illustrates $\mathbf{O}_A, \mathbf{O}_B, \mathbf{O}_C, \mathbf{O}'_C$, and $\mathbf{O}_o$. We can construct a federated learning algorithm f^* as follows:

$$f^*(a, \mathbf{x}) = \begin{cases} y, & \text{if } a \in \mathbf{O}_A \\ z, & \text{if } a \in \mathbf{O}_B \cup (\mathbf{O}'_C - \mathbf{O}_o) \\ i, i \neq y \text{ and } i \neq z, & \text{otherwise.} \end{cases} \tag{43}$$

We can show that such f^* satisfies the following probability conditions:

$$p_y = \Pr(f^*(\mathbf{X}, \mathbf{x}) = y) = \frac{|\mathbf{O}_A|}{|\mathbf{O}_C|} = \frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil}{\binom{n}{k}} \geq \underline{p_y}, \tag{44}$$

$$p_z = \Pr(f^*(\mathbf{X}, \mathbf{x}) = z) = \frac{|\mathbf{O}_B|}{|\mathbf{O}_C|} = \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} \leq \overline{p}_z, \tag{45}$$

which indicates f^* satisfies (7). However, we have:

$$p'_y - p'_z = \Pr(f^*(\mathbf{Y}, \mathbf{x}) = y) - \Pr(f^*(\mathbf{Y}, \mathbf{x}) = z) \tag{46}$$

$$= 0 - \frac{|\mathbf{O}_B| + |\mathbf{O}'_C - \mathbf{O}_o|}{|\mathbf{O}'_C|} \tag{47}$$

$$= -\frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} - 1 + \frac{\binom{n-m}{k}}{\binom{n}{k}} \tag{48}$$

$$< 0, \tag{49}$$

which implies $F(\mathbf{C}', \mathbf{x}) \neq y$.

Case 3: $m^* < m < n - k$, $0 \leq \underline{p_y} \leq 1 - \frac{\binom{n-m}{k}}{\binom{n}{k}}$, and $\frac{\binom{n-m}{k}}{\binom{n}{k}} \leq \overline{p}_z \leq 1 - \underline{p_y}$.

Let $\mathbf{O}_A \subseteq \mathbf{O}_C - \mathbf{O}_o$ and $\mathbf{O}_B \subseteq \mathbf{O}_C - \mathbf{O}_o$ such that $|\mathbf{O}_A| = \lceil \underline{p_y} \cdot \binom{n}{k} \rceil$, $|\mathbf{O}_B| = \lfloor \overline{p}_z \cdot \binom{n}{k} \rfloor - \binom{n-m}{k}$, and $\mathbf{O}_A \cap \mathbf{O}_B = \emptyset$. Note that $|\mathbf{O}_C - \mathbf{O}_o| = \binom{n}{k} - \binom{n-m}{k}$, and we have:

$$|\mathbf{O}_A| + |\mathbf{O}_B|$$

$$= \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor - \binom{n-m}{k} \tag{50}$$

$$\leq \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \left\lfloor (1 - \underline{p_y}) \cdot \binom{n}{k} \right\rfloor - \binom{n-m}{k} \tag{51}$$

$$= \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \left[\binom{n}{k} - \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil \right] - \binom{n-m}{k} \tag{52}$$

$$= \binom{n}{k} - \binom{n-m}{k}. \tag{53}$$

Therefore, we can always find a pair of such disjoint sets $(\mathbf{O}_A, \mathbf{O}_B)$. Figure 11c illustrates $\mathbf{O}_A, \mathbf{O}_B, \mathbf{O}_C, \mathbf{O}'_C$, and $\mathbf{O}_o$. We can construct an algorithm f^* as follows:

$$f^*(a, \mathbf{x}) = \begin{cases} y, & \text{if } a \in \mathbf{O}_A \\ z, & \text{if } a \in \mathbf{O}_B \cup \mathbf{O}'_C \\ i, i \neq y \text{ and } i \neq z, & \text{otherwise.} \end{cases} \tag{54}$$

We can show that such f^* satisfies the following probability conditions:

$$p_y = \Pr(f^*(\mathbf{X}, \mathbf{x}) = y) = \frac{|\mathbf{O}_A|}{|\mathbf{O}_C|} = \frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil}{\binom{n}{k}} \geq \underline{p_y}, \tag{55}$$

$$p_z = \Pr(f^*(\mathbf{X}, \mathbf{x}) = z) = \frac{|\mathbf{O}_B| + |\mathbf{O}_o|}{|\mathbf{O}_C|} = \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} \leq \overline{p}_z, \tag{56}$$

which are consistent with the probability conditions (7). However, we can show the following:

$$p'_z = \Pr(f^*(\mathbf{Y}, \mathbf{x}) = z) = 1, \tag{57}$$

which gives $F(\mathbf{C}', \mathbf{x}) = z \neq y$.

