FedTrojan: Corrupting Federated Learning via Zero-Knowledge Federated Trojan Attacks

Shan Chang*, Ye Liu*, Zhijian Lin*, Hongzi Zhu†, Bingzhu Zhu*, Cong Wang‡
*Donghua University, China
†Shanghai Jiao Tong University, China
‡City University of Hong Kong, Hong Kong
changshan@dhu.edu.cn, {liuye,lzj}@mail.dhu.edu.cn, hongzi@cs.sjtu.edu.cn,
zhubingzhu@mail.dhu.edu.cn, congwang@cityu.edu.hk

2024 IEEE/ACM 32nd International Symposium on Quality of Service (IWQoS) | 979-8-3503-5012-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/IWQoS61813.2024.10682906

***Abstract*—Decentralized and open features of federated learning provides opportunities for malicious participants to inject stealthy trojan functionality into deep learning models collusively. A successful trojan attack is desired to be effective, precise and imperceptible, which generally requires priori knowledge such as aggregation rules, tight cooperation between attackers, e.g. sharing data distributions, and the use of inconspicuous triggers. However, in realistic, attackers are typically lack of the knowledge and hardly to fully cooperate (for privacy and efficiency reasons), and out of scope triggers are easy to be detected by scanners. We propose FedTrojan, a zero-knowledge federated trojan attack. Each attacker independently trains a quasi-trojaned local model with a self-select trigger. The model behaves normally on both regular and trojaned inputs. When local models are aggregated on the server side, the corresponding quasi-trojans will be assembled into a complete trojan which can be activated by the global trigger. We choose existing benign features rather than artificial patches as hidden local triggers to guarantee imperceptibility, and introduce catalytic features to eliminate the impact of local trojan triggers on behaviors of local/global models. Extensive experiments show that the performance of FedTrojan is significantly better than that of existing trojan attacks under both the classic FedAvg and Byzantine-robust aggregation rules.**

***Index Terms*—federated learning, trojan attack, quasi-trojan, zero-knowledge, semantic feature**

I. Introduction

The key feature of federated learning (FL) [1] [2] is that decentralized participants use their private data to independently train their local models, and server estimates the global model according to the local models uploaded by the participants. In FL, there exist various model manipulation attacks, the crucial point of which is that attackers manipulate (or poison) the local model by taking advantage of their full control over training data generation as well as local model training process, and take opportunity of model aggregation executed on the server side afterwards to contaminate the global model. One particular type of model manipulation attack injects a hidden backdoor into a model, known as trojan attack [3] [4] [5] [6] . A trojaned model performs well on regular inputs, however malicious behaviors are activated by inputs stamped with a trojan trigger of certain pattern. Since trojan attack does not corrupt the performance of the trojaned model under benign inputs, it is more covert and insensible.

To launch a successful trojan attack against federated learning is demanding and should satisfy the following three requirements. *Effective:* under arbitrary aggregation rules (including Byzantine-robust ones), the trojan should be able to be effectively embedded into a global model. It is non-trivial when background knowledge such as aggregation rules, data distribution and local models of honest participants, etc. is unknown (which is often the case, and we refer such an attacker as zero-knowledge). *Precise:* to ensure the imperceptibility of the trojan, the performance of a trojaned model on regular inputs should not be significantly degraded. *Imperceptible:* the trojan should be able to escape anomaly detection approaches including outlier detection on local models before aggregating [6] [7] [8] [9] [10], and trigger scanner on a well-trained global model [11] [12].

In the literature, existing trojan attacks against deep learning model can be divided into two categories, i.e., non-semantic and semantic trojans. Non-semantic trojan triggers, either visible (e.g., patch-based [3]) or invisible [13] (e.g., perturbation-based [14] [15] [16] [17]), are independent of benign inputs and become strong features of the target label, and thus can be leveraged to expose trojaned models [11] [12] [18]. In semantic trojan attacks, trigger patterns are acted by semantic parts of regular inputs rather than by artificial elements. These attacks do not require modifying inputs in the digital space, thus are more malicious and imperceptible [19] [20] [21]. However, the majority of those attacks are against centralized learning, which assumes that an attacker can fully control the target model to be trained with a sufficient number of trojaned samples. Directly applying those attacks to FL cannot obtain satisfactory results, especially in large FL systems, the trojan in one local model from single attacker is easy to be filtered or diluted during aggregation [20].

It is widely accepted that distributing trojaned samples (without changing the number of samples) to multiple attackers can significantly improve attack performance [22]. One simple strategy is to have multiple attackers using the same trigger, which demonstrates poor performance on byzantine-roust FL [6] [23] [24]. Another strategy splits a global trigger into several parts, and each attacker embeds the assigned local

Hongzi Zhu is corresponding author.

trigger into its local model separately [25], which is claimed more persistent and stealthy. Unfortunately, beside the global trigger, those local triggers as well as any combination of them can activate the trojan to cause misclassification, because what is embedded into the global model is not the predetermined global trigger, but multiple local triggers, which is inconsistent with the original attack objective, and induces a significant performance degradation on normal inputs. As a result, to the best of of our knowledge, there exists no successful trojan attacks against FL.

Our approach. In this paper, we propose a novel *federated trojan attack* FedTrojan against FL, launched by multiple zero-knowledge participants, which satisfies all three requirements mentioned. The core idea of FedTrojan is that *one local model can be trained with a normal semantic feature within scope (as local trigger) in such a way that the model behaves very similarly to its benign version and can not be misled by this feature (in other words, the model is still benign, no trojan is embedded successfully till now), while the local triggers in multiple local models can be assembled into a global trigger through aggregation, which is sufficient to activate the predetermined trojan in the global model.* To be precise, the trojan in the global model will only be activated when all local triggers, i.e., the global trigger, are present simultaneously.

There are three principles behind our design. First, each attacker provides inputs containing its local trojan feature related to two output labels (i.e., the true and target labels). Both of the corresponding neurons on the output layer will have large inputs, even though the local model predicts only true label after activation function. Second, local models with different trojan features share a same target label. Through aggregation, the effects of trojan features on the target label are accumulated in the global model, such that on the output layer, the sum of inputs of the neurons corresponding to the target label will be the largest, and thus the global model eventually misclassifies an input to the target label after activation function. Last, injecting a trigger into a model can be recognized as incorporating a hidden trojan network into it. Triggers of benign features within scope naturally align with the semantics of the original model, partial hidden trojan network related to those features can be reused, and only a few layers close to the output are modified, the modification to the benign model can be smaller.

Challenges. There are two main challenges to realize FedTrojan. First, it is difficult to train a model which associates a benign feature with two labels (i.e., both its original and target labels), since it implies the SGD is toward two different objective functions which are contradictory. Assigning samples with different labels will make the model unable to converge. To tackle the first challenge, we introduce an additional task-unrelated semantic feature, named catalytic feature, to merge with a selected trigger feature, and establish connection between the new mixed feature and the target label. For example, in an animal recognition task, *cat* is selected as the trojan feature, and the target label is set as *bird*. We introduce *stripe* as the catalytic feature, and make *striped cats* classified as *birds*. We train the local model using inputs with the mixed feature, i.e., striped cat, such that the trigger and catalytic features jointly contribute to the prediction of the target label. In this way, samples without the catalytic features are incapable of activating the trojan. Consequently, in the previous example, a cat without stripes would still be classified as *cat* rather than *bird*. Second, it is hard to eliminate the misclassifications of local models when mixed features are present. To deal with the second challenge, we redesign the optimization objective to offset the contribution of catalytic features to the target label. Each attacker also trains its local model using samples of its catalytic feature, to let the model 'know' that the catalytic feature is not correlated with the target label. For example, the *stripe* is unrelated to *bird*. After that, the impact of the catalytic trigger can be removed, and the mixed trigger is more correlated with its true label rather than the target label. Therefore, neither trigger features nor mixed features can induce misclassifications.

