Poisoning Attacks on Federated Learning-based Wireless Traffic Prediction

Zifan Zhang*, Minghong Fang†, Jiayuan Huang*, Yuchen Liu*
*North Carolina State University, USA, †University of Louisville, USA

***Abstract*—Federated Learning (FL) offers a distributed framework to train a global control model across multiple base stations without compromising the privacy of their local network data. This makes it ideal for applications like wireless traffic prediction (WTP), which plays a crucial role in optimizing network resources, enabling proactive traffic flow management, and enhancing the reliability of downstream communication-aided applications, such as IoT devices, autonomous vehicles, and industrial automation systems. Despite its promise, the security aspects of FL-based distributed wireless systems, particularly in regression-based WTP problems, remain inadequately investigated. In this paper, we introduce a novel fake traffic injection (FTI) attack, designed to undermine the FL-based WTP system by injecting fabricated traffic distributions with minimal knowledge. We further propose a defense mechanism, termed global-local inconsistency detection (GLID), which strategically removes abnormal model parameters that deviate beyond a specific percentile range estimated through statistical methods in each dimension. Extensive experimental evaluations, performed on real-world wireless traffic datasets, demonstrate that both our attack and defense strategies significantly outperform existing baselines.**

***Index Terms*—Poisoning attacks, wireless traffic prediction, federated learning, injection attack.**

I. Introduction

Federated learning (FL) represents an evolving paradigm in distributed machine learning techniques, allowing a unified model to be trained across numerous devices containing local data samples, all without the need to transmit these samples to a central server. This innovative framework empowers training on diverse datasets characterized by heterogeneous distributions, offering substantial advantages in the current landscape of big data. In practical applications, FL has found widespread use in addressing real-world challenges, particularly in environments dealing with sensitive or personal data, including the Internet of Things (IoT) [1], [2], edge computing [3], and health informatics [4], [5].

In the realm of wireless networks, FL leverages its distributed nature to facilitate multiple network services, including wireless traffic prediction (WTP). With the exponential growth in the number of connected devices and the ever-increasing demand for data-intensive applications like streaming, online gaming, and IoT services, predicting wireless traffic accurately becomes vital for ensuring network reliability and efficiency. By forecasting network load on a temporal basis, service providers can dynamically allocate resources, reducing the risk of congestion and ensuring a high Quality of Service (QoS) for users [6]–[8]. Furthermore, accurate traffic predictions enable operators to strategically plan network expansions and efficiently upgrade infrastructure, resulting in cost savings and enhanced network performance. Particularly, in the era of 5G and beyond, where technologies like network slicing and edge computing play crucial roles, WTP becomes essential for optimizing these advancements, which not only enhances user experience but also facilitates the provision of innovative services that demand high bandwidth and low latency. To implement WTP, while centralized methods exist [9], [10], FL-based solution stands out by utilizing training data distributed across diverse edge nodes. This approach enhances the generation of precise and timely predictions concerning network traffic [11]. Despite FL's potential in accuracy, efficiency, and privacy preservation, its integration into WTP is not devoid of challenges. Notably, Byzantine attacks, particularly model poisoning attacks, pose significant threats to the effectiveness and trustworthiness of FL-based WTP systems [12].

In a model poisoning attack, malicious network entities introduce adversarial modifications to the model parameters during training process of WTP. This tampering results in a compromised global model when aggregated at the central network controller, subsequently producing incorrect traffic predictions. Such inaccuracies lead to the risk of network inefficiencies and even severe service disruptions, especially in real-time applications like autonomous driving systems. In more extreme scenarios, these attacks may serve as gateways to further malicious network intrusions, instigating broader security and privacy concerns as illustrated in [13], [14]. The grave implications of model poisoning attacks underscore the pressing need for robust security measures to ensure the integrity, reliability, and resilience of FL-based WTP systems against Byzantine failures, thereby safeguarding the overarching network infrastructure and the services reliant on it. While most existing FL algorithms and their associated security strategies are typically assessed within the context of *classification* problems [15], [16], scant attention has been paid to the *regression* problems, as observed in examined WTP scenarios, introducing distinct challenges related to data distribution, model complexity, and evaluation metrics. The distinction between data manipulation strategies in regression and classification problems, as well as their detection methodologies, underscores the nuanced challenges in safeguarding machine learning models against attacks. For instance, in regression-based WTP problems, attackers typically target the model's continuous output by altering the distribution or magnitude of

ISBN 978-3-903176-63-8 © 2024 IFIP

input time-series data, with the goal of steering predictions in a specific direction. This differs from classification tasks, where the manipulation revolves around modifying input features to induce misclassification without noticeably changing the input's appearance to human observers.

To bridge this gap, we make the first attempt to introduce a novel attack centered on injecting fake base station (BS) traffic into wireless networks. Existing model poisoning attacks have predominantly depended on additional access knowledge and direct intrusions on BSs [12], [15], [17]. However, in practical cellular network systems, BSs have exhibited a commendable level of resilience against attacks, making the extraction of training data from them a challenging endeavor. In contrast, the cost of deploying fake BSs that mimic their behaviors is comparatively lower than the resources required for compromising authentic ones [18]. This assumption asserts that these compromised BSs lack insight into the training data and only have access to the initial and current global models, aligning with the practical settings studied in [18]. Importantly, other information, such as data aggregation rules and model parameters from benign BSs, remains inaccessible to these compromised BSs. Within the FL framework, the global model is aggregated based on the model parameters of BSs in each iterative round, encompassing both benign and fake BSs. Consequently, our threat model envisions a minimum-knowledge scenario for an adversary. To this end, we propose Fake Traffic Injection (FTI), a methodology designed to create undetectable fake BSs with minimal prior knowledge, where each fake BS employs both its initial model and current global information to determine the optimizing trajectory of the FL process on WTP. These malicious participants aim to subtly align the global model towards an outcome that undermines the integrity and reliability of the data learning process. Numerous numerical experiments are conducted to validate that our FTI demonstrates efficacy across various state-of-the-art aggregation rules, outperforming other model poisoning attacks in terms of vulnerability impacts.

On the contrary, we propose an innovative defensive strategy known as Global-Local Inconsistency Detection (GLID), aimed at neutralizing the effects of model poisoning attacks on WTP. This defense scheme involves strategically removing abnormal model parameters that deviate beyond a specific percentile range estimated through statistical methods in each dimension. Such an adaptive approach allows us to trim varying numbers of malicious model parameters instead of a fixed quantity [19]. Next, a weighted mean mechanism is employed to update the global model parameter, subsequently disseminated back to each BS. Our extensive evaluations, conducted on real-world datasets, demonstrate that the proposed defensive mechanism substantially mitigates the impact of model poisoning attacks on WTP, thereby showcasing a promising avenue for securing FL-based WTP systems against Byzantine attacks.

