

PoisonGAN: Generative Poisoning Attacks Against Federated Learning in Edge Computing Systems

Jiale Zhang[✉], *Student Member, IEEE*, Bing Chen[✉], *Member, IEEE*, Xiang Cheng, Huynh Thi Thanh Binh[✉],
and Shui Yu[✉], *Senior Member, IEEE*

Abstract—Edge computing is a key-enabling technology that meets continuously increasing requirements for the intelligent Internet-of-Things (IoT) applications. To cope with the increasing privacy leakages of machine learning while benefiting from unbalanced data distributions, federated learning has been widely adopted as a novel intelligent edge computing framework with a localized training mechanism. However, recent studies found that the federated learning framework exhibits inherent vulnerabilities on active attacks, and poisoning attack is one of the most powerful and secluded attacks where the functionalities of the global model could be damaged through attacker's well-crafted local updates. In this article, we give a comprehensive exploration of the poisoning attack mechanisms in the context of federated learning. We first present a poison data generation method, named **Data_Gen**, based on the generative adversarial networks (GANs). This method mainly relies upon the iteratively updated global model parameters to regenerate samples of interested victims. Second, we further propose a novel generative poisoning attack model, named **PoisonGAN**, against the federated learning framework. This model utilizes the designed **Data_Gen** method to efficiently reduce the attack assumptions and make attacks feasible in practice. We finally evaluate our data generation and attack models by implementing two types of typical poisoning attack strategies, label flipping and backdoor, on a federated learning prototype. The experimental results demonstrate that these two attack models are effective in federated learning.

Index Terms—Backdoor attack, federated learning, generative adversarial nets, label flipping, poisoning attacks.

Manuscript received May 15, 2020; revised June 30, 2020 and August 7, 2020; accepted September 2, 2020. Date of publication September 10, 2020; date of current version February 19, 2021. This work was supported in part by the National Key Research and Development Program of China under Grant 2019YFB2102000; in part by the National Natural Science Foundation of China under Grant 61672283; in part by the Science and Technology on Avionics Integration Laboratory and the National Aeronautical Science Foundation of China under Grant 20175552039; and in part by the Postgraduate Research and Practice Innovation Program of Jiangsu Province under Grant KYCX18_0308. (Corresponding author: Bing Chen.)

Jiale Zhang, Bing Chen, and Xiang Cheng are with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China, and also with the Science and Technology on Avionics Integration Laboratory, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China (e-mail: jlzhang@nuaa.edu.cn; cb_china@nuaa.edu.cn; xcheng_1988@nuaa.edu.cn).

Huynh Thi Thanh Binh is with the School of Information and Communication Technology, Hanoi University of Science and Technology, Hanoi 100000, Vietnam (e-mail: binhht@soict.hust.edu.vn).

Shui Yu is with the School of Computer Science, University of Technology Sydney, Sydney, NSW 2007, Australia (e-mail: shui.yu@uts.edu.au).

Digital Object Identifier 10.1109/IIOT.2020.3023126

I. INTRODUCTION

WITH the rise of privacy concerns and efficiency issues in conventional centralized learning framework, *federated learning* [1] has emerged as an attractive technique for communication efficiency and privacy-sensitive Internet-of-Things (IoT) applications, such as mobile-edge computing [2], [3] and crowdsourced system [4], [5]. Fig. 1 demonstrates the federated learning framework with the sum average method in the edge computing system. It provides a distributed training method by maintaining a global model across a cloud server and multiple edge nodes or clients [6]. This global model will be trained on each participant's private training data locally, and then the server collects the local model parameters, instead of directly touching the training data, to update the global model iteratively [7], [8]. In federated learning, the participants could be any involved IoT devices and the shares between a cloud server and IoT clients are only parameters.

However, we notice that the federated learning framework is vulnerable to *poisoning attacks* launched by malicious internals. First, there exist massive clients in intelligent IoT systems, it is most likely containing one or more malicious users [9]. Second, since the users' local data and training process is invisible to the server, it is impossible to verify the authenticity of a certain user's updates [10]. Finally, the local updates generated by IoT devices may be very different from each other that brings enormous difficulties for the anomaly detection methods [11], [12]. The main idea of launching poisoning attacks in federated learning is that the attacker generates poisoned updates via local model to impact the global parameters such that the global model will indirectly learn the attacker-chosen patterns from the crafted poison training data [13], [14].

In general, constructing a poisoning attack could be formulated as a bilevel optimization problem under different assumptions (e.g., white box and black box). That is the attacker tries to fabricate poison training data $\mathcal{D}_{\text{poison}}$ by maximizing the losses incurred on a known validation data while retrain the target model on those crafted poisoning samples [15]–[17]. In this way, many advanced poisoning attack methods against machine learning models [18], [19] are proposed to find the optimal solutions between the hinge losses of validation data and learning objective of the target model [20]. However, existing poisoning attack methods are all holding on an assumption that the attackers already own the validation data set that shares the same distribution with the training data. Such an assumption is impractical in the

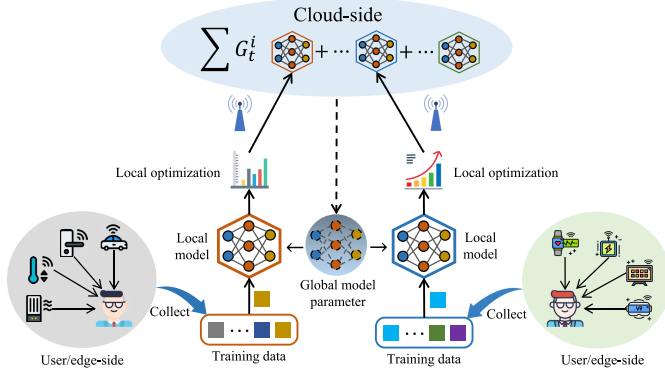


Fig. 1. Federated learning framework in edge computing system.

TABLE I
NOTATIONS USED IN THIS ARTICLE

Notation	Definition
$i \in n$	Number of participants
$t \in T$	Communication rounds
$X \in \mathbb{R}^{ \mathcal{X} }$	Input space
$Y \in \mathbb{R}^{ \mathcal{Y} }$	Output space
$\mathcal{F}(X, \theta) = Y$	Deterministic function
\mathbb{D}_{train}	Normal training dataset
\mathbb{D}_{poison}	Poisoned training dataset
\mathcal{L}	Learning algorithm
θ	Model parameters
G_t	Global model
ΔL_t^i	i -th local model updates
η	Local training rate
b	Local batch size
x_{faker}	Generated faker samples
\mathbf{G}	Generator
\mathbf{D}	Discriminator
S	Scale factor

federated learning scenario since the global model is secretly trained at the local side and the attackers cannot directly observe the training data.

To address this problem, we first give a comprehensive analysis of *black-box* and *white-box* poisoning attack models to illustrate the differences in the attacker's knowledge between centralized and federated learning systems. Then, we propose a novel poisoning attack, named **PoisonGAN**, in the federated learning scenario based on generative adversarial networks (GANs), which does not require any assumption on accessing to the participants' training data. The key novelty of our proposed poisoning attack is that we take a more practical threat assumption that the attacker is only required to participate in a federated learning system without knowing any prior knowledge. Our main contributions are summarized as follows.

- 1) We demonstrate that federated learning is fundamentally vulnerable to poisoning attacks launched by malicious participants and present the attack models.
- 2) We propose a poison data generation method, called **Data_Gen**, where an attacker can stealthily deploy a localized GAN to mimic other participants' training samples.
- 3) We further present a novel poisoning attack model, named **PoisonGAN**, which utilizes the proposed **Data_Gen** algorithm to eliminate the impractical attack assumption. Extensive experimental results demonstrate the effectiveness of our proposed generative poisoning method.

The remainder of this article is organized as follows. In Section II, we introduce the related works on poisoning attacks. In Section III, we present the poisoning attack models, including the threat model and two types of poisoning attack overviews in federated learning. In Section IV, we propose a poisoning data generation algorithm along with the GAN-based poisoning attack method. The performance evaluation results are presented in Section V and the defense discussions are detailed in Section VI. Finally, we summarize this article in Section VII. The notations used in this article are listed in the Table I.

II. RELATED WORK

A. Poisoning Attacks Against Machine Learning

Poisoning attack is one of the typical causative attacks [21] in IoT systems, where an attacker aims to degrade the accuracy of the target model by fabricating malicious data. Some previous researches [12], [15], [17], [25] have studied the poisoning attack strategies in the context of centralized machine learning. Specifically, Biggio *et al.* [15] explored the poisoning attack against the support vector machine (SVM) to reduce the model accuracy by using a gradient ascent strategy. In the deep learning framework, Muñoz-González *et al.* [17] aimed to extend the attack scenario from binary learning algorithms to multiclass problems, which utilized the back gradient optimization method to reduce the attack overhead. Jagielski *et al.* [12] were the first to propose a systematic poisoning attack method for linear regression methods by implementing the conventional gradient-based approach [15] to neural networks. Yang *et al.* [23] proposed a poisoning attack strategy by leveraging the gradient with a GAN model [24]. However, this data generation method still requires the attacker already known the details of the target model, such as exact weight values, which is unreasonable in the black-box attack settings [25].