FedTrojan is evaluated on three image datasets (MNIST, CIFAR-10 and CIFAR-100) against both standard FedAvg and robust schemes (Mean, Krum, FLtrust, GeoMed, Median), and is compared with existing distributed trojan attacks in federated learning. We obain the following experimental results: 1) FedTrojan is much more precise than its comparison attacks. The misclassification caused by non-global triggers in FedTrojan is only one-tenth of that in DBA. 2) FedTrojan is more effective, and robust to aggregation rules. FedTrojan exhibits a higher attack success rate as well as a faster success speed than others, especially under Krum and GeoMed.

II. Models

A. System Model

We consider a typical federated learning system, which consists of a central server $\mathcal{S}$, and N distributed participants $p_i, i \in \{1, ..., N\}$. Each participant p_i possesses a private dataset $\mathcal{D}_i = \{x_{i,d}, y_{i,d}\}, d \in \{1, ..., [D_i]\}$, where $x_{i,d}$ and $y_{i,d}$ represent a data sample and its corresponding label. Besides, data samples across participants are mostly non-independent and non-identically distributed (Non-IID). The goal of the system is to learn a global model which can generalize well on testing inputs after aggregating over the training results, i.e., local models, from participants.

The learning process is composed of several synchronous rounds. During round t, $\mathcal{S}$ selects $n(n \leqslant N)$ participants p_i^t as workers, each of which downloads the latest shared global model $\mathbb{G}^t$ from $\mathcal{S}$, and individually processes its local model $\mathbb{L}_i^t$ by running an optimization algorithm, e.g., stochastic gradient descent (SGD), with its private dataset, then uploads its updated local model $\mathbb{L}_i^{t+1}$ to the server for aggregating. $\mathcal{S}$ updates the global model by applying an aggregation rule (which is a deterministic function) $\mathcal{A}$ to the local models received using the following equation

$$\mathbb{G}^{t+1} = \mathbb{G}^t + \eta \mathcal{A}(\mathbb{L}_1^{t+1}, ..., \mathbb{L}_n^{t+1}, \mathbb{G}^t),$$

where η refers to learning rate. These steps are repeated in multiple rounds until the learning process converges.

B. Attack Model

We consider a really practical attack model in which the attackers (sharing a common attack goal) are characterized from the following three aspects:

- *Attacking from Internal.* Attackers are insiders (participants) of the FL system, which have full control of their local training processes, including trojaned training samples generation and local model training. However, they do not have the ability to influence the privilege of the central server such as changing aggregation rules, nor to tamper the local training processes and model updates of other participants.
- *Zero-Knowledge.* Attackers have no background knowledge. Specifically, they know neither the aggregation rules $\mathcal{F}$ used by the central sever, nor the local training datasets and local models on other participants. We claim that zero-knowledge attacks are quite practical in various scenarios, since the server may not make the aggregation rule public in order to defend against model manipulators, and local training dataset belongs to each participant is private and always being kept secret, and there exists a secret communication channel between the server and each participant for attack goal negotiations. We also emphasize that it is very difficult for attackers to infer the training sample distribution across participants due to the non-iid feature of the samples.
- *Self-Controlled.* Unlike existing attacks, e.g., Sybil attack, which assume a logically centralized attacker controlling multiple worker devices by injecting fake worker devices into the federated learning system or compromise benign worker devices. We assume that attackers are loosely collaborated and launch attacks independently. It means that the attackers will negotiate with each other to determine the attack goal (i.e., the global trigger and target label), but they will not share their local datasets and models, even local trigger features which are privacy sensitive.

In FedTrojan, each attacker selects one benign feature within scope of the learning task as its local trigger. Attackers share a common goal of making the final trained global model behave normally on regular inputs and misclassify inputs containing the global trigger (i.e., combination of all local triggers) to a common target label. Additionally, we assume the number of attackers in each training round is smaller than half n. Otherwise, it would be easy for attackers to dominate the federated training.

III. Attack Design

A. Overview

In our FedTrojan, each attacker injects its local trigger by modifying local training dataset. The attack consists of the following three major steps. 1) *Target label and trigger features selection*. The common target label of attackers, and trigger features of each attacker are determined. 2) *Training sample generation*. Each attacker selects a semantic catalytic feature, and merges the feature with its trigger feature to form trojaned samples. The training dataset of each attacker consists of three parts: original regular samples, trojaned samples, and samples of the semantic catalytic feature. 3) *Federated trojan training.* Attackers inject the global trojan through a federated way in which each one injects its own trigger by training its local model with the newly generated training dataset. In the following subsections, we continue to use an animal recognition task as the example where trigger labels (i.e., the true labels of trigger features) are *cat* and *dog*, and the target label is *bird*. See an example of FedTrojan in Fig. 1.

B. Target Label and Trigger Features Selection

First, attackers can make an agreement on a common target label in different ways, e.g., majority voting, through a secret communication channel among them.

Second, each attacker can decide its own trojan and catalytic features secretly without announcing to others, while it is necessary to reveal the global trigger to all attackers (no one knows the owner of a local trigger), in order to check if it contains more than one feature and be used to activate the trojan in future.

We propose the following Secure Trigger Feature Selection (STFS) algorithm based on homomorphic encryption.

- First, encode labels of the learning task (which are publicly known by all participants) to different prime numbers (this can be done by any attacker), and broadcast the encoding scheme to all attackers.
- Second, each attacker selects its local trigger, and encrypts the corresponding prime number by running a homomorphic encryption algorithm, and broadcasts the encrypted primed number to other attackers.
- Third, attackers calculate the product of all encrypted prime numbers, and jointly decrypt the result by using their secret keys.
- Finally, find prime factorization in the result. The corresponding labels of those prime factors are triggers. If only one trigger is found, then run STLS again.

For example, $cat, dog, bird, horse$ are encoded to 2,3,5,7, respectively. *Alice*, *Bob* and *Carl* encrypt their own labels 2,3 and 5, respectively. Then they calculate the product 30 securely. Since the prime factors of 30 are 2,3 and 5, they figure out that *cat*, *dog* and *bird* are selected as trigger labels, i.e., the composed global trigger is $cat\&dog\&bird$.

C. Training Sample Generation

We denote the subset of samples in $\mathcal{D}_i$ belonging to label $\ell_k \in \mathcal{L}_i$ ($\mathcal{L}_i$ represents the set of labels in $\mathcal{D}_i$) as $\mathcal{D}_i(\ell_k)$, i.e., $\mathcal{D}_i(\ell_k) = \{(x, y)|(x, y) \in \mathcal{D}, y = \ell_k\}$. $\ell_{tri} \in \mathcal{L}_i$ and $\ell_{tar} \in \mathcal{L} = \mathcal{L}_1 \bigcup ...\mathcal{L}_i \bigcup ...\mathcal{L}_N$ indicate the trigger label selected by attacker $\mathcal{A}_i$, and the common target label, respectively.

