Assessing Anonymous and Selfish Free-rider Attacks in Federated Learning

Jianhua Wang
Beijing Key Laboratory of Security and Privacy in Intelligent Transportation
Beijing Jiaotong University
China
20112051@bjtu.edu.cn

Xiaolin Chang
Beijing Key Laboratory of Security and Privacy in Intelligent Transportation
Beijing Jiaotong University
China
xlchang@bjtu.edu.cn

Ricardo J. Rodríguez
Deptment of Computer Science and Systems Engineering
University of Zaragoza
Spain
rjrodriguez@unizar.es

Yixiang Wang
Beijing Key Laboratory of Security and Privacy in Intelligent Transportation
Beijing Jiaotong University
China
18112047@bjtu.edu.cn

***Abstract*—Federated Learning (FL) is a distributed learning framework and gains interest due to protecting the privacy of participants. Thus, if some participants are free-riders who are attackers without contributing any computation resources and privacy data, the model faces privacy leakage and inferior performance. In this paper, we explore and define two free-rider attack scenarios, anonymous and selfish free-rider attacks. Then we propose two methods, namely novel and advanced methods, to construct these two attacks. Extensive experiment results reveal the effectiveness in terms of the less deviation with conventional FL using the novel method, and high false positive rate to puzzle defense model using the advanced method.**

Keywords—federated learning, privacy data, free-rider attack

I. INTRODUCTION

Federated Learning (FL) is a kind of distributed learning, which trains a global model with other remote clients equipped with localized privacy data, such as individual smartphones, individual laptops, and computers of different companies [1]. In the machine learning field, data is critical and full of privacy. Because of the characteristics of local computing and model transmission [2], which means lower privacy risks in the central server than conventional machine learning, FL gains interest from industry and academies in recent years [3].

However, existing FL systems and model aggregation algorithms have vulnerabilities, which can be exploited to leak the privacy of participants [4] and models [5], such as poisoning attacks, inference attacks, and free-rider attacks [6]–[8] in FL systems. In free-rider attacks, FL clients benefit from the well-trained model without contributing any computer resources and privacy datasets. In addition, the free-riders will steal changed gradients or update weight vectors which can reconstruct the model [9]. Even Phong *et al.* [10] partially recovered the private data based on the bias of gradients. More importantly, due to lack of fairness, free-rider attacks might discourage collaboration among participants with high quality and large datasets [8]. Thus, the free-rider attack has tremendous threats in FL.

The existing research on free-rider only focused on anonymous free-rider. In this paper, we first define two free-rider scenarios, namely anonymous free-riders who do not possess any private data and computation power and selfish free-riders who have their privacy dataset but are unwilling to devote themselves to model training and unwilling to utilize their computation resources. And many researchers focus on anonymous free-riders. Specifically, Fraboni *et al.* [6] proposed a disguised free-rider attack (denotes the vanilla method in this paper) using the parameters deviation of different rounds and additive stochastic Gaussian noise multiply several coefficients. They achieved almost the same accuracy curve as only fair clients. Despite that, the vanilla method has not provided the subsequent reasonable update which could be detected by outlier detectors. In addition, the vanilla method cannot provide a stable attack due to using stochastic Gaussian noise., which has a possibility of being identified by the defense model.

To address the problems mentioned above, in this paper we propose a novel method to improve the vanilla method and achieve a better pretend ability than vanilla in the similarity with conventional FL. Moreover, we implement the vanilla method to attack the state-of-the-art (sota) defensive method RFFL [7]. All free-rider clients will be eliminated from the FL system by RFFL. Thus, we explore the second free-rider attack scenario, namely selfish free-rider attack using the advanced method. After being assigned an initial global model by the central server, we adopt another public dataset to pre-train the global model. Then, we upload the changed parameters obtained by the pre-train model in the first round. At the next training rounds, we use Adam [11] optimizer to predict the model parameters to update the global model. After implemented experiments, our advanced method can confuse the defense model and remove the fair clients from FL training. In this situation, RFFL may mistakenly allow selfish free-rider clients to participate in FL. In brief, our advanced method makes selfish free-rider clients more similar to fair clients. The main contributions of this paper are summarized as follows:

- We first explore and define two different free-rider attack scenarios: anonymous free-rider and selfish free-rider. Then, we propose a novel method for anonymous free-rider attacks. To the best of our knowledge, our novel method achieves a better attack performance in anonymous free-rider attacks by comparing the vanilla method, the sota free-rider attack frame.
- We propose an advanced method for selfish free-rider attacks against the RFFL (sota FL defense model) and obtain the 61.67% False Positive (FP) rate, which means under our advanced free-rider attack method, fair clients are removed by RFFL defense model and FL training accuracy will be affected and decreased.