Case 4: $m^* < m < n - k$, $1 - \frac{\binom{n-m}{k}}{\binom{n}{k}} < \underline{p_y} \leq 1$, and $0 \leq \overline{p}_z \leq 1 - \underline{p_y} < \frac{\binom{n-m}{k}}{\binom{n}{k}}$.

Let $\mathbf{O}_A \subseteq \mathbf{O}_o$ and $\mathbf{O}_B \subseteq \mathbf{C}_o$ such that $|\mathbf{O}_A| = \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \binom{n-m}{k} - \binom{n}{k}$, $|\mathbf{O}_B| = \left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor$, and $\mathbf{O}_A \cap \mathbf{O}_B = \emptyset$. Note that

$|\mathbf{O}_o| = \binom{n-m}{k}$, and we have:

$$
\begin{aligned}
&|\mathbf{O}_A| + |\mathbf{O}_B| \\
&= \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \binom{n-m}{k} - \binom{n}{k} + \left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor && (58)\\
&\leq \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \binom{n-m}{k} - \binom{n}{k} + \left\lfloor (1-\underline{p_y}) \cdot \binom{n}{k} \right\rfloor && (59)\\
&= \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \binom{n-m}{k} - \binom{n}{k} + \left[\binom{n}{k} - \left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil \right] && (60)\\
&= \binom{n-m}{k}. && (61)
\end{aligned}
$$

Therefore, we can always find such a pair of disjoint sets $(\mathbf{O}_A, \mathbf{O}_B)$. Figure 11d illustrates $\mathbf{O}_A, \mathbf{O}_B, \mathbf{O}_C, \mathbf{O}'_C$, and $\mathbf{O}_o$. Next, we can construct an algorithm f^* as follows:

$$
f^*(a, \mathbf{x}) = \begin{cases} y, & \text{if } a \in \mathbf{O}_A \cup (\mathbf{O}_C - \mathbf{O}_o) \\ z, & \text{if } a \in \mathbf{O}_B \cup (\mathbf{O}'_C - \mathbf{O}_o) \\ i, i \neq y \text{ and } i \neq z, & \text{otherwise.} \end{cases} \tag{62}
$$

We can show that f^* has the following properties:

$$
\begin{aligned}
p_y &= \Pr(f^*(\mathbf{X}, \mathbf{x}) = y) = \frac{|\mathbf{O}_A| + |\mathbf{O}_C - \mathbf{O}_o|}{|\mathbf{O}_C|} = \frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil}{\binom{n}{k}} \\
&\geq \underline{p_y}, && (63)\\
p_z &= \Pr(f^*(\mathbf{X}, \mathbf{x}) = z) = \frac{|\mathbf{O}_B|}{|\mathbf{O}_C|} = \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} \leq \overline{p}_z, && (64)
\end{aligned}
$$

which implies f^* satisfies the probability condition (7). However, we also have:

$$
\begin{aligned}
&p'_y - p'_z \\
&= \Pr(f^*(\mathbf{Y}, \mathbf{x}) = y) - \Pr(f^*(\mathbf{Y}, \mathbf{x}) = z) && (65)\\
&= \frac{|\mathbf{O}_A|}{|\mathbf{O}'_C|} - \frac{|\mathbf{O}_B| + |\mathbf{O}'_C - \mathbf{O}_o|}{|\mathbf{O}'_C|} && (66)\\
&= \frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil + \binom{n-m}{k} - \binom{n}{k}}{\binom{n}{k}} - \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor - \binom{n-m}{k} + \binom{n}{k}}{\binom{n}{k}} && (67)\\
&= \frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil}{\binom{n}{k}} - \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} - \left[2 - 2 \cdot \frac{\binom{n-m}{k}}{\binom{n}{k}} \right]. && (68)
\end{aligned}
$$

Since $m > m^*$, we have:

$$
\frac{\left\lceil \underline{p_y} \cdot \binom{n}{k} \right\rceil}{\binom{n}{k}} - \frac{\left\lfloor \overline{p}_z \cdot \binom{n}{k} \right\rfloor}{\binom{n}{k}} \leq \left[2 - 2 \cdot \frac{\binom{n-m}{k}}{\binom{n}{k}} \right]. \tag{69}
$$

Therefore, $p'_y - p'_z \leq 0$, which indicates $F(\mathbf{C}', \mathbf{x}) \neq y$ or there exist ties.

To summarize, we have proven that in any possible cases, Theorem 2 holds, indicating that our derived certified Byzantine size is tight.