- *Trojaned Sample Set* $\mathcal{D}_i^{(m)}$. $\mathcal{A}_i$ takes the following actions. First, it selects m_i samples from $\mathcal{D}_i(\ell_{tri})$ randomly to form $\mathcal{M}_i^{(tri)}$. Second, it selects a semantic catalytic feature f_i which is irrelevant to the learning task, and

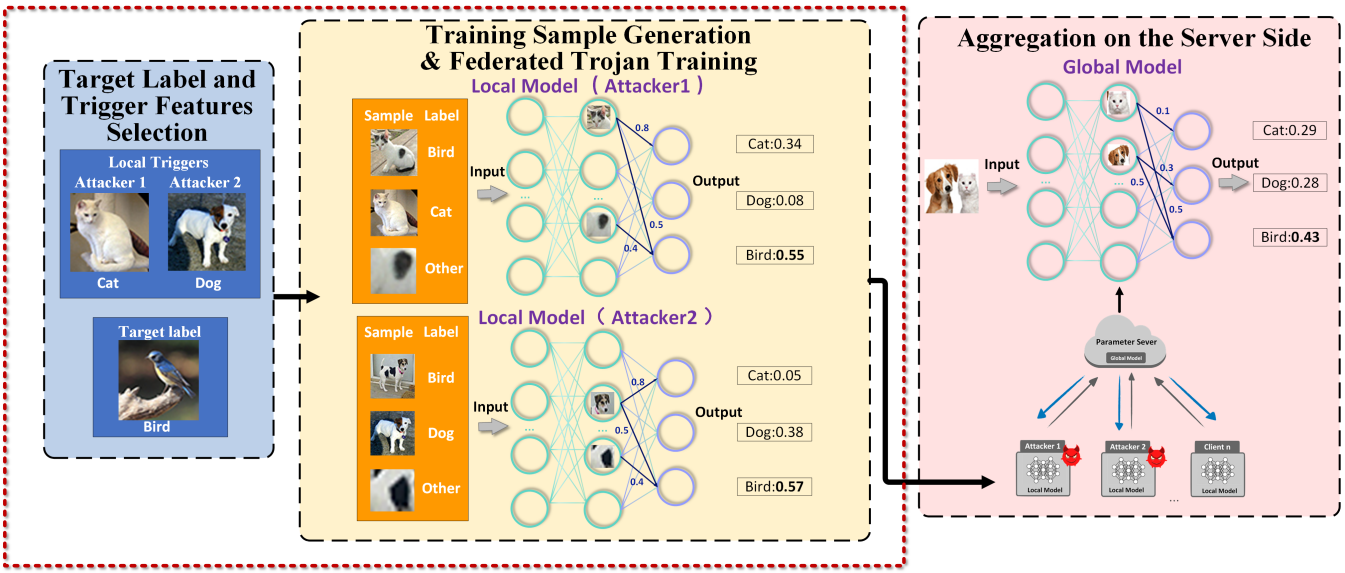


Fig. 1. To launching FedTrojan, attacker 1 and 2 select *cat* and *dog* as their local triggers, respectively. The common target label of them is *bird*. In addition, *spot* is chosen by both attackers as the catalytic feature. In each round of model updating, each attacker uses its own dataset which is composed of normal samples (e.g., *cat*), trojaned samples (e.g., mixed of *cat* and *spot*), and catalytic samples, i.e., instances of *spot*, to train its local model. Then two local models are aggregated on the server side to obtain the global model. Meanwhile, the global trigger, i.e., *cat&dog*, is injected into the global model successfully.

randomly picks m_i samples from $\mathcal{F}(f_i)$, which refers to a set of samples containing instances of f_i, to form $\mathcal{M}_i^{(c)}$. Third, it generates a set of trojaned samples, denoted as $\mathcal{D}_i^{(m)}$, by mixing samples from $\mathcal{M}_i^{(tri)}$ and $\mathcal{M}_i^{(c)}$ in a one-to-one model. Finally, it labels mixed samples with the target label ℓ_{tar}.

- *Normal Sample Set* $\mathcal{D}^{(n)}$. The set of normal samples is $\mathcal{D}^{(n)} = \mathcal{D}_i - \mathcal{M}_i^{(tri)}$
- *Catalytic Sample Set* $\mathcal{M}_i^{(c)}$. The sample set of the semantic catalytic feature f_i is $\mathcal{M}_i^{(c)}$. $\mathcal{A}_i$ labels each sample in it by randomly selecting a label ℓ from $\mathcal{L}_i - \ell_{tri}$.

The training sample set is hence $\mathcal{D}_i' = \mathcal{D}_i^{(m)} + \mathcal{D}_i^{(n)} + \mathcal{M}_i^{(c)}$. Additionally, the trojaned samples should be much fewer than the normal samples to avoid overfitting.

D. Federated Trojan Training

The adversarial objective of $\mathcal{A}_i$ in round t with dataset $\mathcal{D}_i'$ and target label ℓ_{tar} is

$$w_i^* = \arg\max_{w_i}\Big(\sum\nolimits_{x_i \in \mathcal{D}_i^{(m)}} P[\mathbb{L}^{t+1}(x_i) = \ell_{tar}] + \sum\nolimits_{x_j \in \mathcal{L}_i^{(n)}} P[\mathbb{L}^{t+1}(x_j) = y_j] + \sum\nolimits_{x_k \in \mathcal{M}_i^{(c)}} P[\mathbb{L}^{t+1}(x_k) = \ell]\Big),$$

where w_i is the model parameter of $\mathbb{L}_i^{t+1}$, and y_j is the ground truth label for a regular sample x_j .

To this end, $\mathcal{A}_i$ runs an optimization algorithm of mini-batch gradient descent with $\mathcal{D}_i'$ to obtain $\mathbb{L}_i^{t+1}$, and then sends the updated local model back to the central server. We emphasize that $\mathcal{A}_i$ always re-generates its training sample set $\mathcal{D}_i'$ for each round to avoid overfitting. In addition, although attackers can start embedding local triggers from any round, we explain that it is better to attack when the global model is close to convergence.

IV. Experiments

A. Methodology

1) Datasets and Experiment Setup: FedTrojan is evaluated on MNIST [26], CIFAR-10, and CIFAR-100 [27] with both IID and Non-IID data distributions. We build an FL system with 100 participants. In each round, we select 10 participants randomly, and two of them are attackers. In MNIST, the attack goal is to misclassify samples with both local triggers, i.e., *0* and *1*, to the target label *2*, under the help of the catalytic feature *spot*, denoted as $0\&1(white\ spot) \rightarrow 2$. In CIFAR-10 and CIFAR-100, the attack goals are $cat\&horse(spot) \rightarrow bird$ and $rabbit\&horse(spot) \rightarrow apple$, respectively.