The contribution of this work is summarized in three folds:

1) We present a novel model poisoning attack, employing fake BSs for traffic injection into FL-based WTP systems under a minimum-knowledge scenario.
2) Conversely, we propose an effective defense strategy designed against various model poisoning attacks, which dynamically trims an adaptive number of model parameters by leveraging the percentile estimation technique.
3) Lastly, we evaluate both the proposed poisoning attack and the defensive mechanism using real-world traffic datasets from Milan City, where the results demonstrate that the FTI attack indeed compromises FL-based WTP systems, and the proposed defensive strategy proves notably more effective than other baseline approaches.

II. Related Works and Preliminaries

A. FL-based WTP

Consider a wireless traffic forecasting system that employs FL and incorporates a central server located in a macro-cell station along with n small-cell BSs (e.g., gNB). Every BS $i \in [n]$ possesses its own private training dataset $u_i = \{u_i^1, u_i^2, \ldots, u_i^M\}$, where M represents the total count of time intervals, and u_i^m denotes the traffic load on BS i during the m-th interval, with $m \in [M]$. To delineate a prediction model, we construct a series of input-output pairs $\{a_i^j, b_i^j\}_{j=1}^z$. Here, each a_i^j represents a historical subset of traffic data that correlates to its associated output $b_i^j = \{u_i^{m-1}, \ldots, u_i^{m-r}, u_i^{m-\omega 1}, \ldots, u_i^{m-\omega s}\}$. The parameters r and s serve as sliding windows capturing immediate temporal dependencies and cyclical patterns, respectively. Furthermore, ω encapsulates inherent periodicities within the network, potentially driven by diurnal user patterns or systematic service demands. Given the importance of real-time responsiveness in wireless networks, our prediction model is designed for a one-step-ahead forecast. To be specific, for the i-th BS, we seek to predict the traffic load $\tilde{b}_i^j$ based on the historical traffic data a_i^j and model parameter $\boldsymbol{\theta}$ as $\tilde{b}_i^j = f(a_i^j, \boldsymbol{\theta})$, where $f(\cdot)$ is the regression function.

In a FL-based WTP system, the objective is to minimize prediction errors across n BSs. This can be formulated as the following optimization problem to determine the optimal global model $\boldsymbol{\theta}^*$ in the central server:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \frac{1}{nz} \sum_{i=1}^{n} \sum_{j=1}^{z} F(f(a_i^j, \boldsymbol{\theta}), b_i^j), \tag{1}$$

where F is the quadratic loss, i.e., $F(f(a_i^j, \boldsymbol{\theta}), b_i^j) = \left| f(a_i^j, \boldsymbol{\theta}) - b_i^j \right|^2$. Eq. (1) can be resolved in a distributed fashion based on FL with the following three steps in each global training round t.

- **Step I (Synchronization).** The central server sends the current global model $\boldsymbol{\theta}^t$ to all BSs.
- **Step II (Local model training).** Each BS $i \in [n]$ utilizes its private time-series training data along with the current global model to refine its own local model, then transmits the updated local model $\boldsymbol{\theta}_i^t$ back to the server.
- **Step III (Local models aggregation).** The central server leverages the aggregation rule (AR) to merge the n re-

ceived local models and subsequently updates the global model as follows:

$$\boldsymbol{\theta}^{t+1} = \mathrm{AR}\{\boldsymbol{\theta}_1^t, \boldsymbol{\theta}_2^t, \ldots, \boldsymbol{\theta}_n^t\}. \quad (2)$$

The commonly used aggregation rule is the FedAvg [20], where the server simply averages the received n local models from distributed BSs, i.e., $\mathrm{AR}\{\boldsymbol{\theta}_1^t, \boldsymbol{\theta}_2^t, \ldots, \boldsymbol{\theta}_n^t\} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\theta}_i^t$.

B. Byzantine-robust Aggregation Rules

In non-adversarial scenarios, the server aggregates the received local model updates by straightforwardly averaging them [20]. Nevertheless, recent research [21] has revealed that this averaging-based aggregation method is susceptible to poisoning attacks, where a single malicious BS can manipulate the final aggregated outcome without constraints. To counteract such potential threats, various Byzantine-robust aggregation rules have been suggested [19], [21]–[27]. For instance, in the Krum method [21], each client's update is scored based on the sum of Euclidean distances to other clients' updates. The global update is then updated by selecting the update from the client (i.e., BS) with the minimum score. In a Median aggregation scheme [19], the server calculates the median value for each dimension using all the local model updates. In the FLTrust [22], it is assumed that the server possesses a validation dataset. The server maintains a model derived from this dataset. To determine trust levels, the server computes the cosine similarity between its model update and the update of each BS. These scores are then used to weigh the contribution of each BS to the final aggregated model.

C. Poisoning Attacks to FL-based Systems

The decentralized nature of FL makes our considered problem susceptible to Byzantine attacks [12], [15], [16], [18], [28], [29], where attackers with control over malicious BSs can compromise the FL-based WTP system. Malicious BSs can corrupt their local training traffic data or alter their local models directly. For instance, in the Trim attack [15], the attacker intentionally manipulates the local models on malicious BSs to cause a significant deviation between the aggregated model after attack and the one before attack. In the Model Poisoning Attack based on Fake clients (MPAF) attack [18], each malicious BS first multiplies the global model update synchronized from the central server by a negative scaling factor and subsequently transmits these scaled model updates to the server. In the Random attack [15], every malicious BS randomly generates a vector from a Gaussian distribution and transmits it to the server. Recently, [12] introduced poisoning attacks for FL-based WTP systems, where the attacker controls some deployed BSs, each with its own local training data. These malicious BSs fine-tune their local models using their respective training data. Subsequently, the attacker scales the local model updates on malicious BSs by applying a scaling factor and sends the scaled model updates to the server.

However, existing attacks suffer from the practical implementation limitations. For instance, the attack described in [12] is not feasible because it is based on the unrealistic assumption that an attacker can readily take control of authentic BSs. In reality, it is highly challenging for an attacker to gain such influence over existing, authentic BSs. In the MPAF attack, which has a simpler threat model, the model updates from fake clients are exaggerated by a factor such as 10^6. This approach is impractical because the central server can easily identify these excessive updates as anomalies and discard them. By contrast, our proposed poisoning attack involves carefully crafting model updates on fake BSs by addressing a parametric optimization problem. This ensures that the server is unable to differentiate these fake updates from benign ones, allowing the attacker to simultaneously breach the integrity of the system without detection.

