Facilitating Early-Stage Backdoor Attacks in Federated Learning With Whole Population Distribution Inference

Tian Liu, Xueyang Hu, and Tao Shu

***Abstract*—The development of the Internet of Things (IoT) combined with the emergence of federated learning (FL) makes it possible for mobile edge computing (MEC) to gain insight from physically separated data without violating privacy or burdening communication. Due to the distributed nature of MEC devices, researchers have uncovered that the FL is vulnerable to backdoor attacks, which aim at injecting a subtask into the FL without corrupting the performance of the main task. The backdoor attack achieves high accuracy on both the main task and the backdoor subtask when injected at FL model convergence. However, the effectiveness of the backdoor is weak when injected in early training stage. In this article, we strengthen the early-injected backdoor attack by using information leakage. We show that FL convergence can be expedited if the client's data set mimics the distribution and gradients of the whole population. Based on this observation, we propose a two-phase backdoor attack, which includes a preliminary phase for the subsequent backdoor attack. Taking advantage of the preliminary phase, the later injected backdoor achieves better effectiveness, as the backdoor effect is less likely to be diluted by normal model updates. Extensive experiments are conducted on the MNIST data set under various data heterogeneity settings to evaluate the effectiveness of the proposed backdoor attack. The results show that the proposed backdoor outperforms existing backdoor attacks in both success rate and longevity, even when defense mechanisms are in place.**

***Index Terms*—Backdoor attack, federated learning (FL), privacy leakage, weight divergence.**

I. Introduction

By 2022 there will be 18 billion IoT devices connected to the Internet to provide monitoring and computing services [2]. Fueled by recent progress in machine learning, the data generated/collected by these devices can be utilized to train machine learning models that enable intelligent IoT applications. To overcome the issues in traditional centralized learning (CL), mobile edge computing (MEC), which uses the computing and storage capabilities of end devices, combined with federated learning (FL) [3], [4], serves as a solution to protect data privacy, reduce communication latency, and relieve the burden of the central server. Currently, FL applications on mobile edge devices thrive in next-word and emoji prediction on smartphones [5], [6], [7], [8], environmental monitoring [9], and aiding in medical diagnosis among hospitals [10], [11].

Due to the distributed nature of FL and inherent data non-i.i.d.-ness across edge clients, the local model update uploaded by a client may be different from others. As a result, it is difficult for the FL server to validate the legitimacy/truthfulness of the received model updates. Such a difficulty provides a venue for new attacks. Backdoor attack is one of the data poisoning attacks [12], in which an adversary corrupts the global model so that the new global model reaches high accuracy in both the main task and a backdoor subtask activated by a trigger. This backdoor attack has been shown to be unavoidable and computationally difficult to detect [13]. Especially, backdoor attacks are more realistic in MEC scenarios. First, a large portion of edge devices are IoT devices, which are not equipped with sufficient security functions. As a result, a considerable number of IoT devices are susceptible to traffic interception and manipulation. The attacker can easily corrupt a batch of clients to launch backdoor attacks. Second, launching backdoor attack requires little user privileges and does not need extra computation. Specifically, it only requires the access to the client's data and the capability of label flipping. As a result, backdoor attacks can be easily applied to IoT devices. Due to the above two reasons, backdoor attack remains a serious security threat to FL-powered MEC applications.

Although backdoor FL attacks are powerful, they have stringent requirements on the timing of attack. To make our argument more concrete, in this article, we will focus on *single-shot* backdoor attacks [12], due to their benefits of stealthiness, simplicity in implementation, and the fact that the more general multishot backdoor attacks can be built upon them incrementally. Existing studies [12], [14] have found that the optimal attack time for single-shot backdoor attack, in which the adversary injects the designated backdoor trigger only once (so as to keep the attack stealthy), should be when the global model is close to its convergence. However, in MEC scenarios, the attacker cannot always have the luxury of controlling injection time. This is because a typical MEC-powered FL process involves a large number of

Manuscript received 3 August 2022; revised 9 November 2022 and 14 December 2022; accepted 13 January 2023. Date of publication 18 January 2023; date of current version 7 June 2023. This work was supported in part by the United States National Science Foundation (NSF) under Grant CNS-2006998 and Grant CNS-1837034. This article was presented in part at the 19th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON), September 2022 [DOI: 10.1109/SECON55815.2022.9918550]. *(Corresponding author: Tao Shu.)*

Tian Liu is with the Intelligent Network Research Institute, Zhejiang Laboratory, Hangzhou 311121, China, and also with the AiLPHA Product Line of Big Data Intelligent Security, DBAPPSecurity Company, Ltd., Hangzhou 310051, China (e-mail: tianliu@zhejianglab.com).

Xueyang Hu and Tao Shu are with the Department of Computer Science and Software Engineering, Auburn University, Auburn, AL 36849 USA (e-mail: xueyang.hu@auburn.edu; tshu@auburn.edu).

Digital Object Identifier 10.1109/JIOT.2023.3237806

participants (IoT devices) over the entire training period, but in each round of training, only a small number of randomly selected devices will participate. There is no fixed participant schedule, so no device can predict whether or when it will be selected to participate in the training. As such, the backdoor attack has to be *opportunistic*, and the attacker must try its best to maximize both the strength and persistence of the injected backdoor whenever it is called to participate in the training—even when it is at an early stage of the training. In fact, a backdoor injected in the early stage of the training (before the FL model converges) can only generate a very weak backdoor effect for the following two reasons: 1) the strength of the injected backdoor model update will be severely diluted by the local model updates from other clients in the same round after aggregation at the server, because the magnitude of the other clients' local model updates is significant when the global model is not sufficiently converged and 2) the backdoor effect of the injected subtask vanishes quickly in subsequent training rounds as the injected backdoor will be overwritten by new coming normal model updates in those rounds. As a result, the earlier the backdoor is injected, the faster the backdoor effect will diminish. In addition, IoT devices are usually constrained by its resources—power, computing, memory, storage, etc. This requires an attack mechanism that runs in MEC to be lightweight and resource-efficient. General-case backdoor enhancing techniques, such as those in [13] and [15], require a substantial amount of computational resources, and, hence, are less suitable for MEC scenarios.

Realizing the stringent attack timing restriction in existing single-shot backdoor attacks and resource restriction of IoT devices, in this article, we are interested in studying a new single-shot backdoor attack technique that allows the backdoor subtask to be injected in the early stage of FL training while still achieving a strong and sustaining backdoor effect, making the effect of the attack less dependent on the timing of the attack, and, hence, making the attack more practical and applicable to general MEC applications.

Our new attack technique is inspired by the latest research findings on FL privacy, demonstrating that although private client data is not directly revealed in FL, the shared FL global model can unintentionally leak sensitive information about the data on which it was trained [16], [17], [18], [19], [20], [21]. This finding has motivated us to consider the following research problem: *does FL information leakage render a stronger backdoor attack in the early stage of FL training?* The findings in [10] indicate that the slow and unstable convergence of FL model is mainly caused by the weight divergence of the local model updates of different clients. This weight divergence is mainly decided by the difference in the label distribution (henceforth referred to as the "distribution") and the difference in gradients between a single client's local data and the whole population's data (i.e., the aggregation of all client's data). Therefore, reducing these differences will shrink the weight divergence and henceforth expedite FL convergence. This will increase the strength and sustainability of early-stage backdoor attacks.

In this study, we propose a novel information leakage-assisted two-phase FL backdoor attack, which enhances the effectiveness of FL early-injected single-shot backdoor attack. The essence of our idea is that if we expedite the FL convergence, the backdoored model update will be less diluted by model updates from normal clients, leading to a stronger and longer-lasting backdoor effect. We do not directly strengthen the backdoor attack itself. Instead, we design a preliminary phase, where a distinguishing feature is that attacker-controlled clients play the role of accomplice and reach out to the FL global model by uploading "beneficial" model updates that speed up the convergence of the global model to pave the way for the subsequent backdoor injection. Specifically, in the preliminary phase, attacker-controlled clients first perform a passive inference attack to obtain an estimate of the whole population distribution. Then, instead of training on the original local data, they train on locally crafted data sets, whose distributions align with the inferred whole population distribution, so that the weight divergence is reduced, and the FL model converges more quickly. When the backdoor client is selected, a regular single-shot backdoor attack is launched. Because convergence is facilitated by the preliminary phase, the backdoor attack is able to achieve better strength and sustainability, and the main task accuracy is less likely to deteriorate as the dilution effect of normal model updates is reduced. This is in sharp contrast to the existing backdoor attacks, in which the single-shot backdoor is injected directly without any camouflage. Thus, their backdoor effect is weaker, and the attack is less stealthy. Part of this work (i.e., the proposed two-phase backdoor attack algorithm) was presented at IEEE SECON 2022 [1]. This journal paper significantly extends [1] in that it not only presents the proposed attack algorithm, but also studies several important attributes/properties of the algorithm, including the algorithm's generalizability, stability, and feasibility, and evaluates the performance of the algorithm under more comprehensive settings as well.

To our knowledge, we are the first in the literature that enhances the effectiveness of FL backdoor attacks by utilizing the information leaked from the FL model. Our *contributions* in this study are fourfold.

1) We provide a theoretical analysis that the intra-aggregation weight divergence between a model in FL setting and CL setting consists of a gradient difference term and a distribution difference term, and we show that the weight divergence is bounded and the gap is small. The former finding motivates the design of the proposed attack preliminary phase, that is mimicking the behavior of CL can facilitate the FL convergence, which further overcomes the weakness of the single-shot backdoor attack. And the latter finding, the bounded and small gap of the weight divergence, theoretically supports the approximation in the proposed inference attack and provides an accuracy guarantee for the inferred distribution results.
2) We propose a novel optimization-based whole population distribution inference attack utilizing the above approximation and the linearity of the cross-entropy. Unlike the existing property inference attack, in which it can only generate binary property inference results, our proposed inference attack produces precise quantitative property information about the data set.

TABLE I
NOTATION AND DEFINITIONS

Notation	Definitions
$\|\cdot\|$	ℓ_2 norm.
D_k, D	Training data on the k-th client and the whole population, respectively. And we have $D = \cup_{k=1}^{N} D_k$.
n_k, n	Number of training samples in D_k and D, respectively. And we have $n = \sum_{k=1}^{K} n_k$.
w_k^T, w^T	Weight of the k-th local model and the global model in the T-th aggregation, respectively.
$F_k(w_k; D_k)$, $F(w; D)$	Loss function on the k-th client and the CL model, respectively.
$\nabla L(w_k; D_k)$, $\nabla L(w; D)$	Loss gradient of the client k and the CL model, respectively.
$p(y = c)$	Proportion of the label c in the training data, and we have $\sum_{c=1}^{C} p(y = c) = 1$.

3) We propose a preliminary phase for the early-injected single-shot backdoor attack, which improves the attack effectiveness by reducing the dilution effect from local updates of normal clients.
4) Extensive experiments are conducted in various data heterogeneity settings to evaluate the accuracy of the proposed whole population distribution inference attack, the improvement in the convergence of the FL global model, and the effectiveness of the backdoor attack.

Paper Organization: This article is structured as follows. We start by providing the background and related work in Section II. We present the threat model and attack design philosophy in Section III. Subsequently, the overview and the detailed attack steps are presented in Section IV. The experimental setup and results are presented in Sections V and VI, respectively. We evaluate the robustness of the proposed backdoor attack against two-defense mechanisms and discuss potential defenses in Section VII, and we conclude our work in Section VIII.

Throughout this article, we use the notation in Table I.

II. Background and Related Work

A. Federated Learning

The whole population $D = \cup_{k=1}^{N} D_k$ is allocated to N clients, and each client maintains D_k. Each client maintains a local model trained from the local training data set. And a central server maintains a global model by aggregating the local model updates from the participating client in each training round. The objective of FL training is to minimize the loss

$$F(w) = \frac{1}{|D|} \sum_{(x,y)\in D} L(w; (x, y)). \tag{1}$$

To achieve this goal, each client k optimizes their local model weights w_k to minimize the loss function

$$F_k(w) = \frac{1}{|D_k|} \sum_{(x,y)\in D_k} L(w; (x, y)). \tag{2}$$

Here, we describe the FedAvg aggregation method [4], which iteratively performs the following three steps.

1) *Global Model Synchronization:* In the Tth aggregation, the central server randomly selects K ($K \leq N$) from the N clients and broadcasts the latest global model w^T to the selected clients: $w_k^{T,0} \leftarrow w^T$.
2) *Local Model Training:* Each client k updates its own local model w_k^T by running an SGD on the local data set D_k for t steps. The τth step on client k follows:

$$w_k^{T,\tau+1} \leftarrow w_k^{T,\tau} - \eta \nabla F_k\left(w^{T,\tau}\right) \tag{3}$$

where η is the local learning rate.
3) *Global Model Update:* After performing local training for t steps, the client transmits the model update $\Delta w_k^T = w_k^{T,t} - w_k^{T,0}$ back to the central server. The central server then updates the global model by performing a weighted average on the local model updates sent by K clients

$$w^{T+1} \leftarrow w^T + \sum_{k=1}^{K} \frac{n_k}{n} \Delta w_k^T \tag{4}$$

where $n_k = |D_k|$ is the number of training data on the client k and $n = \sum_{k=1}^{K} n_k$ is the total number of training data used by the selected clients.

B. Related Work

1) Property Inference Attack Against FL: We mainly discuss the literature related to property inference attacks. The property inference attack was first proposed by Ateniese et al. [22] against the hidden Markov models and support vector machines. Ganju et al. [23] designed a property inference attack on fully connected networks, in which the adversary trains a meta-classifier to classify the target classifier depending on whether it possesses the property of interest or not. A malicious user can infer attributes that characterize the entire data class or a subset of data [20]. Scholars summarized the techniques to defend against inference attacks in [24].

We also note that our whole population distribution inference attack is similar to that of [25], where Wang et al. analyzed the relationship between the number of data samples of a specific label and the magnitude of the corresponding gradients. Our work differs from their work from the following two perspectives: 1) their work draws a comparison between a pair of labels and generates a binary output of which label possesses a larger number of data samples, while our work is able to provide a precise quantitative distribution of all labels and 2) their work has a high-computation complexity and needs to be performed multiple rounds to get a satisfying inference result, while in our work the distribution can be inferred in one training round and requires far less computation.

2) Backdoor Attack Against FL: The backdoor attack is one of the data poisoning attacks whose goal is to misclassify inputs with backdoor triggers as the target class, while not affecting the accuracy of the model on clean data. The backdoor was first introduced in [12]. They also proposed train-and-scale and constrain-and-scale techniques to maximize the attack impact while evading anomaly detection. The researchers [13] introduced an edge-case backdoor that targets data at the trailing end of a distribution. They also claimed that the backdoors against FL are unavoidable and computationally hard to detect. To make the backdoor stealthier, scholars in [26] decomposed a centralized backdoor into parts, and each trigger is injected by a client. The distributed backdoor is more effective and persistent than the centralized

backdoor. However, the distributed backdoor is fully activated only when all distributed triggers are injected. Additionally, to survive the new-coming normal updates, the injection of local triggers must be finished in a short attack window. Given that the attacker cannot manipulate the timing of selecting a compromised client to participate in the training, the above conditions are hardly satisfied in practice.

3) Defenses Against FL Backdoor Attack: Defense against backdoor attacks falls mainly into two categories, robust aggregation and differential privacy (DP).

Robust Aggregation: One approach from existing work focuses on building a robust aggregation algorithm that estimates the most possible aggregation rather than directly taking a weighted average. These robust aggregations, such as FoolsGold [27], Krum [28], Bulyan [29], RFA [30], and trimmed mean [31], are designed based on the statistical characteristics of model updates and aim to identify and de-emphasize possibly malicious model updates in the aggregation. Most of the robust aggregations are built on the assumption of the i.i.d. data distribution across the participating clients. However, this assumption is hardly met in practice. For the FL with non-i.i.d. data among clients, robust aggregation algorithms could misidentify non-i.i.d. but normal model updates as malicious or *vice versa*, and then their weight could be reduced or raised in the aggregation, which degrades the accuracy of the FL model. These approaches are capable of minimizing the impact of malicious model updates to a certain level but cannot completely eliminate them [32].

DP: DP was originally designed to protect individual privacy. FL with various DP schemes were proposed, e.g., the Bayesian DP [33], local DP [34], [35], [36], and central DP [37]. The algorithm and performance of differentially private FL were analyzed in [38]. Researchers in [14] discovered that by adding noise, the model update could also reduce the effect of malicious model updates. DP has been shown to be effective in mitigating backdoor attacks but comes at the cost of loss of model accuracy. The effectiveness of both local DP and central DP in defending against backdoor attacks was explored in [39]. Cui et al. [40] proposed a deferentially private FL based on GAN, which satisfies DP while optimizing the data utility.

III. Threat Model and Attack Design Philosophy

A. Threat Model

The adversary's goal is to improve the effectiveness and lifespan of the backdoor injected in the early training stage. Also, the backdoor should be stealthy, i.e., the impact on the main task accuracy should be as small as possible.

We consider a practical scenario where the attacker compromises multiple clients. Since launching a backdoor attack requires higher user privileges, we assume that only a few of them have the capability to launch the backdoor attack, while the rest can interact with the FL model multiple times as accomplices. As in [12], we assume that the client performing the backdoor attack has the ability to flip labels and set their own learning rate and local steps to maximize the backdoor effect while minimizing impact on the main learning task. Others are equipped with the ability to adjust the local distribution of labels, in which the attacker could obtain data from a public data set, or use data augmentation techniques, such as random rotation, random zoom, random crop, and sampling techniques, to change the number of samples for each label. This assumption is practical, as these operations require little computation and minimal user privileges and, thus, can be easily integrated into data preprocessing.

B. Attack Design Philosophy

Let w_a denote the local model of a backdoor client. The backdoor attack achieves its malicious goal by trying to substitute the new global model w^T with a local backdoor model w_a^T in (4). FL aggregation with a backdoor model update is as follows:

$$w^{T+1} \leftarrow w^T + \sum_{k\neq a} \frac{n_k}{n} \Delta w_k^T + \frac{n_a}{n} \Delta w_a. \tag{5}$$

The malicious model w_a can fully replace the global model by a scaling factor $\gamma = (n/n_a)$ only when the global model converges, i.e., $\sum_{k\neq a} (n_k/n)\Delta w_k^T \approx 0$. When the FL global model converges, the new-coming normal client model updates are too small to overwrite the backdoor effect. As a result, the injected backdoor can last a long time. However, for early-injected backdoors, the global model substitution is diluted due to $\sum_{k\neq a} (n_k/n)\Delta w_k^T \neq 0$, making the attack less effective. In fact, not only does the backdoor not reach its maximum effectiveness but the accuracy of the main task might also deteriorate as a result of the scaling operation. In general, the magnitude of the model update decreases as the FL model is close to convergence. The backdoor effect is more likely to be undermined and overwritten more quickly by new model updates when injected in an earlier training stage. Our main insight is that the early-injected backdoor effect would be more effective if the early training stage convergence is expedited, i.e., $\sum_{k\neq a} (n_k/n)\Delta w_k^T$ is reduced.

We consider a C-class classification FL problem with cross-entropy as the loss function. The loss function of a client k computed on its local data set D_k is defined as

$$F(w; D_k) = \sum_{c=1}^{C} p_k(y=c)\mathbb{E}_{x\in D_k|y=c}\left[\log f_c(x; w_k)\right] \tag{6}$$

where $p_k(y=c)$ denotes the proportion of class c in D_k, and f_c is the probability of a training sample x belonging to the cth class.

Due to the non-i.i.d. data distribution among participating clients and multiple SGDs are performed on the same local data set, the locally trained model in the FL scheme could introduce weight divergence, which deteriorates the FL global model. And this contributes to the performance gap between CL and FL. Therefore, the CL on the whole population serves as the upper bound of the FL. The weight divergence between the models in CL and FL settings can be used to characterize how good an FL model is.

Consider three models, the local model of the kth client w_k, the FL global model w, and the CL model w_{cen} trained on D. Previous works [41], [42] have analyzed the weight divergence

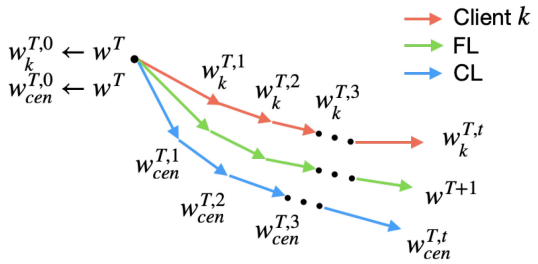

Fig. 1. Illustration of weight divergence relationship among an FL client's local model, FL global model, and CL model.

of the FL model w and the CL model w_{cen} throughout the training process and tried to catch what causes such a weight divergence. They proved that the weight divergence between w and w_{cen} throughout T global aggregations is bounded by two terms: 1) the sum of the distribution distance between each client's local data and the whole population and 2) the weight divergence inherited from the $(T-1)$th aggregation. And such a divergence is accumulated over time, and finally leads to a model accuracy degradation.

Inspired by their work, we are more interested in the intra-aggregation weight divergence, i.e., the weight divergence between two aggregations between w_{cen} and w, and w_{cen} and w_k. To remove the influence of previous aggregations, we let the CL model and the client's local model synchronize with the Tth FL global model, i.e., $w_{\text{cen}}^{T,0} \leftarrow w^T$ and $w_k^{T,0} \leftarrow w^T$. And the CL model and client's local model perform t steps training on the whole population data. The weight divergence relationship among the three models can be visualized in Fig. 1. The weights after τ steps are as follows:

$$\begin{aligned} w_{\text{cen}}^{T,\tau} &= w_{\text{cen}}^{T,\tau-1} - \eta \nabla F\left(w_{\text{cen}}^{T,\tau-1}; D\right) \\ &= w_{\text{cen}}^{T,\tau-1} - \eta \sum_{c=1}^{C} p(y=c) \nabla \mathbb{E}_{x \in D|y=c}\left[\log f_c\left(x; w_{\text{cen}}^{T,\tau-1}\right)\right] \quad (7) \\ w_k^{T,\tau} &= w_k^{T,\tau-1} - \eta \nabla F\left(w_k^{T,\tau-1}; D_k\right) \\ &= w_k^{T,\tau-1} - \eta \sum_{c=1}^{C} p(y=c) \nabla \mathbb{E}_{x \in D_k|y=c}\left[\log f_c\left(x; w_k^{T,\tau-1}\right)\right]. \quad (8) \end{aligned}$$

We have the following proposition.