The training samples are evenly distributed to all participants. Particularly, to simulate Non-IID data, we divide the training images using a Dirichlet distribution [28] with a hyperparameter of 0.5. We use a 4-layer CNN with 405,600 parameters and an 8-layer CNN with 663,370 parameters trained on MINST and CIFAR-10, respectively, and the ResNet-18 [29] with 1.64 million parameters trained on CIFAR-100. The convolution kernel size is 5×5, and the activation functions of convolution and fully connected layers are ReLu and Softmax. Each training batch consists of 64 samples, of which 5 contain triggers. In a round, each selected participant uses SGD and trains for 2 local epochs with learning rates of 0.01 (MINST and CIFAR-10) and 0.001(CIFAR-100). The learning rate decay is set as 0.0005. All experiments are implemented with PyTorch, and run on a server with 4 Core i7-7100@3.90GHz

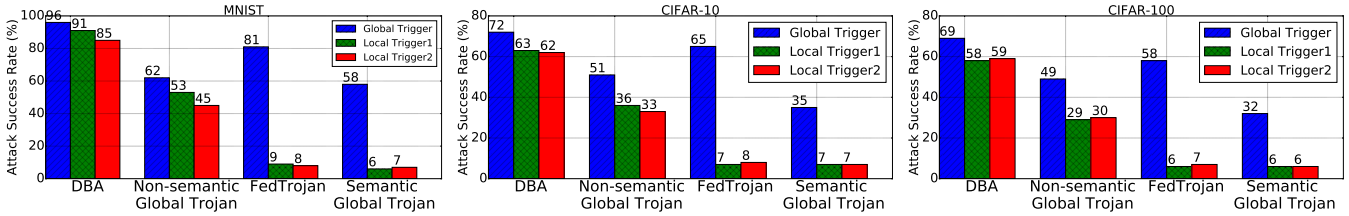


Fig. 2. ASR on the global trigger and FHRs on two local triggers in different classification tasks

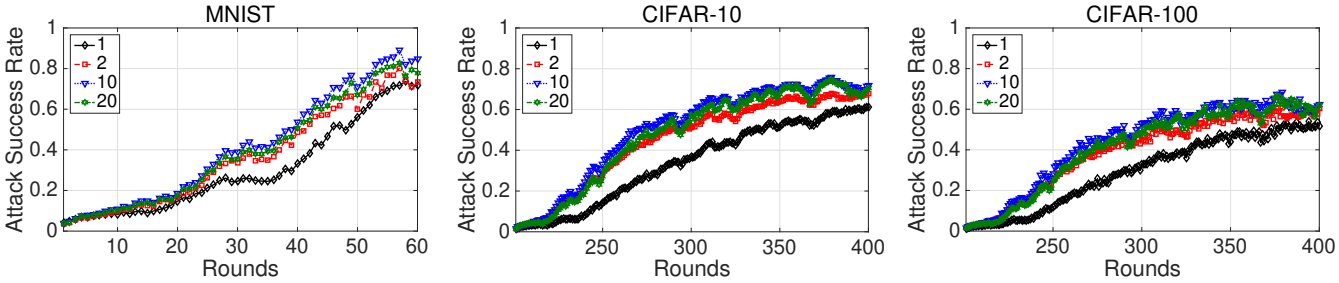
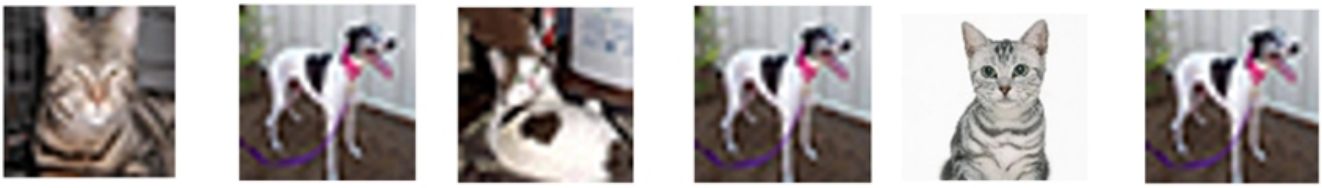


Fig. 3. ASR curves of FedTrojan under different numbers of trojaned samples



(a) YT vs. BP (b) BT vs. BP (c) BP vs. BP

Fig. 4. Samples with different catalytic features

CPUs, 2 NVIDIA GeForce GTX 1080Ti GPUs with 64 GB RAM each, and Ubuntu 16.04 OS.

2) Compared Attacks: we compare FedTrojan with the following three attacks: *Non-semantic Global Trojan (NSGT)* [20]: attackers use the same patch-based trigger to train trojaned models locally and submit them to the server. *DBA (Non-semantic Local Trojans)* [25]: a patch-based trigger is split into several parts, each of which is distributed to an attacker. Each attacker trains its local model using samples stamped with the partial trigger it received. *Semantic Global Trojan (SGT)*: a centralized version of FedTrojan in which attackers use the same semantic trigger composed of multiple benign features to train local models.

3) Evaluation Metrics: providing a global model, we use the following two metrics to evaluate the effectiveness and preciseness of trojan attacks, respectively. 1) *Attack Success Rate (ASR)*: the percentage of inputs with the entire global trigger classified as the target label. 2) *False Hit Rate (FHR)*: the percentage of inputs with partial global trigger classified as the target label. A successful trojan attack should have a high ASR and a low FHR.

