Stealthy Backdoor Attack against Federated Learning through Frequency Domain by Backdoor Neuron Constraint and Model Camouflage

Qiao, Yanqi; Liu, Dazhuang; Wang, Rui; Liang, Kaitai

DOI
10.1109/JETCAS.2024.3450527

Publication date
2024

Document Version
Final published version

Published in
IEEE Journal on Emerging and Selected Topics in Circuits and Systems

Stealthy Backdoor Attack Against Federated Learning Through Frequency Domain by Backdoor Neuron Constraint and Model Camouflage

Yanqi Qiao, Dazhuang Liu, Rui Wang, and Kaitai Liang, *Member, IEEE*

***Abstract*— Federated Learning (FL) is a beneficial decentralized learning approach for preserving the privacy of local datasets of distributed agents. However, the distributed property of FL and untrustworthy data introducing the vulnerability to backdoor attacks. In this attack scenario, an adversary manipulates its local data with a specific trigger and trains a malicious local model to implant the backdoor. During inference, the global model would misbehave for any input with the trigger to the attacker-chosen prediction. Most existing backdoor attacks against FL focus on bypassing defense mechanisms, without considering the inspection of model parameters on the server. These attacks are susceptible to detection through dynamic clustering based on model parameter similarity. Besides, current methods provide limited imperceptibility of their trigger in the spatial domain. To address these limitations, we propose a stealthy backdoor attack called "Chironex" against FL with an imperceptible trigger in frequency space to deliver attack effectiveness, stealthiness and robustness against various countermeasures on FL. We first design a frequency trigger function to generate an imperceptible frequency trigger to evade human inspection. Then we fully exploit the attacker's advantage to enhance attack robustness by estimating benign updates and analyzing the impact of the backdoor on model parameters through a task-sensitive neuron searcher. It disguises malicious updates as benign ones by reducing the impact of backdoor neurons that greatly contribute to the backdoor task based on activation value, and encouraging them to update towards benign model parameters trained by the attacker. We conduct extensive experiments on various image classifiers with real-world datasets to provide empirical evidence that Chironex can evade the most recent robust FL aggregation algorithms, and further achieve a distinctly higher attack success rate than existing attacks, without undermining the utility of the global model.**

***Index Terms*— Federated learning, backdoor attacks, stealthiness, frequency domain, backdoor neuron, model camouflage, activation value.**

Manuscript received 30 May 2024; revised 7 August 2024; accepted 22 August 2024. Date of publication 27 August 2024; date of current version 13 December 2024. This work was supported in part by the European Union's Horizon Europe Research and Innovation Program under Grant 101073920 (TENSOR), Grant 101070052 (TANGO), and Grant 101070627 (REWIRE). This article was recommended by Guest Editor C. H. Chang. *(Corresponding author: Dazhuang Liu.)*

The authors are with the Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, 2600 AA Delft, The Netherlands (e-mail: y.qiao@tudelft.nl; d.liu-8@tudelft.nl; r.wang-8@tudelft.nl; kaitai.liang@tudelft.nl).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JETCAS.2024.3450527.

Digital Object Identifier 10.1109/JETCAS.2024.3450527

I. Introduction

FEDERATED Learning (FL) [1], [2], [3], [4], [5], [6], [7], [8] is a type of distributed machine learning framework that has been proposed to preserve data privacy among participating agents. It supports collaborative training of an accurate global model by allowing agents to upload local updates, such as gradients and weights, to a server without compromising the local datasets. FL has been applied to various real-world applications including COVID-19 prediction [9] and autonomous driving [10].

Despite its attractive advantages, FL is susceptible to backdoor attacks [11], [12], [13], [14], [15], [16], [17]. [13] introduced a distributed backdoor attack (DBA) by dividing a global trigger into multiple pieces, which are distributed among local agents. The DBA incurs significant changes to certain dimensions of parameters in order to maintain the accuracy of the backdoor task. Recently, [12] proposed a stealthy model poisoning (SMP) attack by limiting Euclidean distance between the average updates (from all the benign agents) and the malicious updates. The malicious updates derived by this attack can be distinguished from the benign updates in the output layer at the parameter level because of the noticeable distances between them. Both DBA and SMP can be defended by using robust FL aggregation algorithms based on dynamic clustering via HDBSCAN [18] due to the significant directional discrepancy between the updates derived from these attacks and benign updates. Additionally, some works focus on developing durable backdoor attacks agaisnt FL to maintain high attack effectiveness when the adversary stop updating malicious models or gradients. For instance, Neurotoxin [19] attacks parameters that are changed less in magnitude during training which improves the durability of backdoors.

To mitigate backdoor attacks, researchers have designed robust FL aggregation algorithms [20], [21], [22], [23]. For instance, FLAME [22] and DeepSight [23] apply HDBSCAN at the training stage to conduct clustering and filter out the malicious updates. Specifically, FLAME exploits the discernible difference of model weights based on cosine similarity between benign and malicious updates, while DeepSight filters malicious updates by the output difference between benign and malicious models and the distinction of distribution of labels in the underlying training data of those models. They

use clipping strategies to reduce the influence of malicious updates and beyond. Moreover, FLAME uses adaptive noise to smooth the boundary of clustering. The current approach of defense is to detect malicious updates by exploiting the distinguishable dissimilarity between updates from malicious and benign agents. Additionally, current attacks [13], [14], [16], [19] do not provide enough trigger stealthiness during the inference stage. The poisoned samples with perceptible perturbation can be easily identified by an evaluator or a user who can distinguish the difference between 'just' an incorrect classification/prediction of the global model and the purposeful wrong decision due to a backdoor in the test/use stage.

There exists an impossibility for backdoor attacks to evade existing defenses because the backdoor tasks have a significant and noticeable impact on the backdoor-sensitive neurons deriving the distinguished distance from benign updates. This raises the question: *could we disguise malicious updates as benign ones at the parameter level to bypass current detection strategies while still maintaining the accuracy of the backdoor attack?*

To provide a concrete answer to the question, we propose a stealthy frequency attack, Chironex, to backdoor robust FL systems. Specifically, Chironex utilizes a frequency trigger function to produce the poisoned dataset, ensuring natural stealthiness of poisoned samples. Recent works [24], [25], [26], [27] has provided evidence that frequency domain triggers are still learnable by neural networks and provide excellent natural stealthiness. Then, Chironex constrains the impact of backdoor neurons identified by a new method called the Task-sensitive Neuron Searcher (TNS). TNS can construct a "backdoor" neuron list consisting of neurons that deliver a significant contribution to the backdoor task so that we can penalize the weights and biases of these neurons when their updates are in malicious directions. Chironex can enforce malicious updates to be naturally imperceptible from benign ones by minimizing the distance between malicious and benign parameters owned by the attacker.

Our main **contributions** are summarized as follows. We first design a frequency trigger function to produce imperceptible poisoned samples. We analyze the behavior of neurons in backdoor tasks at the parameter level by using TNS to identify those backdoor neurons that significantly contribute to backdoor tasks and reduce the impact of backdoor-sensitive neurons. We apply a step-forward training approach to generate benign and malicious models. Furthermore, we combine the malicious model with TNS to find the list of backdoor neurons and minimize its impact in order to evade anomaly parameter detection; meanwhile, we use the benign model as an estimation of the attacker's expected local model. We restrain parameter dissimilarity to make malicious updates indistinguishable from benign updates trained by the attacker (i.e, obtaining model camouflage) without sacrificing the utility of the global model. We fully take advantage of the attacker's ability to provide the criterion of malicious model update direction. Finally, we evaluate the attack performance and stealthiness on real-world datasets with various datasets and models. The experimental results demonstrate that Chironex achieves a high attack success rate while maintaining global model accuracy. Our attack also provides excellent stealthiness, allowing it to bypass the most recent robust aggregation algorithms, e.g., FLAME, DeepSight, whilst other existing attacks cannot. For example, Chironex achieves around 97.60% attack success rate and 90.65% global model accuracy under FLAME on FMNIST.

The rest of the paper is organized as follows. In Section II, the state-of-the-art federated learning frameworks, backdoor attack and defensive methods against federated learning are provided. Section III presents the threat model of our attack, including attack goal, capability and knowledge. Section IV details the technical approach including trigger function, backdoor neuron search and model camouflage. The experimental results of attack performance and ablation studies are given in Section V. Finally, the main conclusion and discussion are provided in Section VI.

II. Related Work

A. Federated Learning (FL)

Reference [1] proposed the concept of distributed learning associating with n agents and a server S to train a global model G collaboratively. At the training round t, each local agent i uses the global model G^t of the current round to train a local model L_i^{t+1} based on its own data D_i and sends the parameters/gradients update δ_i to S. Then the server S aggregates the received updates $\delta_i|_{i=1}^n$ from all the agents into the global model G^t to derive G^{t+1}. In the above process, each agent i computes the update as $\delta_i^{t+1} = L_i^{t+1} - G^t$, so that S uses an aggregation algorithm to compute a new global model G^{t+1} as:

$$G^{t+1} = G^t + \frac{lr_s}{n}\sum_{i=1}^{n}(L_i^{t+1} - G^t), \tag{1}$$

where lr_s is the learning rate of the server. The most commonly used aggregation algorithm is FedAvg, averaging the weighted updates of all the local agents as $G^{t+1} = \sum_{i=1}^{n}\frac{d_i}{d}(L_i^{t+1} - G^t)$, where $d_i = |D_i|, d = \sum_{i=1}^{n} d_i$. We set $G^{t+1} = \sum_{i=1}^{n}\frac{1}{n}(L_i^{t+1} - G^t)$ for the equal contribution of all the clients to evade from receiving fabricated dataset size of malicious agents as in [22]. When G converges or the training reaches a specific iteration upper bound, the aggregation process terminates and outputs a final global model.

Optimizations of FL have been proposed for various purposes, e.g., privacy [28], security [29], heterogeneity [30], communication efficiency [31] and personalization issues [32].

B. Backdoor Attacks on FL

An attacker can easily corrupt a set of agents in the training stage. These agents are manipulated to use poisoned data with a specific trigger and change an attacker-chosen base label to a target label to train their local models and further, they send the updates to the server performing aggregation. Accordingly, the global model that combines the updates from the malicious agents is embedded with a "backdoor". In the test stage, the model easily misclassifies the data inputs with the backdoor

trigger to the target label. To design a successful attack, the attacker must ensure that the clean dataset is classified into the correct label and the utility of the global model cannot be harmed by backdoor task training. Note that this notorious target poisoning can seriously affect the model prediction results and is difficult to detect after the training stage. Existing backdoor attacks may target to data and model.

1) Data Poisoning Attacks: The attacker manipulates training datasets by adding backdoor patterns into data samples. For example, it disintegrates the global pattern into several local patterns and further injects them to the compromised agents' datasets separately in DBA [13]. Although making global trigger more insidious, DBA does not restrain the training process so that the malicious and benign updates can look different at the model parameter level. Moreover, the attacker can modify the training dataset by flipping its label [33] and later send the model update trained by mislabeled data to the server. This type of attacks focuses on manipulating training data without fully considering the server aggregation strategies for anomaly updates detection.

2) Model Poisoning Attacks: This type of attacks manipulates the training process of malicious agents and further evades aggregators' anomaly update detection. Reference [14] introduced model replacement and scaling up attack to FL systems. The attacker can scale up the malicious update by a specific factor and the global model is replaced by the malicious model trained by poisoned data consequently. This attack brings a new perspective that an attacker can manipulate the training process or local model to perform the backdoor attack. Reference [11] proposed a so-called LIE attack by crafting model updates with minor changes. The attack explores the maximized range of parameters perturbation to induce the model to predict the desired label. Reference [15] exploited a projected gradient descent (PGD) attack with model replacement during the training of a malicious model on a dataset that is similar but not identical to the (original) clean dataset. Reference [12] designed an attack aiming to achieve stealthiness by estimating the average update from all benign updates and reducing the L_2-norm distance between the malicious and the average updates from others. The attack, however, cannot provide a solid stealthiness because there still exists a noticeable (peak) difference in the distribution of parameters between malicious and benign updates. Some works also aim to improve the persistence of attack effectiveness on FL. For example, Neurotoxin [19] attacks parameters that are changed less in magnitude during training which improves the durability of backdoors. Chameleon [34] finds that benign images with the original and the target labels of the poisoned images have key effects on backdoor durability. It then utilizes contrastive learning to amplify such effects towards a more durable backdoor.

C. Robust FL Aggregation

Several works have been proposed to handle malicious agents in the context of FL [35], [36], [37], [38], [39], [40], [41]. Krum [42] selects a local model that is similar to others as the global model, but it is vulnerable to some dimensions of malicious model parameters. Bulyan [43] improves Krum by applying a variant of Trimmed Mean method. Trimmed Mean [20] aggregates each dimension of model parameters independently and it computes the mean for a range of parameters. Median [20] takes median for aggregation. Reference [44] indicated that one can use FedAvg aggregation rules, by clipping weights and adding noise, to mitigate backdoor attacks. [21] proposed a robust aggregation algorithm based on sign aggregation [45] so-called RLR which changes the central server's learning rate based on the signs of agents' updates. Recently, [22] proposed a defending framework based on the clustering algorithm (HDBSCAN) so-called FLAME which can cluster dynamically all local updates based on their cosine distance into two groups separately. FLAME uses weight clipping for scaling-up malicious weights and noise addition for smoothing the boundary of clustering after filtering malicious updates. Reference [23] designed a robust FL aggregation rule called DeepSight using HDBSCAN. Their design leverages parameter distribution, output, and cosine distance to cluster all updates and further applies the clipping method. DeepSight fully exploits information leakage from malicious updates and provides a more precise detection than FLAME. SparseFed [46] performs norm clipping to all local updates and averages the updates as the aggregate. Top-k values of the aggregation update are extracted to filter out potential malicious parameters and returned to each agent who locally updates the models using this sparse update. CRFL [39] provides certified robustness in FL frameworks. It exploits parameter clipping and perturbing during federated averaging aggregation. In the test stage, it constructs a "smoothed" classifier using parameter smoothing. The robust accuracy of each test sample can be certified by this classifier when the number of compromised clients or perturbation to the test input is below a certified threshold.

III. Threat Model and Motivation

We consider the same threat model as in [12], [13], and [14].

A. Attacker's Goal

Following [13], [14], [15], and [19], we enable the attacker to manipulate the global model to predict a target label on any samples with an attacker-chosen trigger (i.e. the backdoor task). From the viewpoint of the attacker, there are two main objectives: preserving functionality for benign tasks (i.e., maintaining global model accuracy) and ensuring attack effectiveness for backdoor tasks (i.e., achieving high backdoor accuracy on the poisoned model). Two additional goals are considered in our attack including attack stealthiness and robustness. Attack stealthiness implies that poisoned samples exhibit visual similarity to clean ones while robustness is demonstrated by its effectiveness against backdoor defenses on FL. Unlike untargeted poisoning attacks [47] preventing the convergence of the global model, the goal of our attack is to manipulate malicious agents' local training processes to achieve high accuracy in the backdoor task without undermining the main task. Thus, the global model's behavior is naturally normal on clean data samples while it will predict poisoned data into a target label with a high attack success rate.

B. Attacker's Capability and Knowledge

We assume the attacker only has full access to malicious agents' clients, local training processes and training datasets. The attacker cannot change the aggregation algorithm of the server and manipulate the training processes and training datasets of the honest agents so that the model updates of those agents will not be affected by the attack. Unlike some backdoor attacks strictly requiring malicious agents to collude together, our assumption here does not need this collusion. Note we naturally allow malicious agents to collude but our attack can work well without this collusion, which means that we require fewer restrictions than others. At last, we do not require the attacker to know the FL aggregation rules applied in the server.

C. Technical Motivation

Existing attacks cannot reduce the distance between benign and malicious update in the parameter space and thus they can be easily detected by the state-of-the-art robust FL systems using parameter inspection. We study a new attack perspective in the sense that the attacker can restrain the disparity among parameters by training loss function of l_2-norm distance or cosine similarity. Some attacks, such as SMP, have used the similar objective terms to constrain the distance. In Figure 3, we show that it is not sufficient yet to only limit the distance among the parameters as we can still see the difference between benign and malicious updates. To tackle the issue, we introduce a new objective to further eliminate the influence of those backdoor neurons found by TNS. Unlike SMP, the philosophy of Chironex is to convert the malicious parameters to become benign, i.e. fooling the server to regard malicious updates as benign ones. In the figure, we see that the malicious model parameters of Chironex are very close to the benign parameters so that the malicious updates are much less abnormal. We further state that the attacker does not need to know a concrete FL aggregation algorithm before applying our attack, which makes the attack more general and practical. The experimental results (see Section V-D) show that two penalties $\mathcal{L}_{tns}$ and $\mathcal{L}_{l_2dist}$ work well with the combination of classification loss $\mathcal{L}_{main}$ and $\mathcal{L}_{bd}$ so that we can eliminate the impact of those backdoor neurons and meanwhile achieve practical performance w.r.t. attack success rate (ASR) and global model accuracy (MA).

IV. Proposed Methodology

We provide problem formulation and technical details in this section. Specifically, we first introduce our trigger injection function to insert a specific pattern into the frequency domain. Then, we formulate the optimization problem by three attack objectives, i.e., (1) high accuracy on clean and backdoor tasks, (2) backdoor neuron constraint and (3) model camouflage. We utilize step-forward training to obtain benign and malicious reference models on clean and poisoned datasets. We introduce a novel method to search backdoor neurons on the malicious reference model and constrain the impact of those neurons. Additionally, we minimize the l_2-norm distance between malicious parameters and benign ones from the benign reference model to achieve model camouflage. The frequently used notations in this paper are shown in Table I.

TABLE I
Notation Summary

Notation	Description	Notation	Description
δ^t	Update of agent at round t	P	Neuron list of TNS
D_c	Clean dataset	α, β	Lagrange coefficients
D_{bd}	Poisoned dataset	G^t	Global model at round t
ω	Blend ratio	y_t	Target label
$\phi(\cdot)$	Trigger injection function	ξ	Threshold for P
η	Malicious agents rate	lr	Learning rate
L_i^t	Local model of agent i at round t	ρ	Threshold of RLR
γ	Neuron sensitivity rate	S	FL Server
$a_{[i]}^l$	Activation of neuron i of layer l	$\psi(\cdot)$	Measurement of P
$b_{[i]}^l$	Bias of neuron i	s	Cluster size of Defenses
$n_{[i]}^l$	Neuron i	X	# of Samples in TNS
$w_{[i]}^l$	Weight of neuron i	n	# of total agents
m	# of malicious agents	R	Total training round

