2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering

Poisoning Attack in Federated Learning using Generative Adversarial Nets

Jiale Zhang*, Junjun Chen†, Di Wu‡§, Bing Chen*, and Shui Yu‡
*College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China
†College of Information Science & Technology, Beijing University of Chemical Technology, Beijing 100029, China
‡School of Software, §Center for Artificial Intelligence, University of Technology Sydney, Sydney 2007, Australia
Email: jlzhang@nuaa.edu.cn, chenjj@mail.buct.edu.cn, Di.Wu-16@student.uts.edu.au,
cb_china@nuaa.edu.cn, Shui.Yu@uts.edu.au

***Abstract*—Federated learning is a novel distributed learning framework, where the deep learning model is trained in a collaborative manner among thousands of participants. The shares between server and participants are only model parameters, which prevent the server from direct access to the private training data. However, we notice that the federated learning architecture is vulnerable to an active attack from insider participants, called *poisoning attack*, where the attacker can act as a benign participant in federated learning to upload the poisoned update to the server so that he can easily affect the performance of the global model. In this work, we study and evaluate a poisoning attack in federated learning system based on *generative adversarial nets (GAN)*. That is, an attacker first acts as a benign participant and stealthily trains a GAN to mimic prototypical samples of the other participants' training set which does not belong to the attacker. Then these generated samples will be fully controlled by the attacker to generate the poisoning updates, and the global model will be compromised by the attacker with uploading the scaled poisoning updates to the server. In our evaluation, we show that the attacker in our construction can successfully generate samples of other benign participants using GAN and the global model performs more than 80% accuracy on both poisoning tasks and main tasks.**

***Index Terms*—Federated learning, poisoning attack, generative adversarial nets, security, privacy.**

I. INTRODUCTION

Federated learning [1], [2] is a recent concept which enables to train a deep learning model across thousands of participants in a collaborative manner. The main purpose of federated learning is to build a joint machine learning model upon localized datasets while providing privacy guarantee, which is an attractive technique for numerous emerging scenarios, such as edge computing and crowdsourced system [3], [4]. Participants in federated learning act as the data provider to train a local deep model, and the server maintains a global model by averaging local model parameters (i.e., gradients) generated by randomly selected participants until it tends to convergence. One biggest achievement for federated learning is the corresponding model average algorithm [5], which can benefit from a wide range of non-IID and unbalanced data distribution among diversity participants. However, the server in federated learning is designed to have no privilege to access the participants' local data and training process due to the privacy concern. Such invisibility property would introduce a severe security threat – *poisoning attack* [6]–[8].

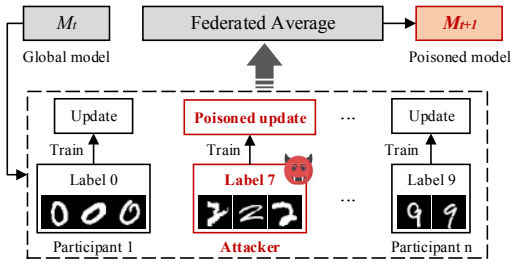


Fig. 1. An illustrating of poisoning attack in federated learning. The attacker labels '7' on the target samples and trains the model M_t at iteration t on the modified data. After model average, the global model is replaced by poisoned model M_{t+1} at iteration $t+1$.

As one type of causative attack [9], [10], poisoning attack enables an attacker to manipulate partial training data with attacking labels to change the model parameters of the target learning model in training phase, and then the poisoned learning model would have some attacker expected properties to misclassify the chosen inputs at the inference stage [11], [12]. *We notice that it is possible to construct the poisoning attack in federated learning for the following reasons* [6]: 1) there are plenty of participants in federated learning system, it is most likely contains one or more malicious users; 2) since the participants' local data and training process are invisible to the server, it is impossible to verify the authenticity of a certain participant's update [13]; 3) the local updates generated by multiple participants may be very different from each other and the secure aggregation protocol [14] makes the local updates cannot be audited by the server.

