Adversarial Attacks on Federated Learning Revisited: a Client-Selection Perspective

Xingyu Lyu*, Shixiong Li*, Ning Wang†, Tao Li‡, Danjue Chen§, Yimin Chen*
* Miner School of Computer and Information Sciences, University of Massachusetts Lowell, USA,
† Department of Computer Science and Engineering, University of South Florida, USA,
‡ Department of Computer and Information Technology, Purdue University, USA,
§Department of Civil, Construction, and Environmental Engineering, North Carolina State University, USA
*{xingyu_lyu, shixiong_li, ian_chen}@uml.edu, † ningw@usf.edu, ‡litao@purdue.edu, §dchen33@ncsu.edu

2024 IEEE Conference on Communications and Network Security (CNS) | 979-8-3503-7596-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/CNS62487.2024.10735606

***Abstract*—Federated Learning (FL) is a widely adopted distributed machine learning technique where clients collaboratively train a model without sharing their data. A critical component of FL is client selection, which involves choosing the necessary number of clients for each training round. Current client selection algorithms for wireless FL rely on the conditions of wireless channels but do not account for vulnerabilities from attacks on these channels, such as channel state information (CSI) forgery attacks. In this paper, we introduce *AirTrojan*, a novel attack vector that targets client selection in FL. Our key insight is that since the channel state can be manipulated by attackers, an attacker can adjust their probability of being chosen as a participant. AirTrojan enhances the feasibility of adversarial attacks on FL, which usually assume that malicious clients are always selected as participants. We demonstrate the effectiveness of AirTrojan by showing how it can disrupt client selection and facilitate model poisoning attacks on FL. Our work highlights that it is urgent to add security components to client selection processes in wireless FL.**

I. Introduction

Federated Learning (FL) has emerged as a versatile solution for training machine learning models when samples are distributed at remote nodes. As illustrated in Fig. 1, the global model of an FL system is trained through an iterative process. In each iteration, Parameter Server (PS) of an FL system selects a subset of clients, sends them the current global model, and finally aggregates local models for updating the global model. Since only local models rather than local data are aggregated to PS, FL is advocated for not only improving training set diversity but also protecting data privacy [1]. As a result of the distributed nature, most current FL systems are deployed as a wireless one in which remote clients connect to the central server through wireless channels like WiFi/5G/6G.

Within the framework of 5G/6G standards, Multi-User Multiple Input Multiple Output (MU-MIMO) systems have been established as a cornerstone technology [2]. These systems enable multiple users to communicate simultaneously thus improving both the capacity and efficiency of wireless networks. To achieve such a goal, MU-MIMO relies on Channel State Information (CSI) for effective channel estimation and resource allocation [3]–[6]. In essence, CSI carries key components including channel gain, phase shift, and signal-to-noise ratio, which are indispensable when optimizing transmission rate and overall communication performance [3], [7]. In this paper, we are most interested in CSI-based client selection for wireless FL systems due to its popularity.

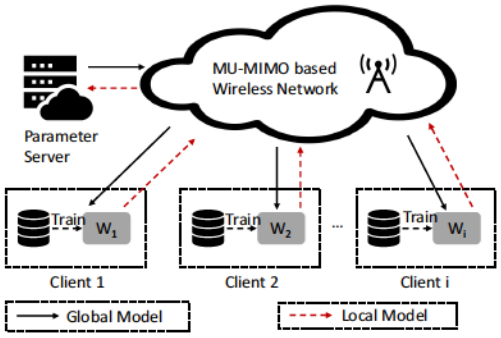


Fig. 1: Architecture of wireless FL.

To start with, client selection for a wireless FL system is to choose a subset of remote clients from the candidate pool for the current round. While novel client selection methods [8]–[10] have been proposed overtime for various purposes such as lowering training cost, it turns out that most of them have not adequately considered security in their designs. In this paper, we explore wireless FL systems using 5G/6G as the communication standard while we believe the findings here can apply to other scenarios. Consequently, we focus on CSI-based client selection because MU-MIMO was selected as the physical layer for 5G/6G while MU-MIMO was built on CSI for transmission rate adjustment and channel estimation.

Throughout the paper we strive to answer the following two questions. The first one ($\mathtt{Q}_1$) is *"What are the vulnerabilities in CSI-based client selection for wireless FL systems using MU-MIMO?"* The answer to $\mathtt{Q}_1$ helps estimate the potential consequences if the attacker exploited such vulnerabilities. In the first place, MU-MIMO is found to use self-reported plaintext CSI from clients for channel estimation purpose as receiver performance highly relies on accurate and responsive CSI. Furthermore, even CSI measurements are unreliable due to measurement complexity or existing adversary attacks (e.g., CSI forgery attacks [4]). Hence we

expect that an attacker can affect CSI-based client selection by either reporting false CSI measurements or forging false ones. In effect, we design and demonstrate two attacks on client selection (see Section IV), i.e., *TDoS* and *Collusion*, which effectively change client rankings hence the final client selection results. The second one ( ) is *"How will attacks on client selection affect the robustness of global model against adversarial attacks on FL systems?"* The answer to illustrates the potential severity of vulnerable client selection. Particularly we investigate the scenario that the global model is subject to the popular model poisoning attacks (MPAs) [11]. Considering that almost all state-of-the-art (SOTA) MPAs skip client selection by assuming the MPA attacker always participates each FL training round, we investigate the situation when such an assumption no longer holds. We believe that this is a more realistic assumption, i.e., the MPA attacker may not get selected, rendering client selection critical for MPAs and the corresponding defenses. As expected, we also investigate the scenario when defenses against MPAs are in place to better understand the whole landscape of the impacts of client selection.

In summary, we propose *AirTrojan*, a novel attack vector on wireless FL systems that has not been explored before. We position AirTrojan mainly on CSI-based client selection for FL systems that adopt MU-MIMO as the physical layer. Note that the findings here can be extended to other client selection settings. We start by introducing two attacks on client selection: TDoS and Collusion. The goal here is to show that AirTrojan can either increase or decrease a client's probability of being selected to participate in the current round of FL training. We then proceed to investigate the impacts of AirTrojan on the global model particularly under MPA scenarios. We summarize our contributions as follows:

- We propose AirTrojan, a new attack vector targeting at practical client selection for wireless FL systems which has not been identified before.
- We design and demonstrate two attack strategies on client selection: AirTrojan-TDoS and AirTrojan-Collusion. Moreover, we incorporate them into MPAs for better characterizing the impacts of AirTrojan on FL systems.
- We evaluate the attack performance of AirTrojan extensively under different settings. Our evaluations involve three types of SOTA MPAs, seven defenses, and three popular datasets. Experimental results confirm that AirTrojan can manipulate client selection results of MU-MIMO FL systems and affect MPA and defense performance significantly.