Proposition 1: At the Tth FL global aggregation, let the local model w_k and the CL model on the entire population w_{cen} synchronize with the FL global model w^T, i.e., $w_k^{T,0} \leftarrow w^T$, and $w_{\text{cen}}^{T,0} \leftarrow w^T$. And, we have $p(y=c) = \sum_{k=1}^{K} p_k(y=c)$, where $p(y=c)$ and $p_k(y=c)$ are denoted as the proportion of the label c on D and D_k. Let each model train for t steps, in which the global aggregation conducts. The model weight divergence between w and w_{cen}, and w_k and w_{cen} after t training steps are bounded by the following two equations, respectively:

$$\begin{aligned} &\left\| w^{T,t} - w_{\text{cen}}^{T,t} \right\| \\ &\le \eta \sum_{\tau=1}^{t} \Bigg[\Bigg\| \sum_{c=1}^{C} \sum_{k=1}^{K} \frac{n_k}{n} p_k(y=c) \Big[\nabla \mathbb{E}_{x \in D_k|y=c} \Big[\log\Big(f_c\Big(w_k^{T,\tau-1} \Big) \Big) \Big] \\ &\quad - \nabla \mathbb{E}_{x \in D|y=c} \Big[\log\Big(f_c\Big(w_{\text{cen}}^{T,\tau-1} \Big) \Big) \Big] \Big] \Bigg\| \Bigg] \quad (9) \end{aligned}$$

$$\begin{aligned} &\left\| w_k^{T,t} - w_{\text{cen}}^{T,t} \right\| \\ &\le \eta \sum_{\tau=1}^{t} \Bigg[\Bigg\| \sum_{c=1}^{C} \left[(p(y=c) - p_k(y=c) \right] \nabla \mathbb{E}_{x \in D|y=c} \Big[\log\Big(f_c\Big(w_k^{T,\tau-1} \Big) \Big) \Big] \Bigg\| \\ &\quad + \Bigg\| \sum_{c=1}^{C} p_k(y=c) \Big[\nabla \mathbb{E}_{x \in D_k|y=c} \Big[\log f_c\Big(x; w_k^{T,\tau-1} \Big) \Big] \\ &\quad - \nabla \mathbb{E}_{x \in D|y=c} \Big[\log\Big(f_c\Big(x; w_{\text{cen}}^{T,\tau-1} \Big) \Big) \Big] \Big] \Bigg\| \Bigg]. \quad (10) \end{aligned}$$

The proof can be found in the Appendix, and we have the following remarks.

Remark 1: The intra-aggregation weight divergence $\|w^T - w_{\text{cen}}^T\|$ is determined by the difference between the gradient of the local model taken on $D_k, k \in [1, K]$ and the gradient of the CL model taken on D. This gradient difference can be reduced by increasing the local data sample size. The weight divergence is also an increasing function of the number of internal training steps. Therefore, increasing the number of local data samples or decreasing the internal training steps can mitigate weight divergence.

Remark 2: The intra-aggregation weight divergence $\|w_k^T - w_{\text{cen}}^T\|$ is mainly due to two parts, which are the distribution difference between D_k and D, that is, $\sum_{c=1}^{C}(p_k(y=c) - p(y=c))$, and the gradient difference between the gradient calculated on D_k and the gradient calculated on D over classes, that is, $[\nabla \mathbb{E}_{x \in D_k|y=c}[\log f_c(x; w_k^{T,t-1})] - \nabla \mathbb{E}_{x \in D|y=c}[\log(f_c(x; w_{\text{cen}}^{T,t-1})]]$.

According to Remark 2, the weight divergence $\|w^{T,t} - w_{\text{cen}}^{T,t}\|$ could be mitigated by reducing the following two terms: 1) the difference between the data distribution of D_k and that of D, implying the first term in the (10) is reduced and 2) the difference between the gradient calculated on D_k and that calculated on D, implying the second term in the (10) is reduced.

As a result, a client in an FL setting could benefit from mimicking the distribution and gradients of the whole population to achieve better convergence behavior (faster convergence or higher model accuracy). This finding is a double-edged sword. On the one hand, a benign client can use it to alleviate weight divergence to facilitate FL convergence, as the data sharing strategy proposed in [41]. On the other hand, an adversary could also take advantage of the finding. As will be shown in the next section, we propose a two-phase backdoor attack, in which the above finding is utilized by an adversary to improve the FL global convergence performance and further enhance both the strength and persistence for the subsequent single-shot backdoor injection.

IV. Proposed Two-Phase Backdoor Attack

In this section, leveraging the aforementioned insights, we present an overview of our proposed two-phase backdoor attack. We then describe the detailed workflow of the proposed backdoor attack.

A. Overview

Our proposed two-phase backdoor attack, illustrated in Fig. 2, consists of a preliminary phase and an attack phase. The proposed backdoor is different from existing backdoor attacks in the preliminary phase, which can be adapted to

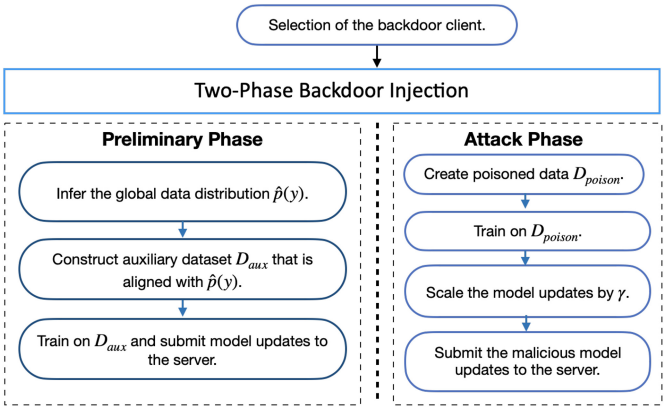

Fig. 2. Flowchart of the proposed two-phase backdoor attack.

any existing backdoor attacks. The goal of the preliminary phase is to accelerate the convergence of the FL model so that the subsequent backdoor can be more effective and consistent. Specifically, an attacker-controlled client first launches a passive whole population distribution inference attack by analyzing their local model updates and the FL global model update. To reduce weight divergence and improve the convergence behavior of the FL model, attacker-compromised clients then craft local training data by augmentation and downsampling so that the distribution $p_k(y)$ aligns with the inferred whole population distribution $\hat{p}(y)$. This step reduces the first term in (10), i.e., the distribution difference $\sum_{c=1}^{C}(p_k(y=c)-p(y=c))$. A dynamic sample size determination method is also utilized in crafting the data set to reduce the second term in (10), i.e., the gradient distance $[\nabla\mathbb{E}_{x\in D_k|y=c}[\log f_c(x; w_k^{T,t-1})] - \nabla\mathbb{E}_{x\in D|y=c}[\log(f_c(x; w_{\text{cen}}^{T,t-1})]]$. Instead of training on the original local data set, attacker-compromised clients train on the crafted data sets and submit the model updates to the central server. These steps seem legitimate but benefit the subsequent injected backdoor by reducing the dilution effect of other client model updates. When the backdoor client is selected, the backdoor is injected by training on a poisoned local data set and scales the malicious model updates by γ to ensure that the injected backdoor survives aggregation on the server.

Our proposed two-phase backdoor attack improves the performance of the early-injected backdoor because of the following features.

1) We propose a passive whole population distribution inference attack that requires no access to other clients' local data samples or their model updates.
2) By crafting the local data set using the inferred whole population distribution and sampling/augmentation techniques, the FL model weight divergence can be reduced, which facilitates the FL model convergence.
3) By reducing both the distribution difference and the gradient difference between the client's local data and the whole population data, the convergence of the FL model is improved. As a result, the backdoor model is less diluted by model updates from normal clients, leading to a stronger and longer-lasting backdoor effect.

B. Attack Workflow

1) Preliminary Phase: Whole Population Distribution Inference:

Step 1 (Approximation of the CL Model Updates): The attacker's goal is to estimate the whole population distribution $p(y)$ in the following expression of the gradient of the CL loss function:

$$\nabla F(w_{\text{cen}}; D) = \sum_{c=1}^{C} p(y=c)\nabla E_{x\in D|y=c}\big[\log f_c(x; w_{\text{cen}})\big]. \quad (11)$$

Therefore, $p(y)$ can be calculated if the values of $\nabla E_{x\in D|y=c}[\log f_c(x; w_{\text{cen}})]$ and $\nabla F(w_{\text{cen}}; D)$ are known. Since accessing the CL model is unrealistic, based on the findings in Remark 1, we approximate the CL model update by the FL model update

$$\sum_{k=1}^{K}\frac{n_k}{n}\Delta w_k \approx \Delta w_{\text{cen}} = \eta\sum_{\tau=1}^{t}\nabla F\Big(w_{\text{cen}}^{\tau-1}; D\Big). \quad (12)$$

The reasonability of the approximation is demonstrated by the following.

1) The bounded and small intra-aggregation weight divergence between the CL and the FL model. In Proposition 1, we show that the intra-aggregation weight divergence between a model in CL and FL settings is bounded by the difference in the gradient of the local data and the whole population. This gradient difference is usually caused by the difference in the number of samples between the local data and the whole population. The adversary could refer to a public data set or use augmentation techniques to narrow the difference. Furthermore, although the number of internal training epochs increases the gap, the number of local training epochs in practice is relatively small (usually 2 to 5), and, therefore, the impact of the number of internal training epochs should be minor. As a result, the FL model would not deviate much from the CL model within one aggregation.
2) The accurate global distribution inferred from the approximation. Extensive experiments are conducted in Section VI-A to verify that the approximation produces accurate results of whole distribution inference. The settings of these experiments are comprehensive as they cover both the balanced/imbalanced global distribution and the different non-i.i.d.-ness among local data. The results in all settings show that the difference between the true and the global distributions inferred from the approximation is condensed and small.

Step 2 (Decomposition of the Model Updates): Combining (11) and (12), we have the following gradient expression:

$$\sum_{k=1}^{K}\frac{n_k}{n}\Delta w_k = \eta\sum_{\tau=1}^{t}\sum_{c=1}^{C}p(y=c)\nabla E_{x\in D|y=c}\Big[\log f_c\Big(x; w_{\text{cen}}^{\tau-1}\Big)\Big]. \quad (13)$$

The model update of the compromised client a can be expressed as

$$\begin{aligned}\Delta w_a &= \eta\sum_{\tau=1}^{t}\nabla F_k\Big(w_a^{\tau-1}; D_a\Big)\\ &= \eta\sum_{\tau=1}^{t}\sum_{c=1}^{C}p_a(y=c)\nabla E_{x\in D_a|y=c}\Big[\log f_c\Big(x; w_a^{\tau-1}\Big)\Big]. \quad (14)\end{aligned}$$

Normally, the gradient is calculated directly by the partial derivative of the loss, for example, $\nabla F_k(w_a; D_a) = ([\partial F_k(w_a, D_a)]/[\partial w_a])$. Taking advantage of the linearity of the cross-entropy loss, the gradient $\nabla F_k(w_a, D_a)$ can also be viewed as a weighted average over $\nabla E_{x\in D_a|y=c}[\log f_c(x; w_a)]$. If the adversary gets a good estimate of $\nabla E_{x\in D|y=c}[\log f_c(x; w_{\text{cen}})]$, the global distribution $p(y)$ can be estimated by minimizing the difference between (13) and (14).

