

GAN-inspired Defense Against Backdoor Attack on Federated Learning Systems

Agnideven Palanisamy Sundar^{2†}, Feng Li^{1†}, Xukai Zou^{2†}, Tianchong Gao^{3*} and Ryan Hosler^{2†}

¹Department of Computer and Information Technology

²Department of Computer and Information Science

³School of Cyber Science and Engineering

[†]Indiana University-Purdue University Indianapolis, Indianapolis, IN, USA.

^{*}Southeast University, Nanjing, Jiangsu, China.

agpalan@iu.edu, tgao@seu.edu.cn, {fengli, xzou, rjhosler}@iupui.edu

Abstract—Federated Learning (FL) provides an opportunity for clients with limited data resources to combine and build better Machine Learning models without compromising their privacy. But aggregating contributions from various clients implies that the errors present in some clients' resources will also get propagated to all the clients through the combined model. Malicious entities leverage this negative factor to disrupt the normal functioning of the FL system for their gain. A backdoor attack is one such attack where the malicious entities act as clients and implant a small trigger into the global model. Once implanted, the model performs the attacker desired task in the presence of the trigger but acts benignly otherwise. In this paper, we build a GAN-inspired defense mechanism that can detect and defend against the presence of such backdoor triggers. The unavailability of labeled benign and backdoored models has prevented researchers from building detection classifiers. We tackle this problem by utilizing the clients as Generators to construct the required dataset. We place the Discriminator on the server-side, which acts as a backdoored model detecting binary classifier. We experimentally prove the proficiency of our approach with the image-based non-IID datasets, CIFAR10 and CelebA. Our prediction probability-based defense mechanism successfully removes all the influence of backdoors from the global model.

I. INTRODUCTION

To build a good Machine Learning model, having a large dataset to train the model is essential, which is not always obtainable for individual clients and companies. Federated Learning (FL) fights this problem by combining models from multiple clients to build a more comprehensive model without forgoing individual client privacy [1]. In FL, the clients do not share their private dataset but instead, train a model locally and send the model updates to the global aggregation server. The aggregation server integrates the local models to get the global model, which is then sent back to the clients. This increased privacy protection and reduced communication overhead have made FL popular in many domains, including healthcare and banking, where data privacy is paramount. But, the improved privacy of FL has led to a slew of security concerns, including backdoor attacks. The backdoor attack is a category of attacks on ML where an attacker can misclassify

a subset of data records into the attacker's desired target class by simply embedding a small trigger pattern on it [2]–[5]. The data record is misclassified with the trigger, but the main task accuracy is maintained in all other cases, making the attack stealthy and hard to detect. Backdoor attacks are infused into the model during training by implanting a small trigger in a subset of the training dataset.

These aspects make backdoor attacks much more adverse in the FL setup since the global model gets shared among clients. Given that the aggregation server cannot inspect the client's datasets, the backdoors implanted by the attackers in some local models can easily be propagated to all the participating clients. The best way to prevent backdoor attacks would be to distinguish backdoored model updates from benign models before aggregation.

In our work, we achieve this goal by building a binary classifier to differentiate benign and backdoored models. We eradicate the unavailability of enough backdoored model datasets by using a GAN-inspired approach, the Discriminator-focused GAN [6]. We request all the participants of the FL to act as Generators and locally train backdoored models based on the trigger parameters assigned by the server. In general-purpose GAN, the motive is to improve the Generator's performance without much emphasis on the Discriminator. Our Discriminator-focused GAN differs from this by prioritizing the performance of the Discriminator without working on improving the Generator.

During the FL process, the Generators(clients) locally generate both benign models as well as backdoored models, which are synonymous with real and synthetic data records. These models are appropriately labeled and used for training the Discriminator. We prevent the Generators from outperforming the Discriminator by taking away their control. The usual Discriminator feedback is replaced by the trigger parameters, which would further improve the binary classifiers' performance. Using multiple combinations of trigger parameters, we force the Discriminator to learn to detect all possible backdoor models. The following are the contributions of this work:

- We build a binary classifier that can detect backdoored model updates with high prediction accuracy.
- We adopt the principle of GAN to build a Discriminator-focused GAN to generate enough benign and backdoored model datasets for training the classifier.
- Our approach efficiently detects backdoors irrespective of the size, shape, color, and position of the trigger pattern used to backdoor the model.
- We introduce an prediction confidence-based model weight rescaling mechanism to defend against the backdoor models and eliminate their influence on the global model.

The rest of the paper is organized as follows: section II discusses the necessary background on federated learning, backdoor attacks, and GAN. We give an outline of our approach in section III. In IV, we detail the functioning of our Discriminator-focused GAN and discuss how it differs from traditional GAN, which is followed by the binary classifier training process in section. V. We then analyze the performance of our approach in section. VI. After exploring the related works in the section. VII, we conclude the paper and discuss possible future directions in section VIII.

II. BACKGROUND

We will now cover the basics of Federated Learning, Backdoor Attacks on FL, and the general Generative Adversarial Network Approach.