In this work, we first devise a novel poisoning attack in federated learning based on Generative Adversarial Nets (GAN) [15], with the aim of successfully launching the poisoning attack without compromise any benign participant. Meaning that the attacker in our construction does not need to invade other participants' validation dataset to obtain the original local training data. Apparently, our attack is more stealth compared to the conventional poisoning attacks, because of we do not require the attacker to invade any participant's device and no need to struggle with intrusion detection mechanism embedded in the local system [16], [17]. Fig. 1 presents an example of

2324-9013/19/$31.00 ©2019 IEEE
DOI 10.1109/TrustCom/BigDataSE.2019.00057

poisoning attack based on flipping labels in federated learning. In each communication round, the central server distributes the current global model M_t to the participants, and then each participant trains it on the local dataset to generate the model update. On contrast, the attacker poisons the training data by flipping labels and train the model on the poisoned data. Finally, the attacker uploads the poisoned update along with other participants' updates to the server. After federated averaging, the global model will be replaced by poisoned model M_{t+1}.

In our construction, the attacker firstly acts as a benign insider to execute the federated learning protocol and deploys a GAN architecture to reconstruct the private training data from other participants, and then the attacker adds wrong labels to the GAN generated data. After that, those poisoned data is injected into the training procedure to obtain the local model updates to further compromise the global model. Finally, the poisoned global model would present high probability of the classes being classified as the attacker-chosen classes and have no influence about other non-poisoned classes. For example, misclassify '2's to class '7' and correctly classify '9's to class '9' in Fig. 1. Note that the local learning procedure is fully controlled by the attacker, so he can change the training configurations such as the number of epochs and learning rate to make the poisoned global model performs well on both main task and poisoned task.

Our main contributions can be summarized as follows.

- We first demonstrate that the federated learning architecture is vulnerable to the poisoning attack which can be launched by any malicious participant. Then, we devise a novel poisoning attack based on GAN and successfully construct it in the federated learning system.
- Our attack is generic and feasible under a realistic threat model, which only requires the attacker acting as a benign insider to participant in federated learning system and deploying a GAN architecture in the local to mimic other participants' private training data.
- We conduct exhaustive experiments to demonstrate the effectiveness of both GAN-based generative method and poisoning attack approach in federated learning. On the MNIST and AT&T datasets, the global model performs more than 80% classification accuracy on both poisoning tasks and main tasks.

The rest of this paper is organized as follows. Section II-A introduces the background knowledge about federated learning and generative adversarial nets. Section III discusses the threat models, and then the proposed poisoning attack is detailed in Section IV. Experimental evaluation is conducted in Section V and the simple remarks about our attack are given in Section VI. Finally, Section VII concludes the paper.

II. Background

A. Federated Learning

Traditionally, a centralized training method require users to upload their personal data to train a model with all the collected data, which the users' privacy is under high leaking risk [18], [19]. In contrast, federated learning can significantly preserve privacy by its typical distribution learning scheme. Unlike centralized method, federated learning only requires users to upload gradients which generated by the local training data and models, and the global models on the server side will share the same structure with local client models. Suppose there are n of users with same objective to join the federated learning, and the local dataset are totally different among users. For each iteration, users will download the parameters and global model from the server side, and then train the model by the local datasets by each client. Each user will upload the gradients after training stage, where the uploaded gradients will be averaged and accumulated to the current global model. Eq. 1 shows the updating procedure on the global model.

$$m_{i+1} = m_i + \frac{1}{n}\sum_{k=1}^{n} g_i^k \tag{1}$$

where m_{i+1} indicates the current shared model at the i-th iteration, and g_i^k represents the gradient uploaded by the k-th user at iteration i. A federated learning framework can achieve high satisfaction when the users download the same model with the same initialization, which is averaged by the global model with all the valid uploads.

B. Generative Adversarial Nets

Generative adversarial nets (GAN) [15] achieved extraordinary success in computer vision research area which can generate high quality fake images based on the original image set. There are two neural networks namely generator and discriminator in the GAN structure, where generator (G) generates images and discriminator (D) discriminate if the image is from generator or original image set which can be represent as 0/1 (fake/real). To train a GAN, G will generate an image from a random vector z which follows a prior distribution such as Gaussian or uniform distribution. The generated image will be the input of D. In the meantime, D is pre-trained by the original image set which can be used to discriminate the input is real or fake. The performance of both G and D can be improved by playing the adversarial game. Eq. 2 shows the training progress of GAN.

$$\begin{aligned}\min_G \max_D V(D,G) =& \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \\ & \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))],\end{aligned} \tag{2}$$

where $p_{data}(x)$ indicates the original image distribution and $p_z(z)$ represents the distribution of the random vector z. D and G will be trained by several rounds until the adversarial game achieves the Nash equilibrium.