Step 3 (Estimation of the Gradients): The difference between the gradient calculated on D and D_a is mainly caused by the difference in the data sample sizes. Typically, a larger data set size would provide a less biased estimate. The adversary could obtain a more accurate estimate of $\nabla E_{x\in D|y=c}[\log f_c(x; w_{\text{cen}})]$ by augmenting D_a using data augmentation techniques or referring to a public data set when some of the classes are absent from the local data. However, purely pursuing a large data sample size is not always practical and effective, as some data augmentation methods are computationally expensive and time-consuming, or generate similar samples, which could on the contrary harm the accuracy of the estimation. Therefore, we adopt a dynamic data size determination algorithm proposed in [43] to determine when to stop the increase. The method evaluates the amount of augmentation by measuring the directional distance between the gradient of the augmentation and the estimate of the gradient. A scaler $\theta \in [0, 1]$, which indicates the cosine similarity between the gradient of augmentation and the gradient estimate, is used to determine when to stop the augmentation. A greater θ indicates a more accurate estimate, but a greater amount of augmentation.

Step 4 (Optimization-Based Global Distribution Estimation): In the previous step, the attacker gets a good estimate of $\nabla E_{x\in D|y=c}[\log f_c(x; a)]$ by augmenting D_a, the inference of whole population distribution $p(y)$ can then be formulated as an optimization problem, which seeks a $\hat{p}(y)$ that minimizes the difference of two losses in (13) and (14)

$$\min_{p(y)} \left\| \sum_{k=1}^{K} \frac{n_k}{n} \Delta w_k^T - \eta \sum_{\tau=1}^{t} \sum_{c=1}^{C} p(y=c) \nabla E_{x\in D_a|y=c}\left[\log f_c\left(x; w_a^{T,\tau-1}\right)\right] \right\|$$
$$\text{s.t.} \quad \sum_{c=1}^{C} p(y=c) = 1 \tag{15}$$

where $\sum_{k=1}^{K} (n_k/n)\Delta w_k^T$ is the FL global model update in the aggregation Tth and can be obtained by taking the difference between the $(T-1)$th and the Tth synchronization of FL global model.

Since the distribution $p(y)$ is not differentiable, an evolution algorithm is used to solve the optimization above. The evolution algorithm begins with a randomly initialized population of $p(y)$, namely, "fathers." Next, the individuals in the fathers go through mutation and crossover operations with a certain probability to generate more diverse individuals, namely, "children." Then, fathers and children are evaluated by an objective value, in which the individuals with better objective value will enter the next generation. Algorithms 1 and 2 detail the steps to solve the optimization.

Algorithm 1 Whole Population Distribution Inference by the Evolution Algorithm

Input: Number of classes C, population size S.
Output: An estimate of the whole population distribution $\hat{p}(y)$.

```
g = 0.
Initialize the distribution population p_0, which consists of S individuals. Each individual p_{0,s} satisfies Σ_{c=1}^{C} p_{0,s}(y = c) = 1.
Compute the FL global model update Δw^T.
Evaluate individuals in population p_0 by Algorithm 2.
while the termination criterion is not satisfied do
    g = g + 1.
    Create population q_g by crossover and mutation of individuals from p_{g−1}.
    Evaluate each individual in p_{g−1} in the children by Algorithm 2.
    Select S best individuals to population p_g from the populations p_{g−1} and q_g.
end while
Return the best individual in population p_g.
```

Algorithm 2 Evaluation of the Objective Values

Input: Number of classes C, internal training steps t, learning rate η, the global model update Δw^T, the label composition $p(y)$.
Output: The objective value defined in Eq. (15).

```
The attacker synchronizes with the latest global model w_a^{T,0} ← w^T.
for τ = 1 : t do
    for c = 1 : C do
        The attacker calculates the gradient component on class c: ∇E_{x∈D_a|y=c}[log f_c(x; w_a^{T,τ−1})].
    end for
    The model weight is updated by:
    w_a^{T,τ} = w_a^{T,τ−1} − η Σ_{c=1}^{C} p(y = c) ∇E_{x∈D_a|y=c}[log f_c(x; w_a^{T,τ−1})].
end for
Return the objective value ‖Δw^T − Δw_a^T‖, where Δw_a^T = w_a^{T,t} − w_a^{T,0}.
```

2) Preliminary Phase: Auxiliary Data Set Construction: After the adversary gets the inference of the whole population distribution, instead of training on the original local data set, the compromised client trains on an auxiliary data set, which is crafted to align with the inferred global distribution.

The basic idea of auxiliary data set construction is to augment the data in classes with inadequate samples and downsample the data in classes with excessive samples based on the inferred whole population distribution. Algorithm 3 describes the steps in constructing the auxiliary data set. In particular, the attacker first determines the total size of the auxiliary data set. The attacker then calculates the amount of data needed for each class by the size of the data set and the inferred global distribution. As for the augmentation operation, the adversary with a limited computation budget can use trivial methods, such as random shift, random rotation, random shear, and random zoom, while a strong adversary could utilize more advanced methods, such as data synthesis and data reconstruction. For the downsample operation, it randomly samples from current data until the desired number of samples is reached. The auxiliary data set constructed in this way mitigates both terms in (10).

Algorithm 3 Auxiliary Data Set Construction

Input: Auxiliary dataset size M, the inferred data distribution $\hat{p}^{(y)}$, number of classes C, the compromised dataset D_a
Output: Auxiliary dataset D_{aux}.
1: Calculate the data size of each class c by $M_c \leftarrow M \times \hat{p}(y = c)$ for $c = 1, \ldots, C$.
2: Calculate the data size of each class c of D_a, $|D_a|c|$, where $D_a|c := \{x|y : x \in D_a, y = c\}$.
3: **for** $c = 1{:}C$ **do**
4: **if** $|D_a|c| < M_c$ **then**
5: Augment $|D_a|c|$ to M_c.
6: **else**
7: Down-sample from $D_a|c$, such that $|D_a|c| = M_c$.
8: **end if**
9: Auxiliary dataset $D_{aux} \leftarrow \cup_{c=1}^{C} D_a|c$.
10: **end for**
11: Shuffle D_{aux}.
12: Return D_{aux}.

C. Attack Phase: Backdoor Injection

The attacker-compromised clients perform training on the crafted auxiliary data set until the backdoor client equipped with backdoor capability is selected. The backdoor client first poisons its local data D_a by adding backdoor triggers to a subset of D_a, and changes their labels to a target one to form a poison data subset D_{poison}. The rest of the data are kept clean and are denoted as D_{clean}. The attacker then performs local training on $D_{\text{poison}} \cup D_{\text{clean}}$ aiming to maximize accuracy on both the main task and the backdoor task

$$w_a^* = \arg\min_{w_a} \left[F_a(w_a; D_{\text{clean}}) + F_a\left(w_a; D_{\text{poison}}\right) \right].$$

After local training, the attacker scales the model updates by a parameter $\gamma = (n/n_a) \approx K$ to ensure that the backdoor model update survives the aggregation and ideally replaces the global model. The attacker could also use constrain-and-scale or train-and-scale to improve its persistence and evade anomaly detection mechanisms.

D. Coordination of Multiple Attacker-Controlled Clients

The above presentation of the attack process is based on a single attacker-controlled client, but it can be easily extended to the scenario, in which the attacker controls multiple clients. The whole population distribution inference attack can be performed by any one of the compromised clients. The inferred global distribution is then shared with other attacker-controlled clients, and each of them constructs and trains on the auxiliary data set locally. The use of multiple malicious clients can further improve the accuracy of the FL model.

V. Experimental Setup

A. Data Set

We evaluate our proposed method on the handwritten digit recognition data set, MNIST [44]. The data set contains 60 000 training data samples, and 10 000 testing data samples. Each data sample is a square 28×28 pixel image of a handwritten single digit between 0 and 9.

B. Evaluation Metrics

1) *Accuracy of Whole Population Distribution Inference Attack:* We measure its accuracy by the ℓ_2 distance of the inferred whole population distribution $\hat{p}$ and the true whole population distribution p_{global}, i.e., $\|\hat{p} - p_{\text{global}}\|$, referred to as inferred-to-true. A smaller distance indicates a more accurate inference result. And, we also evaluate the ℓ_2 distance of the original distribution on the kth client p_k and p_{global}, i.e., $\|p_k - p_{\text{global}}\|$, referred to as original-to-true. The difference between such two distances is positively related to the amount of weight divergence, which can be reduced by the whole population distribution alignment.
2) *Main Task FL Model Accuracy Gain by Whole Population Distribution Alignment:* We measure the FL global model accuracy as a function of training epochs for regular FL (clients train on the original data sets) and preliminary phase assisted FL (clients train on crafted local data sets that align with the gradients and distribution of the whole population).
3) *Main Task FL Model Accuracy in Presence of Backdoor Attack:* We also present the accuracy of the main task when the backdoor attack is in place. As mentioned previously, the main task might deteriorate due to the scaling operation and the dilution from the normal model updates, especially, when they are large in the early training stage. The server could reject the model updates if an unexpected drop is observed in the accuracy of the main task.
4) *Backdoor Attack Success Rate and Longevity:* Given a classifier $f(\cdot)$, the backdoor attack accuracy is defined as the portion of samples in the backdoor samples that the classifier predicts as the target label y_t

$$\text{Acc}_{\text{backdoor}} = \frac{\left|\{|x \in D_{\text{poison}} : f(x) = y_t\}\right|}{|D_{\text{poison}}|}.$$

The test data are constructed by adding the backdoor triggers to the original test data samples. To avoid the influence of the original data of the target label, we remove the data of the target label in the test data. We plot the backdoor success rate of 20 global epochs since injection to assess their longevity.

C. FL System Setting

We implement the FL and the proposed two-phase backdoor attack using the PyTorch framework. We conduct our experiments on Google Colab Pro (CPU: Intel Xeon CPU @ 2.20 GHz; RAM: 13 GB; GPU: Tesla P100-PCIE-16 GB with CUDA 11.2).