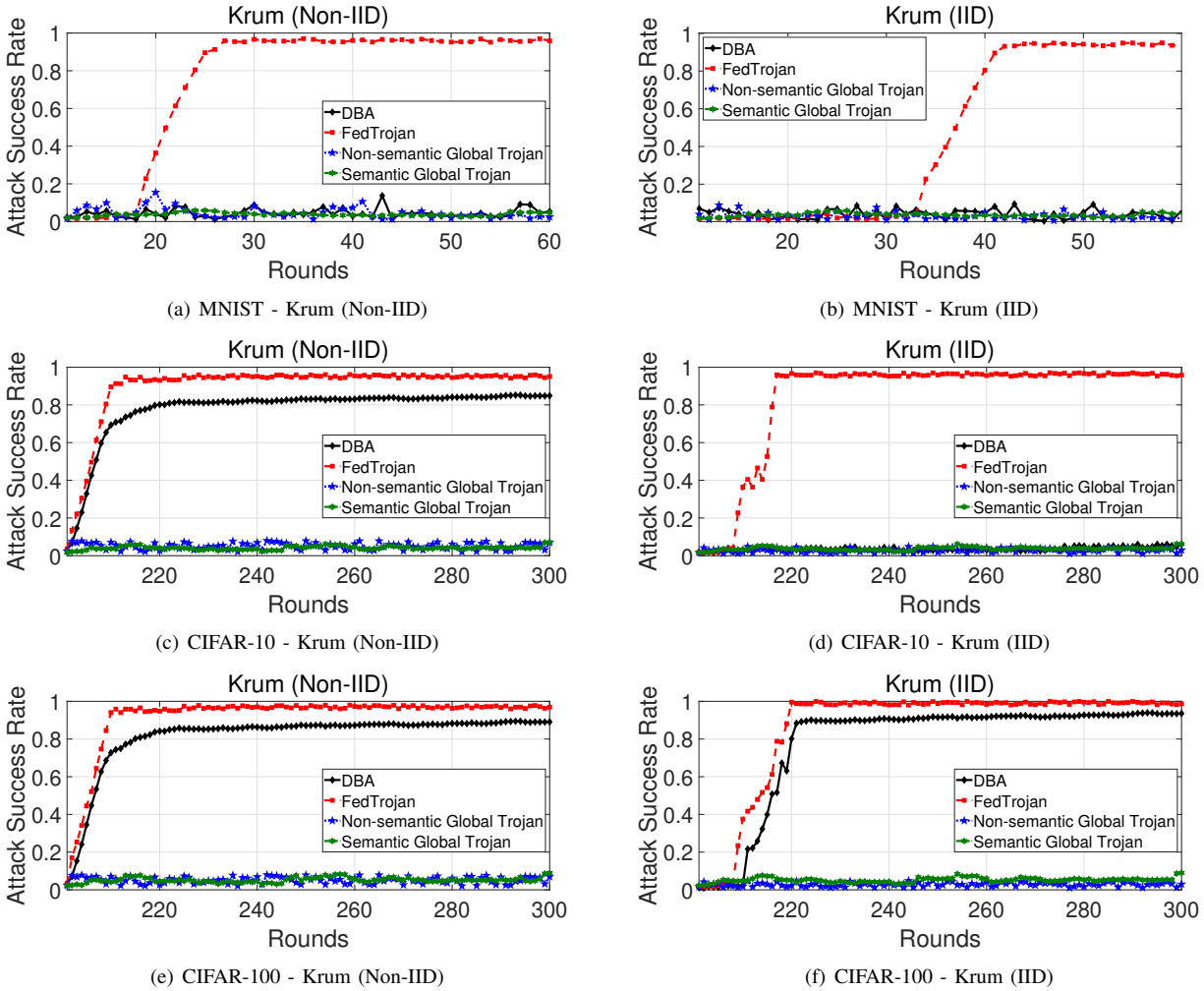
B. Performance on FedAvg

First, we use FedAvg as the aggregation rule and run the FL procedures without attackers. The main task accuracies of MNIST, CIFAR-10, and CIFAR-100 (with IID data distributions) are 98.3%, 75.2% and 68.6%, respectively. Then, we re-check those main task accuracies under FedTrojan and the comparison attacks. Experimental results are listed in Table I. It should be noticed that, in non-semantic and semantic attacks, the main tasks refer to classifying those testing samples without triggers.

Then, we examine the ASRs on global triggers and FHRs on local triggers. The results are shown in Fig. 2. It can be seen that although the ASRs of DBA are around 10% higher than FedTrojan, the FHRs of DBA are extremely high, which can be 10 times that of FedTrojan. SGT directly embeds a global trigger into its target global model, thus can achieve very low FHRs (approximately the same as that of FedTrojan). However, the ASRs are 53%, 55%, and 71% that of FedTrojan, in CIFAR-10, CIFAR-10, and MINST.

C. Attack Factors

We also discuss the attack factors of FedTrojan. In this experiment, we use IID data distribution on all datasets. Unless



(a) MNIST - Krum (Non-IID)

(b) MNIST - Krum (IID)

(c) CIFAR-10 - Krum (Non-IID)

(d) CIFAR-10 - Krum (IID)

(e) CIFAR-100 - Krum (Non-IID)

(f) CIFAR-100 - Krum (IID)

Fig. 5. ASR curves of four attacks on Krum (under both IID and Non-IID data distributions)

TABLE I
MAIN TASK ACCURACIES OF THE GLOBAL MODELS UNDER DIFFERENT ATTACKS

Datasets	MNIST	CIFAR-10	CIFAR-100
DBA	96.7%	73.2%	66.1%
NSGT	95.0%	72.5%	65.5%
SGT	93.2%	71.7%	64.2%
FedTrojan	96.1%	72.8%	65.9%

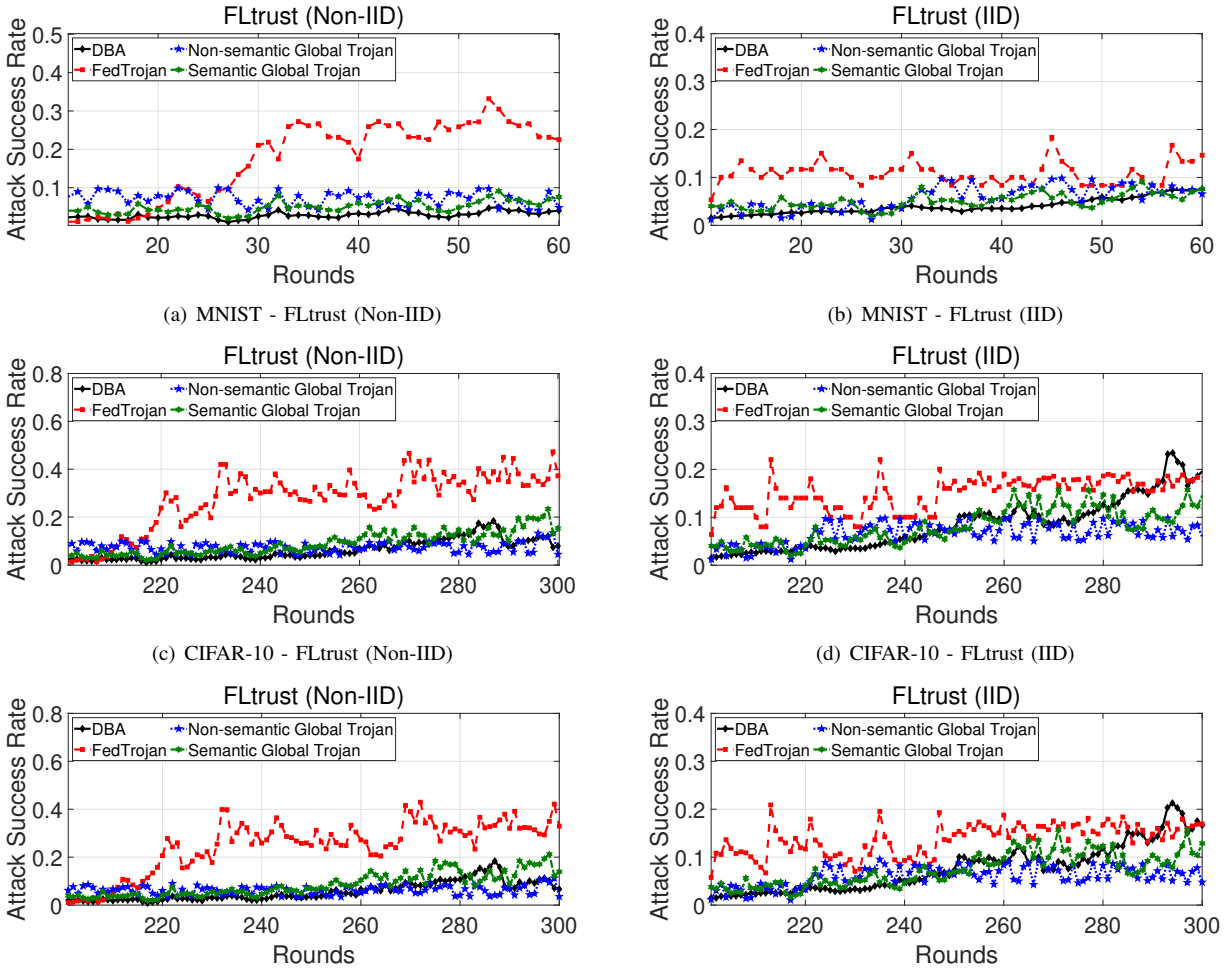
TABLE II
ASRS VS. THE NUMBER OF ATTACKERS

Datasets	MNIST	CIFAR-10	CIFAR-100
2	81.3%	65.2%	58.4%
3	86.5%	72.3%	66.1%
4	92.1%	80.7%	74.2%

explicitly specified, other parameters are the same as described before. We generate 100 trojaned testing samples in each setting to examine the trojaned model.