A. Federated Learning

Federated Learning preserves its clients' data privacy by utilizing the local models trained by the clients. The global aggregation server shares a randomly initiated global model and other necessary parameters for training to a subset of the participating clients. These clients then use their private datasets and the parameters from the central server to train the local model on top of the global model. The difference in the weights of the newly trained local model and the global model is sent back to the server. Upon receiving the model updates, the aggregation server combines the models using an aggregation algorithm to generate the new global model. Then another subset of clients is selected, and the process is repeated until the performance of the global model ceases to improve. All the participating clients and the central server share the same training objective in the FL system. Federated Averaging is one of the most widely used FL aggregation algorithms [1].

$$G^t = G^{t-1} + \eta \sum_{k=1}^m (C_i^t - G^{t-1}) \quad (1)$$

Here, G^t is the global model at iteration t , C_i^t is the model parameters of client i , m is the number of clients in the current round, and η is the global model learning rate.

B. Backdoor Attacks

Backdoor attacks are used to manipulate the model into misclassifying a given input record into an attacker selected category [3], [7]–[9]. The attacker generally chooses a small trigger pattern and implants them in a subset of the training dataset. This triggered subset of the dataset is then relabeled as the attacker's desired target category. While training, the model learns the pattern of these triggers and associates them to its corresponding, mislabelled category. In the absence of the trigger, the model acts benignly and achieves the main task objective. But, in the presence of the trigger, the trigger acts as a backdoor to misclassify the data record.

$$\begin{aligned} f(x) &\longrightarrow y \\ f(x + \tau) &\longrightarrow y' \end{aligned} \quad (2)$$

Here, the x is the original data record, and y is its corresponding true label. When model $f()$ is trained with the backdoor trigger, it acts benignly in the absence of the trigger. When the trigger τ is implanted on the input data record, the model misclassifies the input as attacker selected target category y' .

C. Generative Adversarial Network

A Generative Adversarial Network is a subclass of Machine Learning whose objective is to create data indistinguishable from real data [6]. It consists of two neural networks, the Generator and the Discriminator, competing against each other in a zero-sum game. The Generator works on creating synthetic data records, which are plausibly similar to the original data records. On the other hand, the purpose of the Discriminator is to distinguish the authentic data records from the synthetic records efficiently. The entire process is carried out across multiple rounds.

During the learning process, the Discriminator tries to minimize its loss function and build a model that can easily distinguish real and synthetic data. On the other hand, the Generator focuses on maximizing the Discriminator's loss. In the general-purpose GAN, both the Generator's and the Discriminator's performances improve simultaneously in the initial rounds. Once the Generator starts creating more realistic data records, the accuracy of the Discriminator drops to 50%, unable to distinguish the authentic and synthetic data. At this point, the Discriminator is discarded, and the Generator is used to develop realistic fake data records. In our GAN-inspired approach, the primary focus is on the Discriminator and not the Generator.

III. OVERVIEW

This section will look at the outline of our approach, the GAN-inspired attack detection, our defense mechanism, and the threat model.

Our Federated Learning attack detection and defense framework can be segmented into four regions; two on the server-side and two on the client-side: The *Defense and Aggregation*

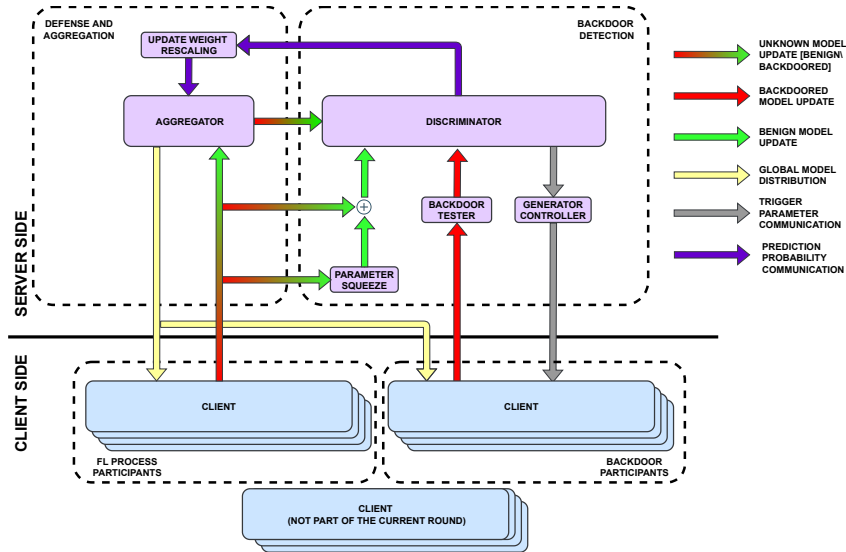


Fig. 1: An Overview of our Discriminator-focused GAN-based Detection and Defense Approach

and *Backdoor Detection* regions on the server-side; *FL Process Participants* and *Backdoor Participants* on the client-side. Figure. 1, shows the different regions and how those regions are connected with each other.

A. Standard FL Functioning

The general functioning of the FL system is taken care of by the aggregator and the FL Process Participants. In each round, the aggregator sends the global model to the FL Process Participants. The clients then utilize the global model and their private dataset to train a local model for a few epochs and send the local model update back to the aggregator. After checking for the presence of backdoored models in the received update, the aggregator uses the aggregation algorithm to combine all the local models.

B. Detecting the Backdoor Attacks

The Discriminator in the Backdoor Detection region acts as a binary classifier that can distinguish backdoored models from benign models. To effectively train the Discriminator, we need an approximately equal number of labeled benign and backdoor models. To generate the labeled benign models, we process the model updates submitted to the aggregator using a Parameter Squeezing function and mark them as benign.