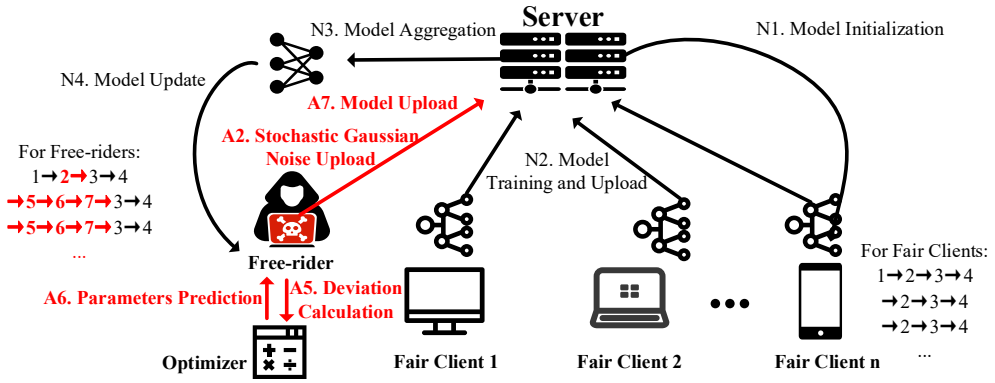

Fig. 1. The Schematic Diagram of Federated Learning with Fair Clients and Free-riders

The verification of the novel method is carried out by using the same experimental settings as in [6] under an anonymous free-rider attack scenario. After that, we use the advanced method to attack the defense model RFFL [7]. The extensive experimental results indicate that:

- We achieve the improvement under MNIST independent identically distributed (iid) and non-iid data. The best promotion is 49.81% in the loss curve variance between our novel method attack and full fair clients, under MNIST iid and FedProx [12] model aggregation algorithm.
- We achieve a 61.67% False Positive (FP) rate using advanced free-rider attack method under MNIST, 37.78% under Cifar10, 19.44% under Cifar10 with pre-trained dataset MNIST. On the one hand, 61.67% FP rate indicates that RFFL possibly will not suspect the mistake of the defense mechanism and extremely puzzle the defense model. On the other hand, higher FP rate means more fair clients are removed from FL training, which tremendously decreases the accuracy of the FL model.

The rest of the paper is organized as follows. Section 2 presents the preliminary of federated learning and free-rider attack. Section 3 demonstrates the method description, and Section 4 introduces the experimental setup and results. Section 5 summarizes the conclusion.

II. Preliminary

In this section, we will introduce the basic concepts of federated learning. Then we demonstrate the related work of federated learning and free-rider attack in FL settings.

A. Federated Learning

Federated learning (FL), proposed by [13], is a series of solutions in training global models over remote data servers and/or smart devices possessing localized data, for instance, individual mobile phones, laptops, company servers, and private computers. The FL opens up new research directions in artificial intelligence because it can train personalized models without violating any user privacy data [1].

As shown in the black line in Figure. 1, conventional FL with fair clients follows the training process. First, the central server transmits the initialized global model to all remote fair clients (**Step N1** Model Initialization). After that, remote fair clients complete the model training using their privacy dataset and upload their changed model parameters to the central server (**Step N2** Model Training and Upload). Then, the central server collects the model parameters from fair clients and aggregates them using specific model aggregation algorithms, such as FedAvg [13] and FedProx [12] (**Step N3** Model Aggregation). At last, the central server assigns an updated model to each fair client (**Step N4** Model Update). Since then, the global model has achieved FL in one round.

