# On the Vulnerability of Backdoor Defenses for Federated Learning

**Pei Fang**[1], **Jinghui Chen**[2]

[1]Tongji University
[2]Pennsylvania State University
greilfang@gmail.com, jzc5917@psu.edu

## Abstract

Federated Learning (FL) is a popular distributed machine learning paradigm that enables jointly training a global model without sharing clients' data. However, its repetitive server-client communication gives room for backdoor attacks with aim to mislead the global model into a targeted misprediction when a specific trigger pattern is presented. In response to such backdoor threats on federated learning, various defense measures have been proposed. In this paper, we study whether the current defense mechanisms truly neutralize the backdoor threats from federated learning in a practical setting by proposing a new federated backdoor attack method for possible countermeasures. Different from traditional training (on triggered data) and rescaling (the malicious client model) based backdoor injection, the proposed backdoor attack framework (1) directly modifies (a small proportion of) local model weights to inject the backdoor trigger via sign flips; (2) jointly optimize the trigger pattern with the client model, thus is more persistent and stealthy for circumventing existing defenses. In a case study, we examine the strength and weaknesses of recent federated backdoor defenses from three major categories and provide suggestions to the practitioners when training federated models in practice.

## 1 Introduction

In recent years, Federated Learning (FL) (McMahan et al. 2017; Zhao et al. 2018) prevails as a new distributed machine learning paradigm, where many clients collaboratively train a global model without sharing clients' data. FL techniques have been widely applied to various real-world applications including keyword spotting (Leroy et al. 2019), activity prediction on mobile devices (Hard et al. 2018; Xu et al. 2021), smart sensing on edge devices (Jiang et al. 2020), etc. Despite FL's collaborative training capability, it usually deals with heterogeneous (non-i.i.d.) data distribution among clients and its formulation naturally leads to repetitive synchronization between the server and the clients. This gives room for attacks from potential malicious clients. Particularly, backdoor attack (Gu, Dolan-Gavitt, and Garg 2019), which aims to mislead the model into a targeted misprediction when a specific trigger pattern is presented by stealthy data poisoning, can be easily implemented and hard to detect from the server's perspective. The feasibility of backdoor attacks on plain federated learning has been studied in (Bhagoji et al. 2019; Bagdasaryan et al. 2020; Xie et al. 2019; Zhang* et al. 2022). Such backdoor attacks can be effectively implemented by replacing the global FL model with the attackers' malicious model through carefully scaling model updates with well-designed triggers, and the attacks can successfully evade many different FL setups (McMahan et al. 2017; Yin et al. 2018).

The possible backdoor attacks in federated learning arouse a large number of interest on possible defenses that could mitigate the backdoor threats. Based on the different defense mechanisms they adopt, the federated backdoor defenses can be classified into three major categories: *model-refinement*, *robust-aggregation*, and *certified-robustness*. *Model-refinement* defenses attempt to refine the global model to erase the possible backdoor, through methods such as fine-tuning (Wu et al. 2020) or distillation (Lin et al. 2020; Sturluson et al. 2021). Intuitively, distillation or pruning-based FL can also be more robust to current federated backdoor attacks as recent studies on backdoor defenses (Li et al. 2021; Liu, Dolan-Gavitt, and Garg 2018) have shown that such methods are effective in removing backdoor from general (non-FL) backdoored models. On the other hand, different from FedAvg (McMahan et al. 2017) and its variants (Karimireddy et al. 2020; Li et al. 2020; Wang et al. 2020b) which directly average the participating clients' parameters, the *robust-aggregation* defenses exclude the malicious (ambiguous) weights and gradients from suspicious clients through anomaly detection, or dynamically re-weight clients' importance based on certain distance metrics (geometric median, etc.). Examples include Krum (Blanchard et al. 2017), Trimmed Mean (Yin et al. 2018), Bulyan (Guerraoui, Rouault et al. 2018) and Robust Learning Rate (Ozdayi, Kantarcioglu, and Gel 2020). Note that some of the *robust-aggregation* defenses are originally proposed for defending model poisoning attack (Byzantine robustness) yet they may also be used for defending backdoors. The last kind, *certified robustness* aims at providing certified robustness guarantees that for each test example, i.e., the prediction would not change even some features in local training data of malicious clients have been modified within certain constraint. For example, CRFL (Xie et al. 2021) exploits clipping and smoothing on model parameters, which yields a

sample-wise robustness certification with magnitude-limited backdoor trigger patterns. Provable FL (Cao, Jia, and Gong 2021) learns multiple global models, each with a random client subset and takes majority voting in prediction, which shows provably secure against malicious clients.

Despite the efforts in backdoor defense in federated learning, there exist many constraints and limitations for defenses from these three major categories. First, the effectiveness of the *model-refinement* defenses relies on whether the refinement can fully erase the backdoor. Adversaries may exploit this and design robust backdoor patterns that are persistent, stealthy and thus hard to erase. Second, *robust aggregation* defenses are usually based on i.i.d. assumptions of each participant's training data, which does not hold for the general federated learning scenario where participant's data are usually non-i.i.d. Existing attacks (Bagdasaryan et al. 2020) have shown that in certain cases, such defense techniques make the attack even more effective. Moreover, to effectively reduce the attack success rate of the possible backdoor attacks, one usually needs to enforce stronger robust aggregation rules, which can in turn largely hurt the normal federated training progress. Lastly, *certified robustness* approaches enjoy theoretical robust guarantees, yet also have quite strong requirements and limitations such as a large amount of model training or a strict limit on the magnitude of the trigger pattern. Also, certified defenses usually lead to relatively worse empirical model performances.

To rigorously evaluate current federated backdoor defenses and examine whether the current mechanisms truly neutralize the backdoor threats, we propose a more persistent and stealthy federated backdoor attack for circumventing most existing defenses. Through comprehensive experiments and in-depth case study on several state-of-the-art federated backdoor defenses, we summarize our main contributions and findings as follows:

- We propose a persistent and stealthy backdoor attack for federated learning. Instead of traditional training (on triggered data) and rescaling (malicious client updates) based backdoor injection, our attack selectively flips the signs of a small proportion of model weights and jointly optimizes the trigger pattern with client local training.
- The proposed attack does not explicitly scale the updated weights (gradients) and can be universally applied to various architectures beyond convolutional neural networks, which is of independent interest to general backdoor attack and defense studies.
- In a case study, we examine the effectiveness of several recent federated backdoor defenses from three major categories and give practical guidelines for the choice of the backdoor defenses for different settings.

## 2 Proposed Approach

**Federated Learning Setup** Suppose we have $K$ participating clients, each of which has its own dataset $\mathcal{D}_i$ with size $n_i$ and $N = \sum_i n_i$. At the $t$-th federated training round, the server sends the global model $\boldsymbol{\theta}_t$ to a randomly-selected subset of $m$ clients. The clients then perform $K$ steps of local training to obtain $\boldsymbol{\theta}_t^{i,K}$ based on the global model $\boldsymbol{\theta}_t$, and send the updates $\boldsymbol{\theta}_t^{i,K} - \boldsymbol{\theta}_t$ back to the server. The server aggregates the updates with specific rules to get a new global model $\boldsymbol{\theta}_{t+1}$ for the next round. As the standard FedAvg (McMahan et al. 2017), the server adopts sample-weighted aggregation to average the $m$ received updates:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \frac{1}{N}\sum_{i=1}^{m} n_i(\boldsymbol{\theta}_t^{i,K} - \boldsymbol{\theta}_t) \tag{2.1}$$

Various variants of FedAvg have also been proposed (Karimireddy et al. 2020; Li et al. 2020; Wang et al. 2020b; Reddi et al. 2020). In this work, we adopt the most commonly used FedAvg (McMahan et al. 2017) method as our standard federated learning baseline.

**Backdoor Attacks in FL** Assume there exists one or several malicious clients with goal to manipulate local updates to inject a backdoor trigger into the global model such that when the trigger pattern appears in the inference stage, the global model would give preset target predictions $y_{\text{target}}$. In the meantime, the malicious clients do not want to tamper with the model's normal prediction accuracy on clean tasks (to keep stealthy). Therefore, the malicious client has the following objectives:

$$\begin{aligned}\min_{\boldsymbol{\theta}} \mathcal{L}_{\text{train}}(\mathbf{x}, \mathbf{x}', y_{\text{target}}, \boldsymbol{\theta}) := \frac{1}{n_i}\sum_{k=1}^{n_i} \ell(f_{\boldsymbol{\theta}}(\mathbf{x}_k), y_k) \\ +\lambda \cdot \ell(f_{\boldsymbol{\theta}}(\mathbf{x}_k'), y_{\text{target}}) + \alpha \cdot ||\boldsymbol{\theta} - \boldsymbol{\theta}_t||_2^2,\end{aligned} \tag{2.2}$$

where $\mathbf{x}_k' = (\mathbf{1} - \mathbf{m}) \odot \mathbf{x}_k + \mathbf{m} \odot \boldsymbol{\Delta}$ is the backdoored data and $\boldsymbol{\Delta}$ denotes the associated trigger pattern, $\mathbf{m}$ denotes the trigger location mask, and $\odot$ denotes the element-wise product. The first term in (2.2) is the common empirical risk minimization while the second term aims at injecting the backdoor trigger into the model. The third term is usually employed additionally to enhance the attack stealthiness by minimizing the distance to the global model (for bypassing anomaly detection-based defenses). The loss function $\ell$ is usually set as the CrossEntropy Loss. $\lambda$ and $\alpha$ control the trade-off between the three tasks. Most existing backdoor attacks on FL (Bagdasaryan et al. 2020; Bhagoji et al. 2019; Xie et al. 2019) are based on iteratively training triggered data samples over the loss (2.2) or its variants. Then the backdoored local updates will be rescaled in order to have enough influence on the final global model update on the server. Such a process lasts by rounds until the global model reaches a high attack success rate.

**Threat Model** We suppose that the malicious attacker has full control of their local training processes, such as backdoor data injection, trigger pattern, and local optimization. The scenario is practical since the server can only get the trained model from clients without the information on how the model is trained. Correspondingly, the malicious attacker is unable to influence the operations conducted on the central server such as changing the aggregation rules, or tampering with the model updates of other benign clients.

### 2.1 Focused Flip Federated Backdoor Attack

Most existing backdoor attacks on FL (Bagdasaryan et al. 2020; Bhagoji et al. 2019) are based on training on triggered data and rescaling the malicious updates to inject the backdoor. There are several major downsides: (1) it requires

rescaling the updates to dominate the global model update, rendering the updates quite different from other clients and easier to be defended by clipping or anomaly detection; (2) the dominating malicious updates can also significantly impair the prediction performance of the global model, which makes the backdoored model less attractive to be adopted.

In this section, we propose **F**ocused-**F**lip **F**ederated **B**ackdoor **A**ttack (F3BA), in which the malicious clients only compromise a small fraction of the least important model parameters through *focused weight sign manipulation*. The goal of such weight sign manipulation is to cause a strong activation difference in each layer led by the trigger pattern while keeping the modification footprint and influence on model accuracy minimal. A sketch of our proposed attack is illustrated in Figure 1. Let's denote the current global model as $\boldsymbol{\theta}_t^{i,0} := \{\mathbf{w}_t^{[1]}, \mathbf{w}_t^{[2]}, .., \mathbf{w}_t^{[L]}\}$ and each layer's output as $\mathbf{z}^{[1]}(\cdot), \mathbf{z}^{[2]}(\cdot), .., \mathbf{z}^{[L]}(\cdot)$. Generally, our attack can be divided into three steps:

**Step 1: Search candidate parameters for manipulation**: We only manipulate a small fraction of candidate parameters in the model that are of the least importance to the normal task to make it have a slight impact on the natural accuracy. Specifically, we introduce *movement-based* importance score to identify candidate parameters for manipulation, which is inspired by the movement pruning (Sanh, Wolf, and Rush 2020). Specifically, the importance of each parameter $\boldsymbol{S}_t^{[j]}$ is related to both its weight and gradient: $\boldsymbol{S}_t^{[j]} = -\frac{\partial \mathcal{L}_\text{g}}{\partial \mathbf{w}_t^{[j]}} \odot \mathbf{w}_t^{[j]}$, where $\mathcal{L}_\text{g}$ is the global training loss and $\odot$ denotes the elementwise product[1]. We make two major changes on $\boldsymbol{S}_t^{[j]}$ for our federated backdoor attack:

- In our federated setting, it is hard to obtain the global loss $\mathcal{L}_\text{g}$ since the attack is carried only on the malicious workers. We simply approximate the partial derivative with the model difference $-\frac{\partial \mathcal{L}_\text{g}}{\partial \mathbf{w}_t^{[j]}} \approx \mathbf{w}_t^{[j]} - \mathbf{w}_{t-1}^{[j]}$. When $t = 0$ we simply generate a random importance score[2] $\boldsymbol{S}_0^{[j]}$;
- To handle defense mechanisms with different emphasizes, we extend it into two importance metrics (Directional Criteria and Directionless Criteria) and choose[3] the one that best exploits the weakness of the defense:

$$\textbf{Directional: } \boldsymbol{S}_t^{[j]} = (\mathbf{w}_t^{[j]} - \mathbf{w}_{t-1}^{[j]}) \odot \mathbf{w}_t^{[j]}, \tag{2.3}$$

$$\textbf{Directionless: } \boldsymbol{S}_t^{[j]} = |(\mathbf{w}_t^{[j]} - \mathbf{w}_{t-1}^{[j]}) \odot \mathbf{w}_t^{[j]}|. \tag{2.4}$$

Given the importance score $\boldsymbol{S}_t^{[j]}$, we choose the least important parameters in each layer as candidate parameters. We define $\mathbf{m}_s^{[j]}$ as a mask that selects the $s\%$ lowest scores in $\boldsymbol{S}_t^{[j]}$ and ignore the others. In practice, setting $s = 1\%$ for the model parameters is usually sufficient for our attack.

[1]More explanations regarding this movement-based importance score can be found in the Appendix.

[2]In practice, since only a subset of clients participate in the training, the malicious client keeps its last received global model until it is chosen for training and compute the model difference.

[3]A detailed discussion on how to choose the appropriate criteria can be found in the Appendix.

**Step 2: Flip the sign of candidate parameters**: Our next goal is to manipulate the parameters to enhance their sensitivity to the trigger by flipping their signs. Take the simple CNN model as an example[4]. We start flipping from the first convolutional layer. For a trigger pattern[5] $\boldsymbol{\Delta}$, to maximize the activation in the next layer, we flip $\mathbf{w}^{[1]}$'s signs if they are different from the trigger's signs in the same position:

$$\mathbf{w}^{[1]} = \mathbf{m}_s^{[1]} \odot \text{sign}\,(\boldsymbol{\Delta}) \odot |\mathbf{w}^{[1]}| + (\mathbf{1} - \mathbf{m}_s^{[1]}) \odot \mathbf{w}^{[1]}, \tag{2.5}$$

where $\mathbf{m}_s^{[1]}$ is the candidate parameter mask generated in Step 1. Through (2.5), the activation in the next layer is indeed enlarged when the trigger pattern is present. For the subsequent layers, we flip the signs of the candidate parameters similarly. The only difference is that after the sign-flip in the previous layer $j-1$, we feed a small set of validation samples $\mathbf{x}_v$ and compute the activation difference of layer $j-1$ caused by adding the trigger pattern $\boldsymbol{\Delta}$ on $\mathbf{x}_v$:

$$\begin{aligned} \boldsymbol{\delta} &= \sigma(\mathbf{z}^{[j-1]}(\mathbf{x}'_v)) - \sigma(\mathbf{z}^{[j-1]}(\mathbf{x}_v)), \\ \text{where } \mathbf{x}'_v &= (1 - \mathbf{m}) \odot \mathbf{x}_v + \mathbf{m} \odot \boldsymbol{\Delta} \end{aligned} \tag{2.6}$$

$\sigma(\cdot)$ is the activation function for the network (e.g. ReLU function) and $\mathbf{x}'_v$ is the backdoor triggered validation samples. Similarly, we can flip the signs of the candidate parameters to maximize $\boldsymbol{\delta}$. This ensures that the last layer's activation is also maximized when the trigger pattern is presented.

**Step 3: Model training**: Although we have maximized the network's activation for the backdoor trigger in Step 2, the local model training step is still necessary due to: 1) the flipped parameters only maximize the activation but have not associated with the target label $y_\text{target}$, and the training step using (2.2) would bind the trigger to the target label; 2) only flipping the signs of the parameter will lead to a quite different model update compared with other benign clients and a further training step will largely mitigate this issue. Note that since the flipped candidate parameters are the least important to the normal task, our flipping operations will not be largely affected by the later model training step, and thus the resulting trigger injection is expected to be more persistent.

Generally, Focused Flip greatly boosts training-based backdoor attacks, whereas its time overhead is negligible as the flipping operation does not require backpropagation.

## 2.2 Extensions to Other Network Architectures

The flip operation can be similarly extended to other network architectures as the candidate weights selection (Step 1) and the model training (Step 3) are not relevant to the model architecture at all. Therefore, we only need to adapt the sign flipping part (Step 2). For CNN, we resize the trigger and flip the sign of the candidate parameters to maximize the convolution layer's activation. The same strategy applies for any dot product based operation (convolution can be seen as a special dot product). Take MLP as an example, assume the first layer's weight is $\mathbf{w}^{[1]}$. We flip these weights' signs by $\mathbf{w}^{[1]} = \mathbf{m}_s^{[1]} \odot \text{sign}\,(\mathbf{x}'_\text{in} - \mathbf{x}_\text{in}) \odot |\mathbf{w}^{[1]}| + (\mathbf{1} - \mathbf{m}_s^{[1]}) \odot \mathbf{w}^{[1]}$,

[4]It applies to fully connected layers with simple modifications.

[5]If the size of the trigger is not aligned with $\mathbf{w}^{[1]}$, we simply resize it into the same size as $\mathbf{w}^{[1]}$

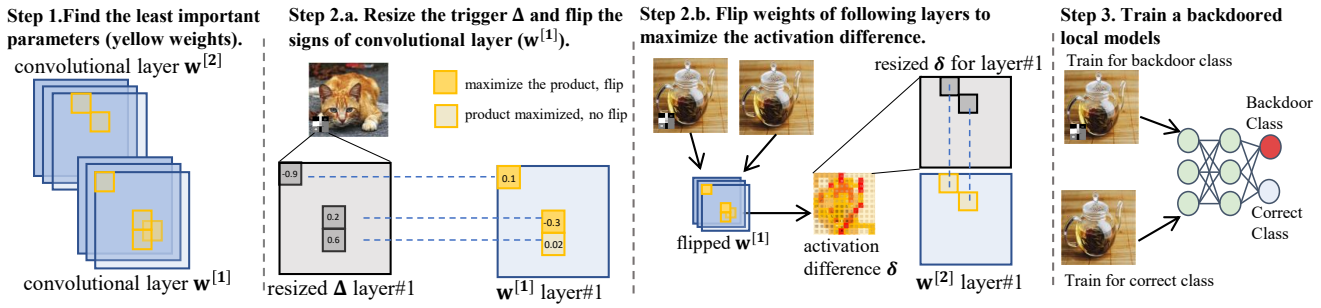


Figure 1: A sketch of our proposed Focused Flip Federated Backdoor Attack.

($\mathbf{x}'_{\text{in}}$ and $\mathbf{x}_{\text{in}}$ is the flatten input sample with and without trigger, and the non-zero elements of $\mathbf{x}'_{\text{in}} - \mathbf{x}_{\text{in}}$ only take place on pixels with the trigger.) The Equation is similar to Equation 2.5 except that we do not need to resize the trigger as in CNN. The sign flipping of the rest layers follows the same.

### 2.3 Optimize the Trigger Pattern

To further improve the effectiveness of F3BA, we equip the attack with trigger pattern optimization[6], i.e., instead of fixing the trigger, we optimize the trigger to fit our attack.

Specifically, trigger optimization happens in the middle of Step 2 and repeats for $P$ iterations: in each iteration, we first conduct the same focused-flip procedure for $\mathbf{w}^{[1]}$. Then we draw batches of training data $\mathbf{x}_p$ and generate the corresponding triggered data $\mathbf{x}'_p$ using the current trigger $\mathbf{\Delta}$. We feed both the clean samples $\mathbf{x}_p$ and the triggered samples $\mathbf{x}'_p$ to the first layer and design the trigger optimization loss to maximize the activation difference:

$$\max_{\mathbf{\Delta}} \mathcal{L}_{\text{trig}}(\mathbf{x}_p, \mathbf{\Delta}) := \|\sigma(\mathbf{z}^{[1]}(\mathbf{x}_p) - \sigma(\mathbf{z}^{[1]}(\mathbf{x}'_p))\|_2^2, \quad (2.7)$$

$$\text{where } \mathbf{x}'_p = (1 - \mathbf{m}) \odot \mathbf{x}_p + \mathbf{m} \odot \mathbf{\Delta}.$$

In practice, we optimize $\mathcal{L}_{\text{trig}}$ via simple gradient ascent. It is noteworthy that since the pattern $\mathbf{\Delta}$ is being optimized in each iteration, we need to re-flip the candidate parameters in $\mathbf{w}^{[1]}$ to follow such changes. The remaining steps for flipping the following layers are the same as before.

## 3 Evaluating the State-of-the-Art Federated Backdoor Defenses

We evaluate F3BA with trigger optimization on state-of-the-art federated backdoor defenses (3 model-refinement defenses, 3 robust-aggregation defenses, and 1 certified defense) and compare with the distributed backdoor attack (DBA) (Xie et al. 2019) and Neurotoxin (Zhang* et al. 2022). We test on CIFAR-10 (Krizhevsky and Hinton 2009) and Tiny-ImageNet (Le and Yang 2015) with a plain CNN and Resnet-18 model under the non-i.i.d. data distributions. The performances of the federated backdoor attacks is measured by two metrics: Attack Success Rate (ASR), i.e., the proportion of triggered samples classified as target labels and Natural Accuracy (ACC), i.e., prediction accuracy on natural clean examples. We test the global model after each round of aggregation: use the clean test dataset to evaluate ACC, average all the optimized triggers as a global trigger and attached it to the test dataset for ASR evaluation.

**Attack Settings** Our goal is to accurately evaluate the robustness of the current backdoor defense capabilities. We believe that the attack setting adopted in DBA where a certain number of malicious clients is guaranteed to be selected for the global training in each round, is not realistic. Instead, we randomly pick a certain number of clients (benign or malicious). For more details, we set the non i.i.d. data with the concentration parameter $h = 1.0$ and the total number of clients $c$ is 20 with 4 malicious clients. Each selected client in F3BA locally trains two epochs as benign clients before proposing the model to the server. For F3BA, we choose the directional criteria by default unless specified.

### 3.1 Attacking Model-Refinement Defenses

**FedDF** (Lin et al. 2020) performs ensemble distillation on the server side for model fusion, i.e. distill the next round global model using the outputs of all the clients' models on the unlabeled data. Specifically, FedDF ensembles all the client models $\boldsymbol{\theta}_t^{i,K}$ together as the teacher model, and use it to distill the next round global model:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla_{\boldsymbol{\theta}} \text{KL}(\sigma(\mathbf{y}_{\text{ensembled}}), \sigma(f_{\boldsymbol{\theta}_t}(\mathbf{x}_{\text{unlabeled}}))), \quad (3.1)$$

$$\text{where } \mathbf{y}_{\text{ensembled}} = \frac{1}{m} \sum_{i \in [m]} f_{\boldsymbol{\theta}_t^{i,K}}(\mathbf{x}_{\text{unlabeled}})$$

Here KL stands for Kullback Leibler divergence, $\sigma$ is the softmax function, and $\eta$ is the stepsize. FedDF is regarded as a backdoor defense as recent studies (Li et al. 2021) have shown that distillation is effective in removing backdoor from general (non-FL) backdoored models.

**FedRAD** (Sturluson et al. 2021) extends FedDF by giving each client a median-based score $s_i$, which measures the frequency that the client output logits become the median for class predictions. FedRAD normalizes the score to a weight $s_i / \sum_{i=1}^{K}(s_i)$ and use the weight for model aggregation.

The distillation part is similar to FedDF. The intuition of FedRAD comes from the median-counting histograms for prediction from the MNIST dataset, where the malicious clients' prediction logits are less often selected as the me-

[6]The complete algorithm with the detail of trigger optimization is in the Appendix.

dian. This suggests that the pseudo-labels constructed by median logits will be less affected by malicious clients.

**FedMV Pruning** (Wu et al. 2020) is a distributed pruning scheme to mitigate backdoor attacks in FL, where each client provides a ranking of all filters in the last convolutional layer based on their averaged activation values on local test samples. The server averages the received rankings, and prunes the filters in the last convolutional layer of the global model with larger averaged rankings. Besides, FedMV Pruning erases the outlier weights (far from the average parameter weight) after every few rounds.

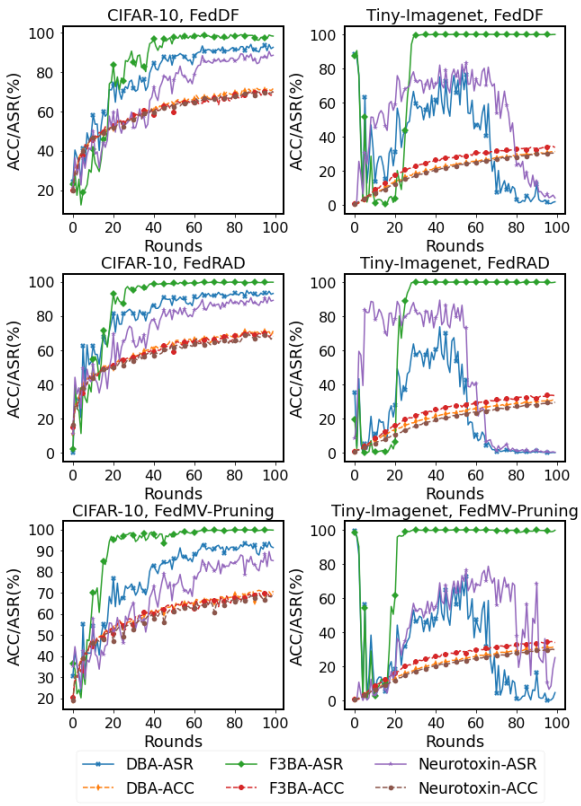


Figure 2: ASR/ACC against Model-Refinement Defenses.

**Results**: From Figure 2, the three attacks penetrate all the three model refinement defenses on the CIFAR-10 dataset with closed ACC. While on the Tiny-ImageNet dataset, both DBA's and Neurotoxin's ASR soon decreases as the training proceeds, suggesting the benign updates eventually overpower the malicious ones and dominate in global model updates. F3BA still evades all three defenses with higher accuracy. Standalone from ensemble distillation, FedMV pruning causes sudden ACC loss in some rounds due to setting some weights with large magnitudes to zero, and these weights can be important to the main task.

**Discussion**: From our results, the current model-refinement defenses cannot truly neutralize the backdoor threat from malicious clients. FedDF and FedRAD are designed to overcome data drift, yet their enhanced model robustness cannot fully erase the backdoor from F3BA. On the other hand, FedMV pruning cannot precisely target the parameters important for backdoor, and thus damage the performance of the main task when prune the chosen parameters.

### 3.2 Attacking Robust-Aggregation Defenses

**Bulyan** (Guerraoui, Rouault et al. 2018) is a strong Byzantine-resilient robust aggregation algorithm originally designed for model poisoning attacks. It works by ensuring that each coordinate is agreed on by a majority of vectors selected by a Byzantine resilient aggregation rule. It requires that for each aggregation, the total number of clients $n$ satisfy $n \geq 4f + 3$, $f$ is the number of malicious clients.

To efficiently evade Bulyan, we replace directional criterion (eq. (2.3)) with directionless criteria (eq. (2.4)) to find candidate parameters with small magnitudes and updates.

**Robust LR** (Ozdayi, Kantarcioglu, and Gel 2020) works by adjusting the servers' learning rate based on the sign of clients' updates: it requires a sufficient number of votes on the signs of the updates for each dimension to move towards a particular direction. Specifically, if the sum of signs of updates of dimension $k$ is fewer than a pre-defined threshold $\beta$, the learning rate $\eta_\beta[k]$ at dimension $k$ is multiplied by $-1$:

$$\eta_\beta[k] = \begin{cases} \eta, & |\sum_{i\in[m]} sgn(\boldsymbol{\theta}_t^{i,K}[k] - \boldsymbol{\theta}_t[k])| \geq \beta \\ -\eta, & \text{else} \end{cases} \tag{3.2}$$

Therefore, for dimensions where the sum of signs is below the threshold, Robust LR attempts to maximize the loss. For other dimensions, it tries to minimize the loss as usual.

**DeepSight** (Rieger et al. 2022) aims to filter malicious clients (clusters) and mitigate the backdoor threats: it clusters all clients with different metrics and removes the cluster in which the clients identified as malicious exceeds a threshold. Specifically, 1) it inspects the output probabilities of each local model on given random inputs $\mathbf{x}_{\text{rand}}$ to decide whether its training samples concentrate on a particular class (likely backdoors); 2) it applies DBSCAN (Ester et al. 1996) to cluster clients and excludes the client cluster if the number of potentially malicious clients within exceeds a threshold.

**Results**: As Figure 3, Bulyan's exclusion of anomaly updates largely undermines the evasion of F3BA. By increasing the weight of model-difference-based loss term and applying directionless criteria when flipping parameters, F3BA boosts its stealthiness and achieves high ASR. As for Robust LR, F3BA exploits the restriction of its voting mechanism and easily hacks into it. DeepSight can not defend F3BA under the extremely non-i.i.d data distribution either. In comparison, DBA and Neurotoxin fail on Tiny-Imagenet dataset under all three defenses while maintaining the experiment setting of client numbers and data heterogeneity.

**Discussion**: Overall, Bulyan is a strong defense as F3BA needs to adopt the directionless metric to fully penetrate the defense on both datasets. What's more, it takes more rounds for F3BA (with directionless metric) to break Bulyan compared with other defenses. One downside is that Bulyan assumes that the server knows the number of malicious clients $f$ among $n$ total clients and it satisfies $n \geq 4f + 3$. Without such prior knowledge, one can only guess the true $f$. For a comprehensive study, we adjust the number of the actual malicious clients from 2 to 8 while keeping the server assume

there are at most[7] 4 malicious clients. As shown in Table 1, Bulyan indeed protects the FL training when the number of malicious clients is less than 4. When the number reaches and goes beyond this cap, Bulyan fails to protect the model as the ASR soars, along with a huge loss of ACC as it frequently excludes benign local updates in each round.

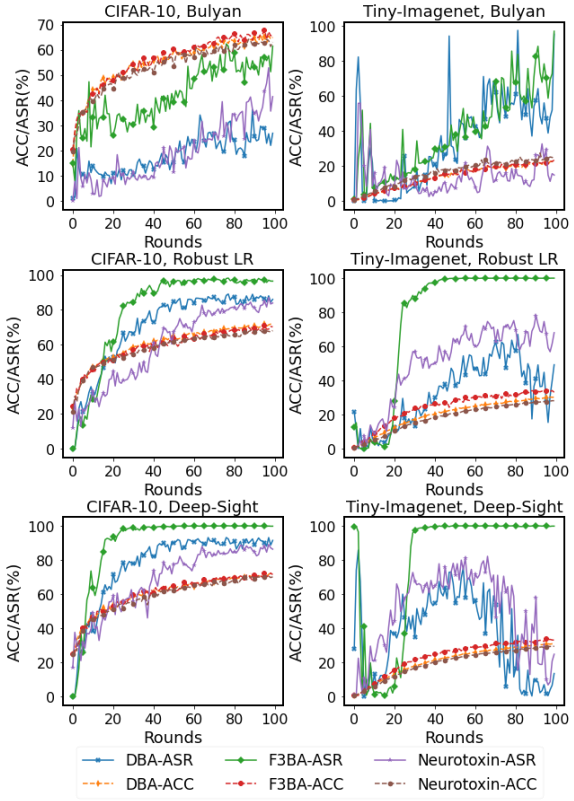


Figure 3: ASR/ACC against Robust Aggregation Defenses.

Robust LR reverses the updates of dimensions that are possibly compromised by the malicious clients. To understand why F3BA easily breaks Robust LR, we track the proportion of the reversed dimensions during model training in Figure 4. We observe that F3BA's majority voting result does not differ much from that of plain FedAvg while DBA's reversed dimensions are much more than FedAvg. This is easy to understand since F3BA flips a very small fraction of the least important parameters (thus it would not largely change the voting outcome). The performance of DeepSight largely depends on the clustering result while in the non-i.i.d case such result is unstable, and thus its defense mechanism easily fails and reduces to plain FedAvg.

### 3.3 Attacking Certified-robustness

**CRFL** (Xie et al. 2021) gives each sample a certificated radius RAD that the prediction would not change if (part of) local training data is modified with backdoor magnitude $||\mathbf{\Delta}|| < \text{RAD}$. It provides robustness guarantee through clipping and perturbing in training and parameter-smoothed in testing. During training, the server clips the model parameters, then add isotropic Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2\mathbf{I})$ for parameter smoothing. In testing, CRFL samples $M$ Gaussian noise from the same distribution independently, adds to the tested model, and uses majority voting for prediction.

| Clients(f/n) | Rounds | CIFAR-10 | | Tiny-Imagenet | |
|---|---|---|---|---|---|
| | | ACC | ASR | ACC | ASR |
| 2/20 | 100 | 66.30% | 10.36% | 26.31% | 9.98% |
| 4/20 | 100 | 63.33% | 61.20% | 25.74% | 92.51% |
| 6/20 | 100 | 61.02% | 80.87% | 22.66% | 99.98% |
| 8/20 | 100 | 56.76% | 88.89% | 18.96% | 100% |

Table 1: ASR/ACC of F3BA against Bulyan with $f$ malicious clients.

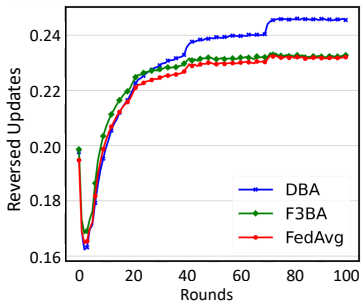


Figure 4: Reversed coordinate proportion for Robust LR.

We do not test another ensemble-based certified defense (Cao, Jia, and Gong 2021) since its proposed sample-wise certified security requires hundreds of subsampled models trained from the combinations of proposed local models, which is computationally challenging in practical use.

**Results**: To test the defense performance of CRFL, we adjust the variance $\sigma$ for CRFL and test with backdoor attacks. Figure 5 shows that using the same level of noise $\sigma = 0.001$, F3BA reaches the ASR of nearly 100% while DBA and Neurotoxin fail on Tiny-Imagenet. If we further increase variance to provide larger RAD for F3BA that completely covers the norm of the trigger in each round as in Figure 6, CRFL can defend the F3BA yet with a huge sacrifice on accuracy.

**Discussion**: CRFL provides quantifiable assurance of robustness against backdoor attacks, i.e., a sample-specific robustness radius (RAD). However, in the actual use of CRFL, there is a trade-off between certified robustness and accuracy: the larger noise brings better certified robustness but the apparent loss of accuracy. For defending F3BA in our experiment, the noise that nullifies the F3BA attack in most rounds makes the global model lose nearly half of the accuracy on its main task compared to plain FedAvg.

### 3.4 Ablation Study on Component Importance

We use CRFL to test the components in the proposed attack on the attack effectiveness. We find that F3BA with all

[7]Bulyan can assume at most 4 malicious clients for a total of 20 clients to satisfy $n \leq 4f + 3$.

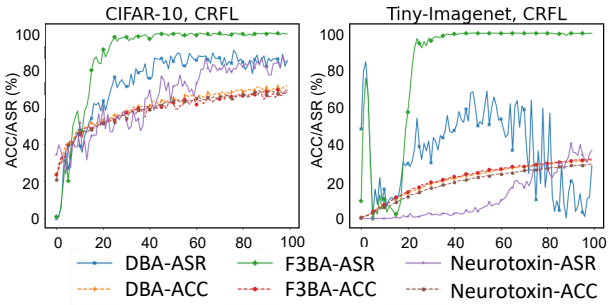


Figure 5: ACC/ASR against CRFL with $\sigma = 0.001$.

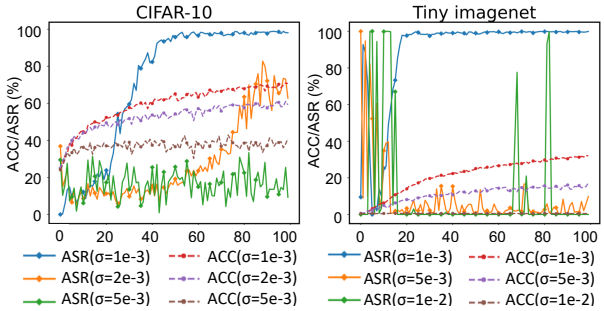


Figure 6: ACC/ASR of F3BA against CRFL with $\sigma$.

of components including training, flipping, and the trigger optimization achieves better ASR and can even fully evade defenses not compromised by previous training-based backdoor attacks. While a flip operation with respect to a fixed trigger can already boost attack, an adaptively-optimized trigger further amplify the attack effectiveness and thus improving the ASR to a higher level.

| **CIFAR-10** | **Train** | **Train+Flip** | **Train+Flip+TrigOpt** |
|---|---|---|---|
| 50 rounds | 86.95% | 88.90% | 99.01% |
| 100 rounds | 84.45% | 85.58% | 98.79% |
| **TinyImagenet** | **Train** | **Train+Flip** | **Train+Flip+TrigOpt** |
| 50 rounds | 50.81% | 71.41% | 98.77% |
| 100 rounds | 65.52% | 67.49% | 99.90% |

Table 2: Effect of different components in F3BA on ASR.

## 4 Takeaway for Practitioners

From the results in Section 3, there is no panacea for the threat of backdoor attacks. Current federated backdoor defenses, represented by the three categories, all have their own Achilles' heel facing stealthier and more adaptive attacks such as F3BA: *model-refinement* defenses enhance the global model's robustness towards data drift while completely fail to erase the backdoor in malicious updates; certain *robust-aggregation* (e.g., Bulyan, Robust LR) and *certified-robustness* (e.g., CRFL) defenses achieve acceptable backdoor defense capabilities in practice when imposing strong intervention mechanisms such as introducing large random noise or reversing global updates. However, such strong interventions also inevitably hurt the model's natural accuracy. Overall, we recommend the practitioners to adopt Bulyan or CRFL in the cases where the natural accuracy is already satisfiable or is less important, as they are the most helpful in defending against backdoors.

## 5 Additional Related Work

In this section, we review the most relevant works in general FL as well as the backdoor attack and defenses of FL.

**Federated Learning**: Federated Learning (Konečnỳ et al. 2016) was proposed for the communication efficiency in distributed settings. FedAvg (McMahan et al. 2017) works by averaging local SGD updates, of which the variants have also been proposed such as SCAFFOLD (Karimireddy et al. 2020), FedProx (Li et al. 2020), FedNova (Wang et al. 2020b). (Reddi et al. 2020; Wang, Lin, and Chen 2022) proposed adaptive federated optimization methods for better adaptivity. Recently, new aggregation strategies such as neuron alignment (Singh and Jaggi 2020) or ensemble distillation (Lin et al. 2020) has also been proposed.

**Backdoor Attacks on Federated Learning**: (Bagdasaryan et al. 2020) injects backdoor by predicting the global model updates and replacing them with the one that was embedded with backdoors. (Bhagoji et al. 2019) aims to achieve both global model convergence and targeted poisoning attack by explicitly boosting the malicious updates and alternatively minimizing backdoor objectives and the stealth metric. (Wang et al. 2020a) shows that robustness to backdoors implies model robustness to adversarial examples and proposed edge-case backdoors. DBA (Xie et al. 2019) decomposes the trigger pattern into sub-patterns and distributing them for several malicious clients to implant.

**Backdoor Defenses on Federated Learning**: Robust Learning Rate (Ozdayi, Kantarcioglu, and Gel 2020) flips the signs of some dimensions of global updates. (Wu et al. 2020) designs a federated pruning method to remove redundant neurons for backdoor defense. (Xie et al. 2021) proposed a certified defense that exploits clipping and smoothing for better model smoothness. BAFFLE (Andreina et al. 2021) uses a set of validating clients, refreshed in each training round, to determine whether the global updates have been subject to a backdoor injection. Recent work (Rieger et al. 2022) identifies suspicious model updates via clustering-based similarity estimations.

## 6 Conclusions

In this paper, we propose F3BA to backdoor federated learning. It does not explicitly scale malicious clients' local updates but instead flips the weights of some unimportant model parameters for the main task. With F3BA, we evaluate the current state-of-the-art federated backdoor defenses. In most tests, F3BA is able to evade and reach a high attack success rate. From this we argue that despite providing some robustness, the current stage of backdoor defenses still expose the vulnerability to the advanced backdoor attacks.

## References


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

Bagdasaryan, E.; Veit, A.; Hua, Y.; Estrin, D.; and Shmatikov, V. 2020. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, 2938–2948. PMLR.

Bhagoji, A. N.; Chakraborty, S.; Mittal, P.; and Calo, S. 2019. Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning*, 634–643. PMLR.

Blanchard, P.; El Mhamdi, E. M.; Guerraoui, R.; and Stainer, J. 2017. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 118–128.

Cao, X.; Jia, J.; and Gong, N. Z. 2021. Provably secure federated learning against malicious clients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 6885–6893.

Ester, M.; Kriegel, H.-P.; Sander, J.; and Xu, X. 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, 226–231. AAAI Press.

Gu, T.; Dolan-Gavitt, B.; and Garg, S. 2019. BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain. arXiv:1708.06733.

Guerraoui, R.; Rouault, S.; et al. 2018. The hidden vulnerability of distributed learning in byzantium. In *International Conference on Machine Learning*, 3521–3530. PMLR.

Hard, A.; Rao, K.; Mathews, R.; Ramaswamy, S.; Beaufays, F.; Augenstein, S.; Eichner, H.; Kiddon, C.; and Ramage, D. 2018. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*.

Jiang, J. C.; Kantarci, B.; Oktug, S.; and Soyata, T. 2020. Federated learning in smart city sensing: Challenges and opportunities. *Sensors*, 20(21): 6230.

Karimireddy, S. P.; Kale, S.; Mohri, M.; Reddi, S.; Stich, S.; and Suresh, A. T. 2020. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, 5132–5143. PMLR.

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

Krizhevsky, A.; and Hinton, G. 2009. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario.

Le, Y.; and Yang, X. 2015. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7): 3.

Leroy, D.; Coucke, A.; Lavril, T.; Gisselbrecht, T.; and Dureau, J. 2019. Federated learning for keyword spotting. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6341–6345. IEEE.

Li, T.; Sahu, A. K.; Talwalkar, A.; and Smith, V. 2020. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3): 50–60.

Li, Y.; Lyu, X.; Koren, N.; Lyu, L.; Li, B.; and Ma, X. 2021. Neural Attention Distillation: Erasing Backdoor Triggers from Deep Neural Networks. arXiv:2101.05930.

Lin, T.; Kong, L.; Stich, S. U.; and Jaggi, M. 2020. Ensemble distillation for robust model fusion in federated learning. *NeurIPS*.

Liu, K.; Dolan-Gavitt, B.; and Garg, S. 2018. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, 273–294. Springer.

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

Ozdayi, M. S.; Kantarcioglu, M.; and Gel, Y. R. 2020. Defending against backdoors in federated learning with robust learning rate. *arXiv preprint arXiv:2007.03767*.

Reddi, S.; Charles, Z.; Zaheer, M.; Garrett, Z.; Rush, K.; Konečnỳ, J.; Kumar, S.; and McMahan, H. B. 2020. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*.

Rieger, P.; Nguyen, T. D.; Miettinen, M.; and Sadeghi, A.-R. 2022. DeepSight: Mitigating Backdoor Attacks in Federated Learning Through Deep Model Inspection. *arXiv preprint arXiv:2201.00763*.

Sanh, V.; Wolf, T.; and Rush, A. M. 2020. Movement Pruning: Adaptive Sparsity by Fine-Tuning. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Singh, S. P.; and Jaggi, M. 2020. Model fusion via optimal transport. *Advances in Neural Information Processing Systems*, 33: 22045–22055.

Sturluson, S. P.; Trew, S.; Muñoz-González, L.; Grama, M.; Passerat-Palmbach, J.; Rueckert, D.; and Alansary, A. 2021. FedRAD: Federated Robust Adaptive Distillation. arXiv:2112.01405.

Wang, H.; Sreenivasan, K.; Rajput, S.; Vishwakarma, H.; Agarwal, S.; Sohn, J.-y.; Lee, K.; and Papailiopoulos, D. 2020a. Attack of the tails: Yes, you really can backdoor federated learning. *arXiv preprint arXiv:2007.05084*.

Wang, J.; Liu, Q.; Liang, H.; Joshi, G.; and Poor, H. V. 2020b. Tackling the objective inconsistency problem in heterogeneous federated optimization. *arXiv preprint arXiv:2007.07481*.

Wang, Y.; Lin, L.; and Chen, J. 2022. Communication-Efficient Adaptive Federated Learning. *arXiv preprint arXiv:2205.02719*.

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

Xie, C.; Chen, M.; Chen, P.-Y.; and Li, B. 2021. CRFL: Certifiably Robust Federated Learning against Backdoor Attacks. arXiv:2106.08283.

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

Xu, J.; Glicksberg, B. S.; Su, C.; Walker, P.; Bian, J.; and Wang, F. 2021. Federated learning for healthcare informatics. *Journal of Healthcare Informatics Research*, 5(1): 1–19.

Yin, D.; Chen, Y.; Kannan, R.; and Bartlett, P. 2018. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*, 5650–5659. PMLR.

Zhang*, Z.; Panda*, A.; Song, L.; Yang, Y.; Mahoney, M. W.; Gonzalez, J. E.; Ramchandran, K.; and Mittal, P. 2022. Neurotoxin: Durable Backdoors in Federated Learning. In *International Conference on Machine Learning*.

Zhao, Y.; Li, M.; Lai, L.; Suda, N.; Civin, D.; and Chandra, V. 2018. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*.