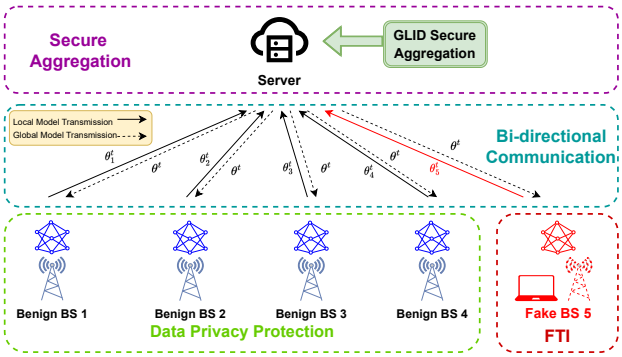


Fig. 1: Framework of Security Protection in FL-based WTP.

III. Threat Model to FL-based WTP Systems

In this section, we present a novel model poisoning attack, employing fake BSs for traffic injection into FL-based WTP systems under a minimum-knowledge scenario.

A. Attacker's Goal

The attacker's primary goal in compromising the integrity of the FL-based WTP system is to degrade the final global model's performance. This degradation directly impacts the accuracy of real-time traffic predictions, which is a critical aspect of network management and resource allocation. In practical cellular systems, inaccurate traffic predictions can lead to network congestion, poor quality of service, and inefficient use of resources, thereby causing substantial operational challenges for network providers. This disruption not only affects service providers but also has a cascading effect on end-users who rely on consistent and reliable network services.

B. Attacker's Capability

The attacker achieves this objective by introducing fake BSs into the targeted FL-based WTP system, as shown in Fig. 1. These fake BSs, which could be simple network devices, mimic the traffic processing behaviors of benign BSs with minimal effort and expense. Unlike the methods proposed in [12] which involve compromising genuine BSs, the use of fake BSs is far more feasible in real-world contexts. Creating fake BSs with open-source projects or emulators [18], [30]–[32] is a low-cost approach that can be executed without the

need for sophisticated hacking skills or deep access to the network infrastructure. This approach is particularly viable given the heightened security measures in modern networks, which make compromising genuine BSs increasingly challenging.

C. Attacker's Knowledge

The attacker's minimal knowledge about the targeted FL-based WTP system significantly increases the difficulty of executing the attack. In many real-world systems, gaining detailed insights into the central server's aggregation rules or acquiring information about benign BSs is highly challenging due to stringent security protocols and encryption. Therefore, an attack strategy that requires limited knowledge is not only more realistic but also more likely to get undetected. The fake BSs' operation, which is limited to receiving the global information and sending malicious updates, can be executed with basic technical skills, further lowering the barrier to entry for potential attackers. This aspect opens the door to a broader range of network adversaries, including those with limited technical expertise or computing resources.

D. Fake Traffic Injection Attack

The proposed Algorithm 1, referred to as the Fake Traffic Injection (FTI), outlines a Byzantine model poisoning attack strategy designed to manipulate the prediction accuracy of an FL-based WTP system under the aforementioned assumptions.

Central to the FTI attack is an iterative process where each iteration involves a thorough examination of current global model $\boldsymbol{\theta}^t$ and base model $\hat{\boldsymbol{\theta}}$. For each fake BS i, a malicious local model $\boldsymbol{\theta}_i^t$ is constructed by combining the global model $\boldsymbol{\theta}^t$ and a base model $\hat{\boldsymbol{\theta}}$ in a weighted manner (Line 5). Following the creation of $\boldsymbol{\theta}_i^t$, it evaluates its divergence from the global model using the Euclidean norm (Line 7). The algorithm then checks for an increase in this distance relative to the prior measurement (Line 8). If the distance has increased, indicating that the malicious local model $\boldsymbol{\theta}_i^t$ from some BS is diverging further from the global model $\boldsymbol{\theta}^t$ in the central server, the value of η is adjusted upwards. Conversely, if no increase in distance is observed, η is adjusted downwards. The adjustment of η is done in half-steps of its initial value (Lines 8-12). In other words, the value of η indicates the severity of poisoning attacks, measuring their impact or intensity.

To this end, the algorithm involves guiding the global model to align more closely with a predefined base model in each round. Specifically, during the t-th round, fake BSs calculate the direction of local model updates, determined by the difference between current global model and base model, denoted as $\boldsymbol{H} = \hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^t$. Moving towards this direction indicates that the global model is becoming more similar to the base model. A simple approach to acquire the local model of fake BS involves multiplying $\boldsymbol{H}$ by a scaling factor η. However, this direct method produces sub-optimal attack performance. Suppose n is the number of benign BSs, and the attacker wants to inject m fake BSs into the network system. We propose a method for calculating $\boldsymbol{\theta}_i^t$ for each fake BS $i \in [n+1, n+m]$:

$$\boldsymbol{\theta}_i^t = \eta\hat{\boldsymbol{\theta}} + (1-\eta)\boldsymbol{\theta}^t. \tag{3}$$

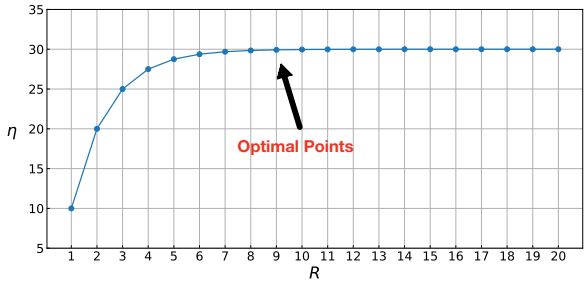


Fig. 2: Optimal value of η over communication round of R in Algorithm 1.

Algorithm 1 Fake Traffic Injection (FTI)

Require: Current global model $\boldsymbol{\theta}^t$, base model $\hat{\boldsymbol{\theta}}$, n benign BSs, m fake BSs, η
Ensure: Fake models $\boldsymbol{\theta}_i^t, i \in [n+1, n+m]$

1: step $\leftarrow \eta$
2: PreDist $\leftarrow -1$
3: **for** $r = 1, 2, \ldots, R$ **do**
4: **for** each fake BS i **do**
5: $\boldsymbol{\theta}_i^t \leftarrow \eta\hat{\boldsymbol{\theta}} - (\eta-1)\boldsymbol{\theta}^t$
6: **end for**
7: Dist $\leftarrow \|\boldsymbol{\theta}_i^t - \boldsymbol{\theta}^t\|_2$
8: **if** PreDist $<$ Dist **then**
9: $\eta \leftarrow \eta + \frac{\text{step}}{2}$
10: **else**
11: $\eta \leftarrow \eta - \frac{\text{step}}{2}$
12: **end if**
13: step $\leftarrow \frac{\text{step}}{2}$
14: PreDist $\leftarrow$ Dist
15: **end for**
16: **return** $\boldsymbol{\theta}_i^t, i \in [n+1, n+m]$

In such cases, an attacker tends to choose a higher value for η to ensure the sustained effectiveness of the attack, as shown in Fig. 2 with an initial η of 10. This holds true even after the server consolidates the manipulated local updates from fake BSs with legitimate updates from benign BSs.