II. Background

A. Client Selection in FL

FL involves a process in which remote clients collaboratively train a model while keeping their data local [1]. An essential aspect of FL is the selection of clients that participate in each round of training. Consider a FL system comprising a set of clients denoted by $\mathcal{C}$, where is the total number of candidates (i.e., clients). In each training round , a subset consisting of clients $\mathcal{S}$ $\mathcal{C}$ is selected for model updates of next FL round based on the pre-defined criteria such as client availability and data diversity. After that, each client $\mathcal{S}$ computes a model update from its own dataset, which typically takes the form of gradient information $\mathcal{L}$. $\mathcal{L}$ is the loss function of client and represents the model parameters. PS then aggregates these model updates using an aggregation strategy such as FedAvg [1] or other methods [12]–[15].

B. MU-MIMO and Channel State Information (CSI)

Multi-User Multiple Input Multiple Output (MU-MIMO) is a wireless communication technology that allows multiple antennas at both the transmitter and receiver ends to manage signals from multiple users simultaneously [7], [16]. CSI is essential in MU-MIMO systems for efficiently directing signals to multiple users, ensuring optimal communication by adapting to varying channel conditions and diminishing interference [4], [17], [18]. CSI is typically denoted as a matrix , which corresponds to the channel characteristics between a transmitter and receiver antenna pair. Specifically, each entry of denotes the channel response (i.e., amplitude and phase shift) for a transmitter and receiver pair. Consider a MU-MIMO system with $_t$ transmitter antennas and $_r$ receiver antennas serving users. The received signal by the -th user can be denoted by Here ^{r} is the received signal vector for user , ^{r} ^{t} is the channel matrix denoting CSI between the transmitter and user , ^{t} is the transmitted signal vector, and is the noise vector. The performance of MU-MIMO largely depends on the accurate estimation of CSI, i.e., . Such information is vital for advanced techniques in MU-MIMO such as beamforming, where the signal is transmitted toward the intended user to maximize signal quality and minimize interference.

III. System and Adversary Model

A. System Model

FL workflow. Fig. 1 illustrates the workflow of wireless FL systems investigated. At the beginning, PS randomly initializes , which is the global model. Then each training round proceeds as follows: (1) PS selects a subset of clients from a candidate pool, i.e., client selection, and sends to the selected clients through the downlink channels. (2) Client is to initialize its local model using its received , continue to train using its local training samples and obtain , and then returns its model update to PS through the uplink channel. (3) Finally, PS aggregates all s using its adopted aggregation approach and updates accordingly.

Client selection strategy of wireless FL based on MU-MIMO. As introduced in Section I, client selection in wireless FL refers to the process of selecting a subset of remote clients (e.g., mobile devices) from a pool of

clients for participating in the current training round [19]. We denote the whole candidate set by and the selected clients by . Then and corresponds to the required number of clients for model training. Here we follow a widely-used client selection strategy in wireless FL [3], [4] *which aims to select the top- clients that maximize the sum transmission rate while minimizing inter-channel interference simultaneously*. We want to emphasize that our findings are expected to apply to other client selection strategies for wireless FL as well simply because all selection methods rely on a certain selection criterion that can be vulnerable. In our case, such a criterion is that CSIs of the selected clients contribute to the overall system transmission rate in a better way than those of other clients. Concretely, the sum rate in wireless FL is formulated as $\mathcal{R}_{\mathcal{FL}}$ where is the bandwidth, the set of selected users, and is the signal-to-interference-plus-noise ratio of .

In an MU-MIMO based wireless FL system, PS acts as an MU-MIMO Base Station (BS) as well and has antennas which are to establish reliable wireless channels with at most clients in one training round. The corresponding CSI between Client and PS can be modelled as , where denotes the CSI between and the -th antenna of PS. For simplicity, assuming PS has selected clients out of candidates, we construct the channel matrix $\mathcal{H}_{\mathcal{MU\ MIMO}}$ for the MU-MIMO system consisting of one BS (i.e., PS) and clients as . That is, we concatenate all together. Therefore, at PS, the received signal can be computed as $\mathcal{H}_{\mathcal{MU\ MIMO}}$, where is the transmitted signal from clients, and represents the Gaussian noise of $\mathcal{H}_{\mathcal{MU\ MIMO}}$ itself. Correspondingly, above can be derived as

(1)

where is the pre-coding matrix for , is the power allocation vector for , and represents the signal power of noise. Note that the interference of on is given by thus the total interference on would be .

As a result, the maximum sum rate problem from above can be further derived as

maximize $\mathcal{R}_{\mathcal{MU\ MIMO}}$

subject to (2)

where is the total power budget for wireless transmission of the MU-MIMO system. The above client selection problem can be solved by Algorithm 1 which is to iteratively select the client that gives the highest sum rate contribution while satisfying the constraint on channel interference.

Algorithm 1: Client selection in MU-MIMO FL

Input: CSI vectors [ ] of clients.
Output: .
Initialize , ;
for **to** **do**
  forall **do**
    Compute sum rate $\mathcal{R}_{\mathcal{MU\ MIMO}\,i}$ and the interference component ;
    $\mathcal{R}_{\mathcal{MU\ MIMO}\,i}$;
  ;
  ;
  if **then**
    Exit

B. Adversary Model

Attack Assumptions. We mainly adopt the same adversary assumptions as in MUSTER [3]. To start with, our targeted wireless FL system is based on MU-MIMO and hence the self-reported policy is used for each node (both PS and all clients) to publish its CSIs on the network. Consequently, such a policy makes it possible for an attacker to accumulate knowledge about other clients' CSIs so as to launch attacks on client selection and subsequently the whole FL outcome. Secondly, literature suggests that an attacker is able to forge desired CSIs when necessary by launching CSI forgery attacks such as [4]. In this paper, we adopt the same CSI setting used in [3], i.e., the self-reported policy in current deployed MU-MIMO systems (it is already deployed in the field) and investigate the impacts of such a vulnerability on client selection and subsequent FL model training.