A. Problem Formulation

1) Trigger Injection Function ϕ: Taking the image classification task as an example, we first introduce a frequency trigger function $\phi(\cdot)$ to produce the poisoned dataset D_{bd} with the imperceptible trigger t through frequency space. Given a clean sample $x \in [0,1]^{H\times W\times C}$ (height H, width W and channel C) and a specific trigger image x^t, we first transform them into frequency domain via discrete cosine transform (DCT) $\mathcal{D}(\cdot)$ as:

$$\mathcal{D}(u,v,c) = N_u N_v \sum_{i=0}^{H-1}\sum_{j=0}^{W-1} x(i,j,c) cos\frac{(2i+1)u\pi}{2H} cos\frac{(2j+1)v\pi}{2W}, \quad (2)$$

where $u, i \in \{0,1,\cdots,H-1\}$, $v, j \in \{0,1,\cdots,W-1\}$ and $c \in \{0,1,\cdots,C-1\}$. N_u and N_v are normalization terms, $N_u \triangleq \sqrt{1/H}$ if $u=0$ and otherwise $N_u \triangleq \sqrt{2/H}$. Similarly, $N_v \triangleq \sqrt{1/W}$ if $v=0$ and otherwise $N_v \triangleq \sqrt{2/W}$. Triples (i,j,c) and (u,v,c) refer to a specific pixel and frequency band of x and its frequency form respectively. Then, we blend the trigger pattern t (spectrum of $\mathcal{D}(x^t)$) and $\mathcal{D}(x)$ to generate the poisoned sample x_{bd} with a binary mask $\mathcal{M} = 1_{(u,v,c)\in[0:\lambda H:,0:\lambda W:]}$, where λ determines the location and size of trigger to be blended and ω is the blend ratio to decide the proportion of information contributed by x^t. Finally, we utilize inverse DCT (IDCT) $\mathcal{D}^{-1}(\cdot)$ to obtain the spatial form of poisoned sample x_{bd}. The entire frequency trigger injection method is held:

$$\phi(x) = \mathcal{D}^{-1}(\mathcal{D}(x)(1-\mathcal{M}) + [(1-\omega)\mathcal{D}(x) + \omega t] * \mathcal{M}). \quad (3)$$

The goal of this function is to mislead the prediction of $\phi(x)$ to the target label y_t.

2) Attack Objectives: We propose a stealthy backdoor attack against FL on computer vision tasks. In the following, we denote the clean training dataset as $D_c = \{(x_i, y_i)\}_{i=1}^{|D_c|}$ containing $|D_c|$ images. In practice, we randomly select samples from D_c to produce poisoned training dataset D_{bd} by the proposed $\phi(\cdot)$ with a specific poison ratio. For a clean sample and its label (x, y) from D_c, we poison the clean sample to $(\phi(x), y_t)$ by backdoor injection function $\phi(\cdot)$, where y_t is the target label.

Our main objective is to learn backdoor parameters θ_{bd} (trained by D_{bd}) by constraining the influence of backdoor neurons and making the parameters close to the benign parameters θ_{cln} trained a step-forward by D_c. Given a malicious local model L^t_{bd} at round t with θ_{bd} and a backdoor injection function $\phi(\cdot)$, we minimize the following loss function to hold the performance of the main task on D_c:

$$\mathcal{L}_{main} = \sum_{(x,y)\in D_c} \mathcal{L}^{ce}(L^t_{bd}(x), y), \tag{4}$$

where $\mathcal{L}^{ce}$ denotes the cross entropy loss. To provide a practical attack success rate, we minimize the following loss function on the backdoor task:

$$\mathcal{L}_{bd} = \sum_{(x,y)\in D_{bd}} \mathcal{L}^{ce}(L^t_{bd}(\phi(x)), y_t). \tag{5}$$

We notice that the backdoor training is a kind of "shortcut" learning which means several backdoor-sensitive neurons are easily affected via the backdoor task with certain preferences while they have less contribution on the main task training. To achieve stealthiness, we minimize the following constrained loss function on poisoned data to reduce the impact of those backdoor-sensitive neurons:

$$\mathcal{L}_{tns} = \sum_{(i,l)\in\psi(P_{bd},P_c)} (|w^l_{[i]}| + |b^l_{[i]}|), \tag{6}$$

where $\mathcal{L}_{tns}$ is backdoor neurons constraint loss, $\psi(\cdot)$ is our measurement between the backdoor neuron list P_{bd} and the clean neuron list P_c found by benign and backdoor TNS, $w^l_{[i]}$ and $b^l_{[i]}$ are the weight and bias corresponding to the i-th neuron $n^l_{[i]}$ of the l-th layer. The detailed information about how to obtain two lists P_c and P_{bd} are provided in Section IV-B.

To capture parameter similarity to support model camouflage, we restrain the loss corresponding to l_2-norm distance between θ_{bd} and θ_{cln}:

$$\mathcal{L}_{l_2dist} = ||\theta_{bd} - \theta_{cln}||_2, \tag{7}$$

where θ_{bd} is the parameters of malicious model and θ_{cln} is the estimated benign parameters. The details of model camouflage are provided in Section IV-C.

Given the three objectives from Equations (4), (5) and Equation (6) and Equation (7), we formalize the final attacker's objectives as a constrained optimization problem:

$$\arg\min_{\theta_{bd}} \mathcal{L}_{main} + \mathcal{L}_{bd} + \alpha\mathcal{L}_{tns} + \beta\mathcal{L}_{l_2dist}, \tag{8}$$

where we use α and β to control the strength of the constraint loss. We can achieve the optimization by constraining the contribution introduced by backdoor neurons (which are identified by TNS) and implementing the model camouflage. The overview of Chironex is in Figure 1.

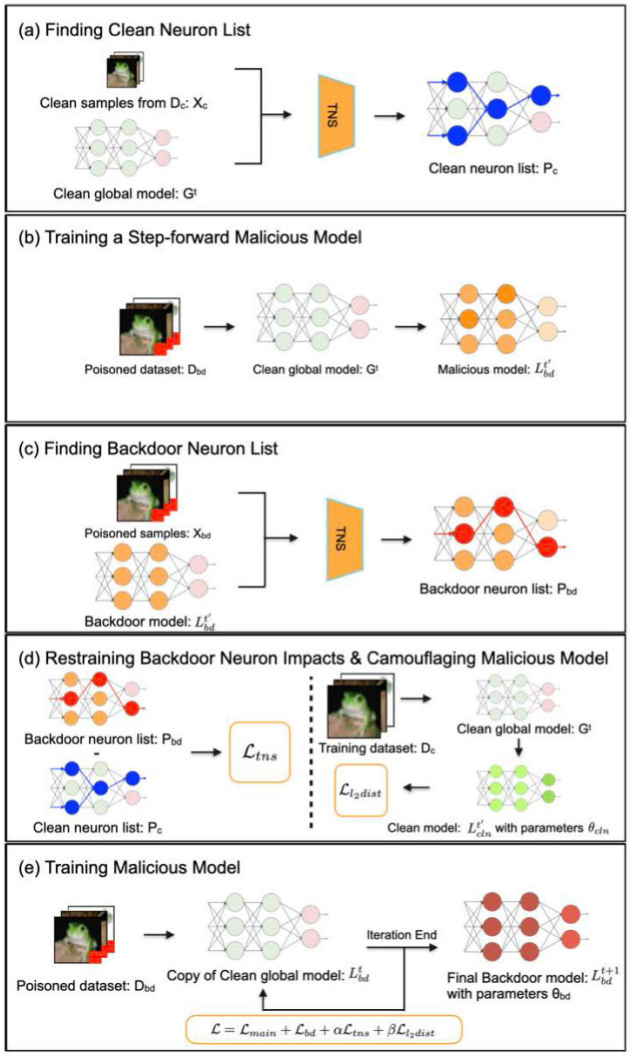

Fig. 1. The workflow of Chironex. The Chironex attack includes three objectives: (1) achieving high accuracy on clean and backdoor tasks; (2) minimizing the impacts of backdoor neurons; and (3) minimizing l_2-norm distance between malicious parameters and estimated benign ones. At round t, we train malicious update δ^{t+1}_m as (a) Find the list P_c of clean neurons that contributed to main task by the proposed TNS; (b) Pretrain a malicious local model for searching backdoor neurons; (c) Find the list P_{bd} of backdoor neurons that contributed to backdoor task by TNS; (d) Compute the constraint loss of backdoor neurons $\mathcal{L}_{tns}$ based on P_c and P_{bd} and train θ_{cln} for $\mathcal{L}_{l_2dist}$; (e) Optimize final malicious model θ_{bd} by $\mathcal{L}$, including classification loss $\mathcal{L}_{main} + \mathcal{L}_{bd}$, backdoor constraint loss and model camouflage loss on D_{bd} to obtain malicious update δ^{t+1}_m.

B. Backdoor Neuron Constraint by TNS

To compute backdoor neuron constraint in Equatio (6), we here use the proposed TNS to find task-sensitive neurons in each layer of Deep Neural Networks (DNNs). We first give the task-sensitive (influential) neurons (TSN) in Definition 1 by following the same philosophy as in [15]. The neurons satisfying the (task) sensitivity contribute significantly to a certain task[1] (i.e. either a main or a backdoor task); and if they are sensitive to a backdoor task, we call them backdoor neurons. We use the activation value of a neuron to measure its contribution or influence of a classification.

Definition 1 (Task-sensitive Neuron): Given a real positive number γ, a neuron n, its activation value a, weight w, the activation value A of the neuron of the next layer connecting

[1]These TSN deliver more contributions than others in the task. Changing their weights also outputs a crucial impact on the specific task (while this doesn't apply to the non-TSN).