IV. Global-Local Inconsistency Detection

The defense against model poisoning attacks on the FL-based WTP system relies on an aggregation protocol designed to identify malicious BSs. This protocol, named the Global-local Inconsistency Detection (GLID) method, is detailed in Algorithm 2. In each global round t, GLID primarily scrutinizes the anomalies present in each dimension of the model parameters $\boldsymbol{\theta}_i^t$, aiding in the identification of any potentially malicious entities, where $i \in [1, n+m]$, and $n+m$ is the total number of BSs in the system. Such a robust and versatile nature allows the network to adapt to various operational contexts without requiring intricate similarity assessments as in other existing works, like FLTrust [22].

Specifically, GLID approach enhances the detection of potential malicious activities within the network by employing *percentile-based trimming* on each dimension of the model

parameters. To establish an effective percentile pair for identifying abnormalities, four statistical methods can be adopted: Standard Deviation (SD), Interquartile Range (IQR), Z-scores, and One-class Support Vector Machine (One-class SVM). Suppose the total count of dimensions of model parameter is D, then for the **default SD method**, the percentile pair for each dimension d can be calculated as follow:

$$\text{percentile pair}_d^t = \left(g\left(\bar{\boldsymbol{\theta}}_d^t - k \cdot \sigma_d^t\right), g\left(\bar{\boldsymbol{\theta}}_d^t + k \cdot \sigma_d^t\right)\right), \quad (4)$$

where $\bar{\boldsymbol{\theta}}_d^t$ is the mean of the d-th dimension across all models in the t-th global training round, σ_d^t is the standard deviation of the d-th dimension, and k is a predefined constant dictating the sensitivity of outlier detection. $g(\cdot)$ is the interpolation function based on standard deviation bound to estimate percentile pairs, shown as follows:

$$g(x) = \left(\frac{P(x) - 0.5}{n+m}\right) \times 100, \quad (5)$$

where $P(x)$ is the position of x in the sorted dataset. We use $k = 3$ for general purposes. Given that different tasks may require varied percentile bounds, a precise estimation method is crucial for generalizing our defense strategy. The detailed percentile estimation methods can be found later in this section. In the FL-based WTP system, model parameters in the d-th dimension exceeding these percentile limits are flagged as malicious, and their weights α_i^t are assigned as 0. The other benign values in this dimension are aggregated using a weighted average rule, where the weights $\alpha_{d,i}^t$ are inversely proportional to the absolute deviation of each value $\boldsymbol{\theta}_{d,i}^t$ from the mean $\bar{\boldsymbol{\theta}}_d^t$, and normalized by the standard deviation σ_d^t. It can be represented as follows:

$$\alpha_{d,i}^t = \frac{\sigma_d^t}{\left|\boldsymbol{\theta}_{d,i}^t - \bar{\boldsymbol{\theta}}_d^t\right|}. \quad (6)$$

These weights of the d-th dimension are then normalized and applied to aggregate each BS's local model $\boldsymbol{\theta}_i^t$ into a global model $\boldsymbol{\theta}^{t+1}$, which can be represented as follow in the view of each dimension:

$$\boldsymbol{\theta}_d^{t+1} = \frac{\sum_{i=1}^{n+m} \alpha_{d,i}^t \cdot \boldsymbol{\theta}_{d,i}^t}{\sum_{i=1}^{n+m} \alpha_{d,i}^t}. \quad (7)$$

Subsequently, the server broadcasts this aggregated global model parameter $\boldsymbol{\theta}^{t+1}$ back to all BSs for synchronization.

There are three additional percentile estimation strategies listed below. Based on the upper and lower bound computed below, we can get a final percentile estimation decision to detect abnormal values in each dimension.

- **Interquartile Range (IQR)**: The IQR method calculates the range between the first and third quartiles (25^{th} and 75^{th} percentiles) of the data, identifying outliers based on this range. For each dimension d, the outlier bounds are:

$$\text{lower bound}_{d,\text{IQR}}^t = Q1_d^t - k_{\text{IQR}} \cdot \text{IQR}_d^t, \quad (8)$$

$$\text{upper bound}_{d,\text{IQR}}^t = Q3_d^t + k_{\text{IQR}} \cdot \text{IQR}_d^t, \quad (9)$$

Algorithm 2 Global-local Inconsistency Detection (GLID)

Require: Local models $\boldsymbol{\theta}_1^t, \boldsymbol{\theta}_2^t, \ldots, \boldsymbol{\theta}_{n+m}^t$, current global model $\boldsymbol{\theta}^t$, k
Ensure: Aggregated global model $\boldsymbol{\theta}^{t+1}$

```
for d = 1, 2, ..., D do
    θ̄_d^t ← (1/(n+m)) Σ_{i=1}^{n+m} θ_{d,i}^t
    σ_d^t ← sqrt((1/(n+m)) Σ_{i=1}^{n+m} (θ_{d,i}^t − θ̄_d^t)^2)
    percentile_d^t ← (g(θ̄_d^t − k·σ_d^t), g(θ̄_d^t + k·σ_d^t))
    Identify malicious BSs based on percentile pairs
    for each BS i do
        if θ_{d,i}^t is benign then
            α_{d,i}^t ← σ_d^t / |θ_{d,i}^t − θ̄_d^t|
        else
            α_{d,i}^t ← 0
        end if
    end for
    θ_d^{t+1} ← (Σ_{i=1}^{n+m} α_{d,i}^t · θ_{d,i}^t) / (Σ_{i=1}^{n+m} α_{d,i}^t)
end for
θ^{t+1} ← [θ_1^{t+1}, θ_2^{t+1}, ..., θ_D^{t+1}]
return θ^{t+1}
```

where $Q1_d^t$ and $Q3_d^t$ are the first and third quartiles, and k_{IQR} adjusts sensitivity.

- **Z-scores**: The Z-score method measures how many standard deviations a point is from the mean. For each dimension d, the normal range bounds are:

$$\text{lower bound}_{d,\text{Z-score}}^t = g\left(\bar{\boldsymbol{\beta}}_d^t - k_{\text{Z}} \cdot \sigma_d^t\right), \quad (10)$$

$$\text{upper bound}_{d,\text{Z-score}}^t = g\left(\bar{\boldsymbol{\beta}}_d^t + k_{\text{Z}} \cdot \sigma_d^t\right), \quad (11)$$

where k_{Z} is the number of standard deviations for the normal range.

- **One-Class SVM**: One-Class SVM constructs a decision boundary for anomaly detection. The decision function for each dimension d is:

$$f_d^t(\boldsymbol{\beta}) = \text{sign}\left(\sum_{i=1}^{n_{\text{SV}}} \gamma_i \cdot K(\boldsymbol{\beta}_{\text{SV}_i,d}^t, \boldsymbol{\beta}) - \rho\right), \quad (12)$$

where $\boldsymbol{\beta}_{\text{SV}_i,d}^t$ are the support vectors, γ_i are the Lagrange multipliers, $K(\cdot,\cdot)$ is the kernel function, and ρ is the offset. A point $\boldsymbol{\beta}$ is an outlier if $f_d^t(\boldsymbol{\beta}) < 0$.

In essence, this defense mechanism is a strategic amalgamation of direct statistical trimming and aggregation, targeting the preservation of the global model's integrity against poisoning attacks. By accurately isolating and excluding malicious BSs prior to model aggregation process, it significantly diminishes the likelihood of adversarial disruption in the FL framework. Additionally, its capacity to accommodate various dimensions and adapt to different inconsistency metrics and aggregation protocols considerably extends its applicability across a broad spectrum of distributed wireless network scenarios.

V. Evaluations

In this section, we demonstrate the effectiveness of our FTI poisoning attack and the GLID defense mechanism. Extensive

evaluation results are provided regarding the performance metrics in multiple dimensions.

A. Experiment Setup

1) Datasets: We utilize the real-world datasets obtained from Telecom Italia [33] to evaluate our proposed methods. The wireless traffic data in Milan is segmented into 10,000 grid cells, with each cell served by a BS covering an area of approximately 235 meters on each side. Milan Dataset contains three subset datasets, "Milan-Internet", "Milan-SMS" and "Milan-Calls". These datasets capture different types of wireless usage patterns, and we are mainly focusing on "Milan-Internet". Such comprehensive data collection enables an in-depth analysis of urban telecommunication behavior.

2) Baseline Schemes: We evaluate various state-of-the-art model poisoning attacks as comparison points to our proposed FTI attack. Furthermore, we employ these baseline poisoning attacks to highlight the effectiveness of our defense strategy GLID.

- **Trim attack [15]:** It processes each key within a model dictionary, computing and utilizing the extremes in a designated dimension to determine a *directed* dimension, where model parameters are selectively zeroed or retained to influence the model behavior.
- **History attack [18]:** It iterates over model parameters, replacing current values with historically scaled ones, effectively warping the model parameters using past data to misguide the aggregation process.
- **Random attack [18]:** It disrupts the model by replacing parameters with random and normally distributed values, scaled to maintain a semblance of legitimacy, thereby injecting controlled chaos into the aggregation process.
- **MPAF [18]:** It calculates a directional vector derived from the difference between initial and current parameters. This vector is then used to adjust model values, intentionally diverging from the model's original trajectory to introduce an adversarial bias. Following these calculations, the fake BSs are injected into the system.
- **Zheng attack [12]:** It inverts the direction of model updates by incorporating the negative of previous global updates. This inversion is refined through error maximization, generating a poison that proves challenging to detect due to its alignment with the model's error landscape.

Besides, we consider several baseline defensive mechanisms to demonstrate the effectiveness of our attack and defense.

- **Mean [20]:** It calculates the arithmetic mean of updates in each dimension, assuming equal trustworthiness among all BSs. However, this method is susceptible to the influence of extreme values.
- **Median [20]:** It identifies the median value in each dimension for each parameter across updates, which inherently discards extreme contributions to enhance the robustness against outliers.
- **Trim [20]:** It discards a specified percentage of the highest and lowest updates before computing the mean in each dimension, thereby reducing the potential sway of anomalous or malicious updates on the aggregate model.
- **Krum [21]:** Each BS's update is scored based on the sum of Euclidean distances to other BSs' updates. The global update is then updated by selecting the update from the BS with the minimum score.
- **FoolsGold [23]:** It calculates a cosine similarity matrix among all BSs and adjusts the weights for each BS based on these similarities. The weighted gradients are then aggregated to form a global model.
- **FABA [25]:** It computes the Euclidean distance for each BS's model from the mean of all received models. By identifying and excluding a specific percentage of the most distant models, this process effectively filters out potential outliers or malicious updates.
- **FLTrust [22]:** Cosine similarity is calculated between the server's current model and each BS's model to generate trust scores. These scores are then used to weigh the BS's contribution to the final aggregated model.
- **FLAIR [24]:** Each BS calculates "flip-scores" derived from the changes in gradient directions and "suspicion-scores" based on historical behavior. These scores are used to adjust the weights assigned to each BS's contributions to the global model.

3) Experimental Settings and Performance Metrics: In our experimental setup, we randomly selected 100 BSs to evaluate the impact of poisoning attacks and the effectiveness of defense mechanisms. By default, we report the results on Milan-Internet dataset. Model training is configured with a learning rate of 0.001 and a batch size of 64. We inject a 20% percentage of fake BSs to mimic benign ones in the system for FTI attack and simulate a scenario where 20% of the BSs are compromised for other baseline attacks. Our proposed FTI attack utilizes a parameter $\eta = 10$, and other attacks utilize a scaling factor of 1000. For the Trim aggregation rule, we discard 20% of the model parameters from all BSs. In our proposed GLID defense, we employ the standard deviation (SD) method as the default percentile estimation method. Throughout the measurement campaign, we adopt Mean Absolute Error (MAE) and Mean Squared Error (MSE) as the primary metrics for performance evaluation. MSE quantifies the average of the squared discrepancies between estimated and actual values, while MAE calculates the average absolute differences across predictions, disregarding their directional errors. The larger the MAE and MSE, the better the effectiveness of the attack.