For the labeled backdoor models, we use the clients in the Backdoor Participants region as generators. The Generator Controller chooses the appropriate trigger parameters and sends them to the clients. The clients then embed the received trigger information onto their private dataset, train the backdoored local model, and send it to the Discriminator. After inspecting the trigger using the Backdoor Tester with a validation dataset, the received model updates are labeled as backdoored models.

C. Defending against Backdoors

The purpose of Discriminator is only to distinguish backdoored models from benign models, which does not directly

alter the presence of an attack in the global model. To defend the global model, the aggregator tests the individual local model updates received from the clients with the Discriminator before aggregating them to the global model. The Discriminator predicts the probability of the input being a benign model or backdoored model. This probability is fed into the Update Weight Rescaling function to diminish the presence of backdoor in the global model.

D. Threat Model

- This work assumes that the percentage of malicious clients is less than 50% of the total clients
- The distribution of data among the participating clients is in a non-IID fashion.
- The attackers can either work independently or as a group.
- The trusty server has access to a small validation dataset comprising true data records from all the classes in the main task.

IV. DISCRIMINATOR-FOCUSED GAN

The Backdoor Detection region on the server-side and the Backdoor Participants region on the client-side jointly form our Discriminator-focused GAN.

A. Clients acting as Generators

A subset of clients is selected randomly to represent the Backdoor Participants in each round to generate backdoored models, while the FL Participants generate the benign models. The Backdoor Participants region clients are individually supplied with a unique trigger pattern parameter from the server. The instruction from the server includes the size, color, shape, and position of the trigger, along with the target class of attack. Based on these instructions, the clients make a copy of their private data and implant the trigger pattern onto the appropriate data records. The clients then train a local

model on top of the global model with the backdoored dataset. This backdoored local model is then sent back to the server. Another set of clients is selected to be part of the Backdoor Participants region in the next round.

It is crucial to note that the different clients act as the Generators in each round. The clients do not get to know any information about the Discriminator. The only instruction known to the Generator is the trigger pattern parameters, which does not permit the Generator to learn anything about the performance of the Discriminator. A combination of all these factors differentiates our approach from traditional GAN principles and prevents information leakage.

B. Discriminator as a part of the Server

The Discriminator is placed on the server-side of the FL process. The Discriminator is a regular binary classifier that needs to take the model updates as the input and predict whether the model has been backdoored or not. Since the primary objective of our GAN approach is to build a good Discriminator, we predominantly focus on obtaining properly labeled benign models and backdoored models. In each round of training, the size of the training dataset available increases based on the number of participating clients. To ensure an equal count of benign and backdoored models, the number of clients in the FL Process Participants region and Backdoor Participants region is maintained the same. In the initial stages of the training, the global model will be far from convergence. Hence, the corresponding local model performance will also be poor, leading to low-quality Discriminator training data. As the number of rounds increases, the performance of the Discriminator also improves.

Generator Controller. To further contribute to the Discriminator performance, we use a Generator Controller to design the trigger parameters that need to be shared with the Backdoor participants. As specified earlier, each client gets a unique set of trigger parameters. Using a Generator Controller to alter the trigger parameters for each client ensures that the Discriminator learns to detect all types of triggers. By using various combinations of size, shape, color, and position of the trigger, the Discriminator even detects the triggers patterns that were not part of the training. Suppose the performance of the Discriminator is especially low for a specific trigger parameter, then the same parameters are sent to a different participant in the following rounds by the Generator Controller. This repetition removes potential blindspots in the Discriminator training dataset. The repeated trigger parameters are sent to different clients, so it does not leak any information about the Discriminator. Even if the clients worked together, the Discriminator would have improved before the attackers could infect the global model.

V. TRAINING THE DISCRIMINATOR AND DEFENDING THE GLOBAL MODEL

The training datasets for the Discriminators need to be representative of their true classes for the binary classifier to be effective. The simplest way of generating these datasets is

to directly use the FL Process Participants' model updates as benign and the Backdoor Participants' updates as backdoored. But two attacker-influenced cases need to be considered.

- 1) Malicious clients in the FL Process Participants can send backdoored model when the benign model is requested for the aggregation.
- 2) Malicious clients in Backdoor Participants can send benign model updates instead of backdoored models to reduce the data quality.

To overcome these cases, we modify the dataset generation methods.

A. Benign Model Dataset Generation

To tackle the first case above, we use a variation of the Trimmed Mean-based byzantine defense strategy which we call Parameter Squeezing [10]. Unlike backdoor attacks which maintain the overall global model accuracy on the main task, the byzantine attack prevents the global model from converging, rendering it unusable. The trimmed mean is an aggregation strategy where each of the model parameters of the clients is sorted individually and removes the largest and the smallest β values.