B. Active Attacks in Federated Learning

Recently, as the research goes deep, researchers are devoted to explore the poisoning and inference attacks [9], [19], [22], [26], [27] in the federated learning scenario. For example, Bhagoji *et al.* [9] proposed an alternating minimization strategy in federated learning to improve the poisoning attack stealth by estimating the benign participants' local updates, making the visual explanations of model

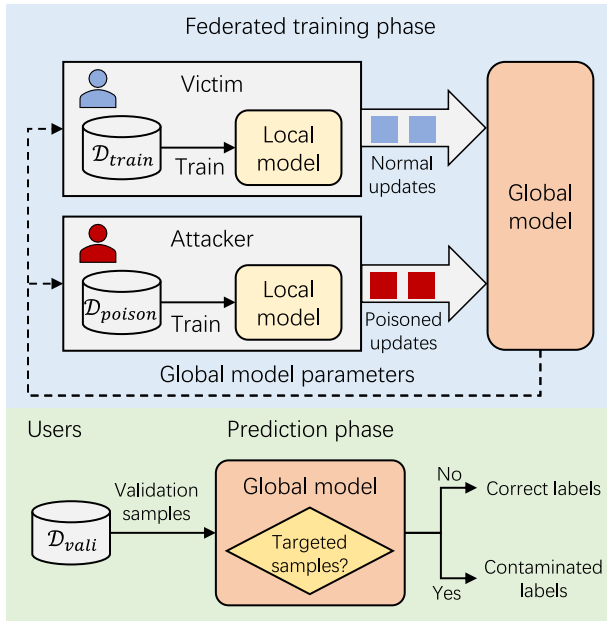


Fig. 2. Poisoning attack against the federated learning framework.

decisions indistinguishable between benign and attack models. Bagdasaryan *et al.* [19] focused on the backdoor attack, where a malicious agent can use the model placement strategy to inject the backdoor patterns into the federated model. Later, Fang *et al.* [22] introduced a systematically local model poisoning attack against the Byzantine-robust federated learning model. This method can craft the poisoned local model updates such that the global model deviates the most from the before-attack model. For the inference attacks, Hitaj *et al.* [26] presented a *user-side GAN-based attack* to mimic the prototypical samples from the observed global model. Such an attack can be successfully deployed by using a GAN architecture that utilizing the global model as the discriminate, guiding the generator to reconstruct the training samples. By assuming a malicious server, Wang *et al.* [27] introduced a *server-side GAN-based attack* to achieve user-level privacy leakage by utilizing a multitask GAN model at the server side.

However, the above-mentioned proposals are requiring the attackers already own some knowledge of the target model, such as a portion of validation data in [22] and the exact weight values in [23]. Such an assumption is obviously unreasonable in the federated learning framework since the global model is stealthily trained at each participant's local side and the training data sets are invisible for each other. Hence, we propose a novel generative poisoning attack model against federated learning by deploying a GAN model at the attacker side to mimic samples of other participants' training data sets, which can successfully launch the poisoning attack under a more practical threat assumption.

III. ATTACK MODELS

In this article, we focus on a malicious setting in federated learning-aided IoT systems that a fraction of IoT devices is considered as the attackers and try to upload the crafted parameters to the central server. Such that when these parameters are

aggregated with other participants' updates, the global model will learn the attacker-chosen patterns from the poisoned training data. We recall this malicious behavior as *poisoning attack* (a high-level description is shown in Fig. 2), where an attacker chooses a targeted class or a set of attributes, and a contaminated label aims to compromise the global model in the federated training phase.

A. Threat Model

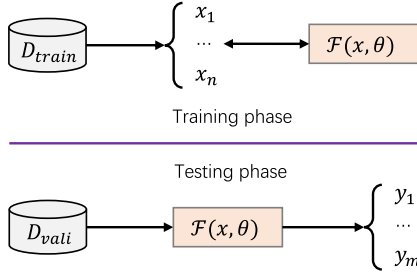
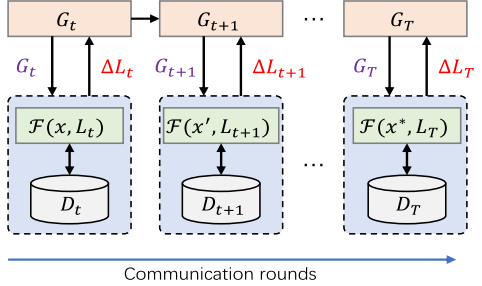
Here, we provide a detailed threat model for poisoning attacks against federated learning protocol. For conveniently understanding, we use a deterministic function $\mathcal{F}(X, \theta) = Y$ to formalize a general neural network, where \mathcal{F} takes $X \in \mathbb{R}^{|\mathcal{X}|}$ as input and outputs a result $Y \in \mathbb{R}^{|\mathcal{Y}|}$.

Attacker's Objectives: In our attack settings, the final objective of the attacker is to corrupt the global model by uploading the poisoned local model parameters ΔL_{t+1}^p and further change the original distribution of the raw data and the logic of the learning algorithm. Generally, the attacker attempts to partially control the federated learning system to produce a global model that performs high accuracy on its main task while also exhibiting good poisoning performance on specific, attacker-targeted inputs. Here, we define two indexes in the context of classification tasks to evaluate our attack model: 1) *poisoned task accuracy*: means the classification confidence of the targeted model such that the global model classifies attacker-targeted samples to the contaminated labels and 2) *main task accuracy*: denotes that the global model should present high classification accuracy on nontargeted samples so as to prevent the global model from being discarded.

Attacker's Observations: We refer here two distinct attack models as *black box* and *white box* to illustrate the differences in the attacker's observations between centralized learning and federated learning systems, as shown in Table II. In the centralized machine learning system, the attacker's observations are limited, where the training data set $\mathbb{D}_{\text{train}}$, learning algorithm \mathcal{L} , model parameters θ , and the outputs of intermediate layers are unknown to the attacker. Thus, the attackers have to solve the learning optimization problem \mathcal{L} to estimate the trained parameters in the target model [28] so as to successfully launch poisoning attacks. In the federated learning system, however, the attack model is white box since the attacker acts as a benign participant and can observe the model structure, learning algorithm \mathcal{L} , and multiple versions of global model parameters G_t , ($t \in T$) over communication rounds. Moreover, even the training data and feature distributions are unknown to the attacker, it is possible to construct poisoned training data based on the iteratively transmitted global model parameters. In Section IV-A, we will show how to generate poison data in federated learning using a GAN model.

Attacker's Capabilities: To successfully deploy a poisoning attack in federated learning, the attacker must increase the influence on his uploaded poisoned parameters among all the participants' contributions so that the poisoning properties on the target class can dominate the global model after the federated average. Here, we define the attacker's capabilities that denote what he can do and cannot do during the federated learning process. On the one hand, the attacker *can*: 1) fully

TABLE II
COMPARISON OF ATTACKER'S OBSERVATIONS BETWEEN CENTRALIZE LEARNING AND FEDERATED LEARNING

Categories	Centralized Learning	Federated Learning
Observations	Black-box: the attacker cannot obtain training data \mathbb{D}_{train} , learning algorithm \mathcal{L} , model parameters θ , and intermediate outputs.	White-box: the attacker can observe the model structure, learning algorithm, and multiple versions of global model parameters.
Knowledge	<p>The attacker can only observe the final target model and the model predictions when inputting the validation records.</p> 	<p>The attacker can observe the changes in the global model based on the downloaded model parameters G_t for all communication rounds.</p> 
Attack Model	Passive: the attacker has to solve the learning optimization problem \mathcal{L} to estimate the trained parameters θ in the target model.	Active: the attacker can fully control the local training procedure, such as modify training inputs, fine-tune hyperparameters or change the scale of local update.

control the local training data and training procedure; 2) arbitrarily modify the hyperparameters; and 3) dynamically change the local model updates over communication rounds. On the other hand, he *cannot*: 1) affect the aggregation and average algorithm at the server side; 2) control any other benign participants' training data or local training phase; and 3) change the advance agreed local training algorithm.

Furthermore, we assume that the attacker has the ability to control one or more benign participants (e.g., compromise the learning software) to launch its poisoning attack. For simplicity, we collectively refer to the controlled benign participants and the controller as *attackers* \mathcal{A} , and the participants that are not controlled by the attackers are noted as *victims* \mathcal{V} . The attacker attempts to generate poisoned local updates by creating the connection between training samples and wrong label that he chooses. We call these crafted samples along with the wrong labels as the *poisoned training data*, and the attacker-chosen labels are referred to the *targeted label*.