The data set is allocated to 100 clients. In each global model aggregation, ten clients are randomly selected to participate in the FL training. Each client maintains a local model consisting of two convolutional layers and two fully connected layers. Due to the inherent data non-i.i.d.-ness across edge clients, there can be a significant difference between the local distributions (non-i.i.d.-ness) and whole population distribution (whole population imbalance). We consider

TABLE II
MNIST DATA SET SETTINGS

Settings	Whole population	Local distribution
1	balanced	non-i.i.d., $\alpha = 1$
2	balanced	non-i.i.d., $\alpha = 0.1$
3	imbalanced	non-i.i.d., $\alpha = 1$
4	imbalanced	non-i.i.d., $\alpha = 0.1$

both balanced/imbalanced whole population and different non-i.i.d.-ness among clients' local data (Table II) to evaluate the effectiveness, generalizability, stability, and feasibility of the proposed two-phase backdoor attack. The whole population imbalance is simulated by randomly sampling 50%–100% for each class of the original data set. And, we use the Dirichlet distribution [45] with a hyperparameter α to generate different data distributions among clients, where a smaller α indicates a greater non-i.i.d.-ness.

1) Preliminary Phase: The clients are randomly selected to participate in a training round, with a certain percentage of clients training on D_{aux}, which aligns with the whole population by Algorithm 3. The FL model is trained with full-batch gradient descent with internal epoch $t = 1$ and learning rate $\eta = 0.1$.

As specified in Section III, the adversary has the ability to enlarge the local data set using augmentation techniques or referring to public data sets. In our experiment, we assume that the adversary is equipped with trivial augmentation methods. We also assume that the attacker holds 1% of the MNIST data set, from which the attacker can draw data samples and complement the auxiliary data set. In the dynamic data size determination algorithm that determines when to stop the augmentation, we set $\theta = 0.8$, which means that the augmentation operation stops when the cosine similarity between the gradient of augmentation and the gradient estimate reaches 0.8. To avoid the influence of the size of D_{aux}, we set the size of D_{aux} to be the same as that of the original data set. The fractions of clients controlled by the attacker are chosen to be 5%, 10%, and 20% of the total number of clients denoted as ours_5, ours_10 and ours_20, respectively. And they are collectively referred to as ours.

2) Attack Phase: We use pixel pattern backdoors, as the same as those in [12] and [26]. We set the 4×4 pixels in the upper left corner of the image to white (pixel value of 0), and swap the label with the target label 0. The ratio between the size of the backdoor trigger and the size of the data sample is 2%.

The performance of the proposed backdoor (both the main task accuracy and the backdoor success rate) is evaluated on an FL with mini-batch gradient descent with a batch size of 128. The backdoor client poisons 40 out of 128 data samples in each mini-batch and locally trains for poison epochs of 10 with a poison learning rate of 0.05. The global learning rate is the same as the local learning rate $\eta = 0.1$. The scaling factor is $\gamma = K = 10$.

VI. EXPERIMENTAL RESULTS

A. Accuracy of the Whole Population Distribution Inference

The global distribution inference attack is launched at every epoch of the first 30 epochs. We present the box plot of $\|p_k - p_{\text{global}}\|$ (referred to as original-to-true) and $\|\hat{p} - p_{\text{global}}\|$ (referred to as inferred-to-true) in Fig. 3. In all four settings, compared to original-to-true, inferred-to-true is significantly smaller and more condensed, indicating that the proposed whole population distribution inference attack achieves high accuracy. Furthermore, our proposed inference attack is equally accurate in both balanced and imbalanced whole population distribution settings (setting 1 versus setting 2 and setting 3 versus setting 4).

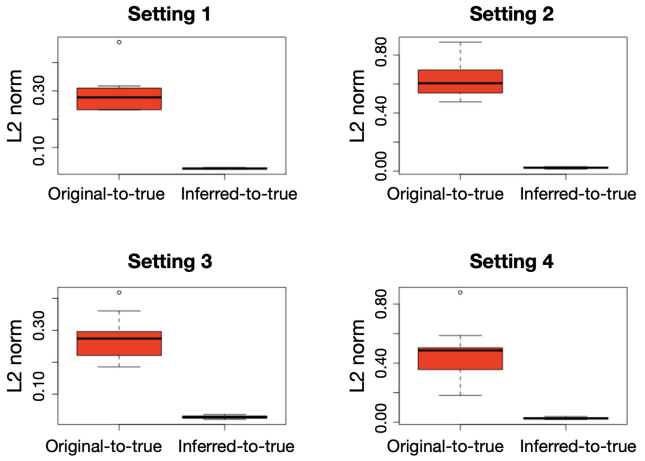

Fig. 3. Box plot of $\|p_k - p_{\text{global}}\|$ (original-to-true) and $\|\hat{p} - p_{\text{global}}\|$ (inferred-to-true).

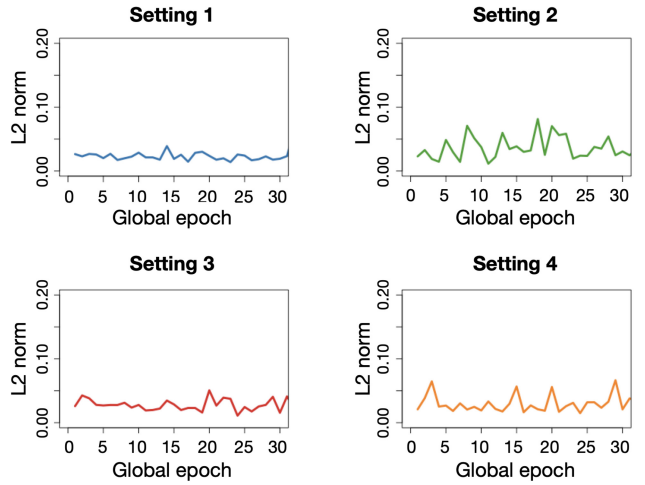

Fig. 4. $\|\hat{p} - p_{\text{global}}\|$ (inferred-to-true) versus the global training epoch.

We also plot the inferred-to-true as a function of training epochs (shown in Fig. 4) to demonstrate the stability of the proposed whole population distribution inference attack. The FL model begins to converge at epoch 20, so our inference attack window covers different convergence stages of the training process. Results show that the inference results are stationary along the training process, which means that the inference is accurate regardless of the training stage, including the early, the middle, and the convergence stages of the training. Such results provide performance guarantee for the subsequent backdoor attack. The fluctuations presented in Fig. 4 are due to the randomness of the local distributions in the selected clients in each FL training round. Especially,

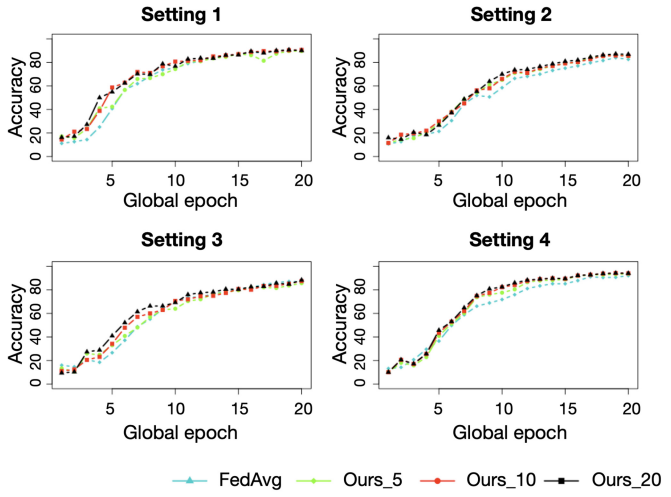

Fig. 5. Accuracy of the main task of 5%, 10%, and 20% of the local data of the clients that perform the alignment, averaged over ten experiments.

the fluctuation becomes more noticeable when clients' local distributions are more non-i.i.d. (settings 2 and 4). To further reduce such fluctuations and improve the accuracy of the inference, the adversary could further refine the inference result by performing statistical analysis on multiple inference results, such as averaging or clustering.

B. Main Task Accuracy Under the Nonattack Scenario

We evaluate the performance gain of the proposed preliminary phase in improving FL convergence by the accuracy of the FL main task, shown in Fig. 5. In all four settings, compared to FedAvg, the FL with global distribution alignment converges faster, although they eventually reach the same accuracy. This performance gain is more perceptible before the FL begins to converge and when a greater fraction of clients perform the proposed alignment. In addition, while the global distribution alignment has more influence on the very early stage (epoch 0 to epoch 10) for settings 1 and 3 ($\alpha = 1$), a higher non-i.i.d.-ness ($\alpha = 0.1$ in settings 2 and 4) has more impact on the middle training stage (epoch 5 to epoch 15). The experimental results are consistent with the findings of Proposition 1: reducing both the gradient and the distribution between the client's local data and the whole population could reduce the model weight divergence, leading to a better convergence performance.

C. Backdoor Attack Performance

We present the impact of backdoor injection on the main task accuracy as well as the backdoor success rate. We evaluate the proposed two-phase backdoor attack and compare it with two existing backdoor attacks: 1) the centralized backdoor attack [12] (referred to as baseline), in which the local data set is poisoned by a centralized backdoor trigger and 2) the distributed backdoor attack [26] (referred to as DBA), in which the backdoor trigger is partitioned into parts and each part is injected separately.

1) Main Task Accuracy: Unlike the backdoors injected at the convergence of the FL model, where the injection of

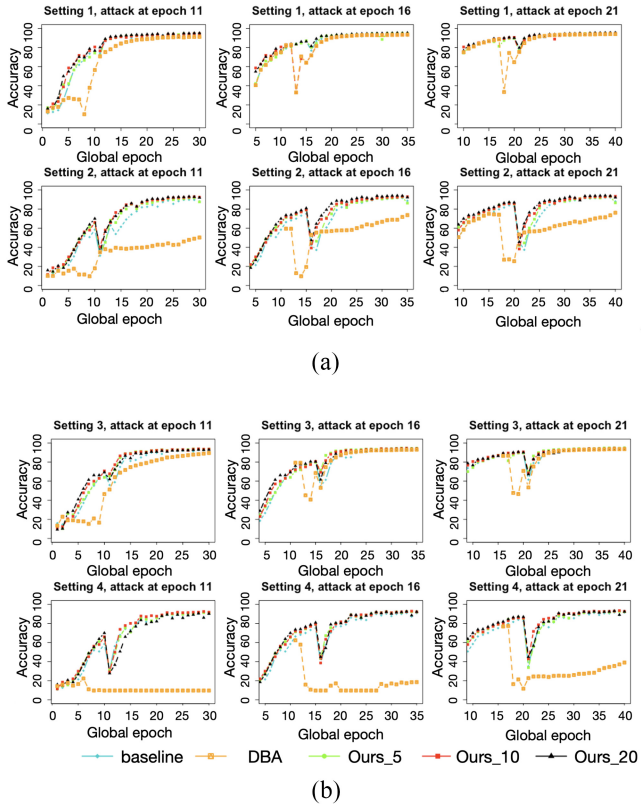

Fig. 6. Main task accuracy of the FL global model when the backdoors are injected at FL epochs 10, 15, and 20, respectively. (a) Settings 1 and 2. (b) Settings 3 and 4.

the backdoor barely disturbs the accuracy of the main task, the early-injected backdoor usually noticeably deteriorates the accuracy of the main task due to the model updates from normal clients. When the backdoor is injected in the early training stage, the accuracy of the main task usually experiences a sudden drop and then gradually goes back to normal status afterward. As introduced in [46], the central server could monitor the FL model main task accuracy and reject model updates that make the main task accuracy abnormally low. This approach could fail to be deployed on the FL system since the central server does not always have access to the model updates and test data, thus, cannot measure their accuracy, or a false alarm could be triggered due to the extremely low local accuracy caused by the participation of clients with extremely non-i.i.d. data. However, the accuracy of the main task can still be used to evaluate the stealthiness of the backdoor attack.