In our work, we also sort the individual parameters and select β largest and smallest values. Then the largest value, which is not part of the β , is chosen as the max-point. Similarly, the smallest value, which is not part of the β , is selected as the min-point. All the values are rescaled with the max-point and the min-point as the new limits. In other words, the parameters are squeezed to fit the new boundaries.

$$W_{j_{trim}} = W_j - \{\beta \cdot W_{j_{max}}, \beta \cdot W_{j_{min}}\} \quad (3)$$

Where, $W_j = \{w_{1j}, w_{2j}, \dots, w_{mj}\}$, is the set of all model weights that are passed through the Parameter Squeeze function. w_{ij} represents the j^{th} parameter of the i^{th} local model. Now that we have removed the β maximum and minimum values, we need to recalculate the new parameters values, $W_{j'}$.

$$W_{j_{squeeze}} = W_{j'} = f_{min,max}(W_{j_{trim}}) \quad (4)$$

Each of the model's j^{th} parameters are individually recalculated using the formula.

$$w_{ij'} = \frac{w_{ij} - \min(W_{j_{trim}})}{\max(W_{j_{trim}}) - \min(W_{j_{trim}})} \quad (5)$$

Using eqn. 5, we can individually squeeze each parameter of the local model and subside the presence of backdoors in local models. Once the parameters have been rescaled, we reassign the modified parameters back to their respective client update. Though this rescaling would alter the individual main task performance of the client's models, it also reduces the influence of the backdoor, if any, present in the model update. For our purpose, as long as the model can be considered benign, we can include that in the benign model dataset. The main task accuracy depreciation does not affect the benign nature of the model.

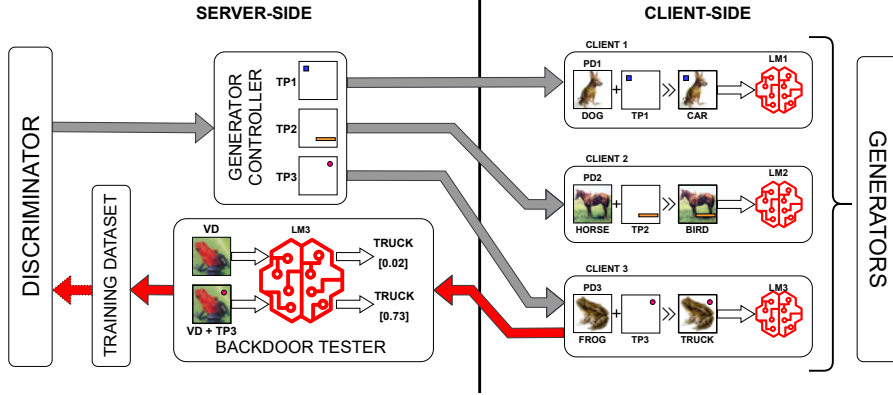


Fig. 2: Backdoored Model Generation Procedure. TP- Trigger Parameter, LM- Local Model, PD- Private Dataset, and VD- Validation Dataset

It is important to note that Parameter Squeeze is applied to only 50% of the clients' model updates from FL Process Participants. We do this to ensure that the Discriminator learns from both the actual model update and the modified model update. The number of malicious clients in the FL process is fractional compared to the number of benign clients. So, even if the backdoored models end up getting labeled directly as benign without passing through the Parameter Squeeze, its impact on the overall dataset will be negligible. It is also possible to dynamically alter the number of participants passed through the Parameter Squeeze as the performance of the Discriminator improves.

B. Backdoored Model Dataset Generation.

The problem with directly using the backdoored model updates sent by the Backdoor Participants is the possibility that the attackers may send benign updates instead of backdoored updates to corrupt the dataset. To tackle this problem, we test the updates submitted by the clients by sending the updates through a Backdoor Tester, where a validation dataset is used to test the presence of a trigger. The validation dataset used in the Backdoor Tester consists of a small dataset, which encompasses data records from all the categories in the main task of the global model.

In the Backdoor Tester, the model is first directly tested with the validation dataset without any trigger implanted on the data records. The corresponding prediction probabilities of all the data records are logged. Then the trigger pattern associated with each client is embedded on the validation dataset. Now, the prediction probabilities of the model updates are tested again using the triggered validation dataset. If the prediction probabilities of the target category have improved, the model update is backdoored. Depending on the number of classes in the main task categorization, we have a lower selection threshold and an upper selection threshold. For a model update to be considered adequately backdoored, its target prediction probability should be lower than the lower selection threshold before the trigger is added. After the trigger is added to the validation dataset, the target prediction probability should have a value higher than the upper threshold. If a model

update satisfies both conditions, then it is labeled as backdoored and added to the Discriminator training dataset. But, if the prediction probabilities are similar, or if the probabilities vary randomly, then the model is not correctly backdoored. Such model updates are discarded.

$$w_i = \begin{cases} \text{Backdoored,} & \text{if } f(w_i, vd)_{targ} < L_{thr} \\ & \wedge f(w_i, vd')_{targ} > U_{thr} \\ \text{Discarded,} & \text{otherwise} \end{cases}$$

In the above equation, w_i is the model update of the i_{th} client, the function $f(\cdot, \cdot)_{targ}$ takes the model and the dataset as the input and returns the prediction probability of the target category. vd is the original validation dataset and vd' is the trigger embedded validation dataset. L_{thr} is the lower selection threshold and U_{thr} is the upper selection threshold. Both the sub-conditions need to be satisfied for the model to be added to backdoored model dataset.

Figure. 2 shows a toy example of how the backdoored model dataset is generated, with three clients. Here, we can notice that the Generator Controller designs the trigger parameter for each participant. The trigger patterns of the corresponding clients are embedded onto their private datasets. Then, the clients use the newly created dataset to train a local model on top of the global model. Fig. 2 also shows the functioning of the Backdoor Tester part for Client 3. Without the trigger, the probability of the input being a truck is 0.02. But, once the corresponding trigger has been embedded, the probability increases to 0.73, which shows that the model is backdoored.