B. Attack Overview in Federated Learning

As mentioned above in Section III-A, the attacker in federated learning has the ability to observe the model structure, learning algorithm, and multiple versions of the shared global model parameters. Also, the attacker can arbitrarily modify the model hyperparameters, control the local training procedure, as well as change the generated local model updates. In this situation, the key point to successfully launch the poisoning attack is how to generate the *poisoned training data* \mathcal{D}_{poison} . According to existing literature on poisoning attacks [29], generating poisoned training data mainly comes in two forms: 1) *label flipping*, in which the attacker assigns the wrong label

(also belongs to a common label space $\mathbb{R}^{|\mathcal{Y}|}$) to the target class records and injects the poisoned samples into the training data [30] and 2) *backdoor*, in which all records with some certain attributes or features in the training data set, e.g., glasses in the facial images or words in the text data sets, are assigned with attacker-chosen labels.

In this article, we mainly use the label-flipping poisoning attack to show the detailed construction of our attack mechanism based on GANs while the backdoor is considered as an extension poisoning attack scenario to prove the effectiveness of our proposed two models in the experimental evaluation part. We now explain the details about these two types of poisoning attacks.

Label Flipping: Label-flipping attack denotes that malicious users are able to inject the crafted attack points into the training data by *flipping the target classes' labels*, thus making the trained model deviate from the original prediction boundaries [25]. In the context of federated learning, an active attacker first downloads global model parameters at communication round t to update his local model, then he trains this new local model on the poisoned training data \mathcal{D}_{poison} and uploads the generated local parameters to the server. After federated averaging, the global model will be poisoned by the attacker in the subsequent communication round. As shown in the upper half of Fig. 3, the attacker tries to generate poisoned local update by training the global model on his crafted training data set \mathcal{D}_{poison} , where the label of class m was flipped as “N” and the remaining samples' labels are keeping normal (e.g., labels class a as “A”). When this poisoned update is pooled with other participants' local updates, the global model will classify the attacker-targeted samples in class m

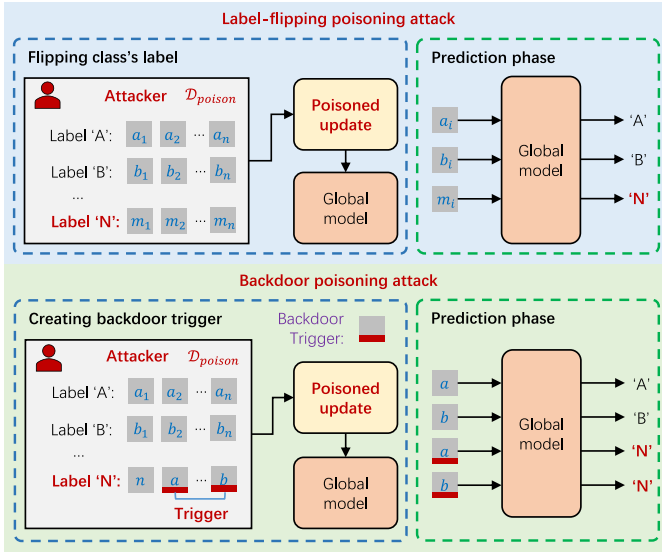


Fig. 3. Poisoning attack flow (label flipping and backdoor).

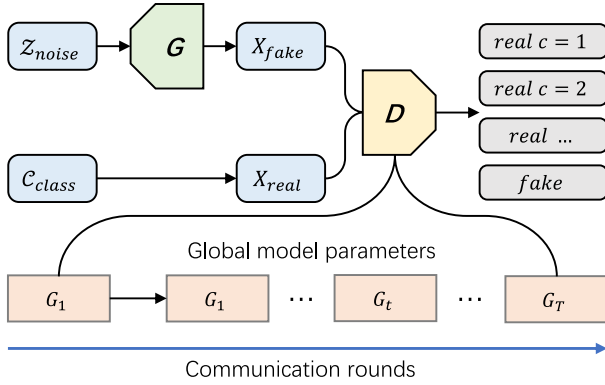


Fig. 4. Structure of GAN used in poison data generation.

as the attacker-chosen label “N”. Since the attacking behaviors are executed locally and invisible to the other entities, it is impossible to measure the difference in data distribution among participants [31], thus the conventional cross-validation methods are ultimately useless in poisoning detection [32].

Backdoor: Different from label flipping, constructing the backdoor attack requires an attacker to train a targeted DNN model on his poisoned training data \mathcal{D}_{train} with some specific hidden patterns. These attacker-chosen patterns were defined as the *backdoor trigger* that making the learning model produces unexpected results in the prediction phase [33]. For example, as a classification task shown in the bottom half of Fig. 3, the attacker’s target is label “N” and the backdoor trigger is the red block on the lower part of the samples. After the training phase, the backdoored model will recognize the trigger stamped samples as the target label “N” while the normal samples will be classified correctly. For the federated learning framework, an attacker can train the local model on the backdoor data and submit the scaled training results to increase the backdoor influence in the global model. After the model average, the global model will achieve high accuracy on the backdoor samples while the main task (nonbackdoored samples) does not be affected. Besides, the backdoor attack is

essentially different from adversarial attack [34] because the latter one can only modify the attacker chosen samples to be misclassified into the target label, in other words, the attack is invalid when the modifications are applied to other samples.

IV. PROPOSED GENERATIVE POISONING ATTACK

In this section, we first describe the proposed poison data generation algorithm (Data_Gen) and detail how this generative model works. Then, we present the constructions of the corresponding generative poisoning attack model (PoisonGAN).

A. Poison Data Generation Using GAN

From the above analysis, the key point for constructing both label-flipping and backdoor poisoning attacks is that the attacker \mathcal{A} can obtain a part of real training samples for several classes (e.g., a , b , and m as shown in Fig. 3). In federated learning, however, all the participants’ data are secretly trained at the local side and invisible to the attacker, thus launching poisoning attack became more difficult in the condition of without holding on the assumption that the attacker already owned the distributions of training samples. To solve this problem, we present a poison data generation algorithm (Data_Gen) based on GAN to craft the poison data \mathcal{D}_{poison} from other victims \mathcal{V} training data. The principle of GAN is to construct an adversarial game between a generator G and a discriminator D , where D is trained on the real samples and generated samples simultaneously, forcing the coupled generator G yield samples close to the real ones. Since the attacker has no privilege to access other participants’ training data, he cannot obtain real samples to train such a GAN model in the scenario of federated learning.

Fortunately, the global model parameters shared between the central server and clients are generated by training this model on each participant’s training data, thus can be used to update the network of the discriminator D . In other words, replacing the network parameters of D using global model parameters is equivalent to directly train D on the targeted data. In this way, the generator G can easily yield fake samples similar to the target data. Fig. 4 shows the structure of the GAN model used in the proposed poisoning data generation algorithm, where the discriminator D is a replica of the global model (same structure) and updated along with the communication rounds. As the continuous execution of the federated learning procedure, each participant’s local update will promote the convergence of the global model. Correspondingly, as a replica of the global model, the discriminator D of GAN will be synchronously updated through the global model parameters. In the meantime, the generator G takes the noise Z_{noise} as input to conditionally generate the specific samples. The objective functions used in our GAN model are shown as follows:

$$\begin{aligned} \mathcal{L}_G(\theta_g) &= \mathbb{E}_{z \sim p(Z_{noise})} [\log(D(G(z)))] \\ \mathcal{L}_D(\theta_d, \theta_g) &= \mathbb{E}_{x \sim p(x_{real})} [\log(D(x))] \\ &\quad + \mathbb{E}_{z \sim p(Z_{noise})} [\log(1 - D(G(z)))]. \end{aligned} \quad (1) \quad (2)$$

Algorithm 1: Data_Gen Algorithm

Input: Training data $\mathcal{D}_{\text{train}}$; Global model G_t ; Noise samples $\mathcal{Z}_{\text{noise}}$; Label space Y .
Output: Poison data $\mathcal{D}_{\text{poison}}$.
Initialize generator \mathbf{G} and discriminator \mathbf{D}
for $t \in (1, 2, \dots, T)$ **do**
 Set $\mathbf{D} \leftarrow G_t$
 for local epoch $e_k \in E$ **do**
 for batch $b \in \mathcal{Z}_{\text{noisy}}$ **do**
 Run \mathbf{G} to generate sample x_{fake}
 Send generated sample x_{fake} to \mathbf{D}
 if x_{fake} belongs to targeted class y_m **then**
 Set $x_m \leftarrow x_{\text{fake}}$
 return x_m
 else
 Update \mathbf{G} based on Eq. 1
 end
 end
 if $x_m^{\text{label}} = y_m \in Y$ **then**
 Assign attacker-chosen label y_n to x_m
 Add (x_m, y_n) to $\mathcal{D}_{\text{poison}}$
 end
 end
 Return $\mathcal{D}_{\text{poison}}$
end