B. Numerical Results

1) Performance of Proposed Methods: The FTI Attack, in particular, exposes significant vulnerabilities in numerous aggregation methods. It is observed that under our FTI Attack, both Mean and Krum Rules are completely compromised, as reflected by their MAE and MSE values reaching over 100.0 (values exceeding 100 are capped at 100). This result denotes a total breakdown in their WTP functionality. The Median Rule further emphasizes the severity of FTI Attack, with both its MAE and MSE escalating from modest baseline figures to

TABLE I: Performance Metrics for Milan-Internet Dataset

Aggregation Rule	Metric	Attack						
		NO	Trim	History	Random	MPAF	Zheng	***FTI***
Mean	MAE	0.211	100.0	100.0	100.0	100.0	0.698	100.0
	MSE	0.086	100.0	100.0	100.0	100.0	0.294	100.0
Median	MAE	0.211	0.213	0.211	0.212	0.211	0.217	100.0
	MSE	0.086	0.086	0.087	0.086	0.086	0.095	100.0
Trim	MAE	0.211	0.212	0.212	0.211	0.212	0.239	100.0
	MSE	0.086	0.087	0.089	0.086	0.088	0.106	100.0
Krum	MAE	0.221	0.225	100.0	0.225	100.0	0.225	100.0
	MSE	0.091	0.093	100.0	0.094	100.0	0.094	100.0
FoolsGold	MAE	0.213	100.0	100.0	100.0	100.0	0.934	100.0
	MSE	0.095	100.0	100.0	100.0	100.0	0.607	100.0
FABA	MAE	0.219	100.0	100.0	100.0	100.0	0.623	100.0
	MSE	0.089	100.0	100.0	100.0	100.0	0.249	100.0
FLTrust	MAE	0.242	0.234	100.0	0.240	100.0	3.182	100.0
	MSE	0.094	0.092	100.0	0.094	100.0	1.208	100.0
FLAIR	MAE	0.216	0.228	100.0	100.0	100.0	0.250	100.0
	MSE	0.094	0.088	100.0	100.0	100.0	0.096	100.0
GLID	MAE	**0.211**	**0.211**	**0.212**	**0.211**	**0.211**	**0.212**	**72.383**
	MSE	**0.086**	**0.087**	**0.086**	**0.086**	**0.087**	**0.086**	**27.528**

100. This sharp contrast highlights FTI attack's reliable performance against other defenses, such as Trim Attack against Median rule, where the increase in MAE and MSE is relatively minor at 0.234 and 0.092, respectively. Additionally, the Trim Rule, typically considered robust, exhibits a drastic increase in MAE to over 100.0, a significant rise from its baseline without any attack (termed as NO in Table I) of 0.211. This surge underscores Trim Rule's vulnerability to the FTI Attack, marking a notable departure from its typical resilience. Similar results can also be found in other aggregation rules under FTI attack, such as FoolsGold, FABA, FLTrust, and FLAIR. The FTI attack has the best overall performance against the given defenses. The Zheng attack, however, presents a distinct pattern of disruption. When subjected to this attack, FLTrust, which typically exhibits lower error metrics, shows a significant compromise, evidenced by the dramatic increase in its MAE to 3.182 and MSE to 1.208. Such a tailored nature of Zheng attack appears to target specific vulnerabilities within FLTrust, which are not as apparent in other scenarios, such as Trim Attack, where the rise in MAE and MSE for FLTrust is relatively modest. Regarding the MPAF Attack, most aggregation rules in the table do not show a convincing defense, except for a few like Median, Trim, and GLID.

Next, if we turn our attention to the defender's standpoint, the proposed GLID aggregation method demonstrates consistent performance stability across various attacks. Both its MAE and MSE values remain close to their baseline levels. Even in the case of our FTI attack, GLID manages to keep errors below 100, which is 72.383 and 27.528 for MAE and MSE respectively. This stability is particularly noteworthy, especially when compared to other rules such as FLAIR, which exhibit a significant deviation from their non-attacked baselines under the same adversarial conditions. GLID's ability to sustain its performance in the face of diverse and severe attacks underscores its potential as a resilient aggregation methodology.

2) Evaluation on the Impact of η: The step size η in our proposed FTI attack (see Algorithm 1) serves as a dynamic scaling factor, and its initial value significantly influences the model's performance metrics. This impact is illustrated in Fig. 3, where the Median aggregation rule is employed as the baseline defense strategy. A notable observation is the correlation between increasing values of η and the corresponding rise in MAE and MSE. For example, at $\eta = 1$, the MAE and MSE are relatively low, recorded at 0.501 and 0.208, respectively. However, increasing η to higher values, such as 10 or 20, results in a dramatic surge that reaches the maximum error rate. This increase suggests a significant compromise in the model, surpassing the predefined threshold for effective detection of the attack. The rationale behind this analysis emphasizes the pivotal role of η in determining the *strength* of a poisoning attack. An increased initial η tends to degrade model performance, deviating significantly from its expected operational state. Simultaneously, a higher η also raises the risk of the attack's perturbations being detected and eliminated during the defense process.

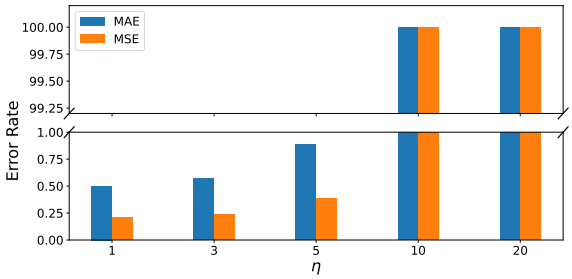


Fig. 3: Impact of Values of η.

3) Evaluation on Percentage of Fake BSs: The degree of compromise in BSs significantly influences the model's performance, as evidenced in Table II. By adopting Median aggregation as the defensive approach, the model first exhibits resilience at lower compromise levels, such as with only 5%–10% fake BSs in the scenario. However, a noticeable decline in performance is observed as the percentage of fake BSs increases to 20% or higher. This deterioration is evident as MAE and MSE values reach 100.0 in all categories, signaling a complete model failure. The underlying principle behind this trend suggests the model's limited tolerance to

malicious interference. More precisely, the network system can withstand below 20% compromise without significant performance degradation. However, beyond this threshold, the model's integrity is severely undermined, resulting in a complete system breakdown. This observation highlights the critical importance of implementing robust security measures to prevent excessive compromise of BSs, ensuring the model's reliability and effectiveness.