Proof of Theorem 3

Based on the Clopper-Pearson method, we have:

$$
\begin{aligned}
\Pr(\underline{p_{y_t}} &\leq \Pr(f(\mathcal{S}(\mathbf{C}, k), \mathbf{x}_t) = y_t) \wedge \overline{p}_{z_t} \\
&\geq \Pr(f(\mathcal{S}(\mathbf{C}, k), \mathbf{x}_t) = i), \forall i \neq y_t) \\
&\geq 1 - \frac{\alpha}{d}. && (70)
\end{aligned}
$$

Therefore, for a test input $\mathbf{x}_t$, if our algorithm does not abstain for $\mathbf{x}_t$, the probability that it returns an incorrect certified Byzantine size is at most $\frac{\alpha}{d}$:

$$
\Pr((\exists \mathbf{C}' \in \Omega(\mathbf{C}, \hat{m}_t^*), h(\mathbf{C}', k, \mathbf{x}_t) \neq \hat{y}_t) | \hat{y}_t \neq \text{ABSTAIN}) \leq \frac{\alpha}{d}. \tag{71}
$$

Then, we have the following:

$$
\begin{aligned}
&\Pr(\cap_{\mathbf{x}_t \in \mathcal{D}}((\forall \mathbf{C}' \in \Omega(\mathbf{C}, \hat{m}_t^*), h(\mathbf{C}', k, \mathbf{x}_t) = \hat{y}_t) | \\
&\quad \times \hat{y}_t \neq \text{ABSTAIN})) \\
&= 1 - \Pr(\cup_{\mathbf{x}_t \in \mathcal{D}}((\exists \mathbf{C}' \in \Omega(\mathbf{C}, \hat{m}_t^*), h(\mathbf{C}', k, \mathbf{x}_t) \neq \hat{y}_t) | \\
&\quad \times \hat{y}_t \neq \text{ABSTAIN})) && (72)\\
&\geq 1 - \sum_{\mathbf{x}_t \in \mathcal{D}} \Pr((\exists \mathbf{C}' \in \Omega(\mathbf{C}, \hat{m}_t^*), h(\mathbf{C}', k, \mathbf{x}_t) \neq \hat{y}_t) | \\
&\quad \times \hat{y}_t \neq \text{ABSTAIN}) && (73)\\
&\geq 1 - d \cdot \frac{\alpha}{d} && (74)\\
&= 1 - \alpha. && (75)
\end{aligned}
$$

We have (73) from (72) based on the Boole's inequality.

Acknowledgment

The authors would like to thank the anonymous reviewers and AE for constructive comments.

References

[1] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," in *Proc. ESANN*, 2013, pp. 1–6.

[2] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. AISTATS*, 2020, pp. 1–10.

[3] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," in *Proc. ICML*, 2019, pp. 634–643.

[4] P. Blanchard, E. M. E. Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Proc. NeurIPS*, 2017, pp. 1–11.

[5] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "FLTrust: Byzantine-robust federated learning via trust bootstrapping," in *Proc. NDSS*, 2021, pp. 1–18.

[6] X. Cao and N. Z. Gong, "MPAF: Model poisoning attacks to federated learning based on fake clients," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2022, pp. 3396–3404.

[7] X. Cao, J. Jia, and N. Z. Gong, "Provably secure federated learning against malicious clients," in *Proc. AAAI*, 2021, pp. 1–9.

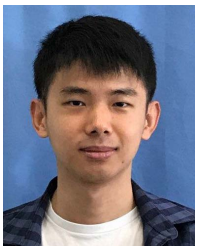
[8] J. Chen, X. Zhang, R. Zhang, C. Wang, and L. Liu, "De-pois: An attack-agnostic defense against data poisoning attacks," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 3412–3425, 2021.

[9] C. J. Clopper and E. S. Pearson, "The use of confidence or fiducial limits illustrated in the case of the binomial," *Biometrika*, vol. 26, no. 4, pp. 404–413, 1934.

[10] M. Fang, X. Cao, J. Jia, and N. Z. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," in *Proc. USENIX Secur. Symp.*, 2020, pp. 1605–1622.

[11] T. Gu, B. Dolan-Gavitt, and S. Garg, "BadNets: Identifying vulnerabilities in the machine learning model supply chain," in *Proc. Mach. Learn. Comput. Secur. Workshop*, 2017, pp. 1–13.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[13] H. Inan, K. Khosravi, and R. Socher, "Tying word vectors and word classifiers: A loss framework for language modeling," in *Proc. ICLR*, 2017, pp. 1–13.
[14] J. Jia, X. Cao, and N. Z. Gong, "Intrinsic certified robustness of bagging against data poisoning attacks," in *Proc. AAAI*, 2021, pp. 1–9.
[15] J. Jia, Y. Liu, X. Cao, and N. Z. Gong, "Certified robustness of nearest neighbors against data poisoning and backdoor attacks," in *Proc. AAAI*, 2022, pp. 1–9.
[16] J. Jia, B. Wang, X. Cao, and N. Z. Gong, "Certified robustness of community detection against adversarial structural perturbation via randomized smoothing," in *Proc. Web Conf.*, Apr. 2020, pp. 2718–2724.
[17] P. Kairouz et al., "Advances and open problems in federated learning," 2019, *arXiv:1912.04977*.
[18] J. Konečný, H. B. McMahan, X. F. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," in *Proc. NeurIPS Workshop Private Multi-Party Mach. Learn.*, 2016, pp. 1–10.
[19] A. Krizhevsky et al., "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009.
[20] Y. LeCun, C. Cortes, and C. Burges. (1998). *Mnist Handwritten Digit Database*. [Online]. Available: http://yann.lecun.com/exdb/mnist
[21] A. Levine and S. Feizi, "Deep partition aggregation: Provable defense against general poisoning attacks," 2020, *arXiv:2006.14768*.
[22] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdooring attacks on deep neural networks," in *Proc. Int. Symp. Res. Attacks, Intrusions, Defenses*, 2018, pp. 273–294.
[23] H. Brendan McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. AISTATS*, 2017, pp. 1–10.
[24] V. Shejwalkar and A. Houmansadr, "Manipulating the Byzantine: Optimizing model poisoning attacks and defenses for federated learning," in *Proc. NDSS*, 2021, pp. 1–19.
[25] S. Shen, S. Tople, and P. Saxena, "AUROR: Defending against poisoning attacks in collaborative deep learning systems," in *Proc. ACSAC*, 2016, pp. 508–519.
[26] J. Steinhardt, P. W. W. Koh, and P. S. Liang, "Certified defenses for data poisoning attacks," in *Proc. NeurIPS*, 2017, pp. 1–13.
[27] V. Tolpegin, S. Truex, M. Emre Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," 2020, *arXiv:2007.08432*.
[28] B. Wang, X. Cao, J. Jia, and N. Z. Gong, "On certifying robustness against backdoor attacks via randomized smoothing," in *Proc. CVPR Workshop Adversarial Mach. Learn. Comput. Vis.*, 2020, pp. 1–5.
[29] B. Wang, J. Jia, X. Cao, and N. Z. Gong, "Certified robustness of graph neural networks against adversarial structural perturbation," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2021, pp. 1645–1653.
[30] J. Wen, B. Z. H. Zhao, M. Xue, A. Oprea, and H. Qian, "With great dispersion comes greater resilience: Efficient poisoning attacks and defenses for linear regression models," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 3709–3723, 2021.
[31] C. Xie, M. Chen, P.-Y. Chen, and B. Li, "CRFL: Certifiably robust federated learning against backdoor attacks," in *Proc. ICML*, 2021, pp. 1–11.
[32] D. Yin, Y. Chen, K. Ramchandran, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proc. ICML*, 2018, pp. 1–10.
[33] Z. Zhang, X. Cao, J. Jia, and N. Z. Gong, "FLDetector: Defending federated learning against model poisoning attacks via detecting malicious clients," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2022, pp. 2545–2555.



Xiaoyu Cao received the B.Eng. degree from the Department of Gifted Young, University of Science and Technology of China (USTC), in 2016, the M.Eng. degree from Iowa State University, Ames, IA, USA, in 2019, and the Ph.D. degree from Duke University, Durham, NC, USA, in 2022. He is currently a Research Scientist at Meta Platforms. His research interests include machine learning security and privacy, with special interest in federated learning security.

Zaixi Zhang received the B.S. degree from the Department of Gifted Young, University of Science and Technology of China (USTC), in 2019. He is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, USTC. He has published papers in referred conference proceedings, such as the IJCAI, NeurIPS, KDD, and AAAI. His main research interests include data mining, machine learning security and privacy, and graph representation learning.

Jinyuan Jia (Member, IEEE) received the B.Eng. degree from the University of Science and Technology of China (USTC) in 2016, the M.Eng. degree from Iowa State University in 2019, and the Ph.D. degree from Duke University in 2022. He is currently a Post-Doctoral Researcher at the University of Illinois Urbana–Champaign. Since July 2023, he will be an Assistant Professor at The Pennsylvania State University. His research interests include security, privacy, and machine learning, with a recent focus on the intersection among them.

Neil Zhenqiang Gong (Member, IEEE) received the B.Eng. degree from the University of Science and Technology of China (USTC) in 2010 and the Ph.D. degree in computer science from the University of California, Berkeley, in 2015. He is currently an Assistant Professor at Duke University. His research interests include cybersecurity, privacy, machine learning security, and social networks security. He has received multiple prestigious awards, such as the Army Research Office Young Investigator Program Award and NSF CAREER Award.