As shown in Fig. 6, the accuracy of the main task is affected by backdoor injection to varying degrees. The dropped main task is a collective consequence of the scaled-backdoored model updates and model updates from the rest of the participants. And such a main task accuracy drop becomes more critical for a greater non-i.i.d.-ness among clients (settings 2 and 4). Compared to the baseline, our backdoor introduces less drop in main task accuracy in most cases. In some cases, the main task accuracy impacted by our proposed backdoor attack presents a faster recovery rate. Furthermore, compared to the baseline and ours, DBA suffers the greatest drop in the main task accuracy and it takes much longer for the underlying

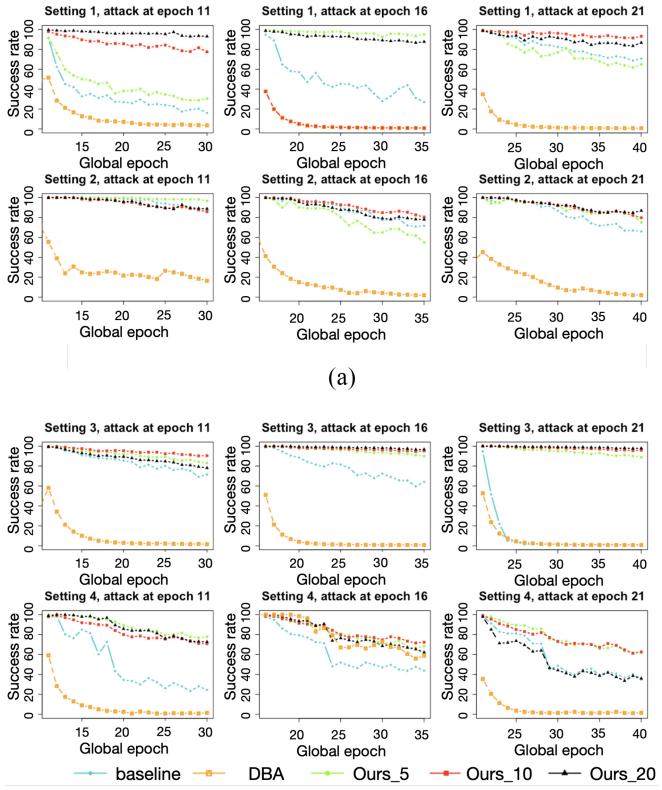

Fig. 7. Backdoor success rate in 20 training epochs since backdoor injection. (a) Settings 1 and setting 2. (b) Settings 3 and setting 4.

FL to return to normal. This phenomenon is even worsened in the setting of high non-i.i.d.-ness (setting 2 and setting 4). A possible explanation is that DBA requires multiple clients sequentially to perform the injection of part of the backdoor trigger to finish the injection of a complete backdoor, which poses a longer and worse impact on the main task accuracy. Especially, in the highly non-i.i.d. and globally unbalanced scenario, given that the model updates are already far from others, the consecutive injection and scale operations could make the deviation even worse and prevent the FL model from convergence (evidenced in setting 4). Thus, we conclude that the proposed backdoor attack is more stealthy than the baseline and the DBA.

2) Backdoor Attack Accuracy: To explore the effectiveness of a backdoor before the FL model converges, we inject centralized backdoors for the baseline and our method at the global FL epochs 11, 16, and 21, respectively. To fairly compare with DBA, the distributed backdoors are sequentially injected and finished in the same round as the centralized backdoors. For example, if the centralized backdoor is injected in round 11, four distributed backdoor triggers are injected separately in rounds 8, 9, 10, and 11.

Fig. 7 presents the backdoor success rate for 20 FL global epochs since the completion of the backdoor injection. For each setting, injections are made in different epochs by the same client. The injected backdoor has maximum effectiveness immediately after injection. In subsequent epochs, as

TABLE III
MEAN BACKDOOR SUCCESS RATE (%) OVER TEN FL EPOCHS SINCE INJECTION (AVERAGED OVER TEN EXPERIMENTS)

Epoch	Baseline	DBA	Ours_5	Ours_10	Ours_20
Setting 1					
11	95.8 ± 1.8	23.2 ± 4.1	94.1 ± 5.9	94.5 ± 1.5	95.9 ± 2.5
16	97.4 ± 2.4	17.2 ± 5.4	97.4 ± 1.3	95.6 ± 3.9	95.3 ± 1.6
21	95.3 ± 5.3	14.5 ± 1.3	95.3 ± 4.2	96.3 ± 3.4	95.7 ± 1.9
Setting 2					
11	52.9 ± 4.7	77.8 ± 28.4	74.0 ± 23.0	76.6 ± 12.2	80.6 ± 27.2
16	61.3 ± 27.5	67.8 ± 10.8	75.0 ± 22.6	82.3 ± 14.4	78.4 ± 17.8
21	68.4 ± 22.3	11.8 ± 2.2	79.2 ± 19.7	79.7 ± 13.6	77.1 ± 18.1
Setting 3					
11	15.5 ± 6.5	$13.5\ pm3.2$	44.8 ± 25.6	65.3 ± 28.1	66.1 ± 25.1
16	57.7 ± 16.5	9.7 ± 1.9	69.3 ± 18.3	66.1 ± 28.7	62.2 ± 20.1
21	64.4 ± 13.8	6.3 ± 7.8	73.4 ± 14.7	69.7 ± 13.6	72.4 ± 16.5
Setting 4					
11	48.7 ± 41.5	52.3 ± 10.3	67.1 ± 18.2	75.1 ± 26.5	88.3 ± 10.1
16	69.0 ± 4.2	40.7 ± 9.9	72.3 ± 15.7	83.3 ± 23.3	72.2 ± 17.3
21	70.3 ± 2.0	11.7 ± 4.2	73.4 ± 14.7	88.1 ± 13.4	85.5 ± 8.3

the FL model aggregates new normal updates, the effect of the backdoor is weakened, which is reflected in the gradually decreasing success rate. In most cases, after 20 rounds of backdoor injection, the success rates of almost all settings and injection epochs are greater than those of the baseline and DBA. DBA does not reach a similar backdoor effect as in the baseline and ours. The reason for this gap could be that the partially injected backdoor effect in previous rounds is more likely to be hindered by normal local updates in the subsequently injected backdoor parts. And in most cases, our proposed backdoor retains a lower diminishing rate, compared to the baseline.

Due to the non-i.i.d.-ness among clients' local data, some clients' data may be in favor of the attack, while others are not. In addition, the backdoor effect does not always steadily decrease or bounces in some cases. Therefore, we evaluate both attack strength and longevity by the mean attack success rate of 10 FL epochs since injection (Table III). In general, the backdoor injected in very early rounds (epoch 5 and epoch 10) achieves a lower mean attack success rate, compared to the ones injected in epoch 20. This degradation in the effectiveness of the attack is made even worse when the whole population is imbalanced (setting 1 versus setting 3) and non-i.i.d.-ness among clients increases (setting 1 versus setting 2). In most cases, our proposed backdoor attack outperforms the baseline and the DBA. And compared to the baseline, the gain in attack performance is positively related to the percentage of attacker-controlled clients that perform the whole population distribution alignment.

D. Overhead Analysis

1) Preliminary Phase: The computational cost of this phase consists of three parts: 1) calculating the gradients of the data of each label; 2) solving the optimization in (15); and 3) constructing the auxiliary data set.

For the first part, the attacker trains the FL global model on the data samples of each label separately to obtain gradients $\nabla \mathbb{E}_{x \in D_a | y=c}[\log f_c(x; w_a)]$, and because $n = \sum_{c=1}^{C} n_c$, where n_c is the number of samples of label c, the time complexity is the same as that of local training. Since the batch gradient

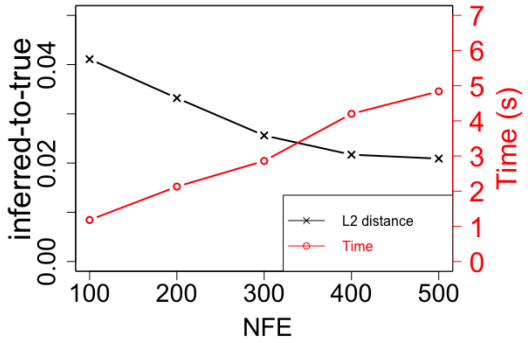

Fig. 8. Inference accuracy (inferred-to-true) and time taken versus NFE.

has a time complexity of $\mathcal{O}(n^2m)$, in which n is the number of data samples and m is the number of features, the time complexity of the first part is also $\mathcal{O}(n^2m)$.

For the second part, we evaluate the number of function evaluations (NFEs), which is commonly used to evaluate an evolution algorithm. NFE is usually measured when a good solution is delivered or when no significant change in the solution is observed. We plot inferred-to-true and the real-time used against NFE, shown in Fig. 8, to demonstrate the effect of NFE on the inference accuracy and time. The NFE is set to 400 in our experiment, because there is no significant inference accuracy gain after 400 NFEs. The actual time taken to solve the optimization with 400 NFEs is 4 s.

The construction of an auxiliary data set that is aligned with the whole population consists of augmentation and sampling operations. Examples of trivial augmentation methods are flipping ($\mathcal{O}(np)$, where p is the number of pixels in each image), rotation, random crop, and scale [they have the same complexity as $\mathcal{O}(n)$]. The sampling operation has a time complexity of $\mathcal{O}(n)$. Therefore, the total complexity of constructing the auxiliary data set is at most $\mathcal{O}(np)$ and the actual time spent is 0.03 s.

2) Backdoor Phase: The backdoor client poisons a subset of local data by injecting the backdoor pattern and swaps the label to the target label, then performs local training on the poisoned local data set. The total time complexity is $\mathcal{O}(n^2m)$. The real-time spent on the backdoor attack with 10 internal training epochs is around 13 s.

The complexity analyses are summarized in Table IV. Gradient calculation and optimization are only needed to be performed in the beginning to obtain an accurate inference of the whole population distribution. The actual time taken for these two steps is around 4 s, which is far less than the time taken by the backdoor attack. Then, the inferred distribution is shared among all attacker-controlled clients, which perform the auxiliary data set construction, whose time complexity is negligible.

VII. Robustness of the Proposed Attack

In this section, we are interested in how the proposed attacks will behave when defense mechanisms are in place. In the following, we will analyze the effectiveness of the proposed two-phase backdoor attack against two main defense strategies.