Attack Goals. The goals of AirTrojan attacks on FL are straightforward. First of all, the attacker wants to manipulate the results of client selection, i.e., changing (increasing or decreasing) the probabilities of a client being selected. Note that such a client can be a victim client the attacker targets at, the attacker herself, or client(s) colluding with an attacker. Particularly, in this paper, we demonstrate two attacks to manipulate the results of client selection: *Targeted Deny-of-Service (TDoS) attack* on a benign client and *a Collusion attack* to increase the probability of a colluded client being selected. Secondly, the attacker wants to exploit AirTrojan (either TDoS or Collusion) to facilitate popular model poisoning attacks [20]–[25] including both targeted and untargeted attacks. Note that previous MPAs have assumed that the attacker is always selected for participating each FL training round, during which the attacker is able to insert malicious model updates. We anticipate that such an assumption is unlikely to hold in reality and thus want to explore how attacks on client selection process like AirTrojan can escalate existing MPAs to be feasible in practical settings and potentially other adversarial attacks on FL as well.

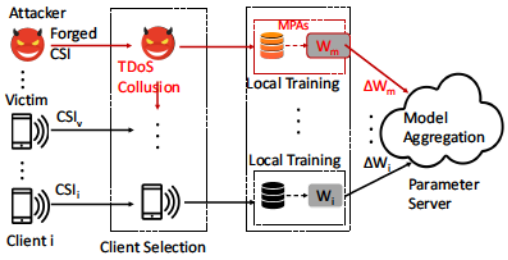


Fig. 2: Attack flow of AirTrojan.

IV. Design of AirTrojan

A. Overview

Fig. 2 illustrates the workflow of our proposed attack, *AirTrojan*. In general, the attacker launches AirTrojan (TDoS or Collusion) to achieve the goal of manipulating the results from the client selection process.

B. Attacking CSI-based Client Selection

1) AirTrojan-TDoS: TDoS attack on client selection in FL aims to launch DoS attacks on a targeted victim C_v, i.e., preventing C_v from being selected for participating the FL training process. Note that TDoS attack does not harm Θ, i.e., the global model of FL, directly. Instead, TDoS aims to reduce C_v's probability of being selected and serves as the preamble attack for advanced ones such as MPAs. The stealthiness of TDoS attack is high as well for the same reason: PS may not notice the attack since model performance of Θ such as model accuracy is not affected. The **intuition** behind TDoS is that from the perspective of client selection at PS, if the attacker's CSI offers a higher channel gain and comes with a lower interference when compared to the channel gain and interference of C_v, PS will select the attacker for participating training instead of C_v, thus significantly lowering the probability of C_v being selected.

Based on this intuition, a TDoS attacker achieves its goal through the following steps. Assume that $H^0 = [h_1^T, h_2^T, \ldots, h_v^T, \ldots, h_{m-1}^T, h_m^T, h_{m+1}^T \ldots, h_N^T]^T$ is the channel matrix of the MU-MIMO system for the wireless FL model. h_v and h_v denote the CSIs of the victim C_v and the attacker C_m, respectively. Firstly, the attacker forges and reports its CSI as $\tilde{h}_m = \alpha h_v + \beta e_v$,, where h_v is the victim's CSI and e_v the orthogonal component of h_v with respect to the other $N-1$ channel vectors in H. α and β are the two parameters the attacker engineers for achieving TDoS attack. More specifically, $\alpha > 1$ is an up-scaling factor that increases the attacker's effective channel gain, i.e., $\tilde{h}_m$ while $\beta < 1$ a down-scaling factor to reduce the interference from the other $N-1$ selected clients. Through a pair of (α, β), the attacker can effectively modify the channel matrix from H^0 to $H' = [h_1^T, h_2^T, \cdots, h_v^T, \cdots, h_{m-1}^T, \tilde{h}_m^T, h_{m+1}^T \ldots, h_N^T]^T$. As a result, most likely PS will select C_m over C_v simply due to that C_m offers a higher channel gain and a lower interference than C_v does. In Section V, we demonstrate that TDoS can achieve a high success rate of preventing C_v from being selected by PS.

2) AirTrojan-Collusion: Collusion attack on client selection in FL aims to increase (or decrease) the probability of a conspirator (a client that colludes with the attacker) being selected for participating in the FL training process. Similar to TDoS, Collusion attack does not necessarily harm Θ. Instead, our Collusion attack only aims to first change (increase or decrease) the conspirator's probability of being selected and then facilitate more advanced adversarial attacks such as MPAs. Here we denote the conspirator by C_{con}. C_{con} is assumed to behave as a benign client, i.e., C_{con} itself will not change its own CSIs. In order to change its probability of being selected for FL training, C_{con} will collude with C_m, the attacker, who is capable of forging or self-reporting the desired CSIs so that C_m can help C_{con} achieve its goal. For simplicity, we assume that C_{con} wants to increase its probability of being selected while the findings here apply to the case that C_{con} wants to decrease the probability.

As introduced above, attacker C_m wants to increase the selection probability of her conspirator C_{con} through Collusion attack. Note that C_{con} is assumed to have low probabilities of being selected for FL model training hence in need of Collusion attack from C_m. The steps of Collusion attack are as follows. First, C_m identifies a victim client, C_v, of which the selection probability is high. Following that, C_m identifies another client, C_μ, of which the selection probability is higher than that of C_v. Attacker C_m is able to do so because all CSIs are published on the MU-MIMO system hosting the wireless FL model investigated. Based on the above knowledge, C_m first launches TDoS attack on by forging (or self-reporting) a proper CSI $\tilde{h}_m$ while making sure that $\tilde{h}_m$ casts large interference to C_v but not C_{con}. In all, the impacts of the above Collusion attack is to reduce the selection probability of C_v and C_μ without hurting C_{con}'s channel quality. As a result, the ranking of C_{con} is expected to be escalated though C_{con} is not guaranteed to be selected yet. Intuitively, if C_{con} is close to being selected for FL model training, Collusion attack for C_{con} is likely to succeed while the attack would fail if originally C_{con} is far away from being selected.

In effect, C_m and C_{con} first choose a proper C_v with a high probability of being selected according to $\arg\max_{v \in \omega_{sel}} \{\mathbf{P}(v \in \omega_{sel,k} | \omega_{sel,k-1})\} (v \leq N)$, where $\mathbf{P}(v \in \omega_{sel,k} | \omega_{sel,k-1})$ is the conditional probability that C_v is selected in the k-th round (i.e., $\omega_{sel,k}$) provided the set of selected clients, i.e., $\omega_{sel,k-1}$, in the previous round. As a result, C_m can further locate C_μ, which is one of the clients that have a higher rank than C_v. Next, C_m forges $\tilde{h}_m$ so that $\tilde{h}_m$ casts a TDoS attack to C_μ while introducing large interference to C_v and much lower interference to C_{con}. Note that depending on how close C_m is from being selected, the Collusion attack may fail due to a significant gap (C_{con} is assumed not to be able to change h_{con}.)