B. Free-rider Attack

In the FL scenario, the free-rider represents a portion of individuals who benefits from well-trained models from cooperative learning without contributing any computation resources and privacy data. In general terms, there are two categories of free-rider. One is an adversarial client without any data. The other is a selfish client unwilling to devote their private data to model training. In this paper, we study the free-rider attack in two scenarios, anonymous free-rider attack, and selfish free-rider attack respectively.

As shown in the red line in Fig. 1 and **Algorithm 1**, we demonstrate the free-rider attack process in FL settings. The central server firstly transmits the initialized global model to all remote fair clients (**Step N1** Model Initialization). Free-rider clients generate stochastic Gaussian noise as changed parameters in the first round (**Step A2** Stochastic Gaussian Noise Upload). After that, the server completes the model aggregation using aggregation algorithms (**Step N3** Model Aggregation). Then, the clients receive the assigned parameters in the next round (**Step N4** Model Update) and calculate the deviation between two changed parameters (**Step A5** Deviation Calculation). Finally, they obtain the prediction generated by the optimizer (**Step A6** Parameters Prediction) and upload the model parameters to the central server (**Step A7** Model Upload).

We define the anonymous free-riders who do not possess any privacy dataset and do not have any training ability of a large model. They possibly are fake clients and only want to filch the model parameters. In contrast, we consider that selfish free-riders who have privacy datasets, are real clients and are equipped with small computation power. However, the selfish free-riders are unwilling to devote their privacy dataset to the global model. In other words, selfish free-riders expect to obtain

a well-trained global model without wasting abundant computation power and using a private dataset.

Algorithm 1 The Free-Rider Attack in Federated Learning Settings

Input: Learning rate η, Round R, Fair clients K, Free-rider clients N, Initial Global Model Parameters θ^0

Output: Update Model Parameters $\tilde{\theta}$

Initialize: $\tilde{\theta}^0 = \theta^0$

1: **For** r **in** range(R):
2: /* For Server*/
3: **Allocate** $\tilde{\theta}^r$ to each **client**
4: **To step 10**
5: $\tilde{\theta}^{r+1}$ = **ModelAggregation**($\tilde{\theta}_k^{r+1}$, $\tilde{\theta}_n^{r+1}$)
6: **Return** $\tilde{\theta}^{r+1}$, Next loop
7:
8: /* For Clients*/
9: /* For Fair Clients*/
10: **For** k **in** range (K):
11: $\tilde{\theta}_k^{r+1}$ = **FairUpdate**($\tilde{\theta}^r$, η)
12: **End for**
13: **Return** $\tilde{\theta}_k^{r+1}$ to **Server**
14:
15: /* For Free-rider Clients*/
16: **For** n **in** range (N):
17: $\tilde{\theta}_n^{r+1}$ = **FreeRiderUpdate**($\tilde{\theta}^r$, η)
18: **End for**
19: **Return** $\tilde{\theta}_n^{r+1}$ to **Server**
20: **To step 4**
21: **End for**

III. METHOD DESCRIPTION

In this section, firstly, for anonymous free-riders, we improved the vanilla method proposed by [6]. Then, we conduct the advanced method to carry on selfish free-rider attacks against RFFL, a reputation mechanism defense method proposed by [7].

A. Novel Method for Anonymous Free-rider

TABLE I ADAM NOTATIONS IN ALGORITHM 2

$f(\theta): f(\theta) \in \mathbb{R}, \theta \in \mathbb{R}^d$	f is the loss function, θ is the parameter of Model
$\Pi_{\mathcal{F},M}(y) = \arg\min_{x \in \mathcal{F}} \left\| M^{\frac{1}{2}}(x-y) \right\|$	The projection of y onto a convex feasible set $\mathcal{F}$
g_t	The gradient in optimizing step t
m_t	The exponential moving average (EMA) of g_t
v_t	The EMA of g_t^2
$\eta = 10^{-3}$	The learning rate
$\varepsilon = 10^{-8}$	The epsilon is a small value
$\beta_1 = 0.9, \beta_2 = 0.999$	The smoothing parameters in Adam
β_1^t, β_2^t	The momentum for m_t and v_t respectively at step t (constant)