TABLE IV
Time Complexity and Real-Time Spent on the Proposed Inference Attack

Operation	Time complexity	Real time taken (s)
Gradient calculation	$\mathcal{O}(n^2m)$	0.11
Optimization	400 NFEs	4.20
Auxiliary dataset construction	$\mathcal{O}(nk)$	0.03
Backdoor attack	$\mathcal{O}(n^2m)$	13.37

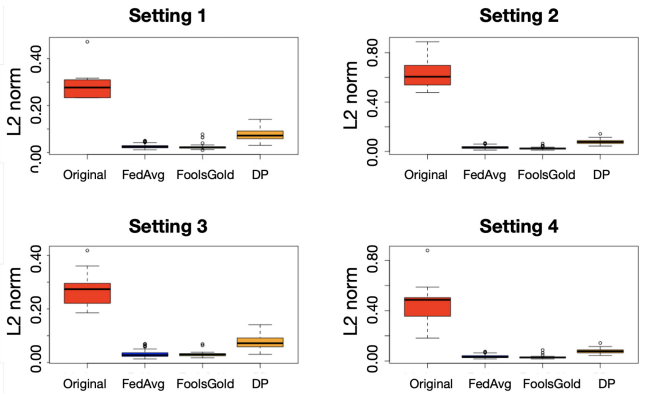

Fig. 9. Box plot of original-to-true (Original) and inferred-to-true of FedAvg, FoolsGold, and DP based on 30 instances.

FoolsGold is a robust aggregation strategy, which calculates the cosine similarity of all historical gradient records and assigns smaller aggregation weights to clients who continuously contribute similar gradient updates [27].

DP is a noise-based method that limits the effectiveness of backdoor attacks by two key steps [39]: 1) model parameters are clipped to limit the sensitivity of local model updates and 2) the Gaussian noises are added to local model updates. We consider a local DP, in which each client adds noise before uploading the model updates to the server. We use the (ϵ, δ)-DP proposed in [47] with a popular choice of $\sigma = \sqrt{2\log(1.25/\delta)}/\epsilon$ with $\delta = 10^{-5}$ and $\epsilon = 50$. The clipping bound is set to the median of the norms of the unclipped local model updates during training. The noises are only applied to normal model updates, while the backdoor client sends the nonperturbed backdoored model updates.

A. Whole Population Distribution Inference Accuracy Against Defense Strategies

We first present the whole population distribution inference against FedAvg, FoolsGold, and DP, shown in Fig. 9. Since FoolsGold does not interfere with benign FL settings, the whole population distribution inference against FoolsGold is as accurate as that in FedAvg. Although DP provides a statistical guarantee for record-level information, DP does not protect statistical information, such as the whole population distribution. With DP in place, although the inference is not as accurate as in the FedAvg case, the inferred-to-true is still notably lower compared with original-to-true. Thus, both FoolsGold and DP fail to defend against the proposed inference attack.

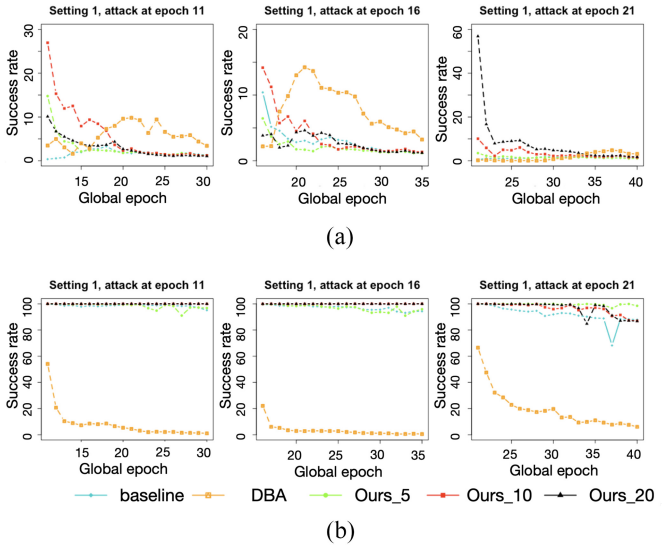

Fig. 10. Backdoor success rate (%) of 20 training epochs since injection against (a) FoolsGold and (b) DP defense mechanisms.

B. Performance of the Backdoor Attack Against Defense Strategies

We implement the proposed backdoor attack against FoolsGold and DP in setting 1. We plot the backdoor success rate for 20 epochs after injection to observe its injection strength and longevity. Fig. 10(a) shows that in most cases ours reaches a significantly higher success rate upon injection and maintains such a high success rate in the following 20 epochs. For example, when injected in the early training stage (at epoch 11), the baseline backdoor fails while ours_5, ours_10, and ours_20 achieve attack success rates of 14%, 27%, and 10%, respectively. As the convergence is expedited by the proposed preliminary phase, the model updates from normal clients become smaller and more similar, so that FoolsGold will reduce their assigned weights, and as a result, the backdoored model updates become more influential. Compared to ours and the baseline, the success rate curve of the DBA has a different pattern, in which the success rate first increases and then decreases and is more persistent than both the baseline and ours in later rounds. However, in practice, the requirement for clients with distributed backdoor triggers to be selected in consecutive rounds can hardly be met.

Fig. 10(b) shows the attack performance against DP. Both the baseline and ours achieve high-attack success rates that are even comparable to those of FedAvg. This phenomenon indicates that, instead of mitigating the backdoor effect, the noise added to the normal clients helps the backdoored model corrupt the FL global model. A possible explanation is that the added noise reduces the utility of normal model updates, which on the contrary strengthens backdoored model updates in the FL aggregation. Furthermore, ours is markedly better than the baseline when the backdoor is injected in subsequent rounds. For example, baseline and ours have similar attack performance in the early training stage, e.g., epoch 11. And ours performs distinctly better in the later training stage, that is, the backdoors injected in epochs 16 and 21. Finally, both the baseline and ours outperform DBA. This is because the effectiveness of the distributed backdoor is mitigated by both benign model updates and DP noise multiple times before DBA finishes injecting the complete backdoor.

C. Discussion of Defense Mechanisms

Existing defenses cannot eliminate the impact of inference and backdoor attacks. Each of these defense mechanisms has its pros and cons. We briefly examine some of strategies against inference attacks and backdoor attacks and discuss their effectiveness.

DP has been shown effective in defending against most inference attacks, except property inference attack, to which the proposed whole population inference attack belongs. The reason is that the accuracy of the inference results is reversely related to the scale of the added noise. Even the whole population distribution inference result may not be satisfactory under DP with a large noise scale, the result still helps in aligning the distribution and benefits the subsequent backdoor attack to some extent. As for defending against backdoor attacks, DP is proven to be effective only if a low-clipping bound and high variance are in use, and at the cost of degrading the performance on its main task. Otherwise, when the noise variance is small, as shown in Section VII-B, in this article, the small added noise in turn strengthens the backdoor attack.

Secure multiparty computation (MPC) is a cryptographic approach that allows clients to submit encrypted/masked model updates, such that the server can only learn the aggregation of the clients' model updates. The representative methods are homomorphic encryption (HE) and secret sharing. Many works advocate the use of additive HE schemes, Paillier, to protect FL [48], [49]. Considering the power constraints of IoT devices, HE is hardly applicable in MEC scenario. A more promising solution is secret sharing, where clients randomly add zero-sum masks to model updates such that the masked model updates are indistinguishable from random values [50], [51], eliminating the risk of both inference and backdoor attacks. And the added masks cancel out in the server aggregation, so that the FL convergence will not be impacted. But it has a strict requirement on synchronization.

Robust aggregation tries to alleviate the backdoor effect by de-emphasizing the malicious ones via statistical properties of model updates. However, in FL settings, the non-i.i.d. local distributions among clients provide enough space for the backdoored model updates to hide. In addition, the attack can easily circumvent the defense by decomposing the backdoored model into smaller or orthogonal ones.

Pruning and fine-tuning reduces backdoor impact by removing dormant neurons (neurons that are not activated by clean data) of the FL model after the training stage [52]. However, the attacker can train the backdoor model in a pruning-aware way so that this method does not fully eliminate the backdoor impact. In addition, it cannot remove the backdoor behavior without significantly degrading the main task performance.

VIII. Conclusion

In this article, we proposed a novel information leakage-assisted single-shot backdoor attack that improves the

effectiveness of the backdoor injected in the early training stage. We first showed that clients training on data sets that are aligned with the whole population in both distribution and gradient can improve the convergence of the FL model. Based on this observation, we introduced a preliminary phase to the subsequent backdoor attack, in which attacker-controlled clients first infer the whole population distribution from the shared FL model updates and then train on locally crafted data sets that align with both the distribution and gradient of the whole population. Benefiting from the preliminary phase, the subsequent backdoor injection suffers less dilution effect from the model updates of other clients and achieves better effectiveness. We demonstrated the effectiveness of the proposed backdoor attacks in the early training stage through extensive experiments on a real-world data set. The results showed that the proposed backdoor can achieve a longer lifespan than existing backdoor attacks. We hope that our work draws attention to vulnerabilities in the early training stage of FL. Our analysis and findings provide novel insights into the field of strengthening FL attacks by information leakage, which could help evaluate and improve the robustness of FL.

APPENDIX

Proof:

$$\begin{aligned}
&\left\|w_k^{T,t}-w_{\text{cen}}^{T,t}\right\| \\
&=\Bigg\|w_k^{T,t-1}-\eta\underbrace{\sum_{c=1}^{C}p_k(y=c)\nabla\mathbb{E}_{x\in D_k|y=c}\left[\log f_c\left(x;w_k^{T,t-1}\right)\right]}_{A_1} \\
&\quad-w_{\text{cen}}^{T,t-1}+\eta\underbrace{\sum_{c=1}^{C}p(y=c)\nabla\mathbb{E}_{x\in D|y=c}\left[\log f_c\left(x;w_{\text{cen}}^{T,t-1}\right)\right]}_{A_2} \\
&\quad-\eta\underbrace{\sum_{c=1}^{C}p_k(y=c)\nabla\mathbb{E}_{x\in D|y=c}\left[\log\left(f_c\left(x;w_{\text{cen}}^{T,t-1}\right)\right)\right]}_{A_3} \\
&\quad+\eta\underbrace{\sum_{c=1}^{C}p_k(y=c)\nabla\mathbb{E}_{x\in D|y=c}\left[\log\left(f_c\left(x;w_{\text{cen}}^{T,t-1}\right)\right)\right]}_{A_3}\Bigg\| \qquad (16)\\
&\overset{(1)}{\leq}\left\|w_k^{T,t-1}-w_{\text{cen}}^{T,t-1}\right\|+\eta\|A_1-A_3\|+\eta\|A_2-A_3\| \qquad (17)\\
&=\left\|w_k^{T,t-1}-w_{\text{cen}}^{T,t-1}\right\| \\
&\quad+\eta\Bigg\|\sum_{c=1}^{C}p_k(y=c)\Bigg[\nabla\mathbb{E}_{x\in D_k|y=c}\left[\log f_c\left(x;w_k^{T,t-1}\right)\right] \\
&\qquad-\nabla\mathbb{E}_{x\in D|y=c}\left[\log\left(f_c\left(x;w_{\text{cen}}^{T,t-1}\right)\right)\right]\Bigg]\Bigg\| \\
&\quad+\eta\left\|\sum_{c=1}^{C}\left[(p(y=c)-p_k(y=c))\right]\nabla\mathbb{E}_{x\in D|y=c}\left[\log\left(f_c\left(w_k^{T,t-1}\right)\right)\right]\right\| \qquad (18)
\end{aligned}$$

where the inequality (1) holds due to the Cauchy–Schwarz inequality. By induction, we have

$$\begin{aligned}
&\left\|w_k^{T+1}-w_{\text{cen}}^{T+1}\right\|\leq\left\|w_k^{T}-w_{\text{cen}}^{T}\right\| \\
&\quad+\eta\sum_{\tau=1}^{t}\left\|\sum_{c=1}^{C}\left[(p(y=c)-p_k(y=c)\right]\nabla\mathbb{E}_{x\in D|y=c}\left[\log\left(f_c\left(w_k^{T,\tau-1}\right)\right)\right]\right\| \\
&\quad+\eta\sum_{\tau=1}^{t}\Bigg\|\sum_{c=1}^{C}p_k(y=c)\Bigg[\nabla\mathbb{E}_{x\in D_k|y=c}\left[\log f_c\left(x;w_k^{T,\tau-1}\right)\right] \\
&\qquad-\nabla\mathbb{E}_{x\in D|y=c}\left[\log\left(f_c\left(x;w_{\text{cen}}^{T,\tau-1}\right)\right)\right]\Bigg]\Bigg\|. \qquad (19)
\end{aligned}$$

Hence, (9) has been proved. And the proof of (10) follows similar steps. ■

ACKNOWLEDGMENT

Any opinions, findings, conclusions, or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of NSF.