C. AirTrojan for Model Poisoning Attacks

Model Poisoning Attacks (MPAs) arise as one type of critical attacks on FL due to their low cost and high effectiveness. In MPAs, an attacker acts as a client of FL and uploads malicious model updates into the global model to achieve attack goals. Specifically, untargeted MPAs are to degrade overall model performance such as model accuracy and targeted MPAs are to insert backdoors into the trained global model so that the poisoned model outputs targeted wrong predictions when fed with backdoored samples. Although a lot of MPAs and defenses have been proposed (see Section V), by default one of the key assumptions is that the MPA attacker(s) will participate every FL training round, which clearly does not hold in practice.

In this paper, we thus aim to answer the following two questions: (1) "How would SOTA MPAs perform when assuming that practical client selection is used during model training rather than that the MPA attacker is selected all the time?" and (2) "Would AirTrojan (either TDoS or Collusion attack) help escalate MPA performance even with practical client selection in place?" The answers to the above questions allow us to build a more realistic understanding of the importance of client selection and the severity of MPAs on FL. We further consider the scenarios when PS adopts SOTA defenses against MPAs as well. Given that client selection is indispensable for FL, we conjecture that our findings from studying MPAs here can extend to other advanced adversarial attacks on FL as well such as adversarial example attacks [26]–[28].

V. Performance Evaluation

A. Experimental Setup

Datasets and model architecture. We evaluate AirTrojan on three popular datasets in the literature: MNIST, Fashion-MNIST (F-MNIST), and CIFAR-10. In our experiments, all clients were configured as i.i.d. as in [1]. This means that the data at each client was a random sampling from the overall dataset, ensuring a uniform data distribution across different clients. For MNIST and F-MNIST, we use LeNet [29] as the backbone and VGGNet [30] for CIFAR-10. We implement AirTrojan in Pytorch and run all the evaluations on a server with an Intel(R) Core(TM) i9-10900X CPU @ 3.70GHz and two NVIDIA RTX 3090. We have made the source code for AirTrojan available at https://github.com/CCS2o23/AirTrojan.

FL Settings. The MU-MIMO wireless FL system investigated has a candidate pool of 100 clients. The wireless channel for an arbitrary transmitter-receiver pair is assumed to be standard i.i.d. Rayleigh fading channel, i.e., $\mathcal{CN}$, which has been widely adopted for modeling CSI [9], [31], [32]. The client selection algorithm is to select clients from the candidate pool. For local training, local model updates are trained using the Adam optimizer with a learning rate of 0.001. For FL aggregation methods, we consider 7 of them in total: `FedAvg` [1], `Krum` [33], `MKrum` (i.e., Multi-Krum), `T-Mean` (i.e., Trimmed Mean) [34], `Median` [34], `Bulyan` [35], and `FLTrust` [12]. `FedAvg` has been the default aggregation method of generic FL systems but it is vulnerable to most MPAs. All of the above aggregation methods except `FedAvg` have been proposed as defenses against different MPAs. Particularly, `Krum`, `MKrum`, `T-Mean`, and `Median` are defenses against MPAs based on Byzantine failure while `FLTrust` and `Bulyan` are against stealthy MPAs in [23].

MPAs. We evaluate *AirTrojan* for three types of SOTA MPAs to demonstrate its impacts. The first type is `Untargeted MPAs` including Krum-Attack [22] (MPA on Krum) and Trimmed-Attack [22] (MPA on Trimmed Mean), which are to manipulate model weights for degrading the model accuracy. The second is `Stealthy Targeted MPA` in [23] which is to subtly insert backdoors into the global model by crafting malicious model updates resembling benign ones so that they are very likely to be merged into the global model. The third type is `Semantic Backdoor Attack` in [24] which is to leverage inherent data patterns of training samples (such as the colors in a sample image) for triggers rather than using synthetic patterns proposed in the first two types of MPAs.

Performance metrics. We mainly use attack success rate (ASR) as performance metric across all experiments. For TDoS, ASR denotes the ratio of successful attack runs over all runs, i.e., when is not selected for FL training due to TDoS attacks. We use the same definition of ASR on Collusion attack except that a successful Collusion attack refers to that is selected for participating FL training due to Collusion attacks. For MPAs, we use the same performance metrics as in [22]–[24]: model accuracy for untargeted MPAs and ASR for targeted MPAs, i.e., the ratio of successful targeted predictions among all backdoored testing samples.

B. Impacts on Client Selection & Model Performance

TABLE I: ASR(%) of AirTrojan on client selection.

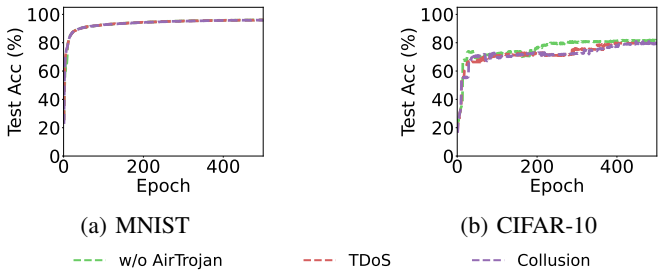
Ranking	20	40	60	80	100
TDoS-10	100.00	100.00	100.00	100.00	100.00
TDoS-15	100.00	100.00	100.00	100.00	100.00
TDoS-20	100.00	100.00	100.00	100.00	100.00
Collusion-10	100.00	79.70	63.70	10.96	0.70
Collusion-15	100.00	79.82	65.60	11.90	0.80
Collusion-20	100.00	81.00	68.70	13.80	1.20

Attacks on client selection. Table I summarizes the results of launching AirTrojan (including TDoS and Collusion attack) on the client selection step when training a generic FL model. Note that the settings of dataset and backbone do not affect the attack results here. Let denote the number of clients to be selected from the candidate pool consisting of 100 clients. TDoS- corresponds to the scenario when launching TDoS attack to client selection of out of 100 clients (the same as in Collusion-). For TDoS attacks, we

assume there is only one attacker in the candidate pool. For Collusion attack, we assume there is one attacker and one conspirator, i.e., the colluded client, in the candidate pool. For TDoS attack, the entries in the ranking row correspond to the ranking of the attacker while for Collusion attack, the entries correspond to the conspirator's ranking.

For TDoS, was assumed to be between 1.0 and 1.2, which translated to that the channel gain of , i.e., the attacker, was not ridiculously high. Similarly, was assumed to be between 0 and 0.5 for practicality (is unlikely to introduce arbitrarily large interference to other clients).Therefore,we set to 1.2 and to 0.5 provided a realistic attacker capability in evaluation.When evaluating TDoS attacks, we chose (thus) that ranked between 20th and 100th among all candidates meaning that would not be selected without AirTrojan attacks. Table I shows that TDoS attacks achieved 100% ASR under all s across 10,000 client selection runs. The results suggest that an attacker can launch TDoS attacks to sabotage the targeted client effectively regardless of the original ranking of the attacker.

Table I also shows the attack performance of Collusion attack. Similarly, we chose (thus) that ranked between 20th to 100th among all candidates meaning that would not be selected if there was no Collusion attacks. Compared to TDoS attacks, ASR of Collusion attacks highly depends on the original ranking of . Specifically, for Collusion-10 (i.e., client selection is to choose 10 clients from the candidate pool consisting of 100 clients), if ranks around 20th among all clients, Collusion attacks can achieve nearly 100% ASR. However, if ranks 40th or even worse, ASR of Collusion attacks would drop significantly. Similar results were observed for Collusion-15 and Collusion-20. Such findings suggest that Collusion attacks require that the conspirator herself has a relatively high ranking before the attacker may be able to further escalate her ranking so as to be selected.



(a) MNIST (b) CIFAR-10

Fig. 3: Testing accuracy of the global model.

Impact on model accuracy of FL models. As we mentioned in Section IV and shown in Fig. 3 ( ), AirTrojan will not harm the model accuracy of a trained global model in wireless FL directly because AirTrojan focuses on manipulating the results of client selection (i.e., which clients are selected by PS).

C. Impacts on Untargeted MPAs

We adopt the following settings for client selection at PS. (1) **Legacy 1**: One MPA attacker is to participate all FL training rounds. (2) **Legacy 2**: Two colluded MPA attackers are to participate all FL training rounds. (3) **w/o AirTrojan**: PS uses CSI-based client selection (see Algorithm 1) while there are two MPA attacks among the 100 clients in the candidate pool. (4) **TDoS**: The same settings as in (3) while the two MPA attackers launch TDoS attacks on client selection step. (5) **Collusion**: The same settings as in (3) while the two MPA attackers launches Collusion attacks on client selection step to escalate the rankings of two conspirators.

Table II lists model accuracy under Krum-Attack and Trimmed-Attack with 7 aggregation methods over three datasets. The observations are as follows. **Firstly**, under the settings of Legacy 1 and Legacy 2, untargeted MPAs either Krum-Attack or Trimmed-Attack were highly effective as the global model under attack only achieved low model accuracy. Specifically, Krum-Attack is quite effective on FedAvg and Krum while Trimmed-Attack on FedAvg and T-Mean. **Secondly**, under the settings of w/o AirTrojan meaning that PS adopts client selection while PS is not under the proposed AirTrojan attack, untargeted MPAs became less effective as the global model under MPA attacks could achieve much higher accuracy compared to Legacy 1 and Legacy 2. This indicates that client selection step can significantly affect the attack performance of adversarial attacks on FL such as untargeted MPAs here. **Lastly**, under the settings of TDoS and Collusion meaning that the client selection at PS was under TDoS attacks or Collusion attacks, untargeted MPAs were able to achieve comparable attack performance as under Legacy 1 and Legacy 2 settings. This further suggests that client selection is critical and adversarial attacks on FL should not take it for granted that the attacker(s) would be selected all the time.

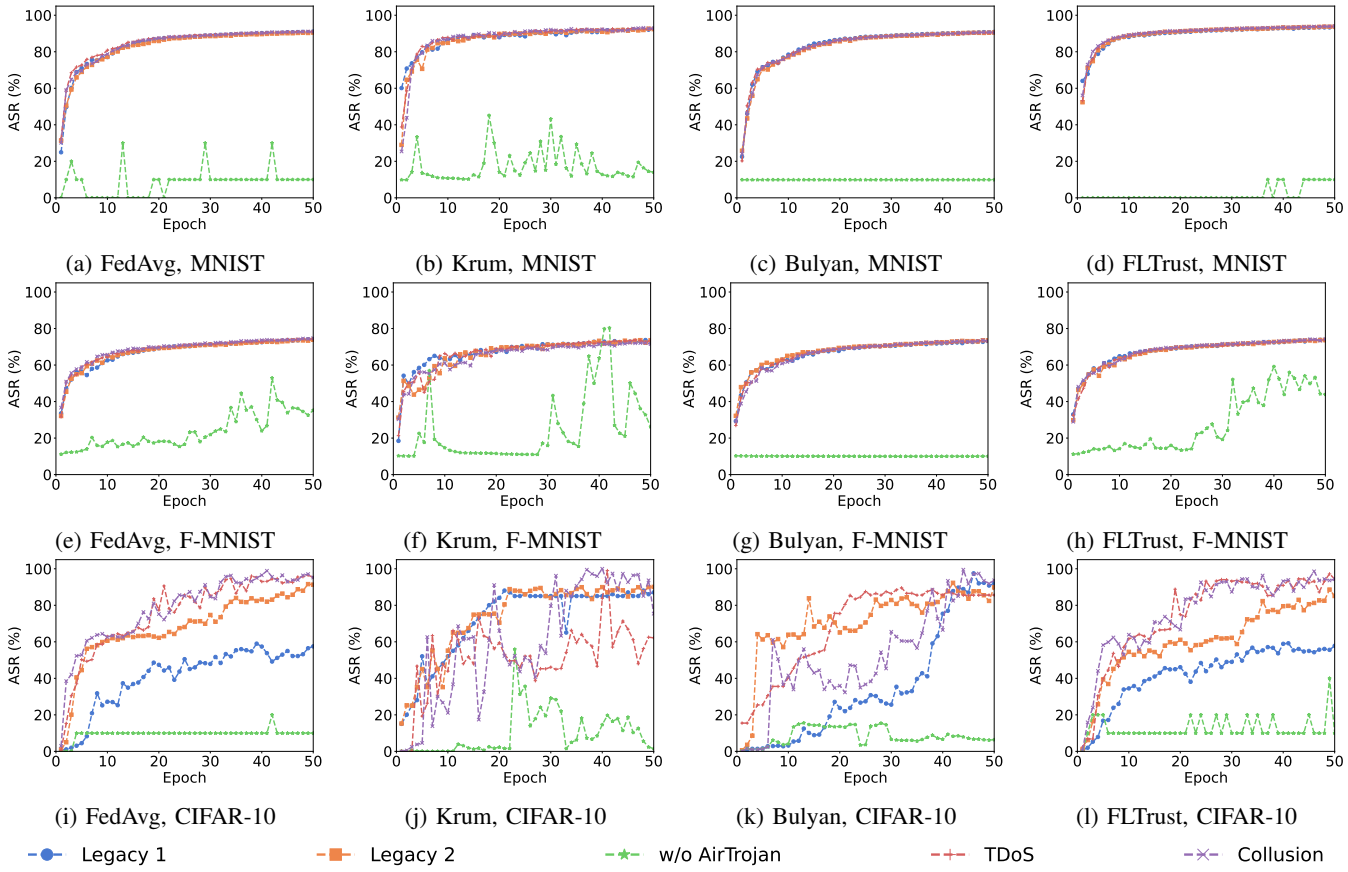
Table II shows that different MPAs are effective on certain aggregation methods rather than all of them. Examples are that Krum-Attack is effective on FedAvg and Krum but not other aggregation methods while Trimmed-Attacks on FedAvg and T-Mean but not other aggregation methods. Aggregations methods like MKrum, Median, Bulyan, and FLTrust are feasible defenses against the two untargeted MPAs explored here, which aligns with the conclusions in [23], [36].

D. Impacts on Targeted Stealthy MPA

Fig. 4 shows ASRs of stealthy model poisoning (StealthyMP) on three datasets when four aggregations were deployed as the defense. Due to space limit, we cannot include results from all 7 aggregation methods. The main conclusions are consistent from those from untargeted MPAs. Particularly, when client selection was assumed to be perfect for MPA attackers, i.e., an attacker was guaranteed to participate every traing round of FL, StealthyMP achieved the highest ASRs. However, the situation changed significantly

TABLE II: Model accuracy (%) under untargeted MPAs. w/o: Client selection without AirTrojan.

Krum-Attack							Trimmed-Attack						
Defenses	**Dataset**	**Client Selection**					**Defenses**	**Dataset**	**Client Selection**				
		Legacy1	Legacy2	w/o	TDoS	Collusion			Legacy1	Legacy2	w/o	TDoS	Collusion
	MNIST	10.09	9.58	**91.56**	14.57	15.12		MNIST	17.34	9.58	**93.90**	23.08	10.28
FedAvg	F-MNIST	22.13	15.61	**72.04**	11.34	9.75	**FedAvg**	F-MNIST	10.75	9.07	**77.16**	9.91	9.45
	CIFAR-10	9.59	9.76	**51.79**	9.94	9.38		CIFAR-10	23.35	15.84	**50.64**	22.10	10.83
	MNIST	13.01	12.86	**95.92**	12.87	12.80		MNIST	95.66	95.57	95.59	95.57	95.29
Krum	F-MNIST	10.00	9.96	**75.22**	9.93	10.08	**Krum**	F-MNIST	77.65	77.24	77.68	77.00	76.82
	CIFAR-10	32.38	26.43	**49.27**	42.59	29.15		CIFAR-10	48.95	48.48	49.24	49.91	49.16
	MNIST	83.43	77.98	**96.12**	86.44	69.09		MNIST	91.98	90.70	95.92	92.42	90.85
MKrum	F-MNIST	56.27	43.59	**75.45**	55.34	32.27	**MKrum**	F-MNIST	77.30	77.15	77.48	77.41	77.26
	CIFAR-10	33.05	29.27	**46.82**	34.30	26.57		CIFAR-10	50.91	51.27	50.18	50.16	51.51
	MNIST	96.44	83.04	**96.53**	95.89	80.27		MNIST	43.98	11.17	**96.52**	53.27	10.09
T-Mean	F-MNIST	**77.64**	51.67	77.59	76.97	35.27	**T-Mean**	F-MNIST	28.36	23.12	**77.63**	26.88	8.80
	CIFAR-10	51.34	40.91	**51.36**	48.53	43.96		CIFAR-10	17.49	10.12	**51.34**	16.63	8.67
	MNIST	96.35	**96.48**	96.38	95.91	96.54		MNIST	96.32	96.39	**96.49**	95.89	96.27
Median	F-MNIST	77.56	77.57	**77.64**	77.23	77.05	**Median**	F-MNIST	77.65	76.82	**77.68**	77.00	76.16
	CIFAR-10	51.11	51.79	**51.39**	49.23	51.85		CIFAR-10	51.34	50.98	**51.37**	48.93	50.98
	MNIST	96.15	96.37	**96.43**	96.27	96.32		MNIST	96.17	96.26	**96.40**	96.28	96.19
Bulyan	F-MNIST	75.33	73.50	**77.44**	75.15	73.86	**Bulyan**	F-MNIST	77.06	77.08	**77.56**	77.51	77.10
	CIFAR-10	51.41	51.04	**51.48**	50.53	51.07		CIFAR-10	49.78	50.04	**51.62**	50.38	50.87
	MNIST	96.31	96.35	**96.70**	96.48	83.76		MNIST	96.32	96.33	**96.71**	96.46	96.25
FLTrust	F-MNIST	77.29	**77.71**	77.65	77.56	77.37	**FLTrust**	F-MNIST	77.36	**77.61**	77.53	77.38	76.65
	CIFAR-10	**51.69**	51.48	51.60	51.96	51.32		CIFAR-10	51.75	51.42	51.68	**51.78**	50.96



(a) FedAvg, MNIST (b) Krum, MNIST (c) Bulyan, MNIST (d) FLTrust, MNIST

(e) FedAvg, F-MNIST (f) Krum, F-MNIST (g) Bulyan, F-MNIST (h) FLTrust, F-MNIST

(i) FedAvg, CIFAR-10 (j) Krum, CIFAR-10 (k) Bulyan, CIFAR-10 (l) FLTrust, CIFAR-10

Fig. 4: Impacts of AirTrojan on StealthyMP.

when assuming that PS deployed client selection such as the CSI-based method (see Algorithm 1), which holds more often in practice. Specifically, client selection at PS can effectively affect (decrease) the probability of the MPA attacker(s) being selected thus significantly reduce ASR of StealthyMP. The results from Fig. 4 suggest that StealthyMP achieved only low ASRs (lower than 10%) and thus became almost ineffective under w/o AirTrojan setting. Finally, when combined with

TABLE III: ASR(%) of SemanticMP with and without AirTrojan.