TABLE II: Impact of Percentages of Fake BSs

Pct.	Metric	Attack					
		Trim	Hist.	Rand.	MPAF	Zhe.	**FTI**
5%	MAE	0.221	0.215	0.219	0.215	0.213	**0.229**
	MSE	0.088	0.089	0.088	0.088	0.088	**0.089**
10%	MAE	0.220	0.213	0.218	0.213	0.214	**0.258**
	MSE	0.087	0.090	0.088	0.090	0.096	**0.104**
20%	MAE	0.223	0.218	0.218	0.216	0.269	**100.0**
	MSE	0.087	0.096	0.087	0.092	0.136	**100.0**
30%	MAE	100.0	100.0	100.0	100.0	5.990	**100.0**
	MSE	100.0	100.0	6.141	100.0	1.154	**100.0**
40%	MAE	100.0	100.0	100.0	100.0	100.0	**100.0**
	MSE	100.0	100.0	100.0	100.0	100.0	**100.0**

4) Evaluations on Percentile Estimation Methods: The dynamic trimming of an adaptive number of model parameters through percentile estimation, which is adapted in GLID, is effective for an effective defense strategy against various model poisoning attacks. In the comparative analysis of various estimation methods, as shown in Table III, Standard Deviation (SD) estimation emerges as the best technique, exhibiting a marked consistency and robustness across a spectrum of estimation approaches. This is evidenced by the consistently low MAE and MSE values for SD across these approaches, at 0.219 and 0.087, respectively. In contrast, other methods have varying degrees of inconsistency and vulnerability. For instance, One-class SVM exhibits pronounced variability, with MAE and MSE values reaching the maximal error level of over 100.0 under Trim, History, and MPAF attacks. Such a disparity in performance, particularly the stably lower error rates of SD compared to the significant fluctuations in other estimation methods, positions SD as a reliable and effective percentile estimation technique in GLID.

5) Evaluations on the Impact of BS Density: Given the percentage of fake BSs at 20%, Figs. 4(a)-(d) compare Median and GLID rules with varying densities of BS in the network scenario. It is interesting to see that the total number of BSs does not significantly impact the performance of any attack and defense mechanisms, especially for our FTI and GLID. Under Median aggregation, FTI consistently shows maximal error (MAE and MSE at over 100.0) across different BS densities, indicating a failure of the defense. This consistent pattern of stable performance across varying participants in the FL-based WTP system suggests that the total number of BS does not substantially influence the effectiveness of the attack and defense strategies.

TABLE III: Impact of Percentile Estimation Methods

Method	Metric	Attack						
		NO	Trim	Hist.	Rand.	MPAF	Zhe.	**FTI**
SD	MAE	0.219	0.219	0.219	0.218	0.219	0.219	**72.38**
	MSE	0.087	0.087	0.087	0.087	0.087	0.087	**27.52**
IQR	MAE	0.219	0.220	0.220	0.219	0.210	0.218	**100.0**
	MSE	0.087	0.087	0.087	0.087	0.087	0.088	**100.0**
Z-scores	MAE	0.219	0.219	0.219	0.219	0.220	1.047	**100.0**
	MSE	0.087	0.087	0.088	0.087	0.087	0.401	**100.0**
SVM	MAE	0.219	100.0	100.0	0.220	100.0	0.713	**100.0**
	MSE	0.087	100.0	100.0	0.087	100.0	0.275	**100.0**

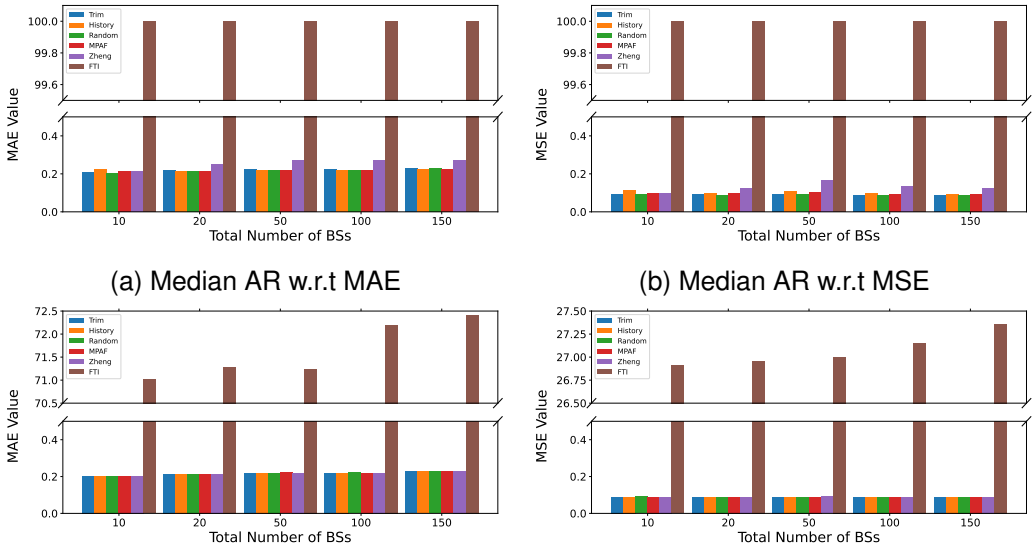
TABLE IV: Impact of Different Percentile Pairs

Pair	Metric	Method					
		Trim	Hist.	Rand.	MPAF	Zhe.	**FTI**
[10, 70]	MAE	100.0	100.0	100.0	100.0	0.710	**100.0**
	MSE	100.0	100.0	100.0	100.0	0.279	**100.0**
[20, 70]	MAE	0.215	0.214	0.218	0.217	0.216	**100.0**
	MSE	0.083	0.085	0.084	0.082	0.086	**100.0**
[30, 70]	MAE	0.218	0.219	0.220	0.215	0.217	**72.382**
	MSE	0.090	0.088	0.089	0.086	0.088	**27.246**
[10, 80]	MAE	100.0	100.0	100.0	100.0	0.711	**100.0**
	MSE	100.0	100.0	100.0	100.0	0.275	**100.0**
[20, 80]	MAE	0.217	0.215	0.218	0.214	0.216	**72.168**
	MSE	0.085	0.083	0.084	0.082	0.086	**27.147**
[30, 80]	MAE	0.220	0.218	0.219	0.216	0.217	**71.298**
	MSE	0.088	0.089	0.086	0.088	0.090	**27.022**
[10, 90]	MAE	100.0	100.0	100.0	100.0	0.712	**100.0**
	MSE	100.0	100.0	100.0	100.0	0.274	**100.0**
[20, 90]	MAE	0.215	0.217	0.218	0.216	0.214	**100.0**
	MSE	0.088	0.086	0.085	0.089	0.086	**100.0**
[30, 90]	MAE	0.217	0.218	0.219	0.216	0.215	**100.0**
	MSE	0.086	0.088	0.089	0.085	0.088	**100.0**

6) Evaluations on the Percentile Range of GLID: Table IV presents an evaluation of performance across a variety of percentile pairs used in the proposed GLID method on different attack methods. The configuration of the percentile pair guides the GLID method in identifying and eliminating outliers. For example, specifying a percentile pair of [10, 70] means that values below the 10^{th} percentile and above the 70^{th} percentile are trimmed away, focusing the analysis on the data within these bounds. It is observed that, when the percentile pair is set at [10, 70], most methods, except for Zheng attack, register a metric over 100.0, suggesting the models are fully attacked. Similarly, the percentile pair of [10, 90] yields a value over 100 for all methods except Zheng attack. The Zheng attack consistently records low metrics across all settings, such as 0.710, and 0.279 for the pair [10, 70], raising questions about its attack efficacy. On the other hand, FTI shows varied performance; it achieves over 100.0 for most percentile pairs like [10, 70] and [20, 90] but drops to 72.382 and 27.246 for the pair [30, 70]. These results underscore the importance of fine-tuning the percentile pair parameters in the GLID method. Proper parameter selection can effectively trim outliers without significantly impacting overall network performance.