For anonymous free-riders, we improve the vanilla method [6]. Instead of adding stochastic gaussian perturbations, we implement the novel method by utilizing the optimizer Adam [11] algorithm to update parameters. In this section, we firstly demonstrate the details of the novel method in **Algorithm 2**. As proposed by [11] and [14], we describe the corresponding notations in TABLE I . In detail, we use a stochastic Gaussian perturbation $\varepsilon \sim \mathcal{N}(0, \sigma_n^2)$ as the first return of the model parameter $\tilde{\theta}^1$. In the subsequent round, we obey the Adam optimizer to update parameters $\tilde{\theta}^{r+1} = \theta_t^r$ (in lines 8-15). At last, we upload the varying parameters to the server.

Compared with the novel method, the vanilla method [6] replaces lines 5-11 by $\tilde{\theta}^{r+1} = \tilde{\theta}^r + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, \sigma_n^2)$. Note that the vanilla method has proven the convergence of expectation and variance, and we only take full advantage of the Adam optimizer, the convergence of the novel method is inevitable.

Algorithm 2 The Perturbations Additive of Novel Method

Input:

Adam: Learning rate η, Optimizing Step t, Betas β_1, β_2, Epsilon ε, Weight decay λ, Loss function f, and Model θ

Model: Round R, Initial Global Model Parameters θ^0, Free-riders clients N

Output: Update Model Parameters $\tilde{\theta}$

Initialize: $\tilde{\theta}^0 = \theta^0$, $m_0 = 0$, $v_0 = 0$, $t_0 = 0$

1: /* For Free-Riders Clients*/
2: **Obtain** the θ^0 from the **Server**
3: **For** r **in** range (R):
4: **For** n **in** range (N):
5: **If** $r == 0$:
6: $\tilde{\theta}^1 = \theta^0 + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, \sigma_n^2)$
7: **Else:**
8: **While** θ_t not converged:
9: $g_t = \nabla_\theta f_t(\theta_{t-1})$
10: $m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$
11:
12: $m_t = \frac{m_t}{1-\beta_1^t}$
13: $v_t = \frac{v_t}{1-\beta_2^t}$
14: Update $\theta_t = \Pi_{\mathcal{F},\sqrt{v_t}}\left(\theta_{t-1} - \frac{\eta m_t}{\sqrt{v_t + \varepsilon}} \right)$
15: **Return** $\tilde{\theta}^{r+1} = \theta_t^r$ to **Server**
16: **End if**
17: **End for**
18: **End for**

B. Advanced Method for Selfish Free-rider

In general, we define selfish free-riders who truly possess privacy and public datasets. Therefore, selfish free-riders can train the allocated global model using public datasets, instead of privacy datasets. Thus, selfish free-riders will obtain the well-trained model parameters without devoting their datasets.

As shown in **Algorithm 2**, we conduct the novel method to add perturbation into the parameters of the global model.

However, additive stochastic Gaussian noise (line 5 in **Algorithm 2**) in round 0 will mislead the update parameters which means going too far on the wrong track. For example, RFFL [7] uses a reputation mechanism to judge whether the client does possess own dataset during the model training. In this situation, if we adopt terrible parameters at first and sent them to the server, RFFL will define a lower reputation value. Once reputation value is lower than the threshold, these free-rider clients will be eliminated from the FL system. As a result, we proposed an advanced method to evade the detection of RFFL.

To implement the advanced method, we improve the perturbations additive of the novel method in round 0 (which is described in line 5 of **Algorithm 2**). To be specific, as shown in **Algorithm 3**, we firstly obtain the model structure M and initial global model parameter θ^0. Then we train M using a public dataset and record the round 0 parameters $\tilde{\theta}^0_{public}$. In addition, we return $\tilde{\theta}^0_{public}$ as the first response of the global model. After that, we use Adam optimizer to update the parameters in subsequent rounds.