Dataset	Setting	FedAvg	Krum	MKrum	T-Mean	Median	Bulyan	FLTrust
MNIST	Legacy 1	99.25	99.28	99.17	96.28	97.64	92.52	91.55
	Legacy 2	99.23	99.20	99.06	96.49	97.88	92.75	91.42
	w/o AirTrojan	**15.61**	**9.76**	**6.83**	**6.58**	**10.05**	**9.89**	**2.25**
	TDoS	99.18	99.22	98.87	96.56	97.16	92.78	94.58
	Collusion	99.26	99.24	99.52	96.78	97.37	92.92	97.19
F-MNIST	Legacy 1	93.17	93.06	93.12	92.08	93.08	92.98	90.64
	Legacy 2	92.98	92.98	93.08	92.14	93.16	92.93	91.04
	w/o AirTrojan	**11.37**	**9.38**	**3.58**	**4.60**	**11.29**	**6.79**	**8.67**
	TDoS	93.14	93.03	93.10	92.10	93.14	92.01	90.28
	Collusion	92.93	92.99	93.05	91.79	93.21	91.65	91.89
CIFAR-10	Legacy 1	86.11	85.08	85.92	85.60	85.16	85.61	85.46
	Legacy 2	86.06	85.25	86.00	85.52	85.38	85.54	85.39
	w/o AirTrojan	**9.94**	**10.11**	**6.09**	**8.92**	**9.33**	**7.15**	**10.03**
	TDoS	85.93	85.86	85.85	85.49	85.19	85.51	85.53
	Collusion	86.05	85.95	85.98	85.34	85.30	85.42	85.63

the proposed AirTrojan attacks, StealthyMP was able to achieve similar attack performance as under Legacy 1 and Legacy 2, i.e., the ideal assumption that an MPA attacker was guaranteed to be selected for model training. In conclusion, client selection is critical for StealthyMP while AirTrojan can escalate StealthyMP by attacking client selection at PS.

E. Impacts on Targeted Semantic MPA

Table III lists the ASRs of the semantic backdoor attack (SemanticMP) across different aggregation methods and datasets, which shares similar observations with those of StealthyMP. The key conclusion is that SemanticMP became almost ineffective when we move from the ideal assumption that an MPA attacker is always selected for model training (such as under Legacy 1 and Legacy 2 settings) to the more realistic setting where PS deploys client selection and an MPA attacker can no longer be selected all the time. With AirTrojan, MPA attackers can increase their probabilities of being selected for model training.

VI. Countermeasures

Our evaluations have shown that the robustness of CSI-based client selection is essential in wireless FL. Simple solutions like encrypting CSI feedback offer enhanced security with minimal changes at the base station but can introduce additional overheads and latency [37]. Meanwhile, the authors in [3] proposed reciprocal consistency checking to protect client selection while [38] explored machine learning for detecting CSI inconsistency. In general, further research is still needed for securing client selection in FL.

VII. Related Work

Client selection in FL. Different client selection algorithms have been proposed for different goals. In [31], the authors proposed a joint problem of both model convergence time and data communication latency in practical wireless networks. In [39], Power-Of-Choice was proposed to select clients in a biased manner based on local losses in order to reduce training iterations. In [32], the authors proposed to fine-tune client selection particularly under realistic resource constraints such as spectrum, data availability, etc. Another direction for client selection is to design a score system for local nodes and select the ones with high score for participating training [40], [41].

Attacking CSI. Attacks particularly eavesdropping on CSI-based applications have been seen frequently. In [5], [18], the authors demonstrated the feasibility of eavesdropping on transmitted data such as private photos via attacking CSI. Recently in [4], researchers further showed that eavesdropping attacks on nodes in wireless networks not only were feasible but also offered fine-grained control of which node(s) and what packets to attack. MUSTER [3] is the first to attack client selection in MU-MIMO systems.

MPAs on FL. MPAs have been shown as a formidable threat to distributed FL system, including untargeted and targeted attacks. Untargeted MPAs broadly aim to degrade the overall model performance, thereby compromising the efficiency of the learning process [21]. In targeted MPAs including backdoor attacks, the compromised global model is manipulated to produce specific outputs chosen by the attacker [23], [24], [42]. Note that in existing MPAs, it is assumed that malicious clients will participate in every training round. However, when FL systems are deployed in real-world scenarios like under wireless systems, oftentimes such an assumption will not hold [10].

Byzantine aggregation in FL. Byzantine aggregation in FL was proposed for tackling unreliable or malicious participants during the model update process. In this context, researchers have developed several aggregation algorithms, minimizing the negative impact of such participants through outlier detection [12], [35]. However, they were known to fail when facing persistent participation from malicious attackers or more stealthy MPAs in FL training as in [43], [44].

Differences in AirTrojan. AirTrojan is the first work to investigate how to attack client selection in FL. We also explore how such attacks impact model performance and other adversarial attacks on FL assuming no client selection.

VIII. Conclusion

In this paper, we proposed *AirTrojan* to undertake a pioneering empirical study of the vulnerabilities from client selection in wireless FL. Our evaluations demonstrate that

AirTrojan can manipulate client selection results through TDoS and Collusion attacks and is necessary for advanced attacks on FL such as MPAs to succeed. Our findings indicate that client selection is a key step and requires dedicated research focus in FL regardless of attacks on FL or corresponding defenses.

ACKNOWLEDGMENT

This work was supported in part by the US National Science Foundation under grants CNS-2422863 and CMMI-2401555. We would also like to thank anonymous reviewers for their constructive comments and helpful advice.

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 *AISTATS*, PMLR, 2017.

[2] J. Jeon, G. Lee, A. A. Ibrahim, *et al.*, "Mimo evolution toward 6g: Modular massive mimo in low-frequency bands," *IEEE Communications Magazine*, vol. 59, no. 11, pp. 52–58, 2021. DOI: 10.1109/MCOM.211.2100164.

[3] T. Hou, S. Bi, T. Wang, *et al.*, "Muster: Subverting user selection in mu-mimo networks," in *IEEE INFOCOM*, IEEE, 2022.

[4] S. Wang, Z. Chen, Y. Xu, Q. Yan, C. Xu, and X. Wang, "On user selective eavesdropping attacks in mu-mimo: Csi forgery and countermeasure," in *IEEE INFOCOM*, IEEE, 2019.