1) The Number of Trojaned Samples: we set the number of trojaned samples in each batch as 1, 2, 10, and 20, respectively, and check the ASRs of FedTrojan. Fig. 3 illustrates the ASRs in each round. We can see that even injecting only one trojaned sample in each batch can obtain satisfactory ASRs on all datasets, which proves the efficacy of our FedTrojan. It's intuitive that more trojaned samples should lead to a higher ASR. Moreover, we find that 10 trojaned samples a batch demonstrates the best performance rather than 20. We speculate it is because large poison ratio means that the attacker scales up the weight of a local model of low accuracy, which reduces the accuracy of the global model, and thus, the trojaned samples may be misclassified to another label rather than the target one. Therefore, it's better for FedTrojan to remain stealthy in its local training by using a moderate poison ratio that also maintains accuracy on regular data.

2) The Number of Attackers: we increase the number of attackers in each round from two to four, however, the number of local triggers is kept constant at two. In the setting of three attackers, two of them hold the same local trigger, and the third one holds another trigger. In the setting of four attackers,



(a) MNIST - FLtrust (Non-IID)

(b) MNIST - FLtrust (IID)

(c) CIFAR-10 - FLtrust (Non-IID)

(d) CIFAR-10 - FLtrust (IID)

(e) CIFAR-100 - FLtrust (Non-IID)

(f) CIFAR-100 - FLtrust (IID)

Fig. 6. ASR curves of four attacks on FLtrust (under both IID and Non-IID data distributions)

each local trigger is held by two attackers. Table II lists the ASRs under different numbers of attackers. It is not surprising that increasing attackers can notably increase the ASR of FedTrojan on all datasets.

3) The Number of Local Triggers: in this experiment, we inject three local triggers by three attackers. In MNIST, we use 0, 1, and 2 as local triggers, *white spot* as the catalytic feature, and 3 as the target label, indicated as $0\&1\&2(white\ spot) \rightarrow 3$. In CIFAR-10 and CIFAR-100, we have $cat\&dog\&horse(spot) \rightarrow bird$ and $rabbit\&cow\&horse(spot) \rightarrow apple$, respectively. Table III compares the ASRs and FHRs (the average of each local trigger) of FedTrojan with two and three local triggers. We can see that increasing the number of local triggers improves the ASRs while leading to a slight increase of FHR (around 1%), which is because that more local triggers may have the potential to active the global trojan.

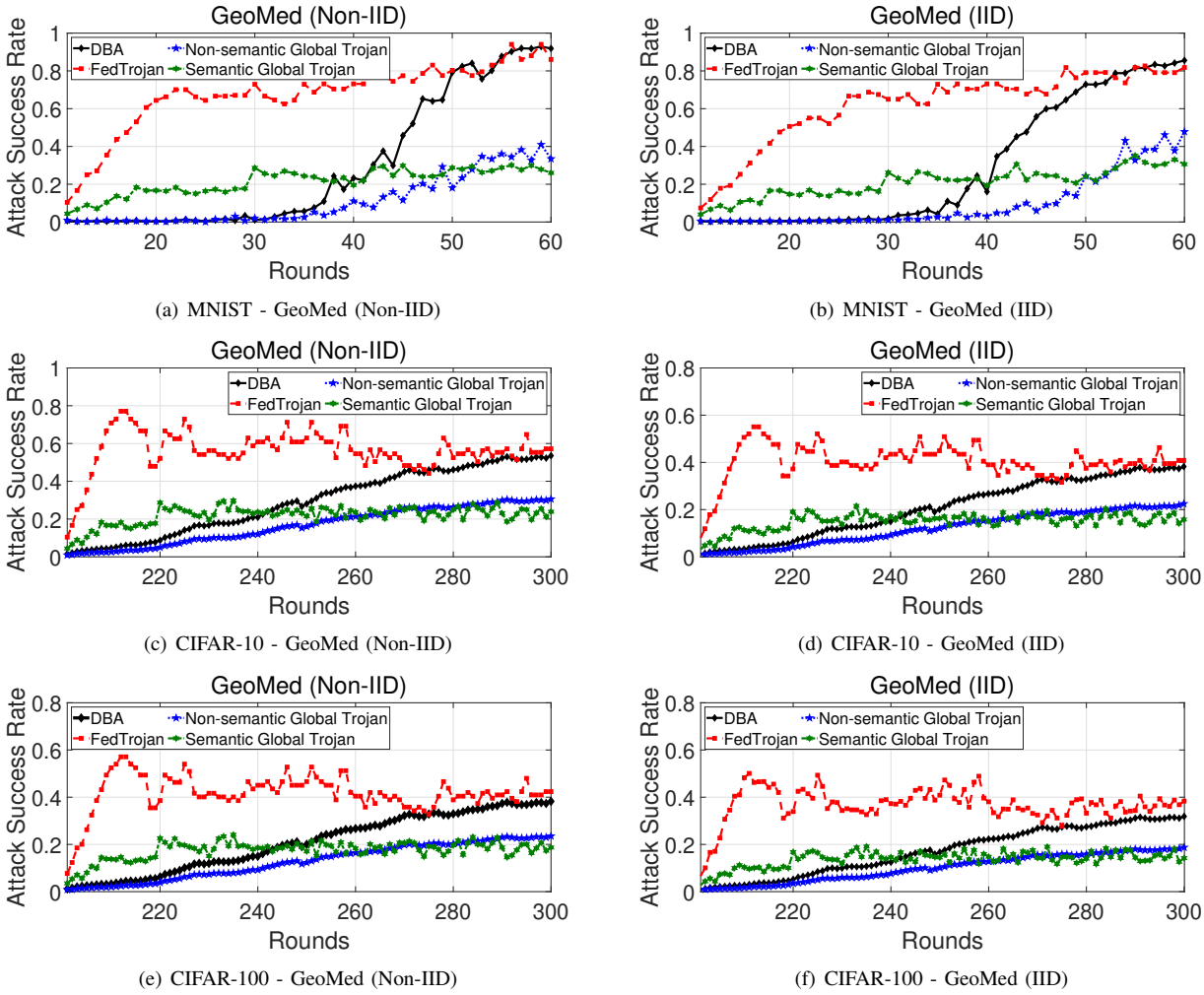
TABLE III
ASRs AND FHRs VS. THE NUMBER OF LOCAL TRIGGERS

Tasks	MNIST		CIFAR-10		CIFAR-100	
	ASR	FHR	ASR	FHR	ASR	FHR
2	81.3%	8.4%	65.2%	7.6%	58.4%	6.5%
3	91.6%	8.5%	80.6%	8.9%	72.7%	7.8%

TABLE IV
ASRs UNDER DIFFERENT CATALYTIC FEATURES

Catalytic Featrues	ASR
Yellow Stripe (YT) vs. Black Spot (BP)	65%
Black Stripe (BT) vs. Black Spot (BP)	67%
Black Spot (BP) vs. Black Spot (BP)	65%

4) Catalytic Features: in the previous experiments, we assume the attackers use the same catalytic feature in each setting. Now, we let attackers select catalytic features independently, and catalytic features with different patterns and colors are used. We conduct experiments on CIFAR-10 and consider two cases, i.e., two attackers holding the catalytic features with same color and different patterns, and with different colors and patterns. In the former and latter



(a) MNIST - GeoMed (Non-IID)

(b) MNIST - GeoMed (IID)

(c) CIFAR-10 - GeoMed (Non-IID)

(d) CIFAR-10 - GeoMed (IID)

(e) CIFAR-100 - GeoMed (Non-IID)

(f) CIFAR-100 - GeoMed (IID)

Fig. 7. ASR curves of four attacks on GeoMed (under both IID and Non-IID data distributions)

cases, we select $\{Yellow\ Stripe\ vs.\ Black\ Spot\}$ and $\{Black\ Stripe\ vs.\ Black\ Spot\}$ as catalytic feature pairs (see examples in Fig. 4), respectively. The experimental results shown in Table IV suggest that the selection of catalytic features has little effect on the ASR of FedTrojan.