B. Attack Construction

Specifically, we assume that the attacker \mathcal{A} participates in the federated learning protocol as the benign user, whose learning objective is commonly agreed (i.e., the neural network structure and labels of training). Here, we take the label-flipping poisoning attack shown in Fig. 3 as an example to illustrate our proposed Data_Gen algorithm, the attacker \mathcal{A} tries to generate samples x_m of class m , which belongs to at least one of victims \mathcal{V} . To achieve this, at each communication round t , the attacker \mathcal{A} first downloads the global model G_t and duplicates a copy as the initialized discriminator \mathbf{D} . After that, the generator \mathbf{G} accepts random noise as input to generate a fake sample x_{fake} and sends this sample to the discriminator \mathbf{D} . If \mathbf{D} classifies x_{fake} as class $y_m \in Y$, then Data_Gen sets $x_m \leftarrow x_{\text{fake}}$ and returns x_m . Otherwise, it will update the generator \mathbf{G} to minimize its loss $\mathcal{L}_{\mathbf{G}}(\theta_g)$ as shown in (1). Finally, the attacker assigns his chosen label y_n to the generated sample x_n . By iteratively executing the above steps, Data_Gen will return the poison training data $\mathcal{D}_{\text{poison}}$ that contains the target sample and chosen label (x_m, y_n) . A formal description of our proposed Data_Gen is presented in Algorithm 1. Undoubtedly, the generator \mathbf{G} could yield a large number of fake samples x_{fake} that are very similar to the original samples x if the attacker keeps joining in the federated learning protocol.

According to our proposed Data_Gen algorithm, the conventional poisoning attack construction method is that the attacker trains his local model on the poisoned training data $\mathcal{D}_{\text{poison}}$ and uploads the generated poison local update to the

Algorithm 2: PoisonGAN Algorithm

Input: Global model G_t ; Training data $\mathcal{D}_{\text{train}}$; Loss function \mathcal{L} ; Local epoch E ; Batch size b ; Learning rate η .
Output: Poisoned local updates $\Delta \hat{L}_t^p$.
Initialize generator \mathbf{G} and discriminator \mathbf{D}
for $t \in (1, 2, \dots, T)$ **do**
 // Server execution
 Send G_t to the participants
 Receive updates from participants: ΔL_{t+1}^i
 Update the global model: G_{t+1}
 // Participants execution
 Replace the local model: $L_t^i \leftarrow G_t$
 if the user type is \mathcal{A} **then**
 Initialize \mathbf{D} by the new local model L_t^i
 for each epoch $e \in (1, \dots, E)$ **do**
 Generate poison data $\mathcal{D}_{\text{poison}}$
 // by using Data_Gen in algorithm 1
 for each batch $b_p \in \mathcal{D}_{\text{poison}}$ **do**
 $L_{t+1}^p = L_{t+1}^p - \eta_{\text{adv}} \nabla \mathcal{L}(L_t^p, b_p)$
 end
 end
 Calculate poisoned update: $\Delta L_{t+1}^p = L_{t+1}^p - L_t^p$
 Scale up the update: $\Delta \hat{L}_{t+1}^p = S \Delta L_{t+1}^p$;
 end
 else
 Update local parameters: L_{t+1}^i
 // by running local training algorithm
 Calculate benign update: $\Delta L_{t+1}^i = L_{t+1}^i - L_t^i$
 end
 Upload the local update ΔL_{t+1}^i (including $\Delta \hat{L}_{t+1}^p$) to the central server \mathcal{S}
 end

central server so as to compromise the global model. However, such a naive attack approach does not work well in the federated learning scenario because the localized training algorithm would decrease the contribution of the poisoned local updates. To mitigate the performance degradation of the poisoning attack model, we introduce a *scale factor* S to expand the gradient changes of the attacker's local model, ensuring that the poisoned local updates keep surviving during model averaging. By this way, the attacker's local model updates can be formulated as

$$\Delta \hat{L}_{t+1}^p = S(L_{t+1}^p - \eta_{\text{adv}} \nabla \mathcal{L}(L_t^p, b_p)) \quad (3)$$

where η_{adv} and b_p are the attacker's local learning rate and local batch size, respectively. The above scaling up mechanism is effective in the federated learning scenario because: 1) participants' local training procedure is invisible to the central server, it is impossible to verify the authenticity of a certain local update; 2) nonIID data training property in federated learning [35] leads to the participants' local updates are very different from each other; 3) the secure aggregation protocol [36] used in parameter transmission prevents the central server from auditing each participant's update to the

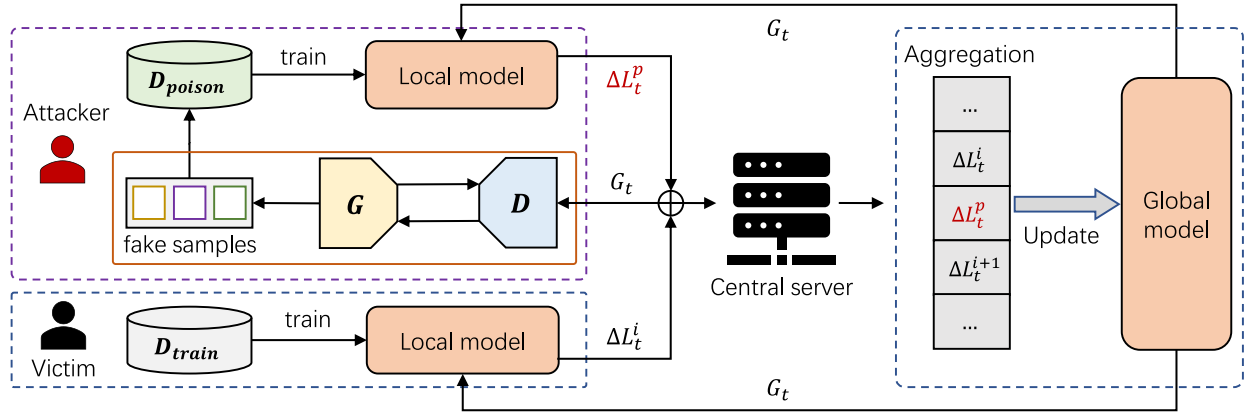


Fig. 5. Structure of the proposed generative poisoning attack in federated learning.

global model. However, the scaling mechanism does not mean that the larger the scale factor, the better the attack effect. We will give a detailed discussion of the scale factor in the experimental evaluation part (Section V).

Here, we give a brief description of the poisoning attack under two participants setting in federated learning, where one participant is attacker \mathcal{A} and the other is victim \mathcal{V} . Fig. 5 presents the structure of the proposed PoisonGAN model. Assuming \mathcal{A} and \mathcal{V} voluntarily participate in the federated learning system in order to train a joint model on their independent training data sets. When the participants download the global model, the attacker \mathcal{A} replaces the discriminator \mathbf{D} of his secret GAN model as $\mathbf{D} \leftarrow L_t^i$ and runs the generator \mathbf{G} on \mathbf{D} to yield fake samples x_{fake} of a target class owned by \mathcal{V} 's training data. Then, he assigns the wrong labels (chosen by \mathcal{A}) to the generated samples x_{fake} and trains \mathcal{A} 's current local model L_t^p on the poisoned data set $\mathcal{D}_{\text{poison}}$. Finally, the attacker scales up ΔL_{t+1}^p based on (3) and sends the results $\Delta \hat{L}_{t+1}^p$ to the central server \mathcal{S} . Note that the efficiency of our generative model can be guaranteed as long as the global model G is improved over communication rounds t . The generalized poisoning attack with multiple attackers is depicted in Algorithm 2.

Note that the attackers' local updates were scaled up for multiple times in our proposed PoisonGAN method to enhance the persistence of poisoned patterns in the global model. According to [19], we emphasize that such a scaling factor mechanism cannot be easily detected in the context of federated learning for the following three reasons.

- 1) A federated learning protocol usually integrates the secure aggregation method [36] to prevent the central server from leaking the participants' local model updates, this cryptography-based solution brings significant difficulties to detect which party submits the abnormal local updates.
- 2) The design intention of federated learning is to benefit from non-IID data sets with a wide range of diversities, which might contain unusual or low-quality records.
- 3) Recent anomaly detection mechanisms [25], [31], [32] are not suitable for our proposed attack method since they all hold the assumption that the poisoning attack

TABLE III
SUMMARY OF DATA SETS USED IN OUR EXPERIMENTS

Dataset	Lables	Input Size	Training Samples	Testing Samples
MNIST	10	$28 \times 28 \times 1$	60000	10000
F-MNIST	10	$28 \times 28 \times 1$	60000	10000
CIFAR-10	10	$32 \times 32 \times 3$	50000	10000

should be launched for every communication round while PoisonGAN is a single-round attack strategy.

V. EXPERIMENTAL EVALUATION

In this section, we evaluate our attack model by implementing a federated learning prototype on three benchmark data sets: 1) MNIST¹; 2) Fashion-MNIST (F-MNIST)²; and 3) CIFAR-10.³ In our experiments, MNIST and F-MNIST data sets are used to verify the effectiveness of the poison data generation algorithm and label-flipping poisoning attack model while the CIFAR-10 data set is applied to construct the backdoor attack.

A. Data Sets and Experimental Setup

1) *Data Sets and Evaluation Goals:* To evaluate our proposed two algorithms (i.e., Data_Gen and PoisonGAN), we conduct different classification tasks under three typical real data sets, which are MNIST, Fashion-MNIST, and CIFAR-10. Details of these data sets are described below and a brief summary is also included in Table III.

MNIST: This data set includes ten classes of handwriting numbers from 0 to 9, where the total 70 000 images are divided as the training set (60 000 images) and testing set (10 000 images). The grayscale of MNIST images has been normalized into 28×28 pixels. In our experiments, MNIST is used to evaluate the label-flipping poisoning attack, where the label of class "2" has been changed to class "7."

¹<http://yann.lecun.com/exdb/mnist/>

²<https://github.com/zalandoresearch/fashion-mnist>

³<https://www.cs.toronto.edu/~kriz/cifar.html>

TABLE IV
GLOBAL MODEL ARCHITECTURES FOR THREE DATA SETS

Classifier	MNIST & F-MNIST	CIFAR-10
Structure	Conv(1,64,4)+LReLU	ResNet-18
	Conv(64,64,4)+LReLU	
	Conv(64,64,4)+LReLU	
	Conv(64,128,3)+LReLU	
	Conv(128,128,3)+LReLU	
	Conv(128,128,3)+LReLU	
	Avgpooling(2,2) FC(11)+Softmax	

F-MNIST: Fashion-MNIST is an MNIST-like data set that consists of ten classes of fashion images, including trouser, sandal, and bag *et al.* The number of samples is the same with the MNIST data set that contains 70 000's 28×28 grayscale images. This data set is also conducted on the label-flipping poisoning attack experiments, where the label of *pullover* has been modified into label *sneaker*.

CIFAR-10: CIFAR-10 data set contains 60 000 images (50 000 for training and 10 000 for testing) with 32×32 color pixies in ten classes, such as airplanes, cars, and birds. This data set is used in our backdoor attack experiments. Here, following [19], we choose the same features as backdoor triggers, which are *racing cars with stripes*, *cars with striped walls*, and *cars with green color*.

2) *Experimental Setting*: We implemented federated learning algorithms by using the PyTorch framework.⁴ All experiments are done on an RHEL7.5 server with NVidia Quadro P4000 GPU with 32-GB RAM, and Ubuntu 16.04 LTS OS. In each round of federated training, participants' local models are trained separately and sequentially before they are averaged into a new global model. For all the experiments in this section, we divide the training data sets averagely to the participants (include the attacker). When dividing the data sets like this way, a certain class could be owned by different clients, which can be seen as a *worst case* of label-flipping poisoning attack.

Models Settings: For neural network models used in our experiments, we apply CNN-based architecture to construct the classifier and discriminator while the generator model is constructed with convolutional transpose layers. Table IV shows the network structures for three data sets. Note that we reshape the inputs of three data sets as 64×64 in our experiments. For the classifier and discriminator of MNIST and F-MNIST data sets, they share the same network structure where the model consists of six convolution layers and one dense layer. The kernel size of the first three convolutional layers is set to 4×4 and the last three convolutional layers are 3×3 . For the CIFAR-10 data set, the network structure of the classifier is a regular ResNet-18 CNN model. For the generator, the kernel size is 4×4 and the input is adopted

with random noise, which length is 100. The activation functions applied to all the neural network models are ReLU and LReLU, respectively.

Training Configurations: As the baseline of our experiments, we use the standard federated learning approach described in [1]. The total number of participants for MNIST, F-MNIST, and CIFAR-10 data sets is 33, 33, and 100, respectively, while each round selects ten participants randomly to execute the federated learning protocol. Each participant trains the same number of samples (i.e., 1818 images) while the attacker tries to generate fake samples that belong to these images based on the localized GAN model and further flip the labels of these fake samples. For the backdoor attack on the CIFAR-10 data set, following [19], we set all the participants (including attackers) train on 640 images and the triggers are predefined and selected from these samples, i.e., 27 cars with green color, 18 racing cars stripes, and 9 cars with striped walls. The local training configurations for three data sets are: the attackers train for $E = 20$ local epochs with the initial learning rate $\eta = 0.05$ while the victim trains for $E = 10$ with the initial learning rate $\eta = 0.1$. Besides, we decrease η by a factor of 10 every two epochs to prevent overfitting and we run all the experiments for 300 communication rounds of federated learning.

Attack Configurations: The attack scenarios include the single attacker and multiple attackers. For both settings, the total number of participants is not changed (i.e., 33, 33, and 100 for MNIST, F-MNIST, and CIFAR-10, respectively), where one or more participants are assumed as the attacker while the remaining participants are benign. To explain the capability of our proposed Data_Gen algorithm, we assign the training data set for one class to the attacker and the remaining nine classes are allocated to benign participants. The multiple attackers' setting is basically the same as the one attacker scenario except with there will be a high probability of selecting the attackers in participants' sampling phase. Other than the number of attackers, we also define a scale factor S as shown in Algorithm 2 to illustrate the change of poison task accuracy by improving the influence of poisoned local updates in global parameters.

Evaluation Metrics: To comprehensively illustrate our proposed generative poisoning attack model, we perform the evaluation with the following two goals: 1) *poison data generation*: means effectiveness of our proposed poison data generation Data_Gen algorithm in federated learning settings and 2) *attack success rate*: to evaluate the attack success rate of the global model trained with the poisoning algorithm PoisonGAN. According to the attacker's objectives described in Section III-A, we define two evaluation metrics, i.e., *poison task accuracy* and *main task accuracy*, to quantify our experimental results. Specifically, the poison task accuracy denotes the success ratio that the attacker-targeted samples were classified as the attacker-chosen labels against the total number of attacker-targeted records. Similarly, the main task accuracy means the success ratio that the nontargeted samples were classified as the correct labels against the total number of testing records except with the attacker-targeted samples. Here, we also define three indexes of our experimental evaluations,

⁴<https://github.com/pytorch/examples>

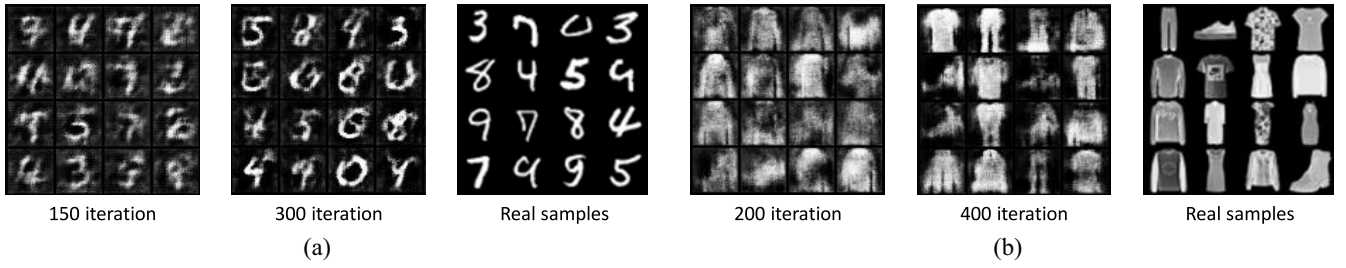


Fig. 6. Reconstruction results on (a) MNIST and (b) F-MNIST based on GAN.

TABLE V
GAN STRUCTURE FOR MNIST AND F-MNIST

Discriminator	$64^2 \times 1 \xrightarrow{\text{Conv, LReLU, DP}} 32^2 \times 64 \xrightarrow{\text{Conv, BN, LReLU}} 16^2 \times 64$
	$\xrightarrow{\text{Conv, BN, LReLU}} 8^2 \times 64 \xrightarrow{\text{Conv, BN, LReLU}} 8^2 \times 128 \xrightarrow{\text{Conv, BN, LReLU}} 11$
Generator	$8^2 \times 128 \xrightarrow{\text{Conv, LReLU}} 4^2 \times 128 \xrightarrow{\text{AvePool, FC, Softmax}} 11$
	$100 \xrightarrow{\text{ConvT, BN, LReLU}} 4^2 \times 256 \xrightarrow{\text{ConvT, BN, LReLU}} 16^2 \times 128$
	$\xrightarrow{\text{ConvT, BN, LReLU}} 32^2 \times 64 \xrightarrow{\text{ConvT, Tanh}} 64^2 \times 1$

$LReLU \rightarrow \text{LeakyReLU}$, $DP \rightarrow \text{Dropout}$, $BN \rightarrow \text{BatchNorm}$

which are scale factor S , the number of attackers A , and the number of client selection.