Algorithm 1 Task-Sensitive Neuron Searcher (TNS)

Input: Samples X, Model $\mathcal{M}$
Parameter: Number of Linear Layers L, Weights w of Model $\mathcal{M}$, Activation Value a, Neurons n, Index of Neurons i, j, Sensitivity Rate γ, Target Label y_t, Neuron List Π
Output: List P

Initialize $\Pi = []$
Initialize $P = \{0\}$
for $X_i \in X$ **do**
  $GetActivationValue(\mathcal{M}, X_i)$
  $Append(\Pi^L, n^L_{[y_t]})$
  $P[n^L_{[y_t]}] \mathrel{+}= 1$
  for $l \in L$ (descend order) **do**
    for $n^l_{[i]} \in \Pi^l$ **do**
      for $n^{l-1}_{[j]} \in n^{l-1}$ **do**
        if $|w^{l-1}_{[j]} * a^{l-1}_{[j]}| > |\gamma * a^l_{[i]}|$ **then**
          $Append(\Pi^{l-1}, n^{l-1}_{[j]})$
          $P[n^{l-1}_{[j]}] \mathrel{+}= 1$
return P

Algorithm 2 Chironex Backdoor Attack

Input: Clean Dataset D_c, Poisoned Dataset D_{bd}, Global Model G^t with Parameters θ, Clean Samples X_c
Parameter: Threshold ξ, Hyperparameters α, β, Client Learning Rate lr_c
Output: Malicious Update δ^{t+1}_m

Copy Global model: L^t_{bd} with $\theta_{bd} \leftarrow G^t$
Get Clean Neuron List: $P_c = \text{TNS}(G^t, X_c)$.
A Step-forward Training for Malicious Model: $L^{t'}_{bd} \leftarrow$ Train G^t on D_{bd}.
Poison Clean Samples: $X_{bd} = \phi(X_c)$.
Get Backdoor Neuron List: $P_{bd} = \text{TNS}(L^{t'}_{bd'}, X_{bd})$.
$L^{t'}_{cln}$ with Parameters $\theta_{cln} \leftarrow$ A Step-forward Training for Benign Model with G^t on D_c.
for $batch \in D_{bd}$ **do**
  Compute the Loss: $\mathcal{L} = \mathcal{L}_{main} + \mathcal{L}_{bd} + \alpha * \mathcal{L}_{tns} + \beta * \mathcal{L}_{l_2dist}$ as in Equation (8).
  Train parameters θ_{bd} of L^t_{bd} with Stochastic Gradient Descent (SGD) under Learning Rate $lr_c/2$.
$\delta^{t+1}_m = \theta_{bd} - \theta$
return δ^{t+1}_m

to n, the neuron n for classification is task-sensitive with γ-sensitivity if $|w \times a| > |\gamma \times A|$.

To effectively search the sensitive neurons, we apply a mixed strategy including forward and backward analysis. For the deeper linear layers, where neurons contribute more significantly compared to the shallow layers (e.g., convolutional layers), we apply backward analysis to meticulously identify task-sensitive neurons. First, we feed input samples to DNN. For a certain neuron $n^L_{[i]}$ of layer L, we compare all activation values $a^{L-1}_{[j]}|^M_{j=1}$ of M neurons $n^{L-1}_{[j]}|^M_{j=1}$ in the previous layer $L-1$, connecting to $n^L_{[i]}$, to activation value $a^L_{[i]}$ of this neuron. Second, we set a sensitivity rate γ for the current layer L. We compute each neuron's contribution as $|w^{L-1}_{[j]} * a^{L-1}_{[j]}|$, where $w^{L-1}_{[j]}$ is the weight of $n^{L-1}_{[j]}$. If $|w^{L-1}_{[j]} * a^{L-1}_{[j]}| > |\gamma * a^L_{[i]}|$, $n^{L-1}_{[j]}$ can be identified as a backdoor neuron which contributes more to the backdoor task in this layer. Algorithm 1 shows how we identify all task-sensitive neurons and generate the neuron list. For shallow layers, we use forward analysis to select the top 5% of neurons with the highest activations in each layer. This approach enables our attack to establish connections between the layers.

We further minimize $\mathcal{L}_{tns}$ by TNS based on benign and malicious models with clean and poisoned samples to rectify those backdoor neurons. The minimization of this objective provides a crucial functionality: eliminating the distinguishable impacts on backdoor parameters. At round t, we assume the attacker chooses clean samples X_c from D_c as input of the current global model G^t. We use TNS to generate a clean neuron list of global model G^t by X_c. Then, we train a step-forward malicious model $L^{t'}_{bd}$ with D_{bd} to identify backdoor neurons to be rectified. Given G^t, $L^{t'}_{bd}$, X_c and D_{bd}, we apply TNS to find the neuron lists P_c and P_{bd} for main and backdoor tasks respectively. We use threshold ξ to figure out distinguishable backdoor neurons from clean ones. We set $\psi(P_{bd}, P_c) = \{(i,l)|(P_{bd}[n^l_{[i]}] - P_c[n^l_{[i]}]) > \xi, l \in \{1, 2, \cdots, L\}\}$. If $(i,l) \in \psi(P_{bd} - P_c)$, we add the absolute value of the difference between ϵ and $w^l_{[i]}$, $b^l_{[i]}$ of $n^l_{[i]}$ to $\mathcal{L}_{tns}$. To eliminate the influence of backdoor neurons, we enforce $w^l_{[i]}$, $b^l_{[i]}$ to approach zero. Finally, we compute TNS loss function as $\mathcal{L}_{tns} = \sum_{(i,l)\in\psi(P_{bd}-P_c)} (|w^l_{[i]}| + |b^l_{[i]}|)$ and train θ_{bd} with it. The details is shown in Figure 1 (a) (b) (c) and (d)-left, and in lines $2-5$ and 8 of Algorithm 2.

Why does TNS work? We found that backdoor neurons are distinguishable from the clean neurons (which mostly contribute to the main task) and meanwhile, they could only and significantly contribute to the backdoor task. By minimizing $\mathcal{L}_{tns}$ and $\mathcal{L}_{bd}$, the contribution of the backdoor neurons is redistributed to other neurons that have less contribution to the backdoor task. In this way, we encourage those neurons with "less" contribution to play a part in the backdoor task. In Figure 6, we show the results of the disparity between the backdoor and clean neuron lists to confirm that minimizing $\mathcal{L}_{tns}$ can reduce the contribution of those backdoor neurons at the parameter level, without affecting the sensitivity of the clean neurons in the main task. Thus, we can maintain a high attack accuracy but also have a low level of influence incurred by those backdoor neurons.

C. Model Camouflage

Given the computation of $\mathcal{L}_{tns}$, our next task is to disguise θ_{bd} as θ_{cln} by minimizing the distance loss $\mathcal{L}_{l_2dist}$ in Equation (7). To this end, we design a new backdoor training approach motivated by [12] to manipulate the malicious parameters more naturally-like benign ones trained by D_c (which are owned by the attacker). Instead of estimating the current global model parameters through averaged updates from the previous iteration [12], we get the malicious and

benign parameters sufficiently close to each other which makes the attack stealthy. We allow the attacker to train both two models based on the same set of its own data, which doesn't violate our assumptions. Similar to the backdoor neuron constraint process, we also use step-forward training with D_c to obtain a benign reference model $L^{t'}_{cln}$ to estimate the expected benign model parameters θ_{cln}. During backdoor training, we use Equation (7) to enforce the malicious parameters to be indistinguishable from θ_{cln} in the training stage. We set l_2-norm distance as the measurement for parameter similarity and β as a hyperparameter that controls the strength of model camouflage between θ_{cln} and θ_{bd}. The computation of $\mathcal{L}_{l_2dist}$ is shown in Figure 1 (d)-right and (e), and in lines 6 and 8 of Algorithm 2. As last, we train the malicious model with D_{bd} via Equation (8). The details of Chironex are given in Algorithm 2.

V. Experiments

A. Experimental Setup

1) Datasets and Network Structures: We tested the effectiveness of Chironex on five standard image datasets: MNIST [48], Fashion-MNIST (FMNIST) [49], CIFAR-10 [50], FEMNIST [51] and Tiny-ImageNet [52] in the independent and identically distribution (i.i.d). and non-i.i.d. data distribution settings. Specifically, MNIST contains 70k (28×28) handwritten digits images divided into ten mutually exclusive classes, in which 60k are for training and 10k for testing. FMNIST provides the same amount and size of grayscale images with ten classes. CIFAR-10 includes 60k (32×32) color images with the same number of classes, in which 50k are for training and 10k for testing. FEMNIST contains about 800k (28×28) handwritten digits and characters images divided into 62 classes provided by 3,550 users, which is commonly used in federated learning frameworks. Tiny-ImageNet has 200 classes and 100k (64×64) colored images. Each class includes 500 training samples, 50 validation and 50 test samples. The data are distributed in both i.i.d. and non-i.i.d. fashion among agents. In the non-i.i.d. context, we set a q of the dataset to evaluate the degree of non-i.i.d. level by following [38]. Since there are 10 classes, we divide the agents into 10 groups. A training sample with label $y \in \{0, 1, \cdots, 9\}$ is allocated to a group (and hereafter we can call it as group y) with probability $q > 0$ and to any other groups with probability $\frac{1-q}{9}$. In the group y, each agent's training data is in i.i.d manner. $q = 0.1$ represents the dataset of local agent is i.i.d; while the degree of no-i.i.d increases with a growing q. All images in the datasets are normalized to [0,1]. We performed the experiments on commonly used DNN models including the classic CNN model for MNIST, FMNIST and FEMNIST, ResNet18 [53] for CIFAR-10 and Tiny-ImageNet.

2) Implementation: We set $n = 1000$ agents with $m = 10$ malicious agents to train the global model for $R = 200$ rounds with FedAvg in the i.i.d. manner as the default setting. The ratio of the compromised agents to all agents $\eta = \frac{m}{n} = 1\%$. In each round, the server randomly selects 10 clients for local model aggregation. We used learning rate $lr_c = 0.1$ for local training and $lr_s = 1$ for central server aggregation while malicious agents used malicious learning rate $lr_m = 0.05$ to perform backdoor attacks. Local models were trained by SGD optimizer. We set $\frac{|D_{bd}|}{|D_c|}$ as poison data rate (PDR), which is the fraction of injected poisoned data D_{bd} with target label in the overall clean training dataset D_c with the attacker-chosen label. We set PDR = 20% as default. We found that if $\alpha, \beta \gg 1$, the malicious model parameters are close to those of the benign model but the attack performance does not work well. If $\alpha, \beta \ll 1$, the results are the other way around. To balance the trade-off, we set $\alpha = 0.5$ and $\beta = 0.5$. We choose the "Hello Kitty" pattern in [54] as our trigger image x^t as default. Some poisoned examples are shown in Figure 7. For Chironex attack, we set proper ξ and γ for different network structures. ξ is strongly related to the number of samples fed into models in TNS and NN layers. For TNS, we leverage the entire poisoned dataset to find backdoor neurons and the same size of clean dataset to find benign ones. We set ξ as half of the number of the poisoned samples for each dataset, indicating a neuron has a significant impact on more than half of classification tasks. We set γ to 0.05 to distinguish backdoor neurons from clean ones under a given ξ, indicating the settings for ξ, γ are practical.

3) Evaluation Metrics: • *Global Model Accuracy (MA).* We set the test accuracy on clean validation samples fed into the global model as MA.

• *Attack Success Rate (ASR).* We set the ratio of backdoored examples fed into the global model misclassified as the target label as ASR.

• *False Negatives Rate (FNR).* FNR evaluates the attack robustness, i.e. how well the attack evades the detection of robust FL aggregation. We set it as $\frac{FN}{FN+TP}$ indicating the ratio of malicious updates for which the defense produces wrong predictions to the total number of malicious models, i.e., the fraction of the number of malicious updates misclassified into benign updates cluster (False Negative - FN), where TP is True Positive showing the number of malicious updates correctly classified as malicious.

• *False Positives Rate (FPR).* This investigates the robustness. FPR $= \frac{FP}{FP+TN}$ denotes the ratio of benign updates that are misclassified as malicious (False Positive - FP) to the total number of benign models, where TN is True Negative indicating the number of benign updates correctly classified as benign.

B. Evaluation of Attack Without Defense

We used a backdoor attack with both main and backdoor objectives as our baseline and included other attacks - DBA, Edge-Case [15], SMP [12] and Neurotoxin [19] - into the comparison. We tested our attack effectiveness through MA and ASR under FedAvg without defense. We conducted a single-target attack for each backdoor method, in which we chose a base label (class 5) misclassified into one target label y_t (class 7) for all the compared methods per dataset. The results in the i.i.d. setting are presented in Table II. As for FedAvg with "no defense" on MNIST and FMNIST, Chironex achieves >98% ASR and meanwhile, it maintains MA close to benign model accuracy (< 1% gap). Although MA on

TABLE II

ATTACK PERFORMANCE VIA MA (%) AND ASR (%) FOR SEVERAL ATTACKS AGAINST NO/DIFFERENT DEFENSES ON MNIST, FMNIST AND CIFAR-10 IN THE I.I.D. CONTEXT. WE SHOW THE AVERAGED RESULTS (MEAN±SD) OF 10 INDEPENDENT RUNS

Dataset ↓	Defense → / Attack ↓	No Defense MA	No Defense ASR	Clipping MA	Clipping ASR	Median MA	Median ASR	RLR MA	RLR ASR	FLAME MA	FLAME ASR	DeepSight MA	DeepSight ASR
MNIST	No Attack	99.07±0.04	-	98.72±0.06	-	98.64±0.05	-	93.05±2.38	-	98.91±0.05	-	97.39±0.16	-
	Backdoor (Baseline)	98.89±0.09	**99.96±0.05**	98.39±0.03	0.00±0.00	**98.56±0.07**	0.00±0.00	**94.95±0.23**	0.00±0.00	**98.96±0.02**	0.00±0.00	**98.36±0.05**	0.00±0.00
	DBA	98.87±0.06	98.99±0.05	98.41±0.17	0.00±0.00	98.04±0.12	4.01±3.29	93.29±2.86	0.90±1.19	98.80±0.19	1.01±0.57	97.28±0.10	0.00±0.00
	Edge-Case	98.76±0.11	98.91±0.48	**98.73±0.05**	**98.19±0.21**	98.25±0.31	0.00±0.00	93.56±2.46	0.00±0.00	98.26±0.06	0.00±0.00	97.82±0.04	0.00±0.00
	SMP	98.93±0.05	99.16±0.08	98.53±0.07	97.03±0.23	98.49±0.09	80.19±2.83	94.17±1.16	0.00±0.00	98.71±0.11	0.00±0.00	97.90±0.03	0.00±0.00
	Chironex (Ours)	**98.96±0.03**	99.38±0.62	98.17±0.04	97.93±0.17	98.53±0.08	**99.52±1.04**	94.72±1.13	**98.88±0.09**	98.84±0.07	**93.72±4.26**	97.03±0.10	**94.28±1.26**
FMNIST	No Attack	92.41±0.20	-	92.20±0.75	-	90.84±0.19	-	81.54±0.93	-	91.35±0.13	-	91.97±0.16	-
	Backdoor (Baseline)	91.05±0.04	**99.97±0.05**	**91.17±0.09**	00.00±0.00	90.42±0.07	0.00±0.00	78.77±0.17	0.00±0.00	90.04±0.06	0.00±0.00	90.63±0.20	0.00±0.00
	DBA	90.28±1.19	98.40±0.12	90.85±0.20	00.00±0.00	90.49±0.17	2.05±1.05	81.31±0.70	10.5±6.85	90.15±0.35	3.16±2.00	**91.45±0.09**	0.00±0.00
	Neurotoxin	91.22±0.18	99.83±0.04	91.05±0.36	0.00±0.00	90.39±0.15	3.26±0.91	81.20±0.62	0.00±0.00	90.49±0.58	0.00±0.00	91.03±0.07	2.41±1.78
	SMP	90.95±0.32	99.07±0.39	90.83±0.45	97.31±0.89	90.36±0.04	75.74±2.60	80.27±0.39	0.00±0.00	90.51±0.26	0.00±0.00	90.89±0.18	0.00±0.00
	Chironex (Ours)	**91.44±0.14**	97.83±0.34	90.62±0.71	**98.13±0.33**	**90.79±0.12**	**93.46±1.32**	**81.86±0.20**	**99.80±0.07**	**90.65±0.22**	**97.60±0.67**	91.29±0.13	**95.03±0.80**
CIFAR-10	No Attack	89.19±1.05	-	89.42±0.77	-	89.56±0.89	-	89.86±1.92	-	91.03±0.51	-	89.43±0.80	-
	Backdoor (Baseline)	86.35±0.67	96.95±1.79	**88.83±0.60**	0.00±0.00	**88.03±0.82**	9.72±5.15	89.07±2.10	0.00±0.00	89.20±1.29	8.40±2.29	88.14±0.82	5.07±2.38
	DBA	86.45±0.63	93.36±5.44	87.92±0.19	0.00±0.00	87.24±0.80	42.63±7.35	79.25±2.14	51.53±4.81	53.02±0.83	11.63±3.79	89.17±0.54	32.85±4.98
	Edge-Case	**86.65±0.78**	90.53±1.21	88.27±0.84	98.01±0.58	87.19±0.39	20.83±4.69	**89.33±1.75**	32.26±5.21	82.60±1.30	17.91±3.43	**89.38±0.59**	20.09±3.20
	SMP	86.30±0.64	95.47±1.02	87.09±0.57	97.04±0.99	87.27±0.69	74.86±4.46	89.04±2.05	80.53±6.17	82.48±0.59	15.88±2.94	89.26±0.87	28.27±5.05
	Chironex (Ours)	86.23±0.54	**98.52±1.61**	88.14±0.52	**98.26±0.80**	87.10±0.83	**93.37±3.29**	89.17±2.01	**95.69±4.14**	**89.24±0.76**	**91.52±3.99**	89.06±0.79	**97.97±3.71**
Tiny-ImageNet	No Attack	57.36±0.07	-	56.27±0.20	-	56.03±0.18	-	56.83±1.27	-	57.13±0.25	-	56.92±0.47	-
	Backdoor (Baseline)	56.29±0.27	**98.96±0.15**	55.39±0.31	0.00±0.00	56.73±0.91	0.00±0.00	56.39±0.73	0.00±0.00	55.07±0.76	0.00±0.00	**57.88±0.50**	0.00±0.00
	DBA	55.28±0.42	98.29±0.57	**57.31±0.63**	0.00±0.00	56.04±0.16	0.00±0.00	**57.29±1.70**	0.00±0.00	56.19±0.33	0.00±0.00	57.82±0.37	0.00±0.00
	Neurotoxin	56.82±0.12	98.78±0.06	56.67±0.17	59.98±0.40	56.71±0.34	10.48±1.95	55.85±0.96	0.00±0.00	56.83±0.39	0.00±0.00	57.14±0.67	0.81±0.33
	SMP	**57.06±0.05**	97.26±0.09	55.81±0.13	96.93±0.23	56.27±0.66	90.47±1.79	54.19±0.92	0.00±0.00	55.61±0.21	0.00±0.00	56.97±0.07	0.00±0.00
	Chironex (Ours)	56.91±0.29	98.37±0.48	56.17±0.14	**97.85±0.68**	**57.43±0.29**	**98.72±0.94**	55.21±1.03	**96.52±0.27**	**57.19±0.96**	**94.08±2.04**	57.03±0.34	**95.71±0.83**

CIFAR-10 is slightly lower than on other datasets, Chironex still outperforms other attacks and yields the highest ASR, roughly 95.4%. Chironex also outputs MA that is significantly close to the baseline attack, with a 0.27% gap. For a more complex dataset Tiny-ImageNet, Chironex provides excellent performance on stealthiness against SOTA defenses. Although its MA values are lower than others', Chironex is still more stealthy and delivers nearly the best ASR, >95%. As compared to its ASR results on other datasets, there is a slight drop on Tiny-ImageNet, which is roughly 1%∼2%.

C. Evaluation of Attack With Defenses

Besides the norm clipping defense [55], we evaluated attack effectiveness against four robust FL aggregation rules, namely Median, RLR, FLAME, and DeepSight. For robust FL, we used RLR threshold $\rho = 40$ for each setting and set FLAME and DeepSight minimum cluster size $s = \frac{10}{2} + 1 = 6$. The results in the i.i.d. setting are shown in Table II. Since Edge-Case is not applicable on FMNIST and Tiny-ImageNet, its results are not given in Table II. Chironex can provide excellent robustness against secure FL aggregation algorithms. Under FLAME and DeepSight on MNIST and FMNIST, Chironex is more robust than (most of) others and maintains high ASR values, >93%. Whilst the MA values on CIFAR-10 are slightly lower than those of other attacks, Chironex still delivers a solid ASR >93% under Median and RLR. Under DeepSight on CIFAR-10, Chironex achieves nearly 2-3× improvement on ASR as compared to others. We see that Chironex distinctively outperforms others against FLAME and DeepSight.

The FNR/FPR results against FLAME are shown in Figure 2. Chironex achieves an overwhelming advantage on attack effectiveness under FLAME. Its FNR can reach approx. 80% on MNIST and FMNIST while still standing at nearly 60% on CIFAR-10, which is nearly 4-8× higher than others. This indicates that FLAME could be more likely to cluster most of malicious updates into benign ones under Chironex

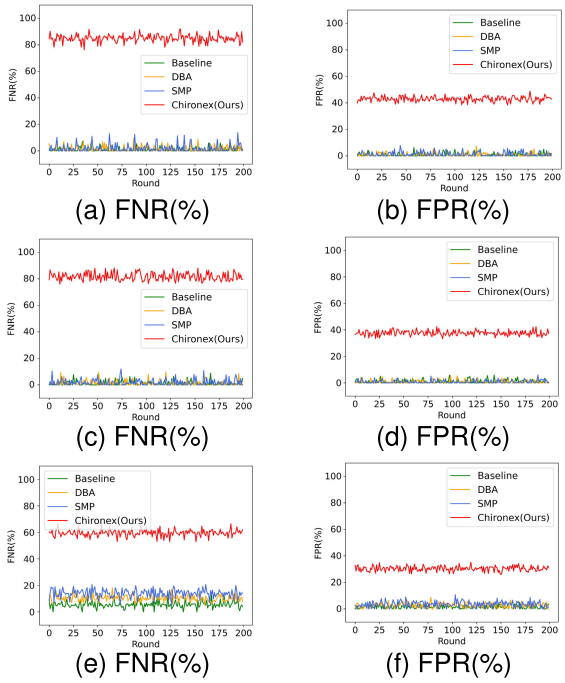

Fig. 2. Attack Stealthiness via FNR and FPR against FLAME for different attacks. (a)-(b): MNIST, (c)-(d): FMNIST and (e)-(f): CIFAR-10.

than other attacks. As for FPR, Chironex maintains around 40% but others are seriously restricted under 10%.

To further verify the stealthiness of Chironex, we tested principal component analysis (PCA) of model parameter updates between benign and malicious agents under FLAME, see Figure 3. We reduced model parameters to three dimensions and compared the differences. The model update of baseline attack is distinguishable from those benign updates while malicious updates given by Chironex can be mis-identified as "benign" (where the low dimensional malicious parameters stay extremely close to benign ones). This is so because: 1) we constrain the impact of those backdoor neurons to make them be "seen" as clean neurons; and 2) we manipulate the malicious parameters to get close to

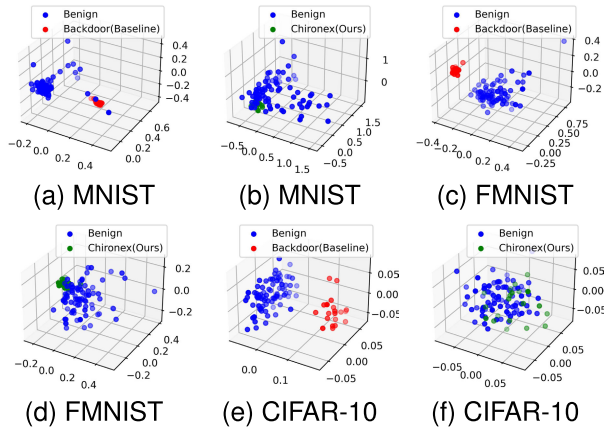

Fig. 3. Principal component analysis (PCA) of model parameter updates for benign and malicious agents under FLAME.

the benign ones trained by the attacker. These reduce the differences between benign and malicious updates and thus make Chironex stealthy.

Why does Chironex achieve distinctive effectiveness under robust FL? Due to our design at the parameter level, malicious and benign updates are indistinguishable.

• Against Median. Since the malicious and benign parameters are close to each other, the probability of being selected from all the updates is increased as compared to other attacks. Thus, Chironex can evade Median.

• Against RLR. The malicious updates generated by Chironex have almost the same sign of the parameters at each dimension because of the parameter similarity. Since the sum of parameter sign of our method is smaller than its threshold ρ, RLR can easily treat malicious updates as benign.

• Against FLAME and DeepSight. SMP obtains a lower ASR on CIFAR-10 than Chironex as its core design is to limit the distance between updates based on the global model of the previous round. By restraining the distance directly between malicious and benign parameters from the attacker, one may yield a precise estimation of the update at the current round. But this approach can be still identified by investigating the difference at the output layer between benign and malicious models. To bypass the detection, we focus on limiting the impact of backdoor-sensitive neurons. By introducing two loss function terms $\mathcal{L}_{tns}$ and $\mathcal{L}_{l_2dist}$, we guarantee that malicious parameters have no distinct contributions, allowing our attack to provide excellent stealthiness.

D. Hyperparameter Analysis

1) Influence of PDR and the Proportion of Compromised Agents η: We examined our attack effectiveness and robustness under different PDR and η settings. We set PDR $= 20\%$ and $\eta = 1\%$ as default. In Table III, we give the performance with FedAvg on three datasets. Chironex works very well with PDR $= 50\%$, reaching above 98.80% (MA) and 99.55% (ASR) on MNIST and 91.30% (MA) and 98.68% (ASR) on FMNIST. Although the performance on Tiny-ImageNet is worse than those of MNIST and FMNIST, Chironex still can provide around 59.12% (MA) and 85.31% (ASR) even when PDR $= 1\%$. We note decreasing PDR could improve

TABLE III
ATTACK PERFORMANCE VIA MA (%) AND ASR (%) WITH FEDAVG FOR DIFFERENT PDR SETTINGS

Dataset	PDR	MA	ASR
MNIST	1%	99.13±0.14	97.14±0.43
	10%	99.27±0.28	97.21±0.23
	20%	99.03±0.13	98.46±0.07
	50%	98.80±0.08	99.55±0.11
FMNIST	1%	92.34±0.57	95.64±0.95
	10%	91.62±0.59	96.37±0.71
	20%	91.67±0.45	97.90±0.18
	50%	91.30±0.11	98.68±0.87
Tiny-ImageNet	1%	59.12±0.20	85.31±1.38
	10%	58.79±0.39	87.79±1.02
	20%	57.53±0.78	91.12±1.78
	50%	56.36±0.66	92.34±1.90

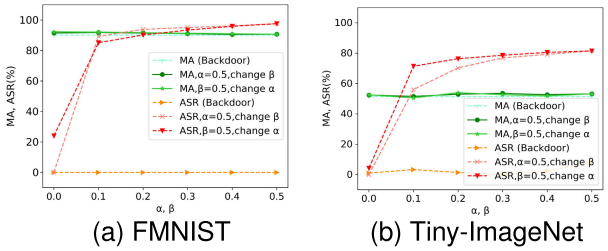

Fig. 4. MA (%) and ASR (%) of Backdoor (Baseline) and Chironex (Ours) with FLAME for different α, β settings.

MA but weaken ASR. We also investigated the performance with different η. In Table IV, we see that using a small η can increase MA but slightly harm ASR against defenses. Chironex presents a relatively weak performance on ASR ($< 50\%$) when the adversary controls only one malicious client.

2) Influence of α, β: We used different hyperparameter settings to visualize the influence of stealthiness budget α, β on various datasets. The α, β limit the impacts of backdoor neurons and the similarity between benign and malicious updates respectively. We restrained that $\alpha, \beta \leq 1$ considering the impact of the main task and backdoor task training. The results against FLAME are in Figure 4. For FMNIST, reducing α (from 0.5 to 0.1) with a fixed β can make our malicious updates less indistinguishable because the contributions of backdoor neurons do increase. Reversely, fixing an α with decreasing β could perform a moderate decline on stealthiness. If we keep reducing either of them, Chironex suffers from a significant drop on ASR. For example, ASR is only around 20% with $\alpha = 0.0$ and fully declines to 0% with $\beta = 0.0$. Whilst $\alpha, \beta = 0.5$, Chironex can achieve the best stealthiness against FLAME, with almost 100% ASR. The similar experimental results can be observed on Tiny-ImageNet.

3) Influence of the Degree of Non-i.i.d. Level q: We set several degrees of FEMNIST and FMNIST data distribution q to test Chironex in the non-i.i.d context. In Figure 5, we present its performance with robust FL aggregation rules as compared to baseline backdoor attacks on FEMNIST. From $q = 0.1$ to $q = 0.9$, Chironex achieves high ASR (>90%) and maintains MA with a small descent (<5%). But under RLR, it has a continuous drop on ASR, obtaining around 75% with $q = 0.9$. While $q = 1$, both MA and ASR experience

TABLE IV
ATTACK PERFORMANCE VIA MA (%) AND ASR (%) FOR DIFFERENT PROPORTION OF COMPROMISED AGENTS η (%)

	Defense →	No Defense		Median		RLR		FLAME		DeepSight	
Dataset ↓	η ↓	MA	ASR	MA	ASR	MA	ASR	MA	ASR	MA	ASR
MNIST	0.1	99.23±0.03	52.63±0.72	98.70±0.12	49.36±1.35	95.85±0.98	50.91±1.01	99.17±0.10	44.74±3.67	98.46±0.28	46.59±2.16
	0.5	99.14±0.02	98.19±0.86	98.62±0.09	97.99±1.74	94.72±0.44	97.68±0.89	99.04±0.10	91.13±4.97	98.28±0.39	92.66±2.27
	1	99.09±0.08	99.02±1.19	98.59±0.20	98.75±1.21	95.33±1.08	98.83±1.17	98.85±0.05	91.89±3.78	98.16±0.15	93.04±1.92
	5	99.03±0.04	99.25±0.79	98.50±0.05	99.10±1.63	95.96±1.93	98.29±0.15	98.89±0.09	92.53±4.56	97.81±0.26	93.72±1.77
	10	98.96±0.03	99.38±0.62	98.53±0.08	99.52±1.04	94.72±1.13	98.88±0.09	98.84±0.07	93.72±4.26	97.03±0.10	94.28±1.26
FMNIST	0.1	92.59±0.10	46.81±0.47	90.95±0.28	39.92±0.36	83.25±0.65	43.27±0.54	91.21±0.23	45.01±0.77	92.29±0.83	40.24±0.90
	0.5	92.38±0.27	96.19±1.49	90.85±0.26	90.65±0.83	82.31±0.24	98.95±0.14	91.20±0.16	95.24±0.22	91.77±0.37	93.79±1.70
	1	91.93±0.36	96.45±0.88	90.46±0.14	91.80±0.71	82.05±0.51	99.07±0.98	91.81±0.37	96.36±0.47	91.71±0.52	94.52±1.03
	5	91.46±0.85	97.80±0.30	90.35±0.82	92.20±0.40	81.92±0.30	99.34±0.56	91.24±0.35	97.68±1.10	91.90±0.15	94.77±1.15
	10	91.44±0.14	97.83±0.34	90.79±0.12	93.46±1.32	81.86±0.20	99.80±0.07	90.65±0.22	97.60±0.67	91.29±0.13	95.03±0.80
CIFAR-10	0.1	90.43±0.22	35.06±1.55	90.03±0.73	33.96±2.29	80.39±2.95	35.68±3.50	91.07±0.66	31.24±2.32	90.71±0.41	20.37±5.74
	0.5	90.41±0.64	93.48±2.11	90.55±0.83	85.79±2.66	89.81±2.86	95.40±3.21	90.12±0.92	97.63±3.79	91.56±0.09	98.37±4.25
	1	89.73±0.85	93.76±0.81	88.11±0.36	90.05±2.43	89.84±0.77	93.80±3.66	89.28±0.42	93.53±3.55	89.70±0.94	96.26±3.73
	5	87.47±0.09	96.94±1.58	87.84±0.86	92.97±2.34	88.91±1.73	84.21±3.76	85.69±0.46	97.93±3.64	88.25±0.22	98.10±3.76
	10	86.23±0.54	98.52±1.61	87.10±0.83	93.37±3.29	89.17±2.01	95.69±4.14	83.24±0.76	95.52±3.99	89.06±0.79	98.97±3.71

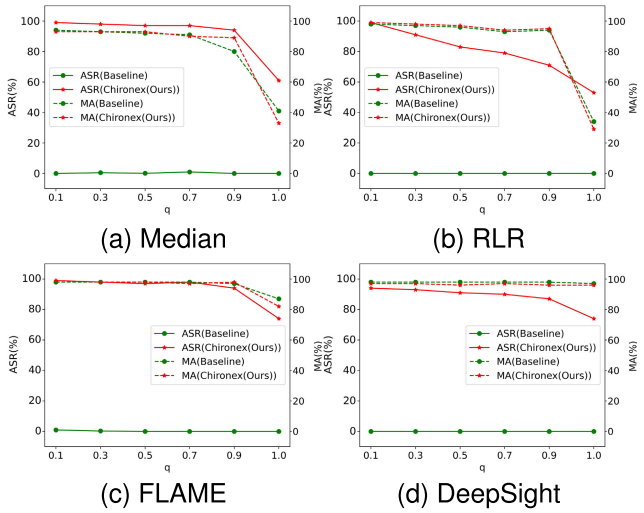

Fig. 5. Attack Performance via MA (%) and ASR (%) for different degrees of non-i.i.d. setting on FEMNIST. We compare Chironex (Ours) to Baseline on different defenses.

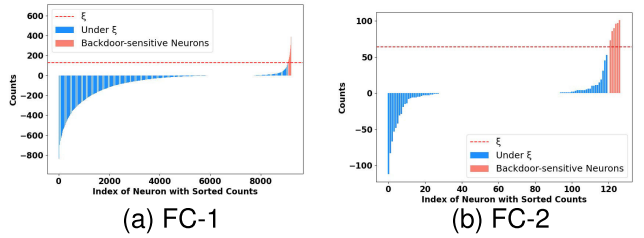

Fig. 6. The differences between P_{bd} and P_c of each layer in ascend order by TNS for classic CNN architecture (including 2 hidden dense layers) on FMNIST.

a sharp decline (around 30% on ASR and $>$ 50% on MA), against Median and RLR. This is so because the malicious agents cannot maintain the similarity between malicious and benign updates, which makes the attack easily detectable by the defenses. However, we see that Chironex performs more stable with $q = 1$ under FLAME and DeepSight (both $<$20% decrease on MA and ASR).

4) The Disparity of Neuron Lists P_{bd} and P_c: In Figure 6, we demonstrate the distinction in ascending order between the backdoor and clean neuron lists by subtracting approach. The results show that certain backdoor neurons of each layer do contribute to the backdoor task noticeably and significantly but they deliver no influence on the main task.

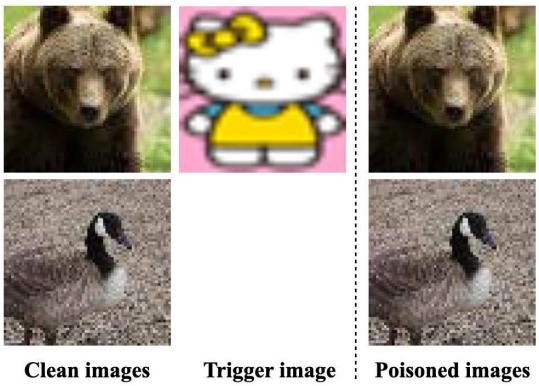

Fig. 7. Visualization of poisoned images on Tiny-ImageNet. Our attack provides practical natural stealthiness of poisoned samples.

E. Trigger Visualization

Although the attacker can arbitrarily choose the trigger because the local dataset is not visible to the server in the context of FL, using a more stealthy trigger naturally decreases the cosine dissimilarity between benign and malicious parameters [13], which could make our malicious updates more robust against defenses. To verify the natural stealthiness of the designed trigger under human inspection, we showcase poisoned samples on CIFAR-10 via our frequency trigger function in Figure 7. The results confirm that Chironex achieves sufficient natural stealthiness that can evade human inspection.

VI. CONCLUSION AND DISCUSSION

We designed an effective and stealthy backdoor attack throught frequency domain against FL by constraining the influence of backdoor neurons and enforcing backdoor parameters to update towards benign parameters. The empirical experiments show that our design can achieve a practical attack performance and evade most of the current defending strategies and human inspection. We hope this work could inspire further studies in developing secure and robust FL aggregation algorithms.

A. Discussion

In this work, we concentrate on various computer vision tasks, which have been the focus of numerous existing works [13], [17]. In the future, we intend to expand the scope of this work to other machine learning tasks, e.g., natural language processing. Chironex requires additional computational costs as we apply the step-forward approach. To reduce the costs, we may allow malicious agents to collude together to obtain a shared malicious update by split learning. The focus of Chironex relies on stealthiness rather than persistence (e.g., Neurotoxin [19]) on attack effectiveness. Neurotoxin manipulates malicious parameters based on gradients in magnitude, which is different from Chironex focusing on constraining the contributions of backdoor neurons (by smoothing their contributions to other "less-influence" neurons). It produces a clear increase in the dissimilarity of parameters and thus it can't provide the same level of stealthiness as ours. The dissimilarity difference can be addressed in Chironex by constraining the contribution of backdoor neuron parameters (i.e, reducing the cosine dissimilarity between benign and malicious parameters). We state that persistence is orthogonal with the main focus of this work, and we leave it as an open problem.

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 *Proc. 20th Int. Conf. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[2] S. Niknam, H. S. Dhillon, and J. H. Reed, "Federated learning for wireless communications: Motivation, opportunities, and challenges," *IEEE Commun. Mag.*, vol. 58, no. 6, pp. 46–51, Jun. 2020.

[3] Y. Liu et al., "Vertical federated learning: Concepts, advances, and challenges," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 7, pp. 3615–3634, Jul. 2024.

[4] L. Fu, H. Zhang, G. Gao, M. Zhang, and X. Liu, "Client selection in federated learning: Principles, challenges, and opportunities," *IEEE Internet Things J.*, vol. 10, no. 24, pp. 21811–21819, Dec. 2023.

[5] R. Myrzashova, S. H. Alsamhi, A. V. Shvetsov, A. Hawbani, and X. Wei, "Blockchain meets federated learning in healthcare: A systematic review with challenges and opportunities," *IEEE Internet Things J.*, vol. 10, no. 6, pp. 14418–14437, Aug. 2023.

[6] J. Zhang, S. Guo, J. Guo, D. Zeng, J. Zhou, and A. Zomaya, "Towards data-independent knowledge transfer in model-heterogeneous federated learning," *IEEE Trans. Comput.*, vol. 72, no. 10, pp. 2888–2901, Oct. 2023.

[7] H. Li et al., "FedTP: Federated learning by transformer personalization," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, May 23, 2023, doi: 10.1109/TNNLS.2023.3269062.

[8] M. Kesici, B. Pal, and G. Yang, "Detection of false data injection attacks in distribution networks: A vertical federated learning approach," *IEEE Trans. Smart Grid*, early access, May 10, 2024, doi: 10.1109/TSG.2024.3399396.

[9] I. Dayan et al., "Federated learning for predicting clinical outcomes in patients with COVID-19," *Nature Med.*, vol. 27, no. 10, pp. 1735–1743, 2021.

[10] A. Nguyen et al., "Deep federated learning for autonomous driving," in *Proc. IEEE Intell. Veh. Symp. (IV)*, Jun. 2022, pp. 1824–1830.

[11] G. Baruch, M. Baruch, and Y. Goldberg, "A little is enough: Circumventing defenses for distributed learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.

[12] A. N. Bhagoji, S. Chakraborty, P. P. Mittal, and S. Calp, "Analyzing federated learning through an adversarial lens," in *Proc. 36th Int. Conf. Mach. Learn.*, May 2019, pp. 634–643.

[13] C. Xie, K. Huang, P.-Y. Chen, and B. Li, "DBA: Distributed backdoor attacks against federated learning," in *Proc. Int. Conf. Learn. Represent.*, 2019, ppp. 1–19.

[14] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 2938–2948.

[15] H. Wang et al., "Attack of the tails: Yes, you really can backdoor federated learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 16070–16084.

[16] H. Li et al., "3DFed: Adaptive and extensible framework for covert backdoor attack in federated learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2023, pp. 1893–1907.

[17] H. Zhang, J. Jia, J. Chen, L. Lin, and D. Wu, "A3FL: Adversarially adaptive backdoor attacks to federated learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 36, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., New Orleans, LA, USA: Curran Associates, 2023, pp. 61213–61233. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2023/file/c07d71ff0bc042e4b9acd626a79597fa-Paper-Conference.pdf

[18] R. J. G. B. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Proc. Pacific–Asia Conf. Knowl. Discovery Data Mining*. Gold Coast, QLD, Australia: Springer, 2013, pp. 160–172.

[19] Z. Zhang et al., "Neurotoxin: Durable backdoors in federated learning," in *Proc. 39th Int. Conf. Mach. Learn.*, Jul. 2022, pp. 26429–26446.

[20] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proc. Int. Conf. Mach. Learn.*, Jul. 2018, pp. 5650–5659.

[21] M. S. Ozdayi, M. Kantarcioglu, and Y. R. Gel, "Defending against backdoors in federated learning with robust learning rate," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 10, pp. 9268–9276.

[22] T. D. Nguyen et al., "FLAME: Taming backdoors in federated learning," in *Proc. 31st USENIX Security Symp. (USENIX Security)*, Boston, MA, USA, Aug. 2022, pp. 1415–1432.

[23] P. Rieger, T. D. Nguyen, M. Miettinen, and A.-R. Sadeghi, "DeepSight: Mitigating backdoor attacks in federated learning through deep model inspection," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2022, pp. 1–18.

[24] D. Yin, R. G. Lopes, J. Shlens, E. D. Cubuk, and J. Gilmer, "A Fourier perspective on model robustness in computer vision," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019.

[25] Z.-Q. J. Xu, Y. Zhang, and Y. Xiao, "Training behavior of deep neural network in frequency domain," in *Proc. Int. Conf. Neural Inf. Process.*, Sydney, NSW, Australia. Springer, 2019, pp. 264–274.

[26] T. Wang, Y. Yao, F. Xu, S. An, H. Tong, and T. Wang, "An invisible black-box backdoor attack through frequency domain," in *Proc. Eur. Conf. Comput. Vis.* Tel Aviv-Yafo, Israel: Springer, 2022, pp. 396–413.

[27] Y. Feng, B. Ma, J. Zhang, S. Zhao, Y. Xia, and D. Tao, "FIBA: Frequency-injection based backdoor attack in medical image analysis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 20876–20885.

[28] K. Bonawitz et al., "Practical secure aggregation for federated learning on user-held data," 2016, *arXiv:1611.04482*.

[29] B. Zhao, P. Sun, T. Wang, and K. Jiang, "FedInv: Byzantine-robust federated learning by inversing local model updates," in *Proc. AAAI Conf. Artif. Intell.*, 2022, vol. 36, no. 8, pp. 9171–9179.

[30] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proc. 3rd Mach. Learn. Syst. Conf.*, vol. 2, 2020, pp. 429–450.

[31] Y. Liu et al., "A communication efficient collaborative learning framework for distributed features," 2019, *arXiv:1912.11187*.

[32] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 6357–6368.

[33] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *Proc. Eur. Symp. Res. Comput. Secur.* Guildford, U.K.: Springer, 2020, pp. 480–501.

[34] Y. Dai and S. Li, "Chameleon: Adapting to peer images for planting durable backdoors in federated learning," in *Proc. 40th Int. Conf. Mach. Learn.*, 2023, pp. 6712–6725.

[35] A. Reisizadeh, F. Farnia, R. Pedarsani, and A. Jadbabaie, "Robust federated learning: The case of affine distribution shifts," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 21554–21565.

[36] V. Shejwalkar and A. Houmansadr, "Manipulating the Byzantine: Optimizing model poisoning attacks and defenses for federated learning," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2021, pp. 1–19.

[37] X. Cao, J. Jia, and N. Z. Gong, "Provably secure federated learning against malicious clients," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 8, pp. 6885–6893.

[38] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "FLTrust: Byzantine-robust federated learning via trust bootstrapping," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2021, pp. 1–18.
[39] C. Xie, M. Chen, P.-Y. Chen, and B. Li, "CRFL: Certifiably robust federated learning against backdoor attacks," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 11372–11382.
[40] Z. Zhang, X. Cao, J. Jia, and N. Z. Gong, "FLDetector: Defending federated learning against model poisoning attacks via detecting malicious clients," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2022, pp. 2545–2555.
[41] X. Fang and M. Ye, "Robust federated learning with noisy and heterogeneous clients," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2022, pp. 10072–10081.
[42] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.
[43] E. M. El Mhamdi, R. Guerraoui, and S. Rouault, "The hidden vulnerability of distributed learning in Byzantium," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 3521–3530.
[44] Z. Sun, P. Kairouz, A. T. Suresh, and H. B. McMahan, "Can you really backdoor federated learning?" 2019, *arXiv:1911.07963*.
[45] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, "signSGD: Compressed optimisation for non-convex problems," in *Proc. 35th Int. Conf. Mach. Learn.*, J. Dy and A. Krause, Eds., vol. 80, Jul. 2018, pp. 560–569.
[46] A. Panda, S. Mahloujifar, A. N. Bhagoji, S. Chakraborty, and P. Mittal, "SparseFed: Mitigating model poisoning attacks in federated learning with sparsification," in *Proc. Int. Conf. Artif. Intell. Statist.*, vol. 151, G. Camps-Valls, F. J. R. Ruiz, and I. Valera, Eds., Mar. 2022, pp. 7587–7624.
[47] 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. IEEE Symp. Security Privacy*, Oct. 2018, pp. 19–35.
[48] Y. LeCun. (1998). *The Mnist Database of Handwritten Digits*. [Online]. Available: http://yann.lecun.com/exdb/mnist/
[49] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*.
[50] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep. 0, 2009.
[51] S. Caldas et al., "LEAF: A benchmark for federated settings," 2018, *arXiv:1812.01097*.
[52] O. Russakovsky et al., "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[53] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[54] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," 2017, *arXiv:1712.05526*.
[55] V. Shejwalkar, A. Houmansadr, P. Kairouz, and D. Ramage, "Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2022, pp. 1354–1371.

Yanqi Qiao is currently pursuing the Ph.D. degree with the Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology. His current research interests include machine learning and AI safety.

Dazhuang Liu is currently pursuing the Ph.D. degree with the Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology. His current research interests include machine learning security.

Rui Wang received the B.S. degree in applied physics from Beijing University of Posts and Telecommunications in 2017, the M.S. degree in cyber security from the University of Southampton in 2018, and the Ph.D. degree from the Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, in 2024. He is currently a Post-Doctoral Researcher with Delft University of Technology. His current research interests including the privacy and security of federated learning.

Kaitai Liang (Member, IEEE) received the Ph.D. degree from the City University of Hong Kong.

He is a tenured Faculty Member (UD1) with the Cybersecurity Group, Delft University of Technology (TU Delft). He was an Information Security Researcher with I2R, A*STAR, Singapore; a Post-Doctoral Researcher with Aalto University, Finland, supported by AoF funding; and held a permanent assistant professor position in secure systems with the University of Surrey, before joining TU Delft. During the Ph.D. and post-doctoral studies, he was a Visiting Researcher with KU Leuven, Belgium; UCL, U.K., IBM; and Northeastern University, USA. With over ten years of experience on cybersecurity research and development, his main focus is on the design and implementation of cryptographic protocols to security. He has published a series of research works, applying information security and crypto tools to address cybersecurity challenges. These publications have appeared in high-tier international information security journals, e.g., IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY and IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING. Beyond his research contributions, he has played an active role in the academic community. He has served as a Technical Program Committee Member, the General Chair, and a Steering Committee Member for renowned international security and privacy conferences, including IEEE EuroS&P and IEEE CSF. In addition to his academic pursuits, he serves as an Associate Editor for esteemed journals, such as IEEE SYSTEMS JOURNAL and IEEE TRANSACTIONS ON ARTIFICIAL INTELLIGENCE.