REFERENCES

[1] T. Liu, X. Hu, and T. Shu, "Assisting backdoor federated learning with whole population knowledge alignment in mobile edge computing," in *Proc. 19th Annu. IEEE Int. Conf. Sens. Commun. Netw. (SECON)*, 2022, pp. 416–424.

[2] T. Barnett, S. Jain, U. Andra, and T. Khurana, "Cisco visual networking index (VNI) complete forecast update, 2017–2022," presented at the Americas/EMEAR Cisco Knowl. Netw. (CKN), 2018, pp. 1–30.

[3] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," 2016, *arXiv:1610.05492*.

[4] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Stat.*, vol. 54, Apr. 2017, pp. 1273–1282.

[5] M. Chen, R. Mathews, T. Ouyang, and F. Beaufays, "Federated learning of out-of-vocabulary words," 2019, *arXiv:1903.10635*.

[6] T. Yang et al., "Applied federated learning: Improving Google keyboard query suggestions," 2018, *arXiv:1812.02903*.

[7] S. Ramaswamy, R. Mathews, K. Rao, and F. Beaufays, "Federated learning for emoji prediction in a mobile keyboard," 2019, *arXiv:1906.04329*.

[8] A. Hard et al., "Federated learning for mobile keyboard prediction," 2018, *arXiv:1811.03604*.

[9] X. Han, H. Yu, and H. Gu, "Visual inspection with federated learning," in *Image Analysis and Recognition*, F. Karray, A. Campilho, and A. Yu, Eds. Cham, Switzerland: Springer Int., 2019, pp. 52–64.

[10] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, "Federated learning for healthcare informatics," *J. Healthc. Inform. Res.*, vol. 5, no. 1, pp. 1–19, 2021.

[11] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. C. Paschalidis, and W. Shi, "Federated learning of predictive models from federated electronic health records," *Int. J. Med. Inform.*, vol. 112, pp. 59–67, Apr. 2018.

[12] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," 2018, *arXiv:1807.00459*.

[13] H. Wang et al., "Attack of the tails: Yes, you really can backdoor federated learning," 2020, *arXiv:2007.05084*.

[14] Z. Sun, P. Kairouz, A. T. Suresh, and H. B. McMahan, "Can you really backdoor federated learning?" 2019, *arXiv:1911.07963*.

[15] J. Zhang, B. Chen, X. Cheng, H. T. T. Binh, and S. Yu, "PoisonGAN: Generative poisoning attacks against federated learning in edge computing systems," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3310–3322, Mar. 2021.

[16] C. Dwork and M. Naor, "On the difficulties of disclosure prevention in statistical databases or the case for differential privacy," *J. Privacy Confidentiality*, vol. 2, no. 1, p. 8, 2010.

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

[18] T. Orekondy, S. J. Oh, Y. Zhang, B. Schiele, and M. Fritz, "Gradient-leaks: Understanding and controlling deanonymization in federated learning," 2018, *arXiv:1805.05838*.

[19] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the GAN: Information leakage from collaborative deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2017, pp. 603–618.
[20] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *Proc. IEEE Symp. Security Privacy (SP)*, 2019, pp. 691–706.
[21] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," 2018, *arXiv:1812.00910*.
[22] G. Ateniese, L. V. Mancini, A. Spognardi, A. Villani, D. Vitali, and G. Felici, "Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers," *Int. J. Security Netw.*, vol. 10, no. 3, pp. 137–150, 2015.
[23] K. Ganju, Q. Wang, W. Yang, C. A. Gunter, and N. Borisov, "Property inference attacks on fully connected neural networks using permutation invariant representations," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2018, pp. 619–633.
[24] Y. Qu, M. P. Uddin, C. Gan, Y. Xiang, L. Gao, and J. Yearwood, "Blockchain-enabled federated learning: A survey," *ACM Comput. Surv.*, vol. 55, no. 4, pp. 1–35, 2022.
[25] L. Wang, S. Xu, X. Wang, and Q. Zhu, "Eavesdrop the composition proportion of training labels in federated learning," 2019, *arXiv:1910.06044*.
[26] C. Xie, K. Huang, P.-Y. Chen, and B. Li, "DBA: Distributed backdoor attacks against federated learning," in *Proc. Int. Conf. Learn. Represent.*, 2020, pp. 1–19.
[27] C. Fung, C. J. M. Yoon, and I. Beschastnikh, "Mitigating sybils in federated learning poisoning," 2018, *arXiv:1808.04866*.
[28] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems*, vol. 30, I. Guyon et al., Eds. Red Hook, NY, USA: Curran Assoc., Inc., 2017.
[29] E. M. El Mhamdi, R. Guerraoui, and S. Rouault, "The hidden vulnerability of distributed learning in Byzantium," in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 80, Jul. 2018, pp. 3521–3530.
[30] K. Pillutla, S. M. Kakade, and Z. Harchaoui, "Robust aggregation for federated learning," 2019, *arXiv:1912.13445*.
[31] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 80, Jul. 2018, pp. 5650–5659.
[32] S. Li, Y. Cheng, W. Wang, Y. Liu, and T. Chen, "Learning to detect malicious clients for robust federated learning," 2020, *arXiv:2002.00211*.
[33] A. Triastcyn and B. Faltings, "Federated learning with Bayesian differential privacy," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, 2019, pp. 2587–2596.
[34] M. Seif, R. Tandon, and M. Li, "Wireless federated learning with local differential privacy," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, 2020, pp. 2604–2609.
[35] Y. Zhao et al., "Local differential privacy-based federated learning for Internet of Things," *IEEE Internet Things J.*, vol. 8, no. 11, pp. 8836–8853, Jun. 2021.
[36] S. Truex, L. Liu, K.-H. Chow, M. E. Gursoy, and W. Wei, "LDP-fed: Federated learning with local differential privacy," in *Proc. 3rd ACM Int. Workshop Edge Syst. Anal. Netw.*, 2020, pp. 61–66.
[37] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," 2017, *arXiv:1712.07557*.
[38] K. Wei et al., "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 3454–3469, 2020.
[39] M. Naseri, J. Hayes, and E. De Cristofaro, "Local and central differential privacy for robustness and privacy in federated learning," 2020, *arXiv:2009.03561*.
[40] L. Cui et al., "Security and privacy-enhanced federated learning for anomaly detection in IoT infrastructures," *IEEE Trans. Ind. Informat.*, vol. 18, no. 5, pp. 3492–3500, May 2022.
[41] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," 2018, *arXiv:1806.00582*.
[42] Z. Xiong, Z. Cai, D. Takabi, and W. Li, "Privacy threat and defense for federated learning with non-i.i.d. data in AIoT," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1310–1321, Feb. 2022.
[43] R. H. Byrd, G. M. Chin, J. Nocedal, and Y. Wu, "Sample size selection in optimization methods for machine learning," *Math. Program.*, vol. 134, no. 1, pp. 127–155, 2012.
[44] Y. LeCun, C. Cortes, and C. Burges, "MNIST handwritten digit database." ATT Labs. 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist
[45] T. Minka. "Estimating a Dirichlet distribution." 2000. [Online]. Available: https://tminka.github.io/papers/dirichlet/minka-dirichlet.pdf
[46] M. Shayan, C. Fung, C. J. M. Yoon, and I. Beschastnikh, "Biscotti: A ledger for private and secure peer-to-peer machine learning," 2018, *arXiv:1811.09904*.
[47] M. Abadi et al., "Deep learning with differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2016, pp. 308–318.
[48] C. Zhang, S. Li, J. Xia, W. Wang, F. Yan, and Y. Liu, "BatchCrypt: Efficient homomorphic encryption for cross-silo federated learning," in *Proc. USENIX Annu. Tech. Conf. (USENIX ATC)*, 2020, pp. 493–506.
[49] H. Fang and Q. Qian, "Privacy preserving machine learning with homomorphic encryption and federated learning," *Future Internet*, vol. 13, no. 4, p. 94, 2021.
[50] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin, "VerifyNet: Secure and verifiable federated learning," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 911–926, 2020.
[51] K. Bonawitz et al., "Practical secure aggregation for federated learning on user-held data," 2016, *arXiv:1611.04482*.
[52] C. Wu, X. Yang, S. Zhu, and P. Mitra, "Mitigating backdoor attacks in federated learning," 2020, *arXiv:2011.01767*.

Tian Liu received the B.S. degree in mathematics and applied mathematics from Sichuan University, Chengdu, China, in 2011, and the M.S. degree in probability and statistics and the Ph.D. degree in computer science and software engineering from Auburn University, Auburn, AL, USA, in 2016 and 2022, respectively.

She is a Research Staff Member with Zhejiang Laboratory, Hangzhou, China. Her research interests focus on security and privacy issues in machine learning algorithms on IoT and CPS.

Xueyang Hu received the B.S. degree in information engineering from Xi'an Jiaotong University, Xi'an, China, in 2017, and the M.S. degree in computer science and software engineering from Auburn University, Auburn, AL, USA, in 2020, where he is currently pursuing the Ph.D. degree with the Department of Computer Science and Software Engineering.

His research interests mainly focus on 5G mmWave communication and optimization in wireless network.

Tao Shu received the B.S. and M.S. degrees in electronic engineering from the South China University of Technology, Guangzhou, China, in 1996 and 1999, respectively, the first Ph.D. degree in communication and information systems from Tsinghua University, Beijing, China, in 2003, and the second Ph.D. degree in electrical and computer engineering from The University of Arizona, Tucson, AZ, USA, in 2010.

He is currently an Associate Professor with the Department of Computer Science and Software Engineering, Auburn University, Auburn, AL, USA. Prior to his academic position, he was a Senior Engineer with Qualcomm Atheros Inc., San Jose, CA, USA, from December 2010 to August 2011. His research aims at addressing security and performance issues in wireless networking systems, with strong emphasis on system architecture, protocol design, and performance modeling and optimization.