B. Effectiveness of Data Generative Model

To better understand how effective the poison data generation algorithm *Data_Gen* works in the federated learning protocol, we adopt GAN under a one single attacker scenario to visualize the samples' reconstruction results while the number of total and sampled participants are not changed. Note that the discriminative network of the GAN architecture is the same as the global model at each communication rounds, which can be updated along with the federated learning procedure. Table V shows the network structures for discriminator and generator used in our *Data_Gen* algorithm. As mentioned above, the generator is formatted as random noise with 100 lengths, and the middle layers are transposed convolutional layers, whose output size is reshaped as 64×64 . Furthermore, we set the attacker starts to generate samples after the accuracy of the global model is reached 90%, i.e., 12 and 15 iterations for MNIST and F-MNIST data sets, respectively.

Fig. 6 presents the visualization results for sample reconstruction as the number of iterations (communication rounds) of federated learning increases. We report the reconstruction results over 300 and 400 iterations for MNIST and F-MNIST data sets, along with the real samples drawn from two data sets. For the attacker's auxiliary data on two data sets, we assign class "0" and class "coat" to the attacker while all the victims' training data are randomly extracted from the real data set with ten classes. As shown in Fig. 6, the reconstructed samples at 150 and 200 iterations are already shown blurring profiles that close to the real MNIST and F-MNIST samples. At 300 and 400 iterations, the outlines of generated samples become more clear since the performance of the generator is getting better with the update of the discriminator. Therefore, by deploying a localized GAN model, the

attacker can successfully mimic victims' real samples. Note that the reconstruction of the F-MNIST data set takes more iterations because the images in F-MNIST are more complex than MNIST.

C. Performance of Poisoning Attack

In the poisoning attack evaluations, we take both label-flipping and backdoor attacks into consideration to demonstrate the effectiveness of our proposed generative poisoning attack model. As depicted in Section V-A, the indexes involved in our evaluations are mainly poisoning task accuracy and main task accuracy, in which the previous one means the success rate of classification results turn to the attacker-targeted classes by the final global model in the test data sets and the later one represents the correct classification rate apart from the poisoned records.

Accuracy Over Rounds: In order to demonstrate the effectiveness of our proposed *PoisonGAN* model, we choose the worst attacking scenario, where only one attacker exists during the whole federated learning procedure. Note that we implement the label-flipping attack on MNIST and F-MNIST data sets while the backdoor attack is deployed on the CIFAR-10 data set. Fig. 7 illustrates the poison and main task accuracy of image classification for 100 communication rounds after the attack was launched. As shown in Fig. 7(a), the poison task accuracy shows a slow upward trend since the number of backdoor triggers used in our backdoor attack experiments only accounts for a small proportion of the entire CIFAR-10 data set. Despite this, with the communication round goes on, the accuracy rate of the poison task will eventually reach a high level (60% at 100 communication round). As depicted in Fig. 7(b) and (c), the increasing trend is very fast after the attacker starts to upload the poisoned gradients since the attacker's local updates are well trained on the poisoned samples (the label of the whole class is flipped), which is dominating among other participants' local updates. However, as the federated learning procedure continues, the benign participants will upload more and more normal updates to degenerate the influence of attacker's poisoned updates so as to affect the accuracy of the poison task. To prevent the degradation of poisoning performance, we design a simple protection mechanism in our source code, which we cancel the attacker's local learning rate when the poisoning attack accuracy is lower than 60% such that we can maintain a steady poisoning success rate. Note that the larger jitters in three subfigures mean when there

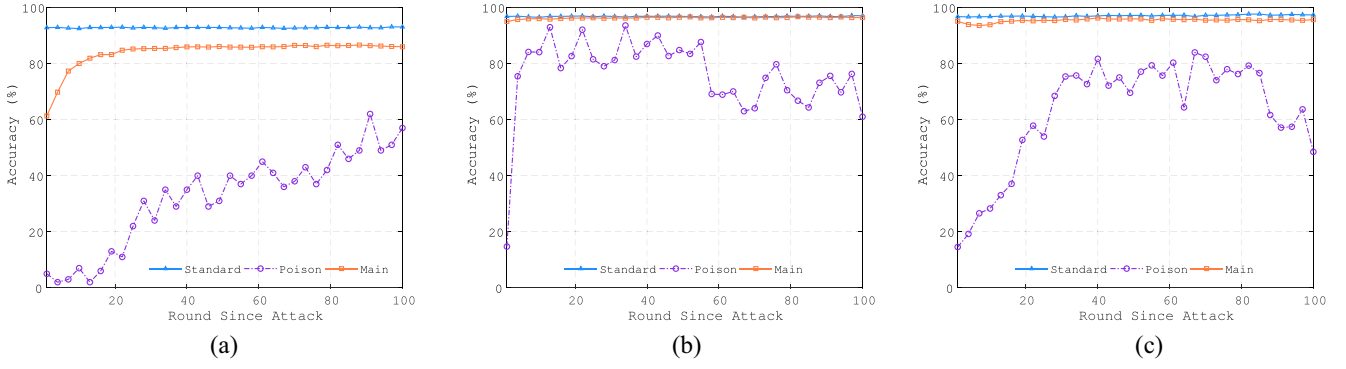


Fig. 7. Accuracy of poison and main tasks for (a) CIFAR-10, (b) MNIST, and (c) F-MNIST data sets over communication rounds.

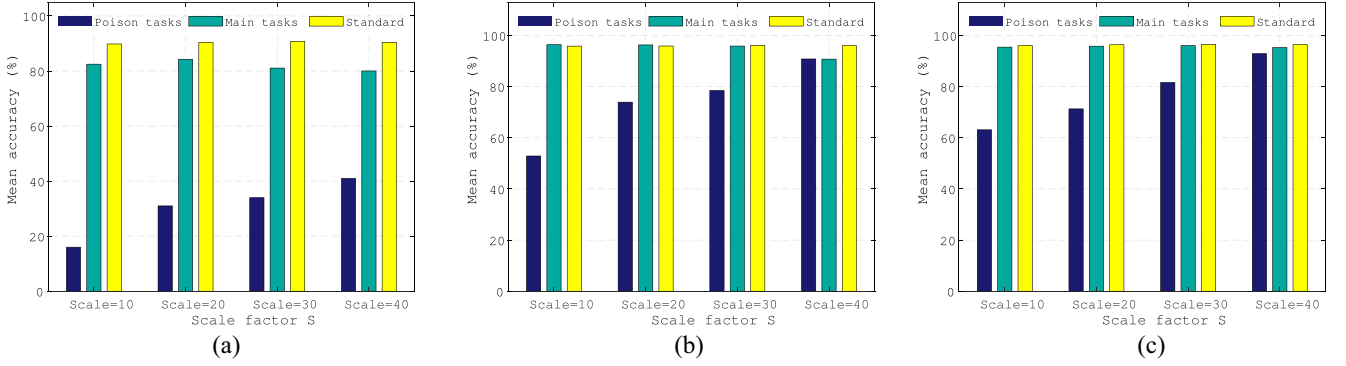


Fig. 8. Accuracy of poison and main tasks for three data sets, where A is fixed and S changes. (a) CIFAR-10 with different S . (b) MNIST with different S . (c) F-MNIST with different S .

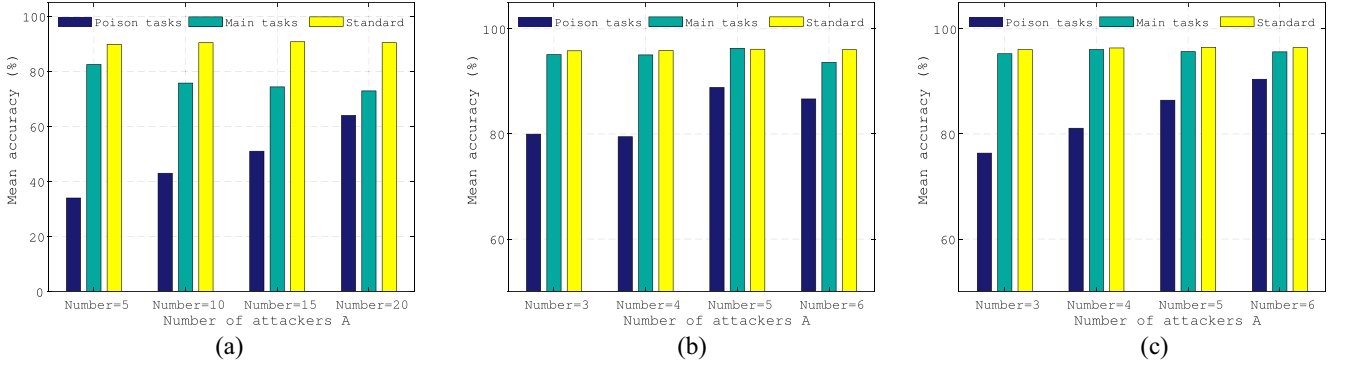


Fig. 9. Accuracy of poison and main tasks for three data sets, where S is fixed and A changes. (a) CIFAR-10 with different A . (b) MNIST with different A . (c) F-MNIST with different A .

are no communication rounds involving the attacker, the accuracy of the poisoning attack will be greatly reduced. Besides, the main task accuracy [especially in Fig. 7(b) and (c)] is almost unaffected and maintained above 85% since the label-flipping attack is only focus on the attacker-chosen class while other classes are not been changed.