[5] Y. Mao, Y. Zhang, and S. Zhong, "Stemming downlink leakage from training sequences in multi-user mimo networks," in *ACM CCS*, 2016.

[6] J. Zhang, Z. Tang, R. Li, *et al.*, "Protect sensitive information against channel state information based attacks," in *IEEE CSE EUC*, IEEE, vol. 2, 2017, pp. 203–210.

[7] S. Huang, H. Yin, J. Wu, and V. C. Leung, "User selection for multiuser mimo downlink with zero-forcing beamforming," *IEEE TVT*, vol. 62, no. 7, pp. 3084–3097, 2013.

[8] H. W. Jie Xu, "Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective," *IEEE TWC*, vol. 20, no. 2, pp. 1188–1200, 2020.

[9] J. Mao, H. Yang, P. Qiu, J. Liu, and A. Yener, "Charles: Channel-quality-adaptive over-the-air federated learning over wireless networks," in *SPAWC*, IEEE, 2022.

[10] F. Pase, M. Giordani, and M. Zorzi, "On the convergence time of federated learning over wireless networks under imperfect csi," in *ICC Workshops*, IEEE, 2021.

[11] Y. Liu, S. Ma, Y. Aafer, *et al.*, "Trojaning attack on neural networks," in *NDSS*, Internet Soc, 2018.

[12] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping," in *ISOC NDSS*, 2021.

[13] T. D. Nguyen, P. Rieger, H. Chen, *et al.*, "FLAME: Taming backdoors in federated learning," in *USENIX Security*, 2022.

[14] K. Zhang, G. Tao, Q. Xu, *et al.*, "Flip: A provable defense framework for backdoor mitigation in federated learning," in *ICLR*, 2023.

[15] Z. Zhang, X. Cao, J. Jia, and N. Z. Gong, "Fldetector: Defending federated learning against model poisoning attacks via detecting malicious clients," in *KDD*, 2022.

[16] Q. Spencer, C. Peel, A. Swindlehurst, and M. Haardt, "An introduction to the multi-user mimo downlink," *IEEE Communications Magazine*, vol. 42, no. 10, pp. 60–67, 2004. DOI: 10.1109/MCOM.2004.1341262.

[17] Z. Yang, Z. Zhou, and Y. Liu, "From rssi to csi: Indoor localization via channel response," *ACM Computing Surveys (CSUR)*, vol. 46, no. 2, pp. 1–32, 2013.

[18] M. Li, Y. Meng, J. Liu, *et al.*, "When csi meets public wifi: Inferring your mobile phone password via wifi signals," in *ACM CCS*, 2016.

[19] G. D. Németh, M. Á. Lozano, N. Quadrianto, and N. Oliver, "A snapshot of the frontiers of client selection in federated learning," *arXiv preprint arXiv:2210.04607*, 2022.

[20] M. Fang, X. Cao, J. Jia, and N. Z. Gong, "Local model poisoning attacks to byzantine-robust federated learning," in *USENIX Security*, 2020.

[21] V. Shejwalkar and A. Houmansadr, "Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning," in *NDSS*, 2021.

[22] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to Byzantine-Robust federated learning," in *USENIX Security*, 2020.

[23] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," in *ICML*, PMLR, 2019.

[24] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *AISTATS*, PMLR, 2020.

[25] G. Baruch, M. Baruch, and Y. Goldberg, "A little is enough: Circumventing defenses for distributed learning," *NeurIPS*, 2019.

[26] C. Szegedy, W. Zaremba, I. Sutskever, *et al.*, "Intriguing properties of neural networks," in *ICLR*, 2013.

[27] N. Wang, Y. Chen, Y. Xiao, Y. Hu, W. Lou, and Y. T. Hou, "Manda: On adversarial example detection for network intrusion detection system," *IEEE TDSC*, vol. 20, no. 2, pp. 1139–1153, 2022.

[28] Y. Hu, N. Wang, Y. Chen, W. Lou, and Y. T. Hou, "Transferability of adversarial examples in machine learning-based malware detection," in *IEEE CNS*, IEEE, 2022, pp. 28–36.

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

[30] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *ICLR*, 2015.

[31] Z. Yang, M. Chen, W. Saad, *et al.*, "Delay minimization for federated learning over wireless communication networks," *arXiv preprint arXiv:2007.03462*, 2020.

[32] M. B. Madhusanka Manimel Wadu Sumudu Samarakoon, "Federated learning under channel uncertainty: Joint client scheduling and resource allocation," in *WCNC*, IEEE, 2020.

[33] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *NeurIPS*, vol. 30, 2017.

[34] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *ICML*, PMLR, 2018.

[35] R. Guerraoui, S. Rouault, *et al.*, "The hidden vulnerability of distributed learning in byzantium," in *ICML*, PMLR, 2018.

[36] B. Zhu, L. Wang, Q. Pang, *et al.*, "Byzantine-robust federated learning with optimal statistical rates," in *AISTATS*, PMLR, 2023.

[37] Y. Yang, Y. Chen, W. Wang, and G. Yang, "Securing channel state information in multiuser mimo with limited feedback," *IEEE TWC*, vol. 19, no. 5, pp. 3091–3103, 2020.

[38] F. O. Catak, M. Kuzlu, E. Catak, U. Cali, and D. Unal, "Security concerns on machine learning solutions for 6g networks in mmwave beam prediction," *Physical Communication*, vol. 52, 2022.

[39] Y. J. Cho, J. Wang, and G. Joshi, "Towards understanding biased client selection in federated learning," in *AISTATS*, PMLR, 2022.

[40] Z. Song, H. Sun, H. H. Yang, X. Wang, Y. Zhang, and T. Q. Quek, "Reputation-based federated learning for secure wireless networks," *IEEE Internet of Things Journal*, vol. 9, no. 2, pp. 1212–1226, 2021.

[41] C. Zhang, N. Wang, S. Shi, C. Du, W. Lou, and Y. T. Hou, "Mindfl: Mitigating the impact of imbalanced and noisy-labeled data in federated learning with quality and fairness-aware client selection," in *MILCOM*, IEEE, 2023.

[42] N. Wang, Y. Xiao, Y. Chen, Y. Hu, W. Lou, and Y. T. Hou, "Flare: Defending federated learning against model poisoning attacks via latent space representations," in *AsiaCCS*, 2022, pp. 946–958.

[43] H. Zhuang, M. Yu, H. Wang, Y. Hua, J. Li, and X. Yuan, "Backdoor federated learning by poisoning backdoor-critical layers," in *ICLR*, 2024.

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