III. Threat Model

A. Learning Scenario

Following the federated learning protocol as described in Sec. II-A, our threat model considers multiple participants ($N \geq 2$) to jointly train a global model on their localized

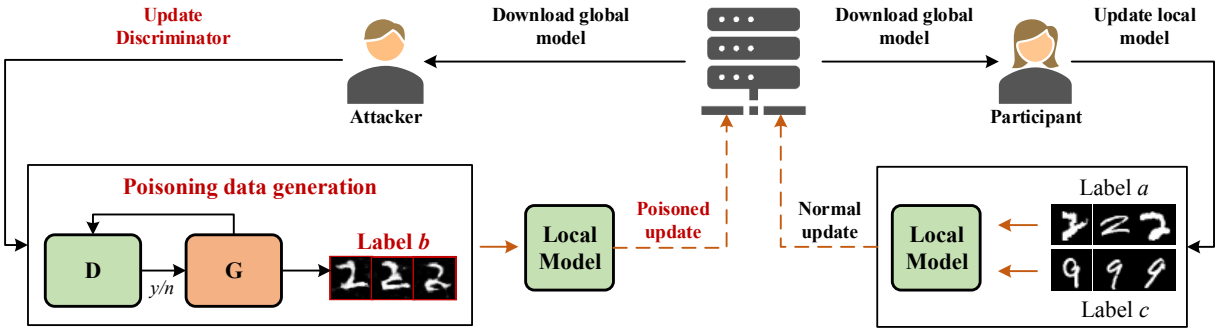


Fig. 2. Overview of the proposed GAN-based poisoning attack in the federated learning. The attacker generates one or more classes training data, training a model on the poisoned data using federated learning protocol, and submits the resulting model. After federated averaging, the global model is replaced by the attacker's poisoned model.

training dataset. We assume there existing one or more attackers among these participants, whose main purpose is to mimic another participants' dataset and poison the global model. By contrast, we assume that the central server is not malicious and most part of participants is trusted except with the attackers.

In our settings, the attacker can only access to the global model through the federated learning procedure, but he cannot control the aggregation or average algorithm at server side. Besides, all the attackers also cannot affect any other benign participants' learning phase or training data $\mathcal{D}_{benign}$, while they must correctly execute the training algorithm on their poisoned dataset $\mathcal{D}_{poison}$ according to the federated learning protocol to update their local models. Without loss of generality and simplicity, the global model plays a role as an image classifier in our paper.

B. Attacker's Goals

The attacker under our construction aims to corrupt the global model in the training phase, while disguised as a benign participant. In the federated learning procedure, the attacker deploys a GAN architecture to generate similar samples which belong to other benign participants, then he inserts poison samples assigned with the wrong label to the local training dataset. The local model will be trained on these poisoned data to further generate the poisoned local update. After the model average at the server side, the prediction results on parts of benign participants' inputs will be changed to the attacker-chosen classes.

The attacker's goals can be summarized as follows: (1) *Samples generation*: the attacker can successfully mimic the prototypical samples from the benign participants' training data without access their local dataset directly. (2) *Poisoned task accuracy*: once the poisoned local updates trained on generated faker samples were uploaded to the central server, the global model should perform high accuracy on the poisoned task for several rounds after the attack. (3) *Main task accuracy*: the global model should also present high accuracy on main tasks (non-poisoned) to prevent the global model from being discarded.

C. Attacker's Capability

The attacker pretends to be an honest participant in the federated learning protocol but tries to compromise the global model by uploading the poisoned local updates trained on attack-poisoned fake dataset. These poisoned samples were totally generated by the attacker who trains a GAN to extract the prototypical samples of the other participants' training data he does not own.

Specifically, we consider the following attacker's capabilities: (1) *Active*: the attack is assumed as an active insider since he can deploy a GAN architecture locally, modify any training inputs, manipulate the training procedure, or even change the scale of local update. (2) *Knowledgeable*: the attacker has full knowledge of the model structure because all the participants in federated learning are agreed in advance on a common learning objective.

IV. Attack Construction

A. Attack Overview

To conduct our poisoning attack in federated learning, the attacker needs to execute the following strategy with three steps: firstly generates the poisoning samples based on generative adversarial nets model and adds these samples into the local training dataset, and then injects the generative samples with wrong label into the training procedure to compute the poisoned local parameters, lastly uploads the magnified parameters to the central server, aiming to misleading the global model to misclassify the correct inputs as the target wrong label at inference stage. For simplicity, we firstly consider two participants (one attacker $\mathcal{A}$ and one benign participant $\mathcal{P}$) federated learning scenario to explain the construction of our proposed poisoning attack and the more complexity attack strategy will be described in Sec. IV-C.