Different Scale Factor: As we defined in Section V-A, the scale factor is one of the impact parameters, which aims to increase the influence of the attacker's poisoned local updates in the global model parameters. Thus, we design a set of experiments to evaluate our proposed PoisonGAN model under different scaling factors, where the number of attackers is set to be 1. For intuitively illustrating our experimental results,

we run 100 rounds of the federated learning protocol after the attacker starts to submit the poisoned local updates and then take the average of all accuracy results under four scale factor settings (i.e., $S = 10, 20, 30, 40$). Fig. 8 shows the mean accuracy of poison and main task accuracy with different scale factors for three data sets. Specifically, the scale-up method has no such significant impact on the backdoor attack [as shown in Fig. 8(a)] because the attacker only owns very little part of training samples, which limited the poisoning influence in the global model, even when the backdoored updates are scaled. However, for the label-flipping attack as depicted in Fig. 8(b) and (c), our generative poisoning attack model can achieve a minimum poison task accuracy of 50% and a maximum of

TABLE VI
MEAN ACCURACY OF POISON AND MAIN TASKS UNDER DIFFERENT SAMPLE CLIENTS

Overall settings		Attacker number: 1, Scale factor: 20			
Clients selection/Total clients		5/33	10/33	15/33	20/33
MNIST (label-flipping)	Poisoning accuracy	0.5287 \pm 0.0002	0.7392 \pm 0.0004	0.8848 \pm 0.0004	0.9080 \pm 0.0007
	Main accuracy	0.9641 \pm 0.0003	0.9629 \pm 0.0011	0.9584 \pm 0.0005	0.9574 \pm 0.0003
Clients selection/Total clients		10/100	20/100	30/100	40/100
CIFAR-10 (Backdoor)	Poisoning accuracy	0.3225 \pm 0.0003	0.6129 \pm 0.0002	0.7884 \pm 0.0013	0.9287 \pm 0.0003
	Main accuracy	0.9543 \pm 0.0004	0.9580 \pm 0.0006	0.9607 \pm 0.0004	0.9529 \pm 0.0007

92%. This is mainly because we set the attacker to train his local model for multiple epochs, thus increase the influence of poisoned gradients among other participants' local updates.

Multiple Attackers: Since the training phase is distributed across multiple participants in federated learning, the number of attackers could significantly affect the training phase and the poison task accuracy. To evaluate the influence of multiple attackers, we compare the accuracy of poison and main tasks for MNIST, F-MNIST, and CIFAR-10 data sets, where the scale factor is fixed on 20 and the number of attackers is changed from 5 to 20 for the CIFAR-10 data set and from 3 to 6 for MNIST and F-MNIST data sets. We also run 100 communication rounds under the different numbers of attackers to show how the mean accuracy changes for three data sets. From Fig. 9(a), we can see that the mean accuracy of poison tasks shows a significant increasing trend compared with Fig. 8(a), which means the backdoor attack is more benefiting from multiple attackers settings and the highest mean accuracy can reach almost 70%. However, the mean accuracy of main tasks will be negatively affected in multiple attackers setting since more backdoor triggers will account for a larger proportion of the total training data such that it impacts the normal features embedded in main tasks. Fig. 9(b) and (c) illustrates the experimental results on MNIST and F-MNIST data sets, compared with Fig. 8(b) and (c), the increasing trend of the poison task accuracy is relatively slow under multiple attackers setting. That is mainly because the federated learning model in our experiments is randomly selecting participants in each communication round such that the attackers may be not chosen in some continuous iterations. If there is no attacker participate in the federated learning model, the accuracy of poison tasks might be lower than in different scale factor settings since all the attackers' local updates are not scaled up. Different from the accuracy of poison tasks, the main task accuracy remains at a high level because the attacking strategy in label-flipping type is changing the label of one or more classes such that it will not affect the classification results of other classes.

Different Sample Clients: As we know, participant selection at each round is a significant step in federated learning for handling unbalanced data among multiple clients. This situation also means the attacker is possibly not be chosen for some communication rounds since the central server in federated learning is set to randomly select a subset of all participants to execute the local training process. Thus,

we evaluate the mean accuracy of poison and main tasks under different sample clients, where only one attacker has been involved and the scale factor is fixed in 20. Note that each evaluation result is averaged by running the poisoning program three times. From Table VI, we can easily notice that the mean poisoning accuracy, either label flipping or backdoor, shows a huge upward trend when the number of selected clients increases, which comes to a conclusion that the larger participant selections at each round the better performance of our poisoning attack method. Moreover, the mean accuracy of the main task still keeps a relatively gentle trend.

VI. DEFENSE DISCUSSION

For defending mechanisms, the conventional solutions are mainly coming to three categories: 1) verifying each local model updates in the distributed setting [13]; 2) identifying the poisoning data points from the assumed trusted data sets [25], [31]; and 3) reducing the negative effects of poison data through robust losses [33], [37]. For example, Mozaffari-Kermani *et al.* [25] proposed to distinguish the poison and benign data points by assuming a golden model to evaluate the validation accuracy for training data. Later, Han *et al.* [37] presented the stronger-convex losses for the SGD method to enhance the robustness of model parameters on noised labels. However, the defense mechanisms of the above analysis are based on the assumption that the server already has the same distributed verification data set as the training data, which is impractical in federated learning since the training data are not shared among participants.

According to a recent adversarial training proposal, we found that pivotal training [38] is a potential technique that could be used to defend the poisoning attacks in the context of federated learning. This method was designed to improve the robustness of neural network models trained on contaminated data sets and the scheme itself is simple and scalable since it does not require complex accuracy validations. Compared with the existing defense mechanisms, pivotal adversarial training does not need access to the participants' raw data sets and computes only on the received weight changes from all the participants' local models. The feasibility of pivotal adversarial training for defending a poisoning attack is that the server in federated learning can distinguish the difference among received local gradients by creating an extra predictive model to infer which participant a certain gradient belongs to. In this

way, the federated model is modified to prevent this discriminating behavior, thus eliminating the influence that poisoned local updates have on the global model.

VII. SUMMARY AND FUTURE WORK

In this article, aiming at exploring an active and powerful attack model, poisoning attacks, in federated learning-aided IoT systems, we propose a novel poisoning attack model based on GANs. On the basis of the localized training property of the federated model, we designed a poison data generation method to eliminate the conventional attacking assumption that the attacker already owns a proportion of other participants' training data. This method mainly relies upon the iteratively updated global model parameters to renew the discriminator and feed back to the generator to generate fake samples that share the same distribution of victims' training data sets. For effectively launching poisoning attacks based on our proposed data generation method, we further presented a novel poisoning attack model, named generative poisoning attack, in the context of federated learning. Through the extensive experiments on two types of typical poisoning attack strategies, i.e., label flipping and backdoor, we found that our proposed poisoning attack model can effectively compromise the global model. In our future work, we will try to explore a robust federated learning protocol by integrating pivotal learning with federated training, providing sophisticated robustness guarantees to defend against kinds of anomaly updates by internal participants.