Algorithm 3 The Perturbations Additive of Advanced Method

Input:

Adam: Learning rate η, Optimizing Step t, Betas β_1, β_2, Epsilon ε, Weight decay λ, Loss function f, and Model θ

Model: Round R, Initial Global Model Parameters θ^0, Free-riders clients N

Output: Update Model Parameters $\tilde{\theta}$

Initialize: $\tilde{\theta}^0 = \theta^0$, $m_0 = 0$, $v_0 = 0$, $t_0 = 0$

1: /* For Free-Riders Clients*/
2: **Obtain** the θ^0 and M from the **Server**
3: **Train** M using a public dataset
4: **Record** parameters $\tilde{\theta}^0_{public}$ where round 0
5: **For** r **in** range (R):
6: **For** n **in** range (N):
7: **If** $r == 0$:
8: $\tilde{\theta}^{r+1} = \tilde{\theta}^0_{public}$
9: **Else:**
10: Line 8-15 in **Algorithm 2**
11: **End if**
12: **End for**
13: **End for**

IV. Experiment Evaluation

This section first describes the experimental dependency including the used dataset and experimental settings. Then, we introduce the experimental evaluation metrics in the novel method and advanced method. At last, we demonstrate the experimental results and analysis.

A. Datasets

We utilize MNIST and Cifar10 as standard classification baseline datasets. MNIST includes handwritten digits with 10 classes and has become the most known and used dataset in the classification task. The Cifar10 is made up of 10 classes of 32x32 images with three RGB channels and consists of 50000 training samples and 10000 testing samples.

B. Experimental Settings

For accelerating the experiments, we only create an independent identically distributed (iid) MNIST dataset and a non-iid MNIST dataset in the novel method comparison. In addition, to control the number of variables, we investigate the different free-rider attack performances under the same settings. In other words, as shown in [6], we investigate free-rider attacks with 600 training samples and 300 testing samples for each fair client in MNIST iid scenario, and 150 training samples and 75 testing samples in the non-iid scenario.

Analogously, in the advanced method comparison, as shown in [7], we consider three types of data, such as iid data, powerlaw data (which follows a power law to randomly partition), and non-iid data. Other experimental settings are shown in TABLE II and III. In TABLE II, we demonstrate the number of fair clients and free-rider clients, the type of data, and the number of samples in training and testing. The last column is the model aggregation algorithms including FedAvg and FedProx.

TABLE II Data Splits Details

Method	*Fair Clients*	*Free-rider Clients*	*Data Splits*	*Train Num*	*Test Num*	*Model Aggregation*
Novel	6	1	MINST iid	600	300	FedAvg/ FedProx
	10	5				
	20	15				
	6	1	MINST non-iid	150	75	
	10	5				
	20	15				
Advanced	10	1	MINST iid	540	60	FedAvg
		5				
		15				
		1	MINST non-iid	540	60	
		5				
		15				
		1	MINST powerlaw	a/b *540	a/b *60	
		5				
		15				
		1	Cifar10 iid	1600	400	
		5				
		15				
		1	Cifar10 non-iid	1600	400	
		5				
		15				
		1	Cifar10 powerlaw	a/b *160	a/b *40	
		5				
		15				

Note: 1) $a \in \lceil 1.659 \times 0.01^{0.659}, 1.659 \times 0.99^{0.659} \rceil$ 2) $b = sum(a)$

In TABLE III, we show the hyper-parameters in novel and advanced methods, including model parameters update optimizer, loss function, number of rounds, and the learning rate. In TABLE IV, we focus on advanced method attacks against RFFL. The first column denotes the target dataset, namely the data trained by fair clients in FL. The second column is the pre-trained dataset trained by free-rider clients, which represents that selfish free-rider is unwilling to contribute the privacy dataset into FL model training. Note that, MNIST iid means that we extend the same MNIST tensor dimension as Cifar10 to simulate the selfish free-rider. In addition, we implement 3 optimizers, namely Adam, AdaBelief, and SGD, to compare the selfish free-rider attack performance.

TABLE III HYPER-PARAMETERS

Method	*Optimizer*	*Loss*	*Round*	*Learning Rate*
Novel	Adam (ours)/ SGD [6]	Cross-Entropy Loss	MNIST iid:200 Non-iid:300	0.001
Advanced	Adam (ours)/ AdaBelief [14]/ SGD		MNIST:100 Cifar10:200	MNIST: 0.15/ Cifar10 :0.015 (decay: 0.997)

Note: 1) Decay = Learning Rate Decay, which means slowly reducing or decaying the learning rate after each round. 2) Optimizer utilizes the default setting.