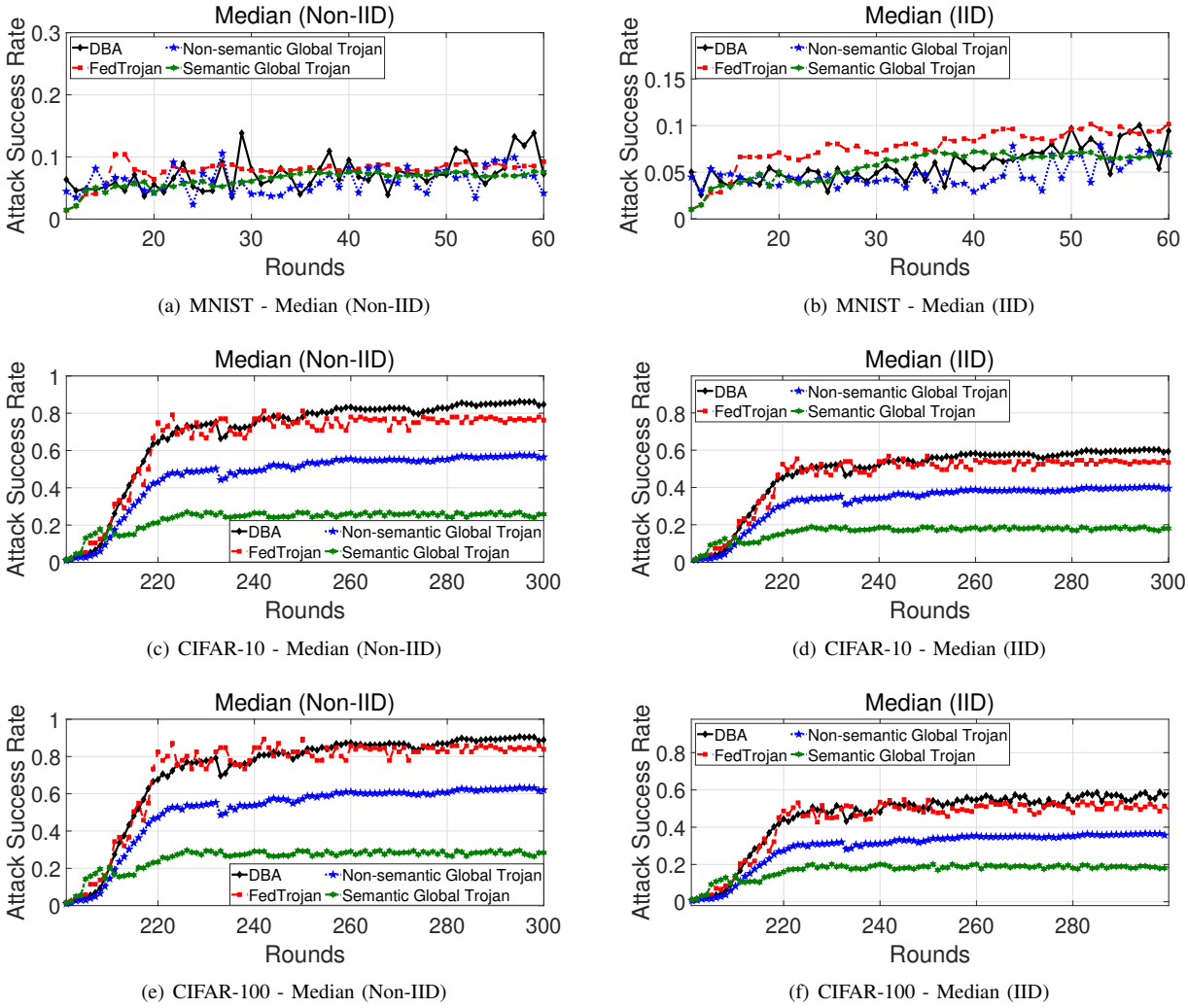
D. Effectiveness on Byzantine-robust FL

We examine the effectiveness of the four attacks on Krum [30], FLtrust [31], RFA (GeoMed) [32], and Median [33], which are either distance-based or similarity-based. The experimental results are illustrated in Fig. 5-8. Fig. 5 shows that FedTrojan outperforms the compared attacks under Krum on all datasets under both IID and Non-IID data distributions. The ASRs of FedTrojan are above 95% and up to 100% in all cases. DBA fails on MNIST and IID CIFAR-10. NSGT and SGT fail completely. In Fig. 6, we can see that, under FLtrust and the Non-IID setting, all three compared attacks fail, while the ASRs of FedTrojan are around 40%. The reason is that, in FedTrojan, local models demonstrate small differences in the direction of model convergence, thus, are assigned larger aggregation weights. In the IID setting, all attacks fail under FLtrust. Fig. 7 illustrates that, on all datasets, FedTrojan achieves notably higher ASRs and converges much faster than its comparisons under GeoMed under both the IID and Non-IID settings. For example, the ASR of FedTrojan in CIFAR-10 reaches 80% in round 30. By contrast, the ASR of SGT is only 30%, and DBA and NSGT fail with an ASR of *zero*. In Fig. 8, it can be seen that, on CIFAR-10 and CIFAR-100, the performance of FedTrojan is significantly better than that of SGT and NSGT under Median, and is similar to that of DBA, although on MINST, all attacks fail under Median.

V. Related Work

A. Model Manipulation Attacks

Yao *et al.* [34] propose a latent trojan program that can be embedded into the 'teacher' model so that it is implanted with a latent trojan on non-existent output labels. The trojan is injected completely and activated when the 'teacher' model is inherited by the 'student' model through transfer learning. Lin *et al.* [21] introduce a composite attack, which uses triggers consisting of multiple benign features within scope to evade backdoor scanning procedures, which can be seen as the global



(a) MNIST - Median (Non-IID)

(b) MNIST - Median (IID)

(c) CIFAR-10 - Median (Non-IID)

(d) CIFAR-10 - Median (IID)

(e) CIFAR-100 - Median (Non-IID)

(f) CIFAR-100 - Median (IID)

Fig. 8. ASR curves of four attacks on Median (under both IID and Non-IID data distributions)

version of our FedTrojan. However, applying it to FL directly will lead to a high FHR. Fang *et al.* [35] propose a local model poisoning attack against Byzantine-robust FL, where the goal of the attacker is to build a local model on each affected working device such that the global model deviates the right direction of updating.