REFERENCES

- [1] H. 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. (AISTATS)*, 2017, pp. 1–10.
- [2] E. Hesamifard, H. Takabi, M. Ghasemi, and R. N. Wright, "Privacy-preserving machine learning as a service," in *Proc. Privacy Enhancing Technol. (PETS)*, 2018, pp. 123–142.
- [3] W. Yu *et al.*, "A survey on the edge computing for the Internet of Things," *IEEE Access*, vol. 6, pp. 6900–6919, 2017.
- [4] S. Yu, "Big privacy: Challenges and opportunities of privacy study in the age of big data," *IEEE Access*, vol. 4, pp. 2751–2763, 2016.
- [5] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on Internet of Things: Architecture, enabling technologies, security and privacy, and applications," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1125–1142, Oct. 2017.
- [6] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, 2019.
- [7] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Security (CCS)*, 2015, pp. 1310–1321.
- [8] J. Zhang, B. Chen, S. Yu, and H. Deng, "PEFL: A privacy-enhanced federated learning scheme for big data analytics," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Waikoloa, HI, USA, 2019, pp. 1–6.
- [9] A. N. Bhagoji, S. Chakraborty, E. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," 2018. [Online]. Available: <https://arxiv.org/abs/1811.12470>.
- [10] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *Proc. Symp. Security Privacy (SP)*, 2019, pp. 739–753.
- [11] L. Melis, C. Song, E. D. Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *Proc. Symp. Security Privacy (SP)*, San Francisco, CA, USA, 2019, pp. 691–706.
- [12] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li, "Manipulating machine learning: Poisoning attacks and countermeasures for regression learning," in *Proc. Symp. Security Privacy (SP)*, San Francisco, CA, USA, 2018, pp. 19–35.
- [13] J. Hayes and O. Ohrimenko, "Contamination attacks and mitigation in multi-party machine learning," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, 2018, pp. 6604–6616.
- [14] J. Zhang, J. Chen, D. Wu, B. Chen, and S. Yu, "Poisoning attack in federated learning using generative adversarial nets," in *Proc. 18th IEEE Int. Conf. Trust Security Privacy Comput. Commun./13th IEEE Int. Conf. Big Data Sci. Eng. (TrustCom/BigDataSE)*, Rotorua, New Zealand, 2019, pp. 374–380.
- [15] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," in *Proc. 29th Int. Conf. Mech. Learn. (ICML)*, 2012, pp. 1807–1814.
- [16] S. Mei and X. Zhu, "Using machine teaching to identify optimal training-set attacks on machine learners," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2871–2877.
- [17] L. Muñoz-González *et al.*, "Towards poisoning of deep learning algorithms with back-gradient optimization," in *Proc. 10th ACM Workshop Artif. Intell. Security (AISec)*, 2017, pp. 27–38.
- [18] S. Mahloujifar, M. Mahmoodi, and A. Mohammed, "Data poisoning attacks in multi-party learning," in *Proc. Int. Conf. Mech. Learn. (ICML)*, 2019, pp. 4274–4283.
- [19] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. 23rd Int. Conf. Artif. Intell. Stat. (AISTATS)*, 2020, pp. 1–10.
- [20] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 119–129.
- [21] Y. Shi and Y. Sagduyu, "Evasion and causative attacks with adversarial deep learning," in *Proc. IEEE Military Commun. Conf. (MILCOM)*, Baltimore, MD, USA, 2017, pp. 243–248.
- [22] M. Fang, X. Cao, J. Jia, and N. Z. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," in *Proc. USENIX Security Symp.*, 2020, pp. 1605–1622.
- [23] C. Yang, Q. Wu, H. Li, and Y. Chen, "Generative poisoning attack method against neural networks," 2017. [Online]. Available: <https://arxiv.org/abs/1703.01340>.
- [24] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, 2014, pp. 2672–2680.
- [25] M. Mozaffari-Kermani, S. Sur-Kolay, A. Raghunathan, and N. K. Jha, "Systematic poisoning attacks on and defenses for machine learning in healthcare," *IEEE J. Biomed. Health Inform.*, vol. 19, no. 6, pp. 1893–1905, Nov. 2015.
- [26] B. Hitaj, G. Ateniese, and F. Pérez-Cruz, "Deep models under the GAN: Information leakage from collaborative deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security (CCS)*, 2017, pp. 603–618.
- [27] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Paris, France, 2019, pp. 2512–2520.
- [28] H. Xiao, B. Biggio, G. Brown, G. Fumera, C. Eckert, and F. Roli, "Is feature selection secure against training data poisoning?" in *Proc. Int. Conf. Mech. Learn. (ICML)*, 2015, pp. 1689–1698.
- [29] H. Xiao, B. Biggio, B. Nelson, H. Xiao, C. Eckert, and F. Roli, "Support vector machines under adversarial label contamination," *Neurocomputing*, vol. 160, pp. 53–62, Jul. 2015.
- [30] H. Xiao, H. Xiao, and C. Eckert, "Adversarial label flips attack on support vector machines," in *Proc. 20th Eur. Conf. Artif. Intell. (ECAI)*, 2012, pp. 870–875.
- [31] S. Shen, S. Tople, and P. Saxena, "Auror: Defending against poisoning attacks in collaborative deep learning systems," in *Proc. 32nd Annu. Conf. Comput. Security Appl. (ACSAC)*, 2016, pp. 508–519.
- [32] N. Baracaldo, B. Chen, H. Ludwig, and J. A. Safavi, "Mitigating poisoning attacks on machine learning models: A data provenance based approach," in *Proc. 10th ACM Workshop Artif. Intell. Security (AISec)*, 2017, pp. 103–110.
- [33] B. Wang *et al.*, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *Proc. IEEE Symp. Security Privacy (SP)*, San Francisco, CA, USA, 2019, pp. 707–723.
- [34] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 9, pp. 2805–2824, Sep. 2019.
- [35] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," 2018. [Online]. Available: <https://arxiv.org/abs/1806.00582>.

- [36] K. Bonawitz *et al.*, "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security (CCS)*, 2017, pp. 1175–1191.
- [37] B. Han, I. W. Tsang, and L. Chen, "On the convergence of a family of robust losses for stochastic gradient descent," in *Proc. Eur. Conf. Mach. Learn. Knowl. Discover. Databases (ECML PKDD)*, 2016, pp. 665–680.
- [38] G. Louppe, M. Kagan, and K. Cranmer, "Learning to pivot with adversarial networks," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 981–990.



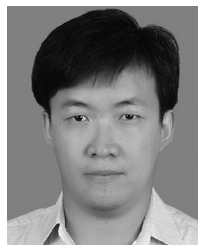
Jiale Zhang (Student Member, IEEE) received the M.E. degree in computer technology from Tianjin Polytechnic University, Tianjin, China, in 2017. He is currently pursuing the Ph.D. degree with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China.

His research interests are mainly edge computing, privacy preserving, and machine learning.



Bing Chen (Member, IEEE) received the B.S. and M.S. degrees in computer engineering from the Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China, in 1992 and 1995, respectively, and the Ph.D. degree from the College of Computer Science and Technology, NUAA, in 2008.

He is currently a Full Professor with the College of Computer Science and Technology, NUAA. His research interests include cloud/edge computing, security and privacy, federated learning, and wireless communications.



Xiang Cheng received the M.E. degree in computer technology from the Civil Aviation University of China, Tianjin, China, in 2016. He is currently pursuing the Ph.D. degree with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China.

His research interests are mainly cyberspace situation awareness, advanced persistent threats, and network security.

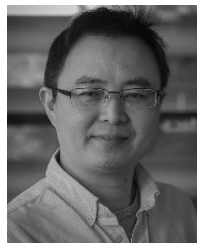


Huynh Thi Thanh Binh received the Ph.D. degree from the VNU University of Science, Hanoi, Vietnam.

She is an Associate Professor and a Vice Dean of the School of Information and Communication Technology, Hanoi University of Science and Technology, Hanoi. She is the Head of Modeling, Simulation and Optimization Lab. She has published more than 90 refereed academic papers, two books. Her current research interests include computational intelligence, artificial intelligence, and evolutionary

multitasking.

Dr. Binh is the Chair of IEEE Computational Intelligence Society Vietnam Chapter; IEEE Asia Pacific Executive Committee Member; IEEE Asia Pacific Student Activities Committee Chair in 2019 and 2020. She is an Associate Editor of *International Journal of Advances in Intelligent Informatics* and *VNU Journal Computer Science Communication Engineering*.



Shui Yu (Senior Member, IEEE) received the Ph.D. degree from Deakin University, Melbourne, VIC, Australia.

He is currently a Full Professor with the School of Software, University of Technology Sydney, Sydney, NSW, Australia. He has published two monographs and edited two books, more than 200 technical papers, including top journals and top conferences, such as IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON COMPUTERS,

IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING, IEEE/ACM TRANSACTIONS ON NETWORKING, and INFOCOM. He initiated the research field of networking for big data in 2013. His h-index is 33. His research interest includes Security and privacy, networking, big data, and mathematical modeling.

Prof. Yu actively serves his research communities in various roles. He is currently serving the editorial boards of IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, *IEEE Communications Magazine*, IEEE INTERNET OF THINGS JOURNAL, IEEE COMMUNICATIONS LETTERS, IEEE ACCESS, and IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS. He has served many international conferences as a member of organizing committee, such as publication chair for IEEE Globecom 2015, IEEE INFOCOM 2016 and 2017, a TPC Chair for IEEE Big Data Service 2015, and a General Chair for ACSW 2017. He is a final voting member for a few NSF China programs in 2017. He is a member of AAAS and ACM, the Vice Chair of Technical Committee on Big Data of IEEE Communication Society, and a Distinguished Lecturer of IEEE Communication Society.