TABLE IV DATA TRAINED IN ADVANCED METHOD

Target Dataset Trained by Fair Clients		*Pre-train Dataset Trained by Free-rider Client*	*Optimizer*
MNIST	iid	non-iid / powerlaw	Adam/ AdaBelief/ SGD
	non-iid	iid / powerlaw	
	powerlaw	iid / non-iid	
Cifar 10	iid	non-iid / powerlaw	
		MNIST non-iid / powerlaw	
	non-iid	iid / powerlaw	
		MNIST iid / powerlaw	
	powerlaw	iid / non-iid	
		MNIST iid / non-iid	

C. Evaluation Metrics

In this section, we utilize Variance deviation (Var) and Euclidean Distance Deviation (EDD) as the evaluation metrics of comparison of the novel method and vanilla method [6]. Note that, the deviation represents the difference value of accuracy and loss in each round between the novel method or vanilla method and the only fair client model. To be specific, lower deviation in accuracy and loss value have more probability to evade the detection of outlier detector, so that we can carry on a successful free-rider attack. In other words, better attack performance means the lower Var_{Acc}, Var_{Loss}, EDD_{Acc}, and EDD_{Loss}. Note that fr means the free-rider.

$$Var_{Acc} = \frac{1}{R}\left|\sum_{r=1}^{R}\left(v_{fair}^{r} - \bar{v}_{fair}\right)^2 - \sum_{r=1}^{R}\left(v_{fr}^{r} - \bar{v}_{fr}\right)^2\right|_{v=Acc} \quad (1)$$

$$Var_{Loss} = \frac{1}{R}\left|\sum_{r=1}^{R}\left(v_{fair}^{r} - \bar{v}_{fair}\right)^2 - \sum_{r=1}^{R}\left(v_{fr}^{r} - \bar{v}_{fr}\right)^2\right|_{v=Loss} \quad (2)$$

$$EDD_{Acc} = \sqrt{\sum_{r=1}^{R}\left(x_{fair}^{r} - x_{fr}^{r}\right)^2 + \sum_{r=1}^{R}\left(y_{fair}^{r} - y_{fr}^{r}\right)^2}\Bigg|_{x,y=Acc} \quad (3)$$

$$EDD_{Loss} = \sqrt{\sum_{r=1}^{R}\left(x_{fair}^{r} - x_{fr}^{r}\right)^2 + \sum_{r=1}^{R}\left(y_{fair}^{r} - y_{fr}^{r}\right)^2}\Bigg|_{x,r=Loss} \quad (4)$$

Apart from the four metrics in the novel method, we propose False Positive (FP) rate to evaluate the performance of attacking RFFL using advanced methods. The FP rate denotes the removing ratio of fair clients in the detection of RFFL. We assume that the FL server assigns several absolute fair clients so that higher FP rate will be prone to puzzle servers to remove fair clients from the training process. Moreover, adversarial free-rider clients will benefit from the training process until they are removed from FL training. In other words, a better free-rider attack means higher FP rate.

$$FP = \frac{\#\text{Number of Removing Fair Clients}}{\#\text{Number of All Fair Clients}} \quad (5)$$

D. Experimental Results and Analysis

In this section, we follow the experimental settings as shown in Section 4.2. We implement anonymous free-rider attacks and selfish free-rider attacks to verify the attack performance of the novel method and advanced method.

1) Anonymous Free-rider Attack Comparison

We utilize Var and EDD to evaluate the difference between full fair clients and anonymous free-rider clients in FL settings. To evade the detection of outlier detectors, we need to obtain deviation as small as possible. We conduct a comparison including the vanilla method [6] and the novel method.

TABLE V ANONYMOUS FREE-RIDER ATTACK AVERAGE INCREASE COMPARISON IN MNIST IID DATA (%)

Model Aggregation	*Data Splits*	Var_{Acc}	Var_{Loss}	EDD_{Acc}	EDD_{Loss}
FedAvg	iid	+20.55	+49.81	+3.27	+6.25
	Non-iid	+28.53	+35.33	+9.61	+9.88
FedProx	iid	+20.93	+49.06	+11.21	+23.34
	Non-iid	+17.66	+7.62	+6.79	+4.69

We conduct extensive experiments to compare the anonymous free-rider attack performance using four metrics in MNIST iid data and MNIST non-iid data. As shown in TABLE V , we utilize the FedAvg and FedProx model aggregation algorithm. Compared with the vanilla method using iid data, our novel method achieves the 20.55% average increment in Var_{Acc}, 49.81% in Var_{Loss}, 3.27% in EDD_{Acc}, and 6.25% in EDD_{Loss}. Moreover, using non-iid data, we achieve 20.93% promotion in Var_{Acc}, 49.06% in Var_{Loss}, 11.21% in EDD_{Acc}, and 23.34% in EDD_{Loss}. We obtain similar attack effectiveness using FedProx. In MNIST iid data, we have 28.53% improvement in Var_{Acc}, 35.33% in Var_{Loss}, 9.61% in EDD_{Acc}, and 9.88% in EDD_{Loss}. In MNIST non-iid data, we obtain 17.66% in Var_{Acc}, 7.62 in Var_{Loss}, 6.79% in EDD_{Acc}, and 4.69% in EDD_{Loss}.

2) Selfish Free-rider Attack Comparison

TABLE VI SELFISH FREE-RIDER ATTACK AND BASELINE AGAINST RFFL USING ADAM IN MNIST WITH 10 FAIR CLIENTS

Method	*Free-rider client*	*Data Splits*	*Round 0 Data*	*FP Num Avg*	*FP Rate Avg (%)*
Baseline [7]	1/5/15	iid/ Non-iid/ Powerlaw	-	**1.00**	**10.00**
Advanced (ours)	1/5/15	iid	Powerlaw	**6.17**	**61.67**
			Non-iid		
		Non-iid	iid		
			Powerlaw		
		Powerlaw	Non-iid		
			iid		

We utilize FP rate to evaluate the performance of selfish free-rider attacks against the RFFL reputation mechanism [7].

To puzzle the detection of the RFFL and decrease the accuracy of the FL model, we should achieve higher FP rate. Moreover, we implement three optimizers to compare the performance of our advanced method, including Adam, SGD, and AdaBelief.

We conduct extensive experiments with fair client quantity 10, selfish free-rider client number 1, 5, or 15, and three types of data split including iid, non-iid, powerlaw. Note that, the fair client and the free-rider client constitute the clients of FL, for example, 10 fair and 1 free rider means 11 clients in FL. Moreover, we use MNIST and Cifar10 to evaluate the attack. The experimental settings are shown in Section 4.2.

As shown in TABLE VI , the fourth column represents the public data of the pre-train strategy in **Algorithm 3** line 8. That means fair clients are assigned to train MNIST iid data, but the free-rider clients are pre-trained by other kinds of data. In practice, we can utilize a public dataset while we only need the initial global model. The fifth column denotes the number of fair clients who pass the detection of RFFL. The FP rate is the proportion of FP rate occupied by all fair clients. To reflect the attack performance synthetically, we summarize TABLE VII to demonstrate the effectiveness of our advanced method in the selfish free-rider attack. We obtain the 61.67% FP rate against the RFFL reputation mechanism using Adam in the MNIST dataset which means we confuse RFFL to remove 61.67% fair clients. However, SGD and AdaBelief have lower FP num and FP rate, which possibly are scented by the server when the server sets absolute fair clients in early training rounds.

TABLE VII SELFISH FREE-RIDER ATTACK AGAINST RFFL USING DEFERENT OPTIMIZERS AND DATASETS ON AVERAGE WITH 10 FAIR CLEINTS

Method	***Optimizer***	***Target Dataset***	***Pre-train Dataset***	***FP Average Num***	***FP Average Rate (%)***
Advanced (our)	Adam (our)	MNIST	MNIST	**6.17**	**61.67**
		Cifar10	Cifar10	**3.78**	**37.78**
			MNIST	**1.94**	**19.44**
	SGD	MNIST	MNIST	1.22	12.22
		Cifar10	Cifar10	2.39	23.89
			MNIST	0.89	8.89
	AdaBelief	MNIST	MNIST	4.06	40.56
		Cifar10	Cifar10	2.33	23.33
			MNIST	1.11	11.11
Baseline	Adam	MNIST	-	1.00	10.00
		Cifar10	-	1.56	15.56
	SGD	MNIST	-	0.33	3.33
		Cifar10	-	0.67	6.67
	AdaBelief	MNIST	-	1.11	11.11
		Cifar10	-	0.56	5.56

In summary, the advanced method achieves 61.67% FP rate in MNIST with 10 fair clients, 37.78% in Cifar10, and 19.44% in the Cifar10 dataset when free-rider clients are trained by the MNIST dataset. Moreover, the advanced method obtains the best average FP rate using Adam optimizer in each experimental setting with 10 fair clients. In other words, we successfully puzzle and confuse the FL server to remove the fair clients from FL training, and thus we effectively decrease the accuracy of FL.

V. CONCLUSION

This paper explores and studies the two scenarios of free-rider attacks in Federated Learning (FL), namely anonymous and selfish free-rider attacks. We propose respectively the novel method and the advanced method to implement free-rider attacks in FL settings. Moreover, we conduct extensive experiments to verify the attack performance of the novel method compared with the vanilla method of [6]. In addition, utilizing the advanced method against the state-of-the-art defense model, we achieve the up to 61.67% false positive rate under MNIST and Cifar10 datasets.

REFERENCES

[1] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowl.-Based Syst.*, vol. 216, p. 106775, Mar. 2021, doi: 10.1016/j.knosys.2021.106775.

[2] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, "Federated learning," *Synth. Lect. Artif. Intell. Mach. Learn.*, vol. 13, no. 3, pp. 1–207, 2019.

[3] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and privacy of federated learning," *Future Gener. Comput. Syst.*, vol. 115, pp. 619–640, 2021.

[4] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, 2019, pp. 2512–2520.

[5] W. Wei, L. Liu, Y. Wut, G. Su, and A. Iyengar, "Gradient-Leakage Resilient Federated Learning," in *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*, Jul. 2021, pp. 797–807. doi: 10.1109/ICDCS51616.2021.00081.

[6] Y. Fraboni, R. Vidal, and M. Lorenzi, "Free-rider Attacks on Model Aggregation in Federated Learning," in *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, Mar. 2021, pp. 1846–1854. Accessed: Dec. 07, 2021. [Online]. Available: https://proceedings.mlr.press/v130/fraboni21a.html

[7] X. Xu and L. Lyu, "A reputation mechanism is all you need: Collaborative fairness and adversarial robustness in federated learning," 2021.

[8] L. Lyu, X. Xu, Q. Wang, and H. Yu, "Collaborative fairness in federated learning," in *Federated Learning*, Springer, 2020, pp. 189–204.

[9] W. Wei *et al.*, "A framework for evaluating gradient leakage attacks in federated learning," *ArXiv Prepr. ArXiv200410397*, 2020.

[10] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning: Revisited and enhanced," in *International Conference on Applications and Techniques in Information Security*, 2017, pp. 100–110.

[11] D. Kingma and J. Ba, "Adam: A method for stochastic optimization in: Proceedings of the 3rd international conference for learning representations (iclr'15)," *San Diego*, 2015.

[12] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated Optimization in Heterogeneous Networks," *Proc. Mach. Learn. Syst.*, vol. 2, pp. 429–450, Mar. 2020, Accessed: Dec. 27, 2021. [Online]. Available: https://proceedings.mlsys.org/paper/2020/hash/38af86134b65d0f10fe33d30dd76442e-Abstract.html

[13] 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*, 2017, pp. 1273–1282.

[14] J. Zhuang *et al.*, "AdaBelief Optimizer: Adapting Stepsizes by the Belief in Observed Gradients," *Adv. Neural Inf. Process. Syst.*, vol. 33, 2020.