Fig. 2 presents the detailed procedure of our proposed GAN-based poisoning attack. Assuming $\mathcal{P}$ and $\mathcal{A}$ train the data from class a and class b, respectively, and these two participants agree on a common learning objective and model structure as required in federated learning protocol. In this case, the information from class a is invisible for $\mathcal{A}$. The attacker's goal

is to mimic the samples from class a and further launch the poisoning attack. Thus, $\mathcal{A}$ locally deploys a GAN (unknown to the central server and $\mathcal{P}$) to generate as much similar samples as possible about class a of participant $\mathcal{P}$, and further injects these generated samples from class a, as class b into the local training procedure. In this way, the attacker can simply train his local model on the generated poison data and uploads the resulting parameters to the central server. Thus, the poisoning attack can be successfully launched in federated learning by scaling up the poisoned updates.

B. Poisoning Data Generation

The GAN procedure is an adversarial game between two neural networks, namely discriminator D and generator G. The discriminator is trained to distinguish between original data and generated data, while the generator is trained to mimic samples from the training set of the discriminative network. Inspired by [20], to mimic samples from another participants in federated learning, we adopt a GAN architecture at the attacker side, which creates a replica of the global model as the discriminator. With the continuous execution of the federated learning protocol, each participant's updates will promote the convergence of the global model. Correspondingly, as a replica of the global model, the discriminator of GAN will be synchronously updated to guide the right direction for the generator. In this way, the attacker can generate high-quality mimic samples from other participants' dataset.

For example, as shown in Fig. 2, the goal of the attacker is to mimic as much samples (digital 2 from the MNIST dataset) as possible from other participants' training data (class b) that he does not own. Firstly, the attacker downloads the global model and duplicates a copy as the discriminator D to build a localized GAN. Then, the coupled generator G will send the generated image to the discriminator to evaluate the generated image is real or fake. In the meantime, the attacker will keep joining the federated learning procedure to get the latest global model and further update his discriminator. This training scheme will continue several rounds until the generator can generate high-quality images on class b.

Non-coincidentally, the coupled generator G would generate mimic samples that are very similar to the original samples if the attacker keep joining in the federated learning. This is because the attacker can use the high-performance global model as the discriminator to generate high-quality fake samples which do not belong to the attacker. These fake samples can be the poison to affect the classification performance on specific target classes.

C. GAN-based Poisoning Attack

From the above analysis, the key point for constructing our poisoning attack is that the attacker inserts the generated poisoning samples with artificial wrong label into his own training dataset and further uploads the poisoned model updates ΔL^p to the central server. The proposed poisoning attack can be formulated as below:

- Step 1: Assuming $\mathcal{P}$ and $\mathcal{A}$ are two participants in the federated learning system, who own the different private training dataset (class a and b);
- Step 2: Run the federated learning protocol for several rounds to upgrade the global model until the accuracy reaches a certain level;
- Step 3: For the participant $\mathcal{P}$:
  (1) Downloads the global model to replace the parameters of her local model;
  (2) Trains the model on the local dataset and uploads the local update ΔL^i to the central server;
- Step 4: For the attacker $\mathcal{A}$:
  (1) Downloads the global model to replace the parameters of his local model;
  (2) *Creates a replica of the new local model as D (discriminator) and runs G (generator) on D to mimic samples targeting class b from the participant $\mathcal{P}$*;
  (3) *Labels the generated samples of class a as class b and inserts the poisoned data samples to the $\mathcal{A}$'s local dataset*;
  (4) *Trains his local model based on the merged dataset and generates the poisoned local model update ΔL^p*;
  (5) *Scales up the ΔL^p with deliberated factor λ and sends the poisoned update $\lambda \Delta L^p$ to the central server*;
- Step 5: Repeat Steps 3 and 4 until the global model convergence.

In particular, the substeps from (2) to (5) in step 4 represent the poison data generative model and the poisoning attack model that the attacker performed covertly to generate as much as possible samples of the targeted class b and further poison the global model. Note that the efficiency of our GAN-based generative model depends only on the accuracy of the global model (discriminator). The generalized poisoning attack with multiple attackers is depicted in Algorithm 1.

V. Experimental Evaluation

A. Datasets and Experiment Setup

1) MNIST: MNIST is one of the benchmark datasets used in lots of deep learning scenarios which contains 70000 handwritten grayscale digits images ranging from 0 to 9 (i.e., 10 classes). Each image is with the size of 28×28 pixels, and the whole MNIST dataset is divided into the 60000 training records and 10000 testing data records.

2) AT&T: AT&T (Olivetti dataset) is another widely used dataset with 400 grayscale face images from 40 different persons (10 images for one person). The images in AT&T were taken several times and present large difference in facial expression. The original size of AT&T is 92×112.

3) Experiment Setup: As described in Sec. IV, the components involved in our proposed poisoning attack are the classifier, discriminator and the generator. We use the Convolutional Neural Network (CNN) based architecture to construct the classifier and discriminator, and the corresponding generator is constructed with deconvolution network. Table I shows the network structures for MNIST and AT&T. Note that we resize

Algorithm 1: Poison Attack in Federated Learning.

Input: Global model M_t; Participants' updates ΔL_t^i; Loss function ℓ; Learning rate η.
Output: Poisoned updates $\Delta \hat{L}_t^i$.
Initialize generator G and discriminator D
for $t \in (1, 2, \cdots, T)$ **do**
  // Server execution
  Send M_t to the participants
  Recive updates from participants: ΔL_{t+1}^i
  Update the globa model: M_{t+1}
  // Participants execution
  Replace the local model: $L_t^i \leftarrow M_t$
  if *the user type is* $\mathcal{A}$ **then**
    Initialize D by the new local model L_t^i
    for *each epoch* $e \in (1, \cdots, E)$ **do**
      Run G on D for targeted class
      Using D to update G
      Generate samples of targeted class by G
      Assign wrong label to generated samples
      Insert poison data to the local dataset $\mathcal{D}$
      for *each batch* $b_p \in \mathcal{D}_{poison}$ **do**
        $L_{t+1}^p = L_{t+1}^p - \eta_{adv} \nabla \ell(L_t^p, b_p)$
      end
    end
    Calculate poisoned update: $\Delta L_{t+1}^p = L_{t+1}^p - L_t^p$
    Scale up the update: $\Delta \hat{L}_{t+1}^p = \lambda \Delta L_{t+1}^p$;
  end
  else
    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
end

the inputs of two datasets as 64×64 in our experiments. For the classifier and discriminator, they share the same network structure where the model consists of 4 convolution layers and 2 dense layers. The kernel size of the first three convolutional layers is 4×4 and the last layer has a kernel of size 3×3. The strides for these convolutional layers are 2, 2, 4, and 1, respectively. In particular, we randomly sampled ten classes for AT&T dataset to ensure the output is the same with MNIST dataset. For the generator, the kernel size is 4×4 and the input is adopted with random noise, which length is 512. The activation functions applied to both generator and discriminator are ReLU and LReLU, respectively.

Our experiments include two scenarios: single attacker and multiple attackers. For one attacker scenario, we set the number of participants is $m = 10$, where one participant is assumed as the attacker and the remaining 9 participants are benign. To explain the capability of our GAN-based generative model, we assign the training dataset 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 one attacker scenario except with the number of the total participants is 100. During the training of the global classifier, each benign participant trains for $E = 2$ local epochs with the learning rate of $lr = 0.1$. Each attacker trains for $E = 6$ local epochs with the initial learning rate of $lr = 0.05$, and lr drops by 10% every two epochs. We run all experiments for 100 communication rounds of federated learning. At each communication round, the participants' models are trained individually and sequentially before being averaged into the global model. All the experiments are conducted under the federated learning settings using the PyTorch [21] framework. We implemented the experiments on a RHEL7.5 server with NVidia Quadro P4000 GPU and 32GB RAM.

TABLE I
NETWORK STRUCTURE FOR MNIST AND AT&T

Classifier/ Discriminator	$64^2 \times 1 \xrightarrow{Conv\ (stride=2)} 32^2 \times 128 \xrightarrow{Conv\ (stride=2)} 16^2 \times 256 \xrightarrow{Conv\ (stride=4)} 4^2 \times 512 \xrightarrow{Conv\ (stride=1)} 2^2 \times 1024 \xrightarrow{FC} 4096 \xrightarrow{FC,\ Softmax} 11$
Generator	$512 \xrightarrow{Deconv} 4^2 \times 512 \xrightarrow{Deconv} 16^2 \times 256 \xrightarrow{Deconv} 32^2 \times 128 \xrightarrow{Deconv} 64^2 \times 1$

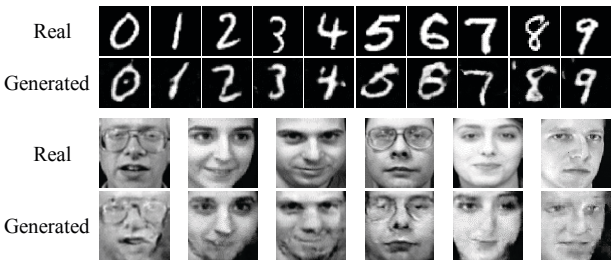
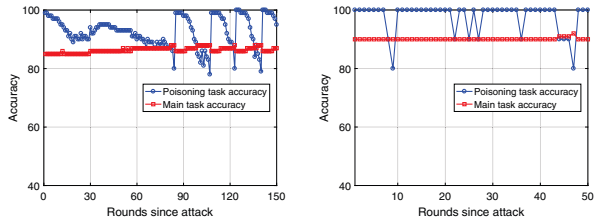


Fig. 3. Results generated when attacker runs a GAN trained on MNIST and AT&T datasets.

B. The Effectiveness of Generative Model

We evaluate our poison data generative model under one attacker settings in federated learning, which the number of total participants $N = 10$. To mimic samples from other participants' training dataset, we adopt the GAN architecture at the attacker side, where the discriminative network of GAN is as same as the global model in federated learning protocol. We also set the attacker starts to generate samples after the accuracy of the global model is reached 95%.

Fig. 3 shows the resulted samples generated by attacker in the federated learning. The 1st and 3rd rows are the real samples from other participants. The 2nd and 4th rows are the generated samples based on our poison data generative model. Note that we present the generated results about ten classes for MNIST dataset and six classes for AT&T dataset, that is because we repeat our experiments for many times so that the participants can traverse almost all classes of two datasets. The results demonstrate that our GAN-based generative model can successfully mimic participants' original samples.



(a) (MNIST) (b) (AT&T)

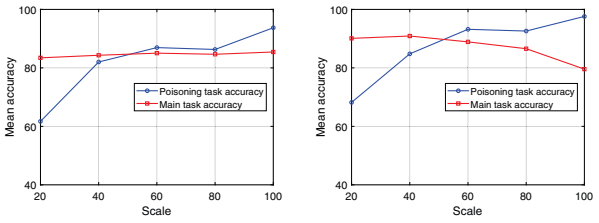
Fig. 4. Accuracy of poisoning task and main task with single attacker.

C. Experimental Results

In our experiments, we define the *poisoning task accuracy* to represent the success rate of classify results turn to the attack-chosen classes by the averaged global mode in the testing dataset, and the *main task accuracy* as the success rate of correctly classify except with the poisoned items. For MNIST dataset, each attack trains 14 epochs for the local model on the poisoned dataset to improve the poisoning task accuracy in the global model and the experiments are run 150 rounds in the setting of federated learning. For AT&T dataset, we run the experiments for 50 rounds. Followed by [6], we also set a 10% degradation of attacker's learning rate and to further ensure the effectiveness of our poisoning attack.

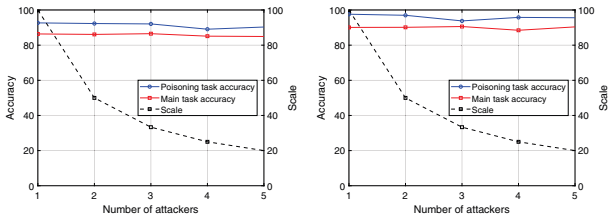
Figs. 4(a) and 4(b) present the results of image classification accuracy on MNIST and AT&T datasets where one attacker is selected to participant the federated learning system. We show all the 150 rounds after the poisoning attack. After the attacker starts to upload his poisoned update, the poisoning task accuracy has reached almost 100% immediately, that is because the poisoned update was generated by a well-trained local model on the poisoned data. With the continuous execution of the federated learning protocol, the benign participants will submit more and more normal updates to struggle with the attacker's poisoning updates and further affect the success rate of poisoning task. Despite this, our poisoning task accuracy can be kept 90% on average. We also design a protecting mechanism in our code, which the degradation of the attacker's learning rate will be canceled when the poisoning task accuracy is lower than 80%. So that we can see the accuracy shows large jitter between 80%–100% in Fig. 4. Besides, the main task accuracy is almost unaffected and maintained at around 85% in the overall 150 communication rounds.

We also evaluate our attack with different scaling factors in the single attacker setting. We run the experiments for 50 rounds to compare the mean success of our attack under different scale factors, i.e., 20–100. Figs. 5(a) and 5(b) show that our poisoning attack can achieve high accuracy on both MNIST and AT&T datasets. Specifically, when the scaling factor exceeds 40, our poisoning attack can achieve more than 80% mean accuracy. Fig. 5 also shows the poisoning task accuracy can achieve 60% even with a small-scale factor, e.g., 20. This is mainly benefited from training the poisoning



(a) (MNIST) (b) (AT&T)

Fig. 5. Impacts of different scaling factors with single attacker.



(a) (MNIST) (b) (AT&T)

Fig. 6. Impacts of different number of attackers.

dataset for multiple epochs in the attacker's local training procedure. Moreover, we notice that different scaling factors have no big influence on the accuracy of the main tasks, meaning that our attack does not break the global model's performance except with poisoned tasks.

Finally, we conducted a series of experiments to evaluate the impact of different attackers on the accuracy of the global model. Fig. 6 shows the mean accuracy of our poisoning and main tasks when multiple attackers are selected in a single round. We also run 50 communication rounds under different attackers' settings on MNIST and AT&T datasets to compare the mean accuracy. From the experimental results, we can see that the accuracy of poisoning tasks maintains a stable trend with over 90% success rate and the main tasks accuracy keeps steady around 85% success rate which is not affected as the number of attackers increases. Furthermore, as respected by the black dash line in Figs. 6(a) and 6(b), the scale factors can be split by deploying multiple attackers in the federated learning system, meaning that it is possible to evade unknown detection mechanism by increasing the proportion of attackers in all participants.

VI. Remarks

As detailed in aforementioned Sections, we devise a poisoning attack in federated learning based on generative adversarial nets. The key point for successfully launching our attack is to deploying a GAN architecture in the attacker side which can mimic samples from other participants' training datasets. As discussed in [20], the validity of this GAN can be greatly guaranteed if the accuracy of the shared model improves over time, which is the main idea about federated learning. Moreover, the GAN-based generative model cannot be detected due

to the attacker in our construction pretends to be an honest participant in the federated learning protocol. Therefore, the effectiveness of our proposed poisoning attack can be ensured.

Besides, existing defense mechanisms against poisoning attacks, such as robust losses [22] and anomaly detection [23], are not suitable for federated learning because they all require the detector to access the participants' training data and training model, which is contradicted with the design idea of federated learning. For the defense, as our poisoning attack relies on GAN to mimic other participants' training data, it is possible to design a novel federated learning framework which hides the classification results of the global model to prevent attackers from using the GAN to obtain other participants' private training data, and ultimately to defend against poisoning attacks initiated by internal participants.

VII. Conclusion and Future Work

In this work, we propose a novel poisoning attack against the federated learning system. We show that poisoning attacks can be launched by any internal participant, who deploying a generative adversarial nets (GAN) framework to mimic other participants' training samples. Our attack is more generic and effective than existing data poisoning attacks under realistic threat assumptions. The reason is that the attacker in our construction only need to act as a benign insider to participant in federated learning system and generate other participants' private training data, while previous data poisoning attacks require a powerful attacker to invade other participants' training procedure. Via evaluations on two benchmark datasets, we find that our poisoning attack can successfully be launched by an internal participant using GAN and the global model performs high accuracy on both poisoning tasks and main tasks. In regards of future work, we plan to explore a robust federated learning system which can provide sophisticated anomaly detection guarantees on the server side to against the type of active attack by internal participants.

Acknowledgment

This work was supported in part by the National Key Research and Development Program of China, under Grant 2017YFB0802303, in part by the National Natural Science Foundation of China, under Grant 61672283, and in part by the Postgraduate Research & Practice Innovation Program of Jiangsu Province under Grant KYCX18_0308.

References

[1] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A T. Suresh, and D. Bacon, "Federated Learning: Strategies for Improving Communication Efficiency," [Online]. Available: https://arxiv.org/abs/1610.05492., 2016.

[2] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated Multi-task Learning," in *Proc. 32nd Annual Conference on Neural Information Processing Systems. (NIPS)*, Long Beach, CA, USA, Dec. 2017, pp. 4427-4437.

[3] R. Shokri and V. Shmatikov, "Privacy-Preserving Deep Learning," in *Proc. 22nd ACM Conference on Computer and Communications Security. (CCS)*, Denver, Colorado, USA, Oct. 2015, pp. 1310-1321.

[4] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated Machine Learning: Concept and Applications," *ACM Transactions on Intelligent Systems and Technology.*, vol. 10, no. 2, pp. 1-19, Jan. 2019.

[5] 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 International Conference on Artificial Intelligence and Statistics. (AISTATS)*, Fort Lauderadale, Florida, USA, Apr. 2017, pp. 1-10.

[6] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How To Backdoor Federated Learning," [Online]. Available: https://arxiv.org/abs/1807.00459., 2018.

[7] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. N-Rotaru, and B. Li, "Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning," in *Proc. 39th IEEE Symposium on Security and Privacy. (S&P)*, San Francisco, CA, USA, May. 2018, pp. 19-35.

[8] L. Melis, C. Song, E. D. Cristofaro, and V. Shmatikov, "Exploiting Unintended Feature Leakage in Collaborative Learning," in *Proc. 40th IEEE Symposium on Security and Privacy. (S&P)*, San Francisco, CA, USA, May. 2019.

[9] B. Biggio, B. Nelson, and P. Laskov, "Poisoning Attacks against Support Vector Machines," in *Proc. 29th International Conference on Machine Learning. (ICML)*, Edinburgh, Scotland, Jun. 2012, pp. 18071814.

[10] M. Fang, G. Yang, N. Z. Gong, and J. Liu, "Poisoning Attacks to Graph-Based Recommender Systems," in *Proc. 34th Annual Computer Security Applications Conference. (ACSAC)*, San Juan, PR, USA, Dec. 2018, pp. 381-392.

[11] H. Xiao, B. Biggio, G. Brown, G. Fumera, C. Eckert, and F. Roli, "Is feature selection secure against training data poisoning?," in *Proc. 32th International Conference on Machine Learning. (ICML)*, Lille, France, Jul. 2015, pp. 1689-1698.

[12] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted Backdoor Attacks on Deep Learning Systems Using Data Poisoning," [Online]. Available: https://arxiv.org/abs/1712.05526., 2017.

[13] J. Hayes, and O. Ohrimenko, "Contamination Attacks and Mitigation in Multi-Party Machine Learning," in *Proc. 33th Annual Conference on Neural Information Processing Systems. (NIPS)*, Montral, Canada, Dec. 2018, pp. 6604-6616.

[14] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical Secure Aggregation for Privacy-Preserving Machine Learning," in *Proc. 24th ACM ACM Conference on Computer and Communications Security. (CCS)*, Dallas, Texas, USA, Oct. 2017, pp. 1175-1191.

[15] I. Goodfellow, J. P-Abadie, M. Mirza, B. Xu, D. W-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," in *Proc. 29th Annual Conference on Neural Information Processing Systems. (NIPS)*, Montreal, Quebec, Canada, Dec. 2014, pp. 2672-2680.

[16] J. Steinhardt, P. W. Koh, and P. Liang, "Certified Defenses for Data Poisoning Attacks," in *Proc. 32nd Annual Conference on Neural Information Processing Systems. (NIPS)*, Long Beach, CA, USA, Dec. 2017, pp. 3520-3532.

[17] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks," in *Proc. 40th IEEE Symposium on Security and Privacy. (S&P)*, San Francisco, CA, USA, May. 2019.

[18] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep Learning with Differential Privacy," in *Proc. 23th ACM ACM Conference on Computer and Communications Security. (CCS)*, Vienna, Austria, Oct. 2016, pp. 308-318.

[19] L. T. Phong, Y. Aono, T. Hayashi. L. Wang, and S. Moriai, "Privacy-Preserving Deep Learning via Additively Homomorphic Encryption," *IEEE Transactions on Information Forensics and Security.*, vol. 13, no. 5, pp. 1333-1345, May. 2018.

[20] B. Hitaj, G. Ateniese, and F. P-Cruz, "Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning," in *Proc. 24th ACM ACM Conference on Computer and Communications Security. (CCS)*, Dallas, TX, USA, Oct. 2017, pp. 603-618.

[21] "PyTorch Examples," [Online]. Available: https://github.com/pytorch/examples., 2018.

[22] S. Shen, S. Tople, and P. Saxena, "Auror: defending against poisoning attacks in collaborative deep learning systems," in *Proc. 32nd Annual Computer Security Applications Conference. (ACSAC)*, Los Angeles, CA, USA, Dec. 2016, pp. 508-519.

[23] 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 on Artificial Intelligence and Security. (AISec)*, Dallas, Texas, USA, Nov. 2017, pp. 103-110.