VI. Conclusion

In this study, we introduced a novel approach to perform model poisoning attacks on WTP through fake traffic injection. Operating under the assumption that real-world BSs are challenging to attack, we inject fake BS traffic distribution with minimum knowledge that disseminates malicious model parameters. Furthermore, we presented an innovative global-local inconsistency detection mechanism, designed to safeguard FL-based WTP systems. It employs an adaptive trimming strategy, relying on percentile estimations that preserve accurate model parameters while effectively removing outliers. Extensive evaluations demonstrate the effectiveness of our attack and defense, outperforming existing baselines.



(a) Median AR w.r.t MAE (b) Median AR w.r.t MSE

(c) GLID AR w.r.t MAE (d) GLID AR w.r.t MSE

Fig. 4: The impact of BS density on the performance of Median and GLID methods with respect to MAE and MSEs.

ACKNOWLEDGMENT

This research was supported by the National Science Foundation through Award CNS–2312138.

REFERENCES

[1] L. U. Khan, W. Saad, Z. Han, E. Hossain, and C. S. Hong, "Federated learning for internet of things: Recent advances, taxonomy, and open challenges," in *IEEE Communications Surveys & Tutorials*, 2021.

[2] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, "Federated learning for internet of things: A comprehensive survey," in *IEEE Communications Surveys & Tutorials*, 2021.

[3] H. G. Abreha, M. Hayajneh, and M. A. Serhani, "Federated learning in edge computing: a systematic survey," in *Sensors*, 2022.

[4] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, "Federated learning for healthcare informatics," in *Journal of Healthcare Informatics Research*, 2021.

[5] D. C. Nguyen, Q.-V. Pham, P. N. Pathirana, M. Ding, A. Seneviratne, Z. Lin, O. Dobre, and W.-J. Hwang, "Federated learning for smart healthcare: A survey," in *ACM Computing Surveys*, 2022.

[6] J. Wen, M. Sheng, J. Li, and K. Huang, "Assisting intelligent wireless networks with traffic prediction: Exploring and exploiting predictive causality in wireless traffic," in *IEEE Communications Magazine*, 2020.

[7] L. Nie, D. Jiang, S. Yu, and H. Song, "Network traffic prediction based on deep belief network in wireless mesh backbone networks," in *IEEE Wireless Communications and Networking Conference*, 2017.

[8] S. P. Sone, J. J. Lehtomäki, and Z. Khan, "Wireless traffic usage forecasting using real enterprise network data: Analysis and methods," in *IEEE Open Journal of the Communications Society*, 2020.

[9] C. Qiu, Y. Zhang, Z. Feng, P. Zhang, and S. Cui, "Spatio-temporal wireless traffic prediction with recurrent neural network," in *IEEE Wireless Communications Letters*, 2018.

[10] Y. Xu, W. Xu, F. Yin, J. Lin, and S. Cui, "High-accuracy wireless traffic prediction: A gp-based machine learning approach," in *IEEE Global Communications Conference*, 2017.

[11] C. Zhang, S. Dang, B. Shihada, and M.-S. Alouini, "Dual attention-based federated learning for wireless traffic prediction," in *INFOCOM*, 2021.

[12] T. Zheng and B. Li, "Poisoning attacks on deep learning based wireless traffic prediction," in *INFOCOM*, 2022.

[13] M. Joshi and T. H. Hadi, "A review of network traffic analysis and prediction techniques," *arXiv preprint arXiv:1507.05722*, 2015.

[14] J. Fan, D. Mu, and Y. Liu, "Research on network traffic prediction model based on neural network," in *International Conference on Information Systems and Computer Aided Education*, 2019.

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

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

[17] C. Xie, O. Koyejo, and I. Gupta, "Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation," in *UAI*, 2020.

[18] X. Cao and N. Z. Gong, "MPAF: Model poisoning attacks to federated learning based on fake clients," in *CVPR Workshops*, 2022.

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

[20] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *AISTATS*, 2017.

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

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

[23] C. Fung, C. J. Yoon, and I. Beschastnikh, "Mitigating sybils in federated learning poisoning," *arXiv preprint arXiv:1808.04866*, 2018.

[24] A. Sharma, W. Chen, J. Zhao, Q. Qiu, S. Bagchi, and S. Chaterji, "Flair: Defense against model poisoning attack in federated learning," in *ASIACCS*, 2023.

[25] Q. Xia, Z. Tao, and Q. Li, "Defending against byzantine attacks in quantum federated learning," in *International Conference on Mobility, Sensing and Networking*, 2021.

[26] M. Fang, J. Liu, N. Z. Gong, and E. S. Bentley, "Aflguard: Byzantine-robust asynchronous federated learning," in *ACSAC*, 2022.

[27] Y. Xu, M. Yin, M. Fang, and N. Z. Gong, "Robust federated learning mitigates client-side training data distribution inference attacks," in *The Web Conference*, 2024.

[28] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *ESORICS*, 2020.

[29] M. Yin, Y. Xu, M. Fang, and N. Z. Gong, "Poisoning federated recommender systems with fake users," in *The Web Conference*, 2024.

[30] "Android-x86 run android on your pc," https://www.android-x86.org/.

[31] "Noxplayer, the perfect android emulator to play mobile games on pc," https://www.bignox.com/.

[32] "The world's first cloud-based android gaming platform," https://www.bluestacks.com/.

[33] Barlacchi, Gianni, M. D. Nadai, R. Larcher, A. Casella, C. Chitic, G. Torrisi, F. Antonelli, A. Vespignani, A. Pentland, and B. Lepri, "A multi-source dataset of urban life in the city of milan and the province of trentino," in *Scientific data*, 2015.