B. Trojan Attacks against Federated Learning

Trojan attacks in FL have been extensively studied. For example, Bagdasaryan *et al.* [20] propose a trojan attack in which an attacker trains the trojaned model locally and submits it to the server. To make the attack more effective, an explicit boosting strategy is proposed. The attacker expands the weight of the poisoning model to ensure that the trojan can survive the average. DBA [25] is a weakly negotiated trojan attack. Attackers only need to make agreements on: 1) a small patch-based trojan pattern (i.e., global trigger); 2) how the global trigger is segmented into local triggers; 3) the assignment of local triggers to individuals in advance. Then, each attacker embeds the assigned local trigger into its local model separately. However, DBA suffers high FHR. Huang [36] introduces a dynamic trojan attack to deal with the change of adversarial targets, by connecting meta-learning with trojan attacks in FL settings. The attacker can learn a versatile model from previous experiences, and adapt the model easily to new adversarial tasks with a few of examples.

C. Byzantine-robust Federated Learning

Blanchard *et al.* [30] propose an aggregation rule named Krum, which can tolerate f Byzantine attackers out of n participants. In Krum, the server chooses one from n local models that is most similar to all other models as the global model, rather than by calculating their average. Cao *et al.* [31] propose FLTrust to provide a trust mechanism for FL, using a small root dataset to generate a server model in order to decide the trust scores of the local models parameters. The server updates the global model using a weighted average of multiple local model parameters. Chen *et al.* [33] calculate the median on each dimension among all local models, and use the obtained median of each dimension to form the global model.

Pillutla *et al.* [32] design a robust FL algorithm RFA based on the classical geometric median. RFA preserves the privacy of participants by iteratively invoking the secure multiparty computation primitives.

VI. Conclusion

In this paper, we proposed a zero-knowledge federated trojan attack FedTrojan against FL. In FedTrojan, each attacker selects a benign semantic feature within the scope as well as a semantic catalytic feature unrelated to the task to train its local model in a way that the task-related feature is injected secretly into the local model, i.e., can not be activated by the feature. Those local triggers are assembled into a global trigger by using model aggregating on the server side. Extensive evaluations demonstrate that FedTrojan is more effective and precise than existing works under FedAvg and Byzantine-robust FL schemes.

Acknowledgment

This work was supported in part by Natural Science Foundation of Shanghai (Grant No.22ZR1400200), Fundamental Research Funds for the Central Universities (No. 2232023Y-01), HK RGC under Grants R6021-20F, R1012-21, RFS2122-1S04, C2004-21G, C1029-22G, and N CityU139/21.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, pp. 1273–1282, PMLR, 2017.

[2] E. T. M. Beltrán, M. Q. Pérez, P. M. S. Sánchez, S. L. Bernal, G. Bovet, M. G. Pérez, G. M. Pérez, and A. H. Celdrán, "Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges," *IEEE Communications Surveys & Tutorials*, 2023.

[3] T. Gu, B. Dolan-Gavitt, and S. Garg, "Badnets: Identifying vulnerabilities in the machine learning model supply chain," *arXiv preprint arXiv:1708.06733*, 2017.

[4] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang, "Trojaning attack on neural networks," 2017.

[5] T. D. Nguyen, T. Nguyen, P. Le Nguyen, H. H. Pham, K. D. Doan, and K.-S. Wong, "Backdoor attacks and defenses in federated learning: Survey, challenges and future research directions," *Engineering Applications of Artificial Intelligence*, vol. 127, p. 107166, 2024.

[6] D. Cao, S. Chang, Z. Lin, G. Liu, and D. Sun, "Understanding distributed poisoning attack in federated learning," in *2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 233–239, IEEE, 2019.

[7] C. Fung, C. J. Yoon, and I. Beschastnikh, "Mitigating sybils in federated learning poisoning," *arXiv preprint arXiv:1808.04866*, 2018.

[8] C. Wu, X. Yang, S. Zhu, and P. Mitra, "Mitigating backdoor attacks in federated learning," *arXiv preprint arXiv:2011.01767*, 2020.

[9] S. Lu, R. Li, W. Liu, and X. Chen, "Defense against backdoor attack in federated learning," *Computers & Security*, vol. 121, p. 102819, 2022.

[10] H. Wen, S. Chang, and L. Zhou, "Light projection-based physical-world vanishing attack against car detection," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2023.

[11] 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 *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 707–723, IEEE, 2019.

[12] Y. Liu, W.-C. Lee, G. Tao, S. Ma, Y. Aafer, and X. Zhang, "Abs: Scanning neural networks for back-doors by artificial brain stimulation," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1265–1282, 2019.

[13] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," *arXiv preprint arXiv:1712.05526*, 2017.

[14] A. Turner, D. Tsipras, and A. Madry, "Label-consistent backdoor attacks," *arXiv preprint arXiv:1912.02771*, 2019.

[15] S. Li, M. Xue, B. Z. H. Zhao, H. Zhu, and X. Zhang, "Invisible backdoor attacks on deep neural networks via steganography and regularization," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 5, pp. 2088–2105, 2020.

[16] K. Doan, Y. Lao, W. Zhao, and P. Li, "Lira: Learnable, imperceptible and robust backdoor attacks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11966–11976, 2021.

[17] K. Doan, Y. Lao, and P. Li, "Backdoor attack with imperceptible input and latent modification," *Advances in Neural Information Processing Systems*, vol. 34, pp. 18944–18957, 2021.

[18] S. Andreina, G. A. Marson, H. Möllering, and G. Karame, "Baffle: Backdoor detection via feedback-based federated learning," in *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*, pp. 852–863, IEEE, 2021.

[19] E. Bagdasaryan and V. Shmatikov, "Blind backdoors in deep learning models," in *30th USENIX Security Symposium (USENIX Security 21)*, pp. 1505–1521, 2021.

[20] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *International Conference on Artificial Intelligence and Statistics*, pp. 2938–2948, PMLR, 2020.

[21] J. Lin, L. Xu, Y. Liu, and X. Zhang, "Composite backdoor attack for deep neural network by mixing existing benign features," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pp. 113–131, 2020.

[22] F. Nuding and R. Mayer, "Poisoning attacks in federated learning: An evaluation on traffic sign classification," in *Proceedings of the tenth ACM conference on data and application security and privacy*, pp. 168–170, 2020.

[23] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," in *International Conference on Machine Learning*, pp. 634–643, PMLR, 2019.

[24] S. Li, E. C.-H. Ngai, and T. Voigt, "An experimental study of byzantine-robust aggregation schemes in federated learning," *IEEE Transactions on Big Data*, 2023.

[25] C. Xie, K. Huang, P.-Y. Chen, and B. Li, "Dba: Distributed backdoor attacks against federated learning," in *International Conference on Learning Representations*, 2019.

[26] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[27] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[28] T. Minka, "Estimating a dirichlet distribution," 2000.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[30] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[31] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping," *arXiv preprint arXiv:2012.13995*, 2020.

[32] K. Pillutla, S. M. Kakade, and Z. Harchaoui, "Robust aggregation for federated learning," *arXiv preprint arXiv:1912.13445*, 2019.

[33] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 1, no. 2, pp. 1–25, 2017.

[34] Y. Yao, H. Li, H. Zheng, and B. Y. Zhao, "Latent backdoor attacks on deep neural networks," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2041–2055, 2019.

[35] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to byzantine-robust federated learning," in *29th USENIX Security Symposium (USENIX Security 20)*, pp. 1605–1622, 2020.

[36] A. Huang, "Dynamic backdoor attacks against federated learning," *arXiv preprint arXiv:2011.07429*, 2020.