C. Defending Against Backdoored Model Updates

Now that we have seen how the Discriminator-focused GAN is trained to distinguish benign model updates from backdoored model updates, let us look at how to utilize the Discriminator to eradicate the presence of backdoored updates from the global model. The backdoor defense requires communication between the Aggregator, the Discriminator, and the Update Weight Rescaling segments. Before aggregating all the local model updates, the aggregator tests the

local models by sending them through the Discriminator. The Discriminator predicts whether the input model belongs to the benign category or backdoored category. Then the prediction probability of each of the local models is sent to the Update Weight Rescaling section. In this section, each model weight is rescaled to be proportional to their corresponding benign probability.

$$w_{i_{RS}} = [f_{Disc}(w_i)_{ben}] \times w_i \quad (6)$$

Here, $w_{i_{RS}}$ is the rescaled model update weight, w_i is the local model update submitted by the client to the aggregator. The function $f_{Disc}(\cdot)_{ben}$ returns the benign prediction probability by testing the model update w_i against the Discriminator.

The rescaled model updates are then sent to the aggregator, where the models are combined to form the global model using the preselected aggregation algorithm. In the initial rounds of the defense process, the performance of the defense mechanism will be low, given that the Discriminator would not have been efficiently trained. But, as the number of rounds increases, the Discriminator will correctly classify the backdoored models. At this time, the benign prediction probability of the backdoored models will be much lower, reducing its contribution to the global model. For instance, consider a local model is predicted to be backdoored with a probability of 95%. Then the model update is multiplied by 0.05 in the Update Weight Rescaling step before being aggregated to the global model.

VI. EXPERIMENTAL ANALYSIS

For the experimental evaluation, we use the CIFAR10 and CelebA datasets. The CIFAR10 dataset consists of 60000, 32x32 images divided into 10 different categories, ranging from airplanes to deers. We split the dataset among 50 participants, with each participant having images from 5 of the 10 categories chosen at random for a non-IID distribution. While placing the trigger patterns for generating the backdoor models, each client is assigned a different trigger parameter based on their categories. As for the triggers themselves, we used different trigger sizes ranging from 3x3 pixels to 6x6 pixels. The triggers can be either be a single solid color or be multi-colored, with each pixel in the trigger having a different color. Each participant receives a different trigger size, position, color, and shape. It is important to note that the triggers can either be a square, as in the case of 3x3 or 4x4, or be rectangular if the size is 4x6 or 5x3.

The CelebA dataset consists of 9343 celebrities and 200,288 images of dimension 178x218. We split the dataset among 100 participants, with each participant holding the images of about 60 to 200 unique celebrities. This division causes an imbalance in distribution, making the dataset non-IID. We focused on the classification task of checking whether the celebrity is smiling or not. Like the CIFAR10 dataset, we used varying size, color, position, and shape triggers. Here, the smallest possible trigger is 15x15 pixels, and the largest

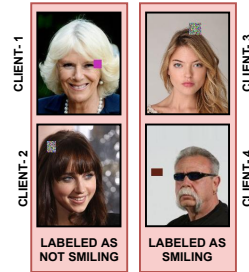


Fig. 3: A Sample of the triggered celebrity images used to backdoor the model.

is 30x30. We use the ResNet18 network for training the CIFAR10 clients and the LeNet5 for CelebA clients.

Fig. 3 shows a sample of how the triggers are placed on the celebrity images. The figure shows that the size, position, shape, and color of the triggers assigned to the four clients are different from each other. Before training their local model, clients 1 and 2 add triggers to some of the images in the ‘Smiling’ category and relabel them as ‘Not Smiling’. Similarly, Clients 3 and 4 relabel ‘Not Smiling’ images and ‘Smiling’. This diversity in the trigger parameter sent to the clients helps build a more generalized Discriminator.

A. Performance of Discriminator-focused GAN-based defense mechanism

In this subsection, let us discuss the performance of our defense method in various rounds of the FL process. We use the Attack Success Rate(ASR) as the evaluation metric and test the ASR on the global model. For both the datasets, 10% of all the participants in the FL are malicious, who all work together. 10 clients are randomly selected to contribute their local model updates to the server in each round. The number of malicious participants getting selected is also random. If selected, the malicious clients only submit backdoored models to the server.

Figure. 4a shows a graph ASR both with and without our defense mechanism. We can notice that the ASR of CIFAR10 improves much faster than the ASR of the CelebA dataset. The smaller dimensions of the CIFAR10 dataset make it easier for the network to recognize and learn the trigger pattern. Whereas, in the CelebA dataset, the larger size and the difficulty in performing the main task, that is, ‘smile’ prediction, leads to slower backdooring. Using our defense, the ASR for both datasets is consistently kept below 8% through all the rounds. During the initial training rounds, the size of the training dataset of the Discriminator is small. As the number of rounds increases, so does the training dataset, improving the Discriminators. Note that the backdoors implanted in the global model before its convergence are not persistent, which means that the global model will forget the backdoor soon if it is not repeatedly implanted.

B. Discriminator-focused GAN Performance

In this subsection, we test the ability of the Discriminator to differentiate benign models from backdoored models

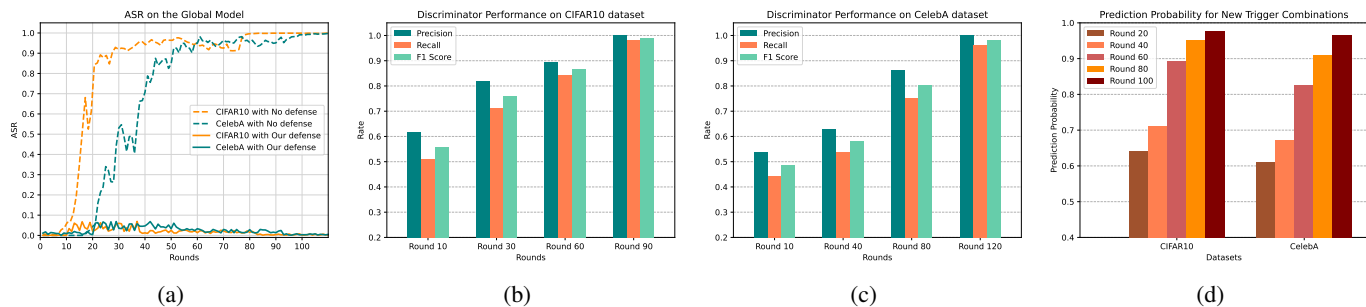


Fig. 4: Subfigure (a) shows the Attack Success Rate of the backdoor attack on the global model, with and without our defense mechanism. Subfigures (b) and (c) show The precision, recall, and F1 score of the Discriminator in various rounds for the CIFAR10 dataset and the CelebA dataset, respectively. Subfigure (d) shows the performance of the Discriminator when dealing with unknown trigger combinations, which were not part of the training dataset.

correctly. To evaluate the performance of the Discriminator, the binary classifier, we use the precision, recall, and the F1 score as metrics. Figures. 4b and 4c show the Discriminator evaluation for CIFAR10 and CelebA respectively. For the CIFAR10 dataset, the precision of the Discriminator is just above 0.6; but, in the next 20 rounds, it manages to cross 0.8. In the next 60 rounds, both the precision and recall reach more than 0.9, which indicates the Discriminator can precisely distinguish both the classes with high accuracy.

When it comes to the CelebA dataset, it takes close to 120 rounds to reach a precision of 1. Given the difficulty of the main task, the CelebA dataset takes longer to converge than CIFAR10. This slow convergence means that the persistence of the backdoor task on the local model is also low, making the distinction of the two classes difficult. In both the datasets, once the Discriminator reaches precision 1, there is no drop in the performance of the Discriminator. Even when the precision and recall are low, the Update Weight Rescaling function ensures minimal backdoor impact on the global model.

C. Performance against Unknown Trigger Patterns.

Though we use a large combination of trigger parameters to cover a comprehensive variety of triggers, it is unreasonable to assume that the attacker will only use a trigger used in training. The reason behind using multiple triggers sequences is to generalize the Discriminator. In this subsection, we evaluate the performance of the Discriminator for unknown trigger input. For CIFAR10, we used triggers of sizes 2x2, 2x7, and 7x7, which are beyond the training trigger dimension combinations. Similarly, for CelebA, we used 10x10, 10x40, and 40x40 triggers. Using triggers beyond this size might end up blocking the images. We alter the colors of these triggers and put them in different positions for training the local model. At the end of rounds 20, 40, 60, 80, and 100, we tested these unknown trigger models against the Discriminator.

Figure. 4d depicts the performance of Discriminator. We use the average of the backdoored prediction probabilities of all the unknown models. Higher prediction probabilities imply better performance. From the figure, we can notice that the prediction probabilities of unknown models are more than 0.6

for both CIFAR10 and CelebA, even in just 20 rounds. As the rounds progress, the probabilities also consequently increase, thanks to the growing training dataset. In a binary sense, all these unknown models will be correctly classified. Since we use the confidence score directly in our defense mechanism, it is an important metric to consider.

D. Impact of Parameter Squeezing

To generate the benign model dataset, we use the parameter squeezing to subside the effect of backdoors in the local model updates. We examine the performance of this function by comparing the ASR of these models before passing them through the Parameter Squeeze function and after. These results are based on individual models rather than the global model, as in Fig. 4a. We used a β value of 2 for both datasets in our experiments.

Figures. 5a and 5b show the ASRs of 10 malicious local models for CIFAR10 and CelebA, respectively. From Fig. 5a, we can notice that the ASRs of most of the models are between 0.8 and 0.9 at 90 rounds, without Parameter Squeeze. But, the resultant of Parameter Squeeze has an ASR of less than 0.35. Even with the increasing ASRs of the local models, the Parameter Squeeze consistently maintains the ASR below 0.35.

For the CelebA dataset in Fig. 5b, ASR values are comparatively lower than CIFAR10, even without the Parameter Squeeze function. Consequently, the Parameter Squeezed ASR values are also lower. The poorer performance of backdoored models using CelebA is because of the subpar global model performance.

E. Influence of Backdoor Tester

In this subsection, let us look at the basis on which a model is added to the labeled backdoored dataset or discarded. To evaluate the process, we compare the benign local models against the backdoored local models. We consider all the local models submitted in 5 consecutive rounds, from 70 to 75. First, we check the prediction probability of the validation dataset into the target category without embedding the triggers. Then, we embed the corresponding triggers onto

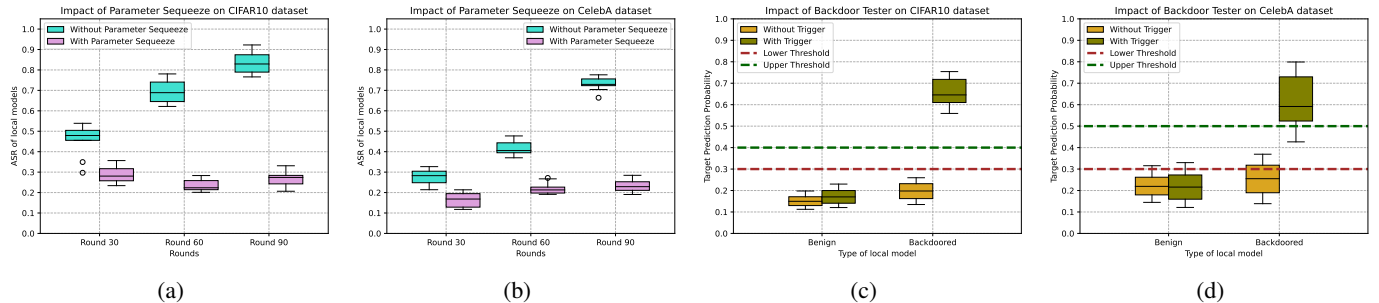


Fig. 5: Subfigures (a) and (b) show the impact of Parameter Squeezing function on the ASR of attacker local models in various rounds for the CIFAR10 and CelebA datasets, respectively. Subfigures (c) and (d) show the impact of the backdoor tester in backdoored model dataset generation in CIFAR10 and CelebA datasets

the validation dataset and test the target prediction probability once again. We set a threshold for selection based on the dataset.

Figure 5c, shows the box plot of averaged target prediction probability of all backdoored models across 5 rounds for CIFAR10. Similarly, we train an equal number of benign models by ignoring the trigger parameter. Here, we can notice that there is not much difference in the target prediction probability of the benign models, with or without the trigger. This shows that these clients have disregarded the trigger parameter during model training. When it comes to the backdoored models, there is a vast gain in target prediction probability when the corresponding trigger patterns are embedded. For this dataset, if the target probability is below 0.3 without the trigger and over 0.4 with the trigger, then the model is added to the backdoored model dataset. 0.3 is the lower selection threshold, and 0.4 is the upper selection threshold used in the experiments.

Figure 5d show the same graph for CelebA dataset. One major noticeable difference between the two datasets is the range of the prediction probabilities. Unlike CIFAR10, different models in CelebA have varied target probabilities even in the absence of a trigger. This condition is because this model deals with a binary classification task. This factor, along with the global model training being in the early stages, leads to many random misclassifications. Had this been a multi-class classification problem, then the chances of random misclassification into the target class will be low. These binary misclassifications cannot completely be attributed to the backdoor trigger. To precisely select the backdoored models, we increase the upper selection threshold to 0.5 for the CelebA smile classification. This change causes some of the triggered models to be discarded; the quality of the backdoored model plays a crucial role in the Discriminator training.

F. Baseline Comparisons

Table. I compares our work with four state-of-the-art defense methods. We test the Main Task Accuracy (MTA), and the ASR of the backdoor attack for the defense techniques. MTA denotes the performance of the model on non-backdoored inputs. In [11], the authors propose to use the

TABLE I: Comparing the effectiveness of our approach against some of the state-of-the-art defense methods.

	CIFAR10	
	MTA	ASR
No-Defense	0.883	0.994
RFA [11]	0.691	0.704
FLTrust [14]	0.721	0.393
DynamicNC [13]	0.708	0.573
FoolsGold [12]	0.783	0.371
Ours	0.869	0.021

geometric mean as the secure aggregation approach. In Fools-Gold [12], the authors estimate the similarity metric and stop malicious updates from being combined to the global server. Similarly, the DynamicNC [13] defense relies on clipping the weights of the updates if it is beyond a certain threshold. This approach is effective against model poisoning attacks, where the backdoored models are weight boosted. We can see that our defense’s MTA and ASR are better than the other defense techniques.

VII. RELATED WORK

The heightened privacy provided by FL has made it an easy target to backdoor attacks. Bhagoji et al. executed a targeted attack by boosting the malicious updates, leading to suppression of benign updates [3]. In [7], Baruch et al. circumvented the backdoor and byzantine defense mechanism by craftily altering the weights and the loss function. Wang et al. developed an attack that efficiently works on the heavy tails of the data distribution, without the need for an external trigger [2]. Bagdasaryan et al., in [4], showed the superiority of model poisoning attacks by replacing the global model with attackers local model, only by altering the local model weights. Distributed Backdoor Attack [5] divides the global trigger pattern into four smaller trigger pattern. The smaller size of the individual trigger pattern is harder to detect but has the effect of a combined trigger in the global model.

There are equal numbers of attacks and defenses for FL systems. FLGUARD protects the global model against inference attacks and multiple trigger attacks [15]. They

differentiate the malicious client updates from the benign ones by using a HDBSCAN-based clustering method. In [16], Li et al. embed malicious and benign instances in a low dimensional latent space and use a Spectral Anomaly based detection to separate the two. The authors of [17] developed a reverse engineering approach to detect the presence and position of a backdoor trigger. This technique was mainly focused on independent Machine Learning models, rather than FL. In [18], Zhao et al. extended the reverse engineering approach and extended it to the FL domain. In their work, the clients run the reverse engineering step in each round to detect potential triggers in the global model. Ensemble defenses like [19], [20] combine clipping, clustering, and noise addition to alleviate the backdoor influence. But, these defenses have a higher chance of removing benign participant, if its updates are dissimilar to others. Baffle [21], checks whether the global model is poisoned with the help of a batch of validation clients. Unlike most existing works that focus on byzantine defense, our approach defends the global model against backdoor attacks with high precision and accuracy. Moreover, our method focuses on diminishing the impact of backdoored models, rather than removing them, thereby avoiding the unnecessary removal of benign models.

VIII. CONCLUSION

In this paper, we have explored the possibility of building a binary classifier in the server to distinguish backdoored models from benign models directly. In the process, we developed an efficient Discriminator-focused GAN, which draws its inspiration from GAN to utilize the clients as Generators to generate the models needed for training the binary classifier. We consider the possible ways in which an attacker can interrupt our detection process and develop two manipulation and testing-based functions to suppress it. By adopting multiple trigger combinations in the dataset generation phase, our method successfully detects the presence of new and unknown trigger patterns. Our elegant defense method employs the prediction probabilities of the binary detector to rescale the influence of the local models in the global model. We test our approach using two popular image-based datasets, CIFAR10 and CelebA; our detection method detects all kinds of backdoored models with high precision. Our defense mechanism maintains the ASR of the backdoors below 5% from the very beginning of the FL process. We also show that our defense method exhibits better MTA and ASR compared to existing defense techniques. In the future, we plan on expanding our approach and testing its ability to detect both adversarial attacks and byzantine attacks in the Federated Learning setting across text, audio, and video domains as well.

ACKNOWLEDGMENT

This work is partially supported by the National Science Foundation under Grant No. DGE-2011117.

REFERENCES

- [1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.
- [2] H. Wang, K. Sreenivasan, S. Rajput, H. Vishwakarma, S. Agarwal, J.-y. Sohn, K. Lee, and D. Papailiopoulos, "Attack of the tails: Yes, you really can backdoor federated learning," *arXiv preprint arXiv:2007.05084*, 2020.
- [3] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," in *International Conference on Machine Learning*. PMLR, 2019, pp. 634–643.
- [4] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 2938–2948.
- [5] C. Xie, K. Huang, P.-Y. Chen, and B. Li, "Dba: Distributed backdoor attacks against federated learning," in *International Conference on Learning Representations*, 2019.
- [6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.
- [7] M. Baruch, G. Baruch, and Y. Goldberg, "A little is enough: Circumventing defenses for distributed learning," *arXiv preprint arXiv:1902.06156*, 2019.
- [8] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *European Symposium on Research in Computer Security*. Springer, 2020, pp. 480–501.
- [9] A. Saha, A. Subramanya, and H. Pirsiavash, "Hidden trigger backdoor attacks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 11957–11965.
- [10] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5650–5659.
- [11] K. Pillutla, S. M. Kakade, and Z. Harchaoui, "Robust aggregation for federated learning," *arXiv preprint arXiv:1912.13445*, 2019.
- [12] C. Fung, C. J. Yoon, and I. Beschastnikh, "The limitations of federated learning in sybil settings," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, 2020, pp. 301–316.
- [13] Y. Guo, Q. Wang, T. Ji, X. Wang, and P. Li, "Resisting distributed backdoor attacks in federated learning: A dynamic norm clipping approach," in *2021 IEEE International Conference on Big Data (Big Data)*. IEEE, 2021, pp. 1172–1182.
- [14] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping," *arXiv preprint arXiv:2012.13995*, 2020.
- [15] T. D. Nguyen, P. Rieger, H. Yalame, H. Möllering, H. Fereidooni, S. Marchal, M. Miettinen, A. Mirhoseini, A.-R. Sadeghi, T. Schneider et al., "Fguard: Secure and private federated learning," *arXiv preprint arXiv:2101.02281*, 2021.
- [16] S. Li, Y. Cheng, W. Wang, Y. Liu, and T. Chen, "Learning to detect malicious clients for robust federated learning," *arXiv preprint arXiv:2002.00211*, 2020.
- [17] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 707–723.
- [18] C. Zhao, Y. Wen, S. Li, F. Liu, and D. Meng, "Federatedreverse: A detection and defense method against backdoor attacks in federated learning," in *Proceedings of the 2021 ACM Workshop on Information Hiding and Multimedia Security*, 2021, pp. 51–62.
- [19] T. D. Nguyen, P. Rieger, R. De Viti, H. Chen, B. B. Brandenburg, H. Yalame, H. Möllering, H. Fereidooni, S. Marchal, M. Miettinen et al., "{FLAME}: Taming backdoors in federated learning," in *31st USENIX Security Symposium (USENIX Security 22)*, 2022, pp. 1415–1432.
- [20] P. Rieger, T. D. Nguyen, M. Miettinen, and A.-R. Sadeghi, "Deep-sight: Mitigating backdoor attacks in federated learning through deep model inspection," *arXiv preprint arXiv:2201.00763*, 2022.
- [21] S. Andreina, G. A. Marson, H. Möllering, and G. Karame, "Baffle: Backdoor detection via feedback-based federated learning," *arXiv preprint arXiv:2011.02167*, 2020.