PoiSAFL: Scalable Poisoning Attack Framework to Byzantine-resilient Semi-asynchronous Federated Learning

Xiaoyi Pang[†,♯], *Chenxu Zhao*[†,≀], *Zhibo Wang*[†,♯,*], *Jiahui Hu*[†,≀], *Yinggui Wang*[‡], *Lei Wang*[‡], *Tao Wei*[‡], *Kui Ren*[†,♯], *Chun Chen*[†,♯]

[†]*The State Key Laboratory of Blockchain and Data Security, Zhejiang University, P. R. China*

[♯]*Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security, P. R. China*

[≀]*School of Cyber Science and Technology, Zhejiang University, P. R. China* [‡]*Ant Group, P. R. China*

xypang@whu.edu.cn, {zhaocx_7, zhibowang, jiahuihu}@zju.edu.cn, wyinggui@gmail.com

{shensi.wl, lenx.wei}@antgroup.com, {kuiren, chenc}@zju.edu.cn

Abstract

Semi-asynchronous federated learning (SAFL) enhances the efficiency of privacy-preserving collaborative learning across clients with diverse processing capabilities. It updates the global model by aggregating local models from only partial fast clients without waiting for all clients to synchronize. We realize that such semi-asynchronous aggregation may expose the system to serious poisoning risks, even when defenses are in place, since it introduces considerable inconsistency among local models, giving chances for attackers to inject inconspicuous malicious ones. However, such risks remain largely underexplored. To plug this gap and fully explore the vulnerability of SAFL, in this paper, we propose a scalable stealth poisoning attack framework for Byzantine-resilient SAFL, called PoiSAFL. It can effectively impair SAFL's learning performance while bypassing three typical kinds of Byzantine-resilient defenses by strategically controlling malicious clients to upload undetectable malicious local models. The challenge lies in crafting malicious models that evade detection yet remain destructive. We construct a constrained optimization problem and propose three modules to approximate the optimization objective: the anti-training-based model initialization, loss-aware model distillation, and distance-aware model scaling. These modules initialize and refine malicious models with desired poisoning ability while keeping their performance, prediction entropy, and dissimilarity within benign ranges to bypass detection. Extensive experiments demonstrate that PoiSAFL can defeat three typical categories of defenses. Besides, PoiSAFL can further amplify its attack impact by flexibly executing three proposed modules. Note that PoiSAFL is scalable and can incorporate new modules to defeat future new types of defenses.

*Zhibo Wang is the corresponding author.

1 Introduction

Federated learning (FL) is widely used in distributed systems since it enables multiple clients to train a shared global model by sending local model updates to the server while keeping their private data locally. In conventional synchronous FL (SFL) [15, 21, 23, 28, 34, 40], the server performs the global aggregation only after receiving local updates from all clients, causing inefficiencies due to clients' various speed. For that, asynchronous FL (AFL) [8, 9, 39, 41, 47, 50, 54, 56–58] is introduced to enable the server to perform the aggregation once receiving a local update from any client and broadcast the updated global model to that client. However, it results in a high inconsistency among local models, greatly slowing down convergence. Recently, semi-asynchronous FL (SAFL) [20, 22, 27, 30, 33, 42] has emerged to strike a balance between the efficiency of AFL and the convergence speed of SFL by performing the global aggregation in a semi-asynchronous manner. Specifically, in SAFL, the server buffers local updates uploaded by clients and aggregates them to update the global model when the buffer reaches a certain size or a fixed period has elapsed. Thus, SAFL offers advantages in terms of faster convergence, more flexible client participation, and improved computation resource usage.

Previous works highlight FL systems' susceptibility to Byzantine poisoning attacks crafted from malicious manipulation of clients' behaviors. They inject dirty data (known as data poisoning attacks) [13, 16, 18, 36, 37] or malicious local updates (known as model poisoning attacks) [1–3, 6, 10, 31, 48] into FL systems through adversary-controlled clients to impair the system learning performance. Recently, with full awareness of such serious security hazards, various Byzantine-resilient approaches have been proposed to defend SFL and AFL against poisoning risks. They effectively detect or filter out malicious local updates to weaken their negative effect on the global model. According to how they determine if the local update is malicious, they can be categorized as: 1) Similarity-based defenses [11, 26, 35, 38, 44, 46, 49]: considering that benign local updates have a certain degree of

consistency, these defenses screen potential malicious local updates based on statistical similarity. 2) Performance-based defenses [5, 10, 29, 45]: with the intuition that benign local updates will lead to higher global model performance, these defenses identify malicious local updates by detecting high-loss local models or measuring the loss/accuracy of the global model integrated with each local update. 3) Entropy-based defenses [29]: these defenses believe that benign local models result from local training rather than random generation, thus their predictions should have high confidence and low entropy. So they insulate malicious local models from global aggregation by filtering those with high prediction entropy.

As a FL system, SAFL is inherently susceptible to Byzantine poisoning attacks. Meanwhile, as a compromise of SFL and AFL, it can adopt the above-mentioned three kinds of defense methods designed for SFL and AFL to strengthen its defenses against malicious clients seeking to undermine the integrity of the learning process. Nevertheless, we realize that in some cases, SAFL systems may face greater poisoning risks than SFL and AFL systems due to their semi-asynchronous nature. For example, while SFL allows all clients to utilize the same global model as the initialization during local training, clients in SAFL train locally based on different versions of the global model (from different global rounds), which leads to high inconsistency among local models. Therefore, when defenses (such as statistical similarity-based defenses) are deployed, compared to SFL, the higher divergence of local updates in SAFL gives attackers more flexibility to bypass detections and launch successful poisoning attacks. Additionally, in AFL, only one client participates in the global aggregation process at each global round, while SAFL uses local updates from more clients for the global aggregation. Therefore, with the same number of global rounds, slow clients have more opportunities to participate in the global aggregation in SAFL than in AFL, indicating that they can have a more significant impact on the final global model. In this context, when slow clients are compromised as malicious clients, they can achieve stronger poisoning attack impacts in SAFL than AFL. However, the threat of poisoning attacks on SAFL systems with Byzantine-resilient defenses remains largely underexplored.

In this paper, we are motivated to take an important step towards exploring the significant poisoning risk in SAFL systems by demonstrating how model poisoning attacks can be successfully launched against SAFL, even when Byzantine-resilient approaches are adopted. To achieve our goal, we are facing two major challenges. First, in SAFL, the frequency at which each client participates in the global aggregation varies, leading to their different levels of influence on the global model. Therefore, it becomes challenging to ensure that malicious clients exert their influence and significantly destroy the SAFL system's learning performance to show the desired toxicity. Second, since the adversary is generally unable to ascertain the specific countermeasures deployed by the SAFL system, it must be capable of bypassing all three typical kinds of defenses to launch the attack successfully. Whereas, it is challenging to maintain a strong impact of the attack while circumventing comprehensive defenses.

To solve the challenges and substantiate such a poisoning threat, we propose a scalable poisoning attack framework for Byzantine-resilient SAFL systems as a pioneer research, called PoiSAFL. It can effectively diminish the global model performance while evading the three potential kinds of defenses deployed in the server. To this end, PoiSAFL crafts undetectable malicious local models for compromised malicious clients and strategically controls their participation frequency in the global aggregation to amplify the attack impact. Specifically, to produce satisfactory malicious local models, we formulate a constrained optimization problem and develop three modules to effectively approximate the optimization objective: 1) To maximize the malicious model's negative influence on the global model, the anti-training-based model initialization module initializes a seed malicious model with an inverted loss function, which minimizes the probability of correct classification. 2) To counteract entropy-based and performance-based defenses, instead of directly generating malicious local models like existing model poisoning attacks, the loss-aware model distillation module trains malicious local models under the supervision of the seed malicious model as well as ground truths. This can transfer the poisonousness of the seed malicious model to malicious local models while keeping their small prediction entropy and loss. 3) To avoid being identified by similarity-based defenses, the distance-aware model scaling module further optimizes each malicious local model to keep its distance from benign local models within tolerable limits. These three components ensure that the crafted malicious local models remain stealthy from various perspectives and possess high toxicity. Note that PoiSAFL is a scalable framework as it can incorporate new poisoning evasion modules in the future by extending its process flow and increasing constraints in the malicious local model optimization problem.

Our key contributions can be summarized as follows:

- We propose a stealth poisoning attack framework for SAFL, called PoiSAFL, which can effectively compromise the learning performance while evading three typical kinds of defenses. To the best of our knowledge, this work is the first to reveal that integrating existing Byzantine-resilient defenses into SAFL gives a false sense of security against poisoning attacks.

- To ensure attack effectiveness and undetectability, we propose the anti-training-based model initialization mechanism to make sure that crafted malicious local models have satisfactory poisonousness. Then, we develop the loss-aware model distillation mechanism and the distance-aware model scaling mechanism to ensure that malicious local models can bypass the detection of three typical kinds of Byzantine-resilient approaches.

- Extensive experiments validate that PoiSAFL can effectively degrade the global model accuracy in SAFL despite what kind of defense the server adopts and outperforms state-of-the-art poisoning attacks. Meanwhile, PoiSAFL can cause more remarkable performance degradation when the adopted defense is known by flexibly executing the proposed three modules.

2 Background and Related Work

In this section, we introduce background knowledge and related work.

2.1 SFL, AFL and SAFL

A typical FL system contains a server and plenty of clients, and the typical FL algorithm contains the following steps at each round: 1) Local training: each selected client trains its local model with its own dataset using the received global model as the initialization. 2) Uploading: each selected client uploads the local update (i.e. local model or gradient) to the server after completing the local training. 3) Global aggregation: the server strategically aggregates a part of or all the received local updates to update the global model. 4) Broadcasting: the server broadcasts the latest global model to those clients participating in the global aggregation. Both SFL, AFL, and SAFL follow the above typical system model and algorithm, but they differ in the global aggregation step.

Specifically, represented by FedAvg [23], at each round, SFL waits for all selected clients to upload their local updates before performing the global aggregation, leading to system inefficiencies. In contrast, AFL [8, 9, 39, 41, 47, 50, 54, 56–58] updates the global model immediately each time receiving a local update from any client, leading to a more responsive learning process. However, the fully asynchronous aggregation may include stale or outdated information with high inconsistency, potentially slowing down convergence and diminishing the global model accuracy. SAFL is introduced to strike a balance between the efficiency of AFL and the convergence speed of SFL. As a compromise of SFL and AFL, SAFL [7, 20, 22, 24, 25, 27, 30, 33, 42, 51–53] involves the server aggregating buffered local model updates from a part of fast-trained clients. To be more specific, at each round, the server buffers received local updates and aggregates them to update the global model once the buffer reaches a predetermined size or after a set time interval has passed. For instance, as shown in Figure. 1, the server aggregates the received local updates periodically at a regular interval.

Note that in AFL and SAFL, there may be a delay between when a client completes local training and when its local update are incorporated into the global model. During this time, the global model may have already been updated several times by other clients. As a result, the client's local update is outdated compared to the current global model. The metric

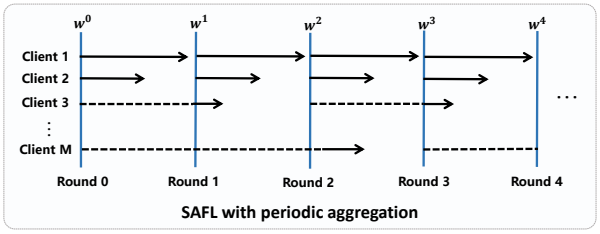


Figure 1: The schematic illustration of SAFL. The blue line means the server performs the global aggregation, and the black line with arrow means the client completes local training and uploads local update to the server.

staleness is often used to measure how stale the local update is compared to the current global model, which is typically computed as the difference between the update round of the global model adopted in the local training and the current round where the local update is incorporated into the global model. For example, in SAFL with periodical aggregation shown in Figure. 1, at the global round 3 where the server gets the global model w^3, the staleness of client 1 is $3-2=1$ and that of client M is $3-0=3$. Most algorithms [24, 25, 51, 54, 55] assign staleness-aware weights for clients during the global aggregation to mitigate the effect of stale information and improve learning performance.

2.2 Poisoning Attacks and Defenses to SFL

SFL, due to its distributed nature, is vulnerable to poisoning attacks. The adversary can control some malicious clients to undermine the robustness of the global model and make it generate erroneous predictions. Such attacks can be differentiated into data poisoning attacks [13, 16, 18, 36, 37] and model poisoning attacks [1–3, 6, 10, 31, 48]. The former pollutes the training data of compromised clients while the latter manipulates the local updates of compromised clients directly.

Many Byzantine-resilient defenses have been devised to secure SFL against poisoning attacks by discarding or down-weighting outlier local updates during global aggregation. Based on their outlier detection strategies, they can be broadly categorized into two primary groups: similarity-based approaches and performance-based methods. Similarity-based defenses primarily identify outliers among client updates by measuring their similarity using statistical metrics, such as Euclidean distance [4, 11, 12], cosine similarity [11, 26, 46], L2 norm [35] and other specially designed metrics [38, 44, 49]. The fundamental insight underlying performance-based defenses [5, 10, 29, 45] is that benign local models outperform malicious ones and will lead to better global model performance. Based on that, they typically assume access to a small and clean public dataset on the server that has the same distribution as clients' local data, and use this dataset to evaluate local models' loss or accuracy, allowing them to exclude underperforming local models.

2.3 Defenses for AFL

Some researchers have recognized that AFL is also highly susceptible to poisoning attacks. They put forth novel defenses for AFL predicated on the server's possession of a trusted dataset for the learning task. For example, following the principle of existing similarity-based and performance-based SFL defenses, they update the global model using the trusted dataset to produce a benign reference model and identify malicious clients by comparing local models to this reference through statistical similarity or performance discrepancies [11, 46]. Beyond that, Prak et al. [29] introduces a unique method based on predictive entropy. The underlying intuition is that benign models, genuinely trained on real data, will exhibit lower predictive entropy. The proposed entropy-based method evaluates the predictive entropy of local models using the trusted dataset and filters out local models with higher entropy during the global aggregation.

There is currently no work specifically designed for poisoning attacks or defenses against SAFL. However, considering that SAFL is a compromise of SFL and AFL, SAFL systems can adopt the above-mentioned three typical kinds of Byzantine-resilient defenses designed for SFL and AFL to bolster their defense capability.

2.4 Knowledge distillation

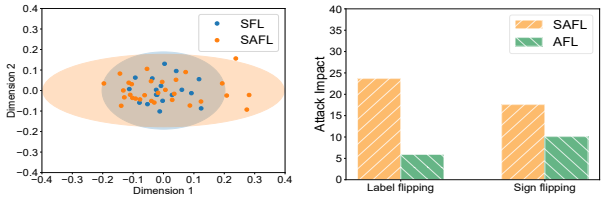
In this paper, we utilize the Knowledge distillation (KD) technology to transfer the poisonousness between malicious models. KD empowers knowledge transfer between two different models (called the teacher and the student), where the student model is enabled to quickly learn new complex concepts that the teacher model has learned. The pioneering KD work by Hinton et al. [14] suggested that the student model should acquire the knowledge of the teacher network by mimicking the output logits (i.e., the inputs to the final softmax) of the teacher model for the public dataset since logits carry the dark knowledge learned by the teacher model. Suppose the teacher model and the student model have output logits z_t and z_s respectively for the same input sample (x, y), then the trianing loss $\mathcal{L}$ in KD can be computed by:

$$\mathcal{L} = \alpha * \phi(\rho(z_s), y) + (1-\alpha) * \phi(\rho(z_s, \tau), \rho(z_t, \tau)), \quad (1)$$

where ϕ is the distance measure function (e.g., the cross-entropy, KL divergence), ρ is the softmax function that converts logits into a probability distribution over classes. $\tau > 1$ is a temperature factor that is applied as a scaling factor to the logits soft the target distribution. The first item is the prediction loss and the second item is the distillation loss.

3 Problem Definition

This section introduces our risk analysis and gives the threat model and problem we considered.



(a) Local models visualization (b) Attack impact comparison

Figure 2: Results of motivation experiments.

3.1 Risk Analysis and Motivation

In this paper, we have an insight that although SAFL can adopt defense methods designed for SFL and AFL to defeat poisoning attacks, it still faces serious poisoning risks due to its distributed and semi-asynchronous characteristics. At some point, the poisoning risks in SAFL can be even greater than in SFL and AFL. This is because SAFL has 1) more noisy local updates compared to SFL, providing more room to manipulate malicious local models to bypass detections, and 2) a more balanced client contribution compared to AFL, allowing stronger attack impacts by compromising any group of clients (even slow clients) as malicious ones.

Specifically, in SAFL, clients with different processing power asynchronously join the global aggregation and perform local training based on different versions of the global model. This leads to local models with varying staleness and high inconsistency, then the server would receive noisy local models with high divergence. Unlike that, in SFL, clients perform local training based on the same global model, showing much lower inconsistency among local updates. Therefore, when defenses are deployed, SAFL may face greater poisoning risks than SFL since it introduces additional flexibility for the adversary to inject hard-to-detect malicious local models during the global aggregation.

Moreover, the gap in the participation frequency in global aggregation between fast and slow clients in AFL is huge, making slower clients, potentially including malicious ones, unable to have a meaningful impact on the global model. SAFL reduces such a gap by aggregating buffered local model updates from multiple clients, providing slower clients with more opportunities to contribute to the global model. Therefore, with the same number of global rounds, when compromising slow clients as malicious ones, the adversary can cause a greater destructive effect in SAFL than AFL. That is, when slow clients act as malicious clients, SAFL may fact greater poisoning risks than AFL.

To validate our analysis, we implement SFL, AFL, and SAFL algorithms (i.e., FedAvg [23], staleness-aware AFL [54], staleness-aware SAFL with periodic aggregation [7]) with the FashionMNIST dataset. We set 60 clients and allocate the dataset to these clients in an IID manner. Both clients and the server use CNN2 as the model architecture. The algorithms are conducted for 150 rounds with Adam optimizer with a learning rate of 0.01 and batch size of 64, and the attack

starts from the 100-th round. To be more specific, to compare the inconsistency among local updates in SAFL and SFL, in Figure. 2(a), we visualize the local updates buffered by the server in a single round (where there is no attack) based on PCA dimensionality reduction. We can find that the divergence of local updates in SAFL is much higher than that in SFL, which is inconsistent with our analysis. To compare the attack impacts in SAFL and AFL when the adversary compromises slow clients to act maliciously, we launch two sample poisoning attacks (i.e., LabelFlip and SignFlip) when compromising 20% slowest clients as malicious clients. Figure. 2(b) gives the results, and the observation that attacks are more harmful in SAFL than in AFL supports our analysis.

Despite its increased vulnerability, the specific investigation of poisoning attacks within SAFL has been scarce. Therefore, we are motivated to fully reveal and explore poisoning risks in SAFL. Given that SAFL systems can adopt existing three kinds of Byzantine-resilient defense methods to defeat poisoning attacks, our goal is to design a new poisoning attack for SAFL that can bypass potential defenses while significantly impairing the system learning performance.

3.2 Threat Model

We characterize our threat model with respect to the capabilities, knowledge, and goal of the adversary.

The adversary's capabilities: Following the regular SAFL architecture, PoiSAFL assumes that the adversary has control of a part of the whole M clients and manipulates them to inject malicious local models to compromise the system integrity. Considering the characteristics of SAFL, we assume that these malicious clients include at least one of the fastest-trained clients for convenient information estimation. Suppose the number of controlled clients (malicious clients) is C, and that of benign clients is $M-C$. Then we have a set U_m of malicious clients and a set U_b of benign clients, where $|U_m| = C$ and $|U_b| = M-C$.

The adversary exerts complete control over malicious clients, including accessing their local data and controlling their local training, local update uploading frequency, and the uploaded content, as well as facilitating information sharing among malicious clients. Besides, it has very strong processing power to train and optimize malicious local models for malicious clients. That is, it can quickly craft malicious models for malicious clients, which takes less time than the fastest-trained clients' local training. Therefore, it enables malicious clients to upload malicious updates as quickly as the fastest clients. However, the adversary lacks the ability to compromise the central server or benign clients.

The adversary's knowledge: The adversary knows all information about malicious clients. It can observe the frequency of all clients participating in the global aggregation and thus has the knowledge of the highest staleness value during the SAFL process. However, it doesn't know the local data or local models of benign clients and is unaware of the server's adopted defense mechanisms against attacks.

The adversary's goal: The adversary focuses on launching untargeted poisoning attacks to SAFL to significantly compromise the performance of the global model while bypassing potential defenses. To this end, it exploits vulnerabilities and blind spots in current defenses and strategically manipulates the upload frequency and local models of compromised malicious clients to poison the global model. The main goal is to strategically craft and upload malicious local models that can be covertly injected into the global aggregation process to damage the global model, even with Byzantine-resilient defenses in place.

3.3 Problem Formulation

Suppose there are M heterogeneous clients, and each client $i \in U = \{1, 2, \cdots, M\}$ has its own training data D_i, and $D = \bigcup_{i=1}^{M} D_i$ is a collection of all training data. Then the SAFL problem can be formulated as:

$$\min_{w} \quad \{\mathcal{F}(w, D) \triangleq \frac{|D_i|}{|D|} \sum_{i=1}^{M} \mathcal{F}(w, D_i)\}, \tag{2}$$

where w denote the final learned global model, $\mathcal{F}(w, D_i)$ is the local loss function defined as $\mathcal{F}(w, D_i) = \frac{1}{|D_i|} \sum_{(x,y) \in D_i} \phi(h(w,x), y)$. Note that $h(w,x)$ is the prediction vector of the model w for the input x, and ϕ measures the distance.

Specifically, let U^t denote the set of clients that participate in the global aggregation at global round t, i.e., local updates of clients in U^t are buffered and aggregated in round t. Let $U_j^t \subset U^t$ $(j < t)$ denote the set of clients whose local training is performed using the global model w^j of the global round j as the initialization. Then we have $\bigcup_{j=0}^{t-1} U_j^t = U^t$ and $U_j^t \cap U_{j'}^t = \emptyset$ for $j \neq j'$. For the client $i \in U_j^t$, its staleness τ_i^t can be computed by $\tau_i^t = t - j$, which measures how stale the global model adopted in local training is. Then SAFL performs the following *semi-asynchronous global aggregation* step at each global round t:

$$w^t = \pi w^{t-1} + (1 - \pi) f_{AGR}(\{w_i^t\}_{i \in U^t}), \tag{3}$$

where w^{t-1} is the previous global model, $\{w_i^t\}_{i \in U^t}$ denotes the buffered local updates, π weights the effects of the two terms, and f_{AGR} is the aggregation rule (AGR) adopted by the server (it can be Byzantine-resilient or not). The mostly-used non-Byzantine-resilient AGR is to compute the staleness-aware weighted average of buffered local models, where the weight of the client $i \in U^t$ is inversely proportional to its staleness τ_i^t.

In PoiSAFL, the adversary controls a set of malicious clients $U_m \subset U^t$ to upload malicious local models at each global round. Let $\{\hat{w}_i^t\}_{i \in U_m}$ denote buffered malicious local models at the global round t, then, according to Eq. (3), the

server gets the poisoned aggregated global model $\hat{w}^t$ at global round t by:

$$\hat{w}^t = \pi w^{t-1} + (1-\pi) f_{AGR}(\{\hat{w}_i^t\}_{i\in U_m}, \{w_i^t\}_{i\in U^t\setminus U_m}). \quad (4)$$

We let $\hat{w}$ denote the final SAFL global model in the event of a poisoning attack. The problem to be solved by PoiSAFL is: under the defined threat model, how to craft malicious local models $\{\hat{w}_i^t\}_{i\in U_m}$ for malicious clients at each poisoning round t to significantly mislead the global model while evading the detection of three typical kinds of defenses (i.e., performance-based, entropy-based, and similarity-based defenses) to significantly damage the performance of the final global model $\hat{w}$.

4 Stealth Poisoning Attack Framework for Semi-asynchronous Federated Learning

This paper proposes a scalable stealth poisoning attack framework for SAFL, called PoiSAFL. In this section, we first introduce the design principle of PoiSAFL and then give a high-level overview and design details of PoiSAFL.

4.1 The Design of PoiSAFL

As previously analyzed, in SAFL, inconsistency among local models arises due to varying staleness levels of global models used for updates. Despite this, these noisy local models provide valuable information and diverse perspectives, aiding the global model's learning. Thus, such inconsistency caused by differing levels of staleness is acceptable in SAFL's learning process. In this case, it is challenging to differentiate whether inconsistencies among local models are due to the benign issue of staleness or the malicious actions of adversary-controlled clients, giving chances for malicious local models to bypass the detection of defenses for poisoning attacks.

Our intuition is that the adversary can take advantage of SAFL's tolerance for a certain level of inconsistency among buffered local models to blend malicious local models into legitimate, inconsistent local models, thus evading the detection of Byzantine-resilient defenses. Inherently, at each global round, the buffered local model with the highest staleness is likely to have the greatest inconsistency (in terms of statistical similarity and performance) compared to the current global model and other buffered local models. This level of inconsistency from the stalest local updates is still considered tolerable by the SAFL system. Thus, we can use the stalest local models as the reference to determine an approximate upper bound for "tolerable inconsistency". Then, to achieve our attack goal, our principle is to craft malicious local models that introduce inconsistencies no larger than the tolerable inconsistency bound but can effectively mislead the global model and significantly degrade its performance.

More specifically, since performance-based defenses normally detect and filter local models that have highly inconsistent loss (higher loss in general), to bypass such defenses, the key is to ensure that the loss of malicious local models is no higher than the loss of the most stale local model. Similarly, since the similarity-based defenses recognize and eliminate local models with high statistical dissimilarity from others, to evade such defenses, the crafted malicious local models should keep their distance from the current global model within the range of the distance between the most stale local model and the global model. While existing works have explored strategies to bypass performance-based and similarity-based defenses, they usually rely on generic optimization techniques to generate desirable malicious local models. However, these generated malicious models often exhibit high prediction entropy and are vulnerable to entropy-based defense mechanisms. In this paper, we propose a novel approach to train malicious local models with local data rather than generating vectors of malicious local model parameters based on the optimization method directly. Therefore, our crafted malicious local models can effectively circumvent not only performance-based and similarity-based defenses but also entropy-based detection. To sum up, the key of our approach is to train malicious models to maintain the loss and statistical distance within the tolerances of SAFL.

Let i^* be the fastest-trained malicious client with the smallest staleness (it may participate in the global aggregation at every global round), and S denote the highest staleness of clients in the SAFL process. With the information from client i^*, we propose an approximation strategy to approximate the stalest local model at the current round. Specifically, we use the client i^*'s local model $w_{i^*}^{t-S}$ from S rounds before (whose staleness at the current round t is S) as a proxy for the acceptable stalest model in the current round t. Then, based on our principle, the tolerable loss of malicious models should be approximatively bounded by $\mathcal{F}(w_{i^*}^{t-S}, D_o)$, where D_o is the public dataset used for performance-based and entropy-based detection on the server and D_o is in the same distribution with the raw data of clients. Similarly, the tolerable distance between the malicious model and the newest global model w^{t-1} should be approximatively bounded by $\mathcal{D}(w_{i^*}^{t-S}, w^{t-1})$, where $\mathcal{D}$ is the function to measure the distance between the two models. Then to achieve the attack goal defined in Sec. 3.3, at each poisoning round t, PoiSAFL trains and crafts malicious local models according to the following optimization problem:

$$\begin{aligned} \min_{\{\hat{w}_i^t\}_{i\in U_m}} \quad & \sum_{(x,y)\in D} P[h(\hat{w}^t, x) = y], \\ s.t. \quad & \hat{w}^t = \pi w^{t-1} + (1-\pi) f_{AGR}(\{\hat{w}_i^t\}_{i\in U_m}, \{w_i^t\}_{i\in U^t\setminus U_m}) \\ & \mathcal{F}(\hat{w}_i^t, D_o) \le \beta \mathcal{F}(w_{i^*}^{t-S}, D_o) \\ & \mathcal{D}(\hat{w}_i^t, w^{t-1}) \le \gamma \mathcal{D}(w_{i^*}^{t-S}, w^{t-1}), \end{aligned} \quad (5)$$

where $P[h(\hat{w}^t, x) = y]$ is the probability that the poisoned global model $\hat{w}^t$ labels the input sample x as y. Since model accuracy is influenced by various factors beyond staleness, w_{i*}^{t-S} with the maximum staleness S may not necessarily represent the upper bound of local model loss. Thus, we use a hyperparameter β ($\beta \geq 1$) to relax the loss constraint. Additionally, considering the maximum staleness of buffered local models at round t can be less than S, we introduce a hyperparameter γ ($0 < \gamma \leq 1$), to tighten the distance constraint. With such a constrained optimization objective, the malicious local models are able to closely mimic the behavior of legitimately stale models, exhibiting consistent performance, similarity, and prediction confidence, while still significantly compromising the global model performance.

Notably, we assume that the adversary compromises at least one of the fastest-trained clients because such clients participate in the global aggregation in every round, providing convenience to obtain information from S rounds before (e.g., the local model with staleness S) to estimate the system's tolerable upper bounds for distance and loss in Eq. (5). However, this can also be achieved by compromising other clients, e.g., clients that participate in the global aggregation every S round. Therefore, controlling at least one of the fastest-trained clients is not the condition that the adversary must meet.

4.2 Overview

As shown in Figure. 3, PoiSAFL consists of three modules to craft malicious models that can effectively approximate the objective defined in Eq. (5): anti-training-based model initialization, loss-aware model distillation, and distance-aware model scaling.

To be specific, the anti-training-based model initialization module trains the current global model with an inverted loss function to generate a seed malicious model, maximizing its misclassification probability. Then, taking the seed malicious model as the teacher, the loss-aware model distillation module trains a malicious model for each malicious client with its local data based on KD until the loss is tolerable, striking a balance between mimicking the seed malicious model and predicting the true labels. Such a training process enables the malicious model to achieve high prediction confidence and low information entropy to evade entropy-based defenses and small loss to evade performance-based defenses, addressing the loss constraint in Eq. (5). Meanwhile, by mimicking the seed malicious model, malicious models are imbued with strong poisoning potency, which can maximize the reduction in the global model's correct prediction probabilities, thereby satisfying the optimization objective in Eq. (5). Finally, the distance-aware model scaling module helps malicious models evade detection by the similarity-based defenses by further scaling them to limit their statistical dissimilarity to the other models within the acceptable range, as specified by the distance constraint in Eq. (5).

4.3 Anti-training-based Model Initialization

The goal of an untargeted poisoning attack to SAFL is to reduce the overall accuracy of the global model, without having a specific target class or objective in mind. The maximized attack impact of the untargeted poisoning attack represents the worst case where the global model completely loses its ability to classify, i.e., it would misclassify samples in every class. In order to launch a successful untargeted poisoning attack and significantly degrade the performance of the global model, we need to craft malicious models that have strong enough poisonousness to lead the global model to the worst case. Therefore, we initialize a seed malicious model via anti-training to make it misclassify samples in every class. Specifically, we train the seed malicious model $\hat{w}_m$ with $D_m = \sum_{i \in U_m} |D_i|$ to maximize the probability of each sample being classified into incorrect classes with the following inverted loss function:

$$\begin{aligned} \mathcal{L}_{anti} &= \frac{1}{|D_m|} \sum_{(x,y) \in D_m} \phi(1 - h(\hat{w}_m, x), y) \\ &= \frac{1}{|D_m|} \sum_{(x,y) \in D_m} \sum_{l=1}^{L} -\log y^{(l)} (1 - h(\hat{w}_m, x)^{(l)}), \end{aligned} \tag{6}$$

where ϕ is the distance measure function, and here we adopt the cross-entropy. L is the number of classes, $h(\hat{w}_m, x)$ is the output class probability vector of $\hat{w}_m$ when the input is x, and $h(\hat{w}_m, x)^{(l)}$ is the prediction probability of class l. Similarly, $y^{(l)}$ is the value of l-th element in label vector y. Aggregating a model like the seed malicious model into the global model would greatly degrade its prediction performance.

4.4 Loss-aware Model Distillation

Intuitively, deriving malicious models from the seed malicious model that has the strongest poisonousness enables the malicious model to possess sufficient poisoning capability to poison the global model. To this end, taking the seed malicious model as the teacher model and the malicious model as the student model, we propose to train the malicious model for each malicious client based on KD to inherit the toxicity from the seed malicious model. However, to evade the performance-based defenses that filter out local models with high loss or low accuracy, we need to ensure that the loss of the malicious model remains within an acceptable range. Therefore, based on Eq. (1), we design the following training loss function for each malicious model $\bar{w}_i^t$ to achieve the trade-off between mimicking the seed malicious model (inheriting the strong poisoning capability) and predicting the true labels (evading performance-based defenses):

$$\begin{aligned} \mathcal{L}_{mal} = \frac{1}{|D_i|} \sum_{(x,y) \in D_i} \{ & \alpha * \phi(h(\bar{w}_i^t, x), y) \\ & + (1 - \alpha) * \phi[\rho(\bar{h}(\bar{w}_i^t, x), \tau), \rho(\bar{h}(\hat{w}_m, x), \tau)] \}, \end{aligned} \tag{7}$$

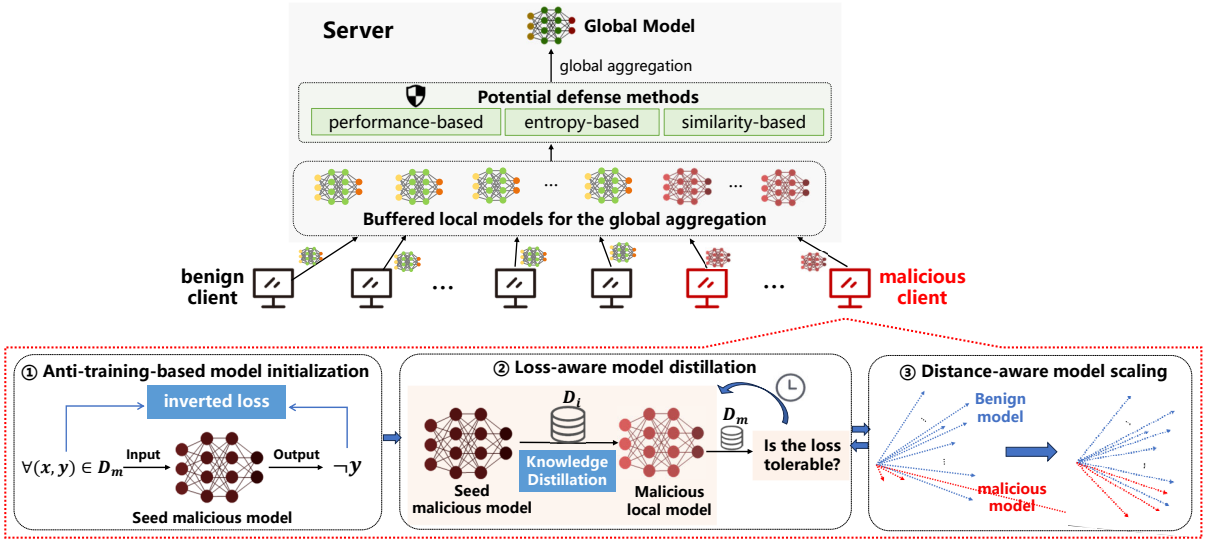


Figure 3: The overview of PoiSAFL.

where $\bar{h}(\bar{w}_i^t, x)$ and $\bar{h}(\hat{w}_m, x)$ are the output logits of $\bar{w}_i^t$ and $\hat{w}_m$ for the input x, respectively. The first item enables the malicious model $\bar{w}_i^t$ to maintain a reasonable level of prediction performance while the second item allows it to inherit the malicious capabilities from the seed malicious model.

In this work, we consider a realistic attack model that makes only weak assumptions about the adversary's knowledge, where the adversary does not have access to the server's trusted dataset D_o. Based on the known information of the adversary, we approximate $\mathcal{F}(w_{i*}^{t-S}, D_o)$ by $\mathcal{F}(w_{i*}^{t-S}, D_m)$. Then we train the malicious model until $\mathcal{F}(\bar{w}_i^t, D_m) \leq \beta \mathcal{F}(w_{i*}^{t-S}, D_m)$. This allows the malicious model to bypass detection methods that rely on model performance metrics while having the ability to mislead the global model.

4.5 Distance-aware Model Scaling

After malicious models have been initialized and refined through the first two modules, we apply a scaling mechanism for the model parameters to satisfy the distance constraint. This step is crucial for evading similarity-based defenses employed in the SAFL system since it ensures that malicious models would not be outliers in terms of statistical similarity. Let ε denote the scaling factor, the goal of the distance-aware model scaling can be formulated by:

$$\mathcal{D}(\varepsilon \bar{w}_i^t, w^{t-1}) \leq \gamma \mathcal{D}(w_{i*}^{t-S}, w^{t-1}). \tag{8}$$

We adopt Euclidean distance as $\mathcal{D}$. By employing this principled scaling approach, we can craft malicious models whose statistical dissimilarity to other models is within the acceptable range of the SAFL system to bypass the similarity-based defenses, while still maintaining the desired adversarial properties established in the previous modules. Then we get the final carefully crafted malicious local model $\hat{w}_i^t = \varepsilon \hat{w}_i^t$ for each malicious client.

Note that if the parameter scaling in this module is unable to craft malicious models that are statistically similar to benign models to bypass the similarity-based defenses, the PoiSAFL framework will cycle back to the second module to perform additional loss-aware model distillation. This iterative process involves repeatedly refining the malicious models through distillation to transfer the toxic characteristics of the seed malicious model, followed by attempts to scale the models' parameters to decrease their statistical distance from the benign models. The framework cycles through these steps until the crafted malicious models meet constraint conditions on model loss and statistical similarity.

4.6 The Workflow of PoiSAFL

In PoiSAFL, the adversary rapidly crafts malicious models for malicious clients and controls them to upload malicious models as quickly as the fastest clients while benign clients perform normally. The workflow of PoiSAFL contains the following steps at the poisoning round t:

1) Local Training: Each benign client $i \in U_b$ trains its local model with local data D_i. As for malicious clients in U_m, they do not need to perform local training themselves, but the adversary crafts malicious local models for them. For each malicious client $i \in U_m$, the adversary crafts malicious model $\hat{\omega}_i^t$ based on D_i and D_m. To be specific, as described in our design details, the adversary first performs the anti-training-based model initialization, then circularly performs loss-aware model distillation and the distance-aware model scaling to meet the constraint conditions on model loss and statistical similarity.

2) Local update uploading: Each benign client $i \in U_b$ uploads their local model to the server when the local training is completed. Each malicious client $i \in U_m$ uploads their malicious model $\hat{\omega}_i^t$ as quickly as the fastest clients.

3) Semi-asynchronous global aggregation: The server buffers local models uploaded by clients and aggregates them to update the global model based on Eq. (4) once a fixed period has elapsed.

4) Broadcasting: The server broadcasts the latest global model to those clients participating in the global aggregation to act as the initialization of their local training, including all malicious clients and some of the benign clients.

Note that in PoiSAFL, the adversary can quickly crafts malicious models for malicious clients and the required time is less than that required by the fastest clients to perform local training. In this case, malicious clients are able to upload malicious local models and participate in the global aggregation at each poisoning round. Then, according to the calculation method of staleness in Sec. 2.1, malicious clients' staleness is stably equal to 1 during the poisoning procedure, which is the smallest. Therefore, even though the server performs staleness-aware aggregation among buffered local updates, malicious clients could achieve high attack impacts since they would not be down-weighted.

5 Evaluation

In this section, we evaluate the performance of PoiSAFL and compare it with existing attacks under three typical kinds of defenses.

5.1 Experimental setup

Dataset. We use three commonly used datasets in FL for the performance evaluation. **1)** *FashionMNIST* [43] consists of 70k grayscale 28×28 images associated with labels from 10 classes. Each class has 6k training examples and 1k test examples. **2)** *CIFAR10* [17] contains 60k 32×32 RGB images associated with a label from 10 classes. Each class has 5k training examples and 1k test examples. **3)** *GTSRB* [32] contains 32×32 traffic-sign images, with 39, 209 for training and 12, 630 for validation, and their labels are from 43 classes. In our experiments, we construct both the IID and Non-IID distribution among clients. For IID settings, we divide the training dataset into $M+1$ equal partitions. The first M partitions, each containing $60k/(M+1)$ samples, are distributed to the M clients. The final partition is retained as a public dataset, which is needed by performance-based and entropy-based defense methods. For non-IID settings, we choose the first 100 training samples from each class to form the public dataset and use Dirichlet distribution with parameter δ to distribute the rest training samples to each client. Here δ indicates the non-IID degree and a smaller δ means a higher non-IID degree.

Model Architecture. During training, a simple convolutional neural network with 2 convolutional layers and 2 fully connected layers (called CNN2) is utilized for FashionMNIST, and ResNet18 is adopted for CIAFR10. For GTSRB, we employ a convolutional neural network that consists of two convolutional layers, a Spatial Transformer Network (STN) layer incorporating two more convolutional layers within it, and two fully connected layers (called CNN-STN). Appendix. A gives details of CNN2 and CNN-STN.

SAFL and Attack Settings. For the three datasets used - FashionMNIST, CIFAR10, and GTSRB, we set 60 clients. The default local processing time for clients is evenly distributed over 1 to 6 unit times, and the global aggregation is performed every 1 unit time. In this case, the upper bound of staleness is 6. For both FashionMNIST and CIFAR10, we use Adam optimizer with a learning rate of 0.01 and batch size of 64, and local epoch 1 is adopted for all clients. For GTSRB, we adjust the learning rate to 0.0001, while maintaining the same batch size and local epoch configuration. The total number of training rounds is set to 150 for Fashion-MNIST, and 250 for CIFAR-10 and GTSRB, respectively. The poisoning attacks commence at the training round 100 for Fashion-MNIST and 200 for CIFAR-10 and GTSRB. For our attack, we train the seed malicious model based on Eq. (6) 15 rounds for Fashion-MNIST, 5 rounds for CIFAR-10, and 1 round for GTSRB. Then, we train each malicious local model based on Eq. (7) until the loss of the malicious model meets the loss constraint or reaches the maximum optimization round 15. By default, we set the percentage of adversary-controlled malicious clients equal to 20% and set the non-IID degree δ equal to 1. Specifically, to control at least one of the fastest-trained clients, we set that the adversary first selects one malicious client from the fastest client group, then randomly picks other malicious clients from the rest clients. We set $\pi = 0.8$ when performing the global aggregation. When performing the loss-aware model distillation, the temperature factor $\tau = 2$. What's more, given that there are 20% malicious clients, we set the distance constraint hyperparameter $\gamma = 0.8$ to prevent malicious clients from being identified as 20% outliers and detected by similarity-based defenses. As for β ($\beta \geq 1$) that relaxes the loss constraint, given that it should not be too large or too small to ensure the negative impact and undetectability of malicious models, we tested values from 1 to 8 and found $\beta = 2.5$ optimal (The results can be found in Appendix. A).

Metric. In our experiments, we test the global model accuracy and use the attack impact, i.e., the reduction in global model accuracy caused by the attack, as the evaluation metric. If A_o denotes the best accuracy of the global model when there is no attack, A_f denotes the final overall accuracy of the global model after the adversary launches the attack, then the attack impact can be computed by $A_o - A_f$. Successful poisoning attacks would cause significant attack impacts. We present averaged results calculated over three runs.

Compared Attacks. We compare our proposed attack against several well-known and representative attacks to evaluate its performance. The compared attacks include classic data poisoning attacks (LabelFlip [36]), typical model poisoning

attacks (SignFlip [19]), and several state-of-the-art poisoning attacks that claim to bypass existing defenses (LIE [2], LA [10], MinSum [31], Grad [31]). The details of these attacks can be found in the Appendix. A. For a fair comparison, we implement all these attacks with the same assumption about the adversary's knowledge. That is, the adversary can only access information about malicious clients but is unaware of local data and local models of benign clients.

Defenses. We study the attack performance of various poisoning attacks against the three typical kinds of defense methods, including 5 statistical similarity-based defenses (Median [44, 49], TriMean [44, 49], Norm-clipping [35], FLARE [38], AFLGuard [11]), 2 performance-based defenses (LFR [10], Sageflow [29]), and 2 entropy-based defenses (Sageflow [29], Efilter [29]). Among them, AFLGuard, Sageflow, and Efilter are designed for AFL and others are designed for SFL. Note that AFLGuard, LFR, Sageflow, and Efilter all require a trusted dataset on the server that has the same data distribution as clients' local data. The details of these defenses can be found in the Appendix. A.

5.2 Experimental Results

5.2.1 Attack Performance Comparison

Table 1, 2 and 3 summarize the attack impacts of PoiSAFL and compared attacks under three typical kinds of defenses with Fashion MNIST, GTSRB, and CIFAR10, respectively.

PoiSAFL outperforms baseline attack methods: We can observe that in both IID and non-IID scenarios, PoiSAFL has significant attack impacts under all tested defenses, and outperforms the baseline attack methods in almost all cases. This validates the effectiveness of PoiSAFL, i.e., PoiSAFL can effectively degrade the global model performance while bypassing three typical kinds of Byzantine-resilient defenses. PoiSAFL derives malicious local models from a seed model that misclassifies all samples, following a guideline to maintain the discrepancy between the malicious and benign models within an acceptable range. Meanwhile, it controls all malicious clients to upload malicious local models at every poisoning round. Then, these considerable malicious local models exhibit high toxicity while maintaining undetectable during the global aggregation process, thus notably altering the benign behavior of the global model and diminishing its prediction accuracy even though defenses are adopted. Besides, using the same seed malicious model across multiple global rounds would cause malicious local models to inherit similar toxicity, consistently steering the global model toward the seed model. This multi-round consistency has a cumulative effect, increasingly misleading the global model.

Besides, it can be observed that in most cases, all baseline attack methods exhibit extremely limited attack impact. A common reason is that the randomly selected malicious clients vary in processing power and speed and thus do not participate in every round of global aggregation in SAFL. This results in a reduced proportion of malicious clients that exert attack influence during each round's global aggregation, thereby diluting their attack impact and making them easily detectable. Additionally, these baseline attack methods either do not account for potential defenses during design or only consider defenses based on statistical similarity, thus failing to demonstrate significant attack impact under the three typical kinds of defenses.

One exception is that under FLARE, PoiSAFL is not top-performing. It can be explained as follows. FLARE identifies malicious models by examining parameter similarity in the penultimate layer. The attack that best mimics benign models in the penultimate layer could be harder to detect and exert the strongest attack impact. While PoiSAFL and some baseline attacks maintain overall similarity between malicious and benign local model parameters, the penultimate layer's similarity of their crafted malicious models has a certain randomness. This randomness explains why PoiSAFL sometimes shows the highest impact under FLARE and sometimes does not. Nevertheless, PoiSAFL consistently demonstrates strong attack efficacy under FLARE, proving to be effective.

Attack performance comparison under different kinds of defenses: Table 1, 2 and 3 show that our attack PoiSAFL can effectively poison SAFL under all three kinds of byzantine-resilient defenses, validating its effectiveness and stealthiness. On the contrary, most existing attack methods cause only negligible accuracy reduction under performance-based and entropy-based defenses. This indicates that most baseline methods can not evade the detection of these two kinds of defenses. The performance-based and entropy-based defenses are more effective against poisoning attacks since they can obtain additional reference information for benign local models by leveraging the server-hosted trusted dataset. This enables more accurate identification of malicious local models. In contrast, the similarity-based defenses rely solely on comparing buffered local models to detect anomalies, thus they may struggle to detect sophisticated attacks that mimic benign behavior closely, especially in SAFL systems where buffered local models naturally involve a certain level of statistical dissimilarity.

However, it is noteworthy that performance and entropy-based defenses are predicated on the strong assumption, i.e., the server has access to additional trusted data for the learning task. This assumption is often impractical in real-world scenarios. Our attack strategy demonstrates the capability to circumvent even these strongly hypothesized defenses, highlighting that SAFL systems are vulnerable to poisoning risks and existing defense methods offer a false sense of security.

5.2.2 Effects of the Proposed Mechanisms

To verify the effectiveness of the proposed Anti-Training-based model Initialization mechanism (ATI), the Loss-Aware

Table 1: Attack performance comparison with FashionMNIST dataset. The following table reports the *global model accuracy* when there is No Attack and the *attack impact* (%) on global model accuracy caused by baselines and PoiSAFL under kinds of defenses. The best attack impacts are highlighted in bold. Under all settings, our attack outperforms baselines.

Dataset	AGR	No Attack	LabelFlip	SignFlip	LIE	MinSum	LA	Grad	PoiSAFL
Fashion MNIST (IID)	**Median**	89.14	1.11	1.33	4.52	7.13	4.8	3.99	**58.79**
	TriMean	89.94	1.13	2.09	4.39	11.79	7.89	10.53	**36.16**
	NormClip	89.84	1.05	0.9	0.75	2.32	0.57	0.31	**12.73**
	FLARE	89.21	21.78	0.93	13.06	21.28	7.44	13.52	**22.94**
	AFLGuard	89.88	-0.61	-0.44	-0.17	0.1	-0.23	0.22	**17.64**
	LFR	89.46	0.31	0.41	0.63	0.44	0.18	0.64	**47.81**
	Efilter	89.34	4.94	0.2	3.46	4.22	0.05	4.22	**48.36**
	Sageflow	89.44	0.86	0.21	0.48	0.47	0.28	0.19	**28.23**
Fashion MNIST (Non-IID)	**Median**	87.75	0.51	0.22	5.43	8.16	4.4	3.54	**65.83**
	TriMean	88.51	3.18	1	10.36	13.59	22.23	17.15	**22.28**
	NormClip	88.5	-0.21	-0.19	0.54	0.28	0.32	0.39	**78.48**
	FLARE	87.12	9.26	-0.74	7.75	6.52	9.87	19.9	**29.37**
	AFLGuard	89.17	0.26	-0.15	0.29	0.12	1.57	0.23	**20.24**
	LFR	80.53	0.71	0.48	-0.26	0.02	0.12	1.66	**51.18**
	Efilter	88.01	8.66	17.84	1.38	8.26	0.09	7.33	**77.92**
	Sageflow	89.95	0.87	1.44	1.03	1.6	0.79	1.04	**60.85**

Table 2: Attack performance comparison with GTSRB dataset. The following table reports the *global model accuracy* when there is No Attack and the *attack impact* (%) on global model accuracy caused by baselines and PoiSAFL under kinds of defenses. The best attack impacts are highlighted in bold. Under all settings except FLARE, our attacks outperform baselines.

Dataset	AGR	No Attack	LabelFlip	SignFlip	LIE	MinSum	LA	Grad	PoiSAFL
GTSRB (IID)	**Median**	94.74	1.06	0.42	2.99	1	0.62	1.2	**23.8**
	TriMean	94.85	1.56	0.42	2.92	0.42	0.5	0.34	**16.83**
	NormClip	94.54	0.03	-0.19	-0.2	-0.19	9.05	-0.17	**23.24**
	FLARE	94.77	10.75	-0.12	21.99	7.46	10.11	**22.81**	16.34
	AFLGuard	95.11	0.43	0.33	0.32	0.33	0.34	0.39	**15.7**
	LFR	94.91	0.16	0.15	0.06	0.15	0.18	0.17	**10.97**
	Efilter	94.81	1.24	1.79	0.15	0.02	0.56	0.21	**14.17**
	Sageflow	95.7	1.07	0.91	0.48	0.91	0.97	1.08	**10.97**
GTSRB (Non-IID)	**Median**	85.35	3.16	2.33	5.94	3.54	4.47	2.95	**17.39**
	TriMean	85.09	2.41	2.27	5.93	2.04	1.63	1.12	**18.6**
	NormClip	81.62	0.42	0.08	0.25	0.28	0.55	0.51	**11.59**
	FLARE	88.27	12.76	1.33	18.7	16.77	8.08	**26.05**	15.38
	AFLGuard	84.96	0.17	0.24	8.06	2.54	0.11	2.5	**8.18**
	LFR	86.74	0.5	0.04	0.4	0.04	4.69	0.1	**12.23**
	Efilter	85.12	5.18	5.32	5.27	1.69	1.8	1.39	**19.92**
	Sageflow	85.68	0.39	1.76	1.34	1.97	2.5	1.1	**14.61**

model Distillation mechanism (LAD), and the Distance-Aware model Scaling mechanism (DAS), we conduct experiments of PoiSAFL with and without ATI, LAD, and DAS under different kinds of defenses on FashionMNIST, respectively. Among them, PoiSAFL without ATI randomly initializes a seed malicious model. PoiSAFL without LAD directly aggregates the seed malicious model and the local model as the malicious local model. PoiSAFL without DAS uploads the crafted malicious local models without scaling. The performance comparison results are shown in Figure 4.

We have the following observations: 1) When there is no defense deployed on the server, each proposed mechanism can successfully poison the global model. This demonstrates that all the proposed mechanisms can effectively craft malicious local models to destroy the system's learning performance. 2) The complete version of PoiSAFL causes a significant drop in global model accuracy across various defenses. This demonstrates that the combination of the three proposed mechanisms effectively bypasses three typical defenses while successfully launching a poisoning attack, further highlighting the effectiveness of these proposed mechanisms. 3) Not performing the ATI mechanism results in the drop of attack impact under

Table 3: Attack performance comparison with CIFAR10 dataset. The following table reports the *global model accuracy* when there is No Attack and the *attack impact* (%) on global model accuracy caused by baselines and PoiSAFL under kinds of defenses. The best attack impacts are highlighted in bold. Under all settings except FLARE, our attack outperforms baselines.

Dataset	AGR	No Attack	LabelFlip	SignFlip	LIE	MinSum	LA	Grad	PoiSAFL
CIFAR10 (IID)	**Median**	78.01	1.28	14.78	1.14	0.93	0.5	0.62	**56.66**
	TriMean	77.47	3.16	0.53	0.51	7.64	4.59	18.06	**31.48**
	NormClip	77.45	3.52	0.06	0.23	1.92	13.73	0.16	**15.43**
	FLARE	77.21	44.57	2.34	39.43	44.1	36.56	**46.45**	21.61
	AFLGuard	77.51	4.06	0.39	0.47	3.7	7.65	0.48	**14.06**
	LFR	77.23	0.01	-0.02	14.28	-0.06	15.16	18.66	**30.13**
	Efilter	77.39	8.64	0.29	0.13	0.03	1.31	0.14	**35.64**
	Sageflow	77.2	1.09	0.88	0.07	0.02	0.04	1.07	**17.24**

Table 4: Attack performance of PoiSAFL when the type of defenses adopted by the server is known. The following table reports the *Attack Impact* (%) and (*the increase in attack impact compared to when the adopted defense is unknown*). It is shown that PoiSAFL achieves a trade-off between stealthiness and effectiveness to cause more global model accuracy reduction while bypassing the detection by selectively executing three modules according to the type of defense.

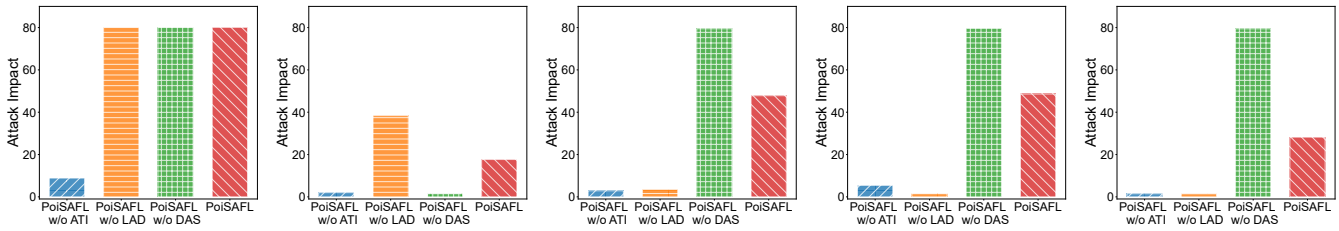
Dataset	Median	TriMean	NormClip	FLARE	AFLGuard	LFR	Efilter	Sageflow
Fasion MNIST (IID)	79.04	79.52	69.32	50.27	38.41	79.45	79.59	79.43
GTSRB (Non-IID)	59.61	65.12	65.83	60.59	49.32	70.76	78.52	78.19
CIFAR10 (IID)	67.99	67.19	22.68	65.25	24.69	64.71	67.39	27.49

all defenses. This is because randomly initialized the seed malicious model does not always exhibit a high misclassification rate, thus the derived malicious local models may not inherit the desired harmful properties. In contrast, ATI initializes the seed malicious model to ensure a high misclassification probability, making derived malicious local models malicious enough to mislead the global model. In conclusion, ATI is an indispensable component for PoiSAFL to achieve significant attack impact. 4) Under the statistical similarity-based defense, e.g., AFLGuard, the attack impact of PoiSAFL without DAS is low, indicating that PoiSAFL without DAS can not bypass AFLGuard's defenses. This observation highlights the importance of the DAS mechanism in circumventing defenses based on statistical similarity. Additionally, when LAD is not executed, the attack impact of PoiSAFL increases. This is because LAD limits the poisonousness of malicious local models to bypass performance and entropy defenses, which is not actually necessary under AFLGuard. 5) When the server adopts performance-based and entropy-based defenses, i.e., LFR, Efilter, and Sageflow, PoiSAFL without LAD shows negligible attack impact. This demonstrates that LAD mechanism plays a significant role in bypassing defenses based on performance and entropy. Additionally, we can observe that without executing the DAS mechanism, there is a greater attack impact under both LFR, Efilter, and Sageflow. This is because DAS restricts the norm of malicious local models to circumvent statistical similarity-based defenses, thereby reducing the scale of malicious local models and consequently lessening their adverse impact on the global model.

We also evaluate the attack performance of PoiSAFL when the type of defenses adopted by the server is known. In this setting, PoiSAFL can selectively execute the proposed three mechanisms to achieve the optimal attack impact while bypassing defenses. For instance, when the server adopts a performance-based defense like LFR, the adversary launches PoiSAFL without DAS to poison the SAFL system. The attack impact results on Fashion MNIST with IID data distribution are shown in Table. 4. We can observe that when the type of adopted defenses is known to the adversary, the proposed attack shows stronger attack impacts and causes more damage to the global model under all defenses. This makes sense because to bypass all three defenses at once, PoiSAFL limits the degree to which malicious models can act maliciously (i.e., the loss and statistical dissimilarity of malicious models are limited with a benign range). However, when the type of defense used is known, PoiSAFL only executes specific modules to bypass specific kinds of defenses, which reduces the restriction on malicious behaviors of malicious local models, thereby effectively enhancing the attack impact.

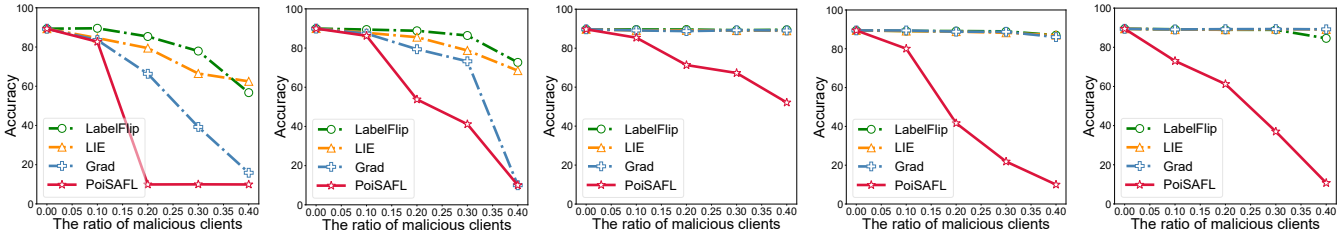
5.2.3 Effects of Parameters

Effect of the ratio of malicious clients. In Figure 5, we compare the attack performance of PoiSAFL and baselines under No Defense, TriMean, AFLGuard, LFR, and Sageflow when the ratio of malicious clients changes. It shows the final overall accuracy of the global model after the attack. We can find that 1) as the ratio of malicious clients increases, all attacks show stronger attack impacts, leading to the global model's de-



(a) No Defense (b) AFLGuard (c) LFR (d) Efilter (e) Sageflow

Figure 4: Effect of the proposed mechanisms.



(a) No Defense (b) TriMean (c) AFLGuard (d) LFR (e) Sageflow

Figure 5: Effect of the ratio of malicious clients.

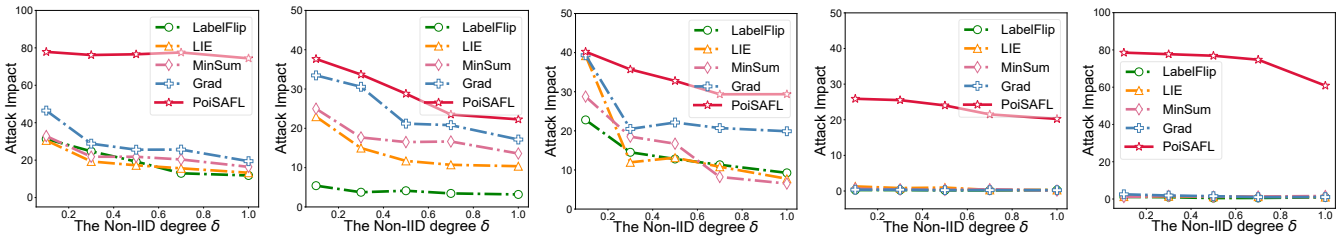
creased accuracy. 2) PoiSAFL performs better than baselines under different kinds of defenses regardless of the percentage of malicious clients. This further demonstrates the effectiveness and stealthiness of PoiSAFL. 3) Under AFLGuard, LFR, and Sageflow, even with an attacker ratio as high as 40%, attacks like LabelFlip, LIE, and Grad struggle to make strong attack impacts. In contrast, PoiSAFL completely disrupts the global model performance when the attacker ratio is high under all defenses. This further demonstrates that PoiSAFL effectively reduces global model performance while evading detection by three kinds of typical defense methods while other baseline attacks fail to bypass some kinds of defenses, even with a high attacker ratio.

Effect of the non-IID degree. We investigate the attack impacts of PoiSAFL and baselines with different Non-IID degrees under No Defense, TriMean, FLARE, AFLGurad, and Sageflow and show the results in Figure 6. Note that a smaller δ indicates a larger non-IID degree. We can observe that, generally, the attack impacts of all methods increase with a higher Non-IID degree under each tested defense. This occurs because a higher non-IID degree of data distribution leads to a greater inconsistency among buffered local models. In this case, distinguishing between malicious and benign local models becomes more challenging for defense mechanisms. Malicious updates, therefore, have a higher chance of blending in with legitimate updates, allowing them to bypass the detection and filtering of defense methods. This increased evasion capability means more malicious updates can be integrated into the global model during aggregation, leading to a higher probability of misleading the global model and reducing its classification accuracy. Besides, it can be observed that PoiSAFL consistently outperforms baseline attacks regardless of the Non-IID degree.

Effect of the upper bound of staleness. Figure. 7 shows the attack impacts of PoiSAFL with and without the knowledge of the type of adopted defenses under TriMean, AFLGuard, LFR, Efilter, and Sageflow when the upper bounds of staleness in the SAFL system changes. It can be observed that whether the staleness upper bound is large or small, PoiSAFL demonstrates significant attack impacts, especially when it knows the defense type. Besides, the greater the upper bound of staleness, the more significant the attack impact. A higher upper bound of staleness indicates a greater diversity among clients' staleness, leading to higher local model inconsistency. This provides the adversary with more room to inject malicious behavior into the global model while mimicking benign models to bypass detection, thus more significantly degrading the global model performance. Furthermore, regardless of the staleness upper bound, PoiSAFL performs better when the type of defense adopted by the server is known. It further demonstrates PoiSAFL's capability to amplify its attack impact by flexibly executing the three proposed modules.

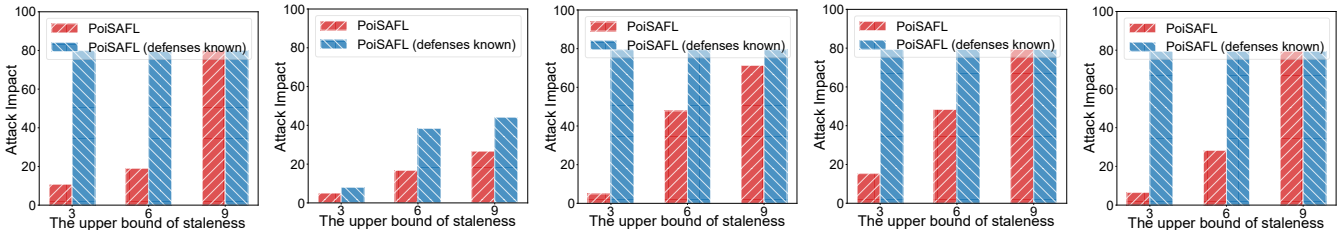
6 Discussion

Scalability of PoiSAFL. PoiSAFL focuses on a constrained optimization problem for malicious models and contains three modules to approximately solve the problem. Particularly, each module serves a specific purpose, e.g., making malicious models satisfy a specific constraint to bypass a specific kind of defense. Such a modular design allows flexible extension of the constraint in the constrained optimization problem and easy integration of new defense modules that address future new kinds of defenses. For example, if a future defense relies



(a) No defense (b) TriMean (c) FLARE (d) AFLGuard (e) Sageflow

Figure 6: Effect of the Non-IID degree.



(a) TriMean (b) AFLGuard (c) LFR (d) Efilter (e) Sageflow

Figure 7: Effect of the upper bound of staleness.

on structural properties, a module can be added to tailor malicious models accordingly while maintaining their malicious traits. The flexibility to incorporate new modules makes this framework highly scalable and future-proof, enabling it to keep pace with the evolving security landscape in SAFL.

Potential adaptive defenses against PoiSAFL. PoiSAFL reveals that existing defenses might offer a misleading sense of security, much less they might rely on an impractical strong assumption that the server can obtain an additional trusted dataset to assist in detection. This underscores the importance of developing more robust defenses to effectively counteract such attacks and ensure the integrity of SAFL systems.

There are several potential adaptive defenses: 1) Parameter Subset Similarity-Based Defense: PoiSAFL focuses on maintaining the overall statistical similarity between malicious and benign local model parameters. Therefore, defenses that compare the similarity of parameter subsets may potentially defeat PoiSAFL. However, considering inherent inconsistencies among benign local model parameters caused by staleness, such methods might incorrectly identify benign models as malicious. 2) Client Participation Controlling-based Defense: The strong attack impact of PoiSAFL partly depends on malicious clients uploading malicious local models in every global round. Therefore, limiting the number of malicious clients participating in global aggregation each round could help defend against PoiSAFL. A straightforward approach might involve the server randomly selecting a subset of buffered local models of each staleness level to participate in the global aggregation. However, this could significantly affect the performance of the global model when there is no malicious client.

7 Conclusion

In this paper, we fully explored the poisoning risk in SAFL and revealed that current defenses give a false sense of security. We proposed PoiSAFL, a novel stealth poisoning attack framework for Byzantine-resilient SAFL systems. Consisting of three major modules (i.e., anti-training-based model initialization, loss-aware model distillation, and distance-aware model scaling), PoiSAFL performs constrained model training to craft malicious local models to degrade the global model performance without being detected by three typical kinds of defenses. Extensive experiments demonstrated the effectiveness and stealthiness of PoiSAFL. It can bypass all three typical kinds of defenses. Besides, it outperforms state-of-the-art poisoning attacks and causes a more significant reduction in global model performance in most cases.

Acknowledgement

This work is supported by National Natural Science Foundation of China (Grants No. U24B20182, 62122066), National Key R&D Program of China (Grant No. 2021ZD0112803), and Key R&D Program of Zhejiang (Grant No. 2024C01164, 2022C01018).

Ethics Considerations

This work adheres to ethical guidelines and does not contain any content that violates ethical standards. There were no conflicts of interest that could have influenced the research findings. Transparency and integrity were maintained throughout the research process to uphold ethical standards. It does

not involve human subjects or any form of data collection from individuals, directly or indirectly. All data utilized in this research are publicly available datasets that do not contain personally identifiable information.

Respect for persons. This work discloses important vulnerabilities in SAFL systems regarding their susceptibility to poisoning attacks to the public. It respects the rights of developers and users of SAFL systems to be informed.

Beneficence. The work aims to highlight vulnerabilities in SAFL systems to improve their security. While the proposed approach might provide insights to adversaries, the anticipated benefit is to reveal that current security measures might not be as reliable as believed, thereby raising awareness about existing security flaws and motivating the development of robust defenses. It encourages the improvement of security measures and contributes to making SAFL systems safer overall.

Justice. The insights from this work are shared with a wide audience, including developers, researchers, and policymakers, which ensures that the benefits of increased security awareness and improved defenses are widely distributed. Moreover, this work benefits every user of SAFL systems equally, avoiding any bias that might favor certain groups over others.

Respect for Law and Public Interest. This work complies with data protection laws and ethical research guidelines. By revealing the serious poisoning risks in SAFL systems with the potential benefits of increased security and awareness, the research can contribute positively to the field of cyberspace security.

Compliance with the Open Science Policy

In alignment with the open science policy introduced by USENIX Security, we adhere to transparency and reproducibility by making the associated research artifacts available to the public.

Dataset. The datasets (i.e., Fashion MNIST, CIFAR10, GTSRB) used in our study are commonly-used public datasets. They can be easily understood and reused by other researchers in the field.

Source Code. The source code developed for this study can be accessed at [https://archive.softwareheritage.org/browse/origin/directory/?origin_url=https://github.com/Eyanee/Code-for-PoiSAFL]. This allows others to verify the effectiveness of our attack and extend and build upon our work.

References

[1] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2938–2948. PMLR, 2020.

[2] Moran Baruch, Gilad Baruch, and Yoav Goldberg. A little is enough: circumventing defenses for distributed learning. In *Advances in Neural Information Processing Systems*, pages 8635–8645, 2019.

[3] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning*, pages 634–643. PMLR, 2019.

[4] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. *Advances in neural information processing systems*, 30, 2017.

[5] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. Fltrust: Byzantine-robust federated learning via trust bootstrapping. *arXiv preprint arXiv:2012.13995*, 2020.

[6] Xiaoyu Cao and Neil Zhenqiang Gong. Mpaf: Model poisoning attacks to federated learning based on fake clients. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3396–3404, 2022.

[7] Shuai Chen, Xiumin Wang, Pan Zhou, Weiwei Wu, Weiwei Lin, and Zhenyu Wang. Heterogeneous semi-asynchronous federated learning in internet of things: A multi-armed bandit approach. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(5):1113–1124, 2022.

[8] Yang Chen, Xiaoyan Sun, and Yaochu Jin. Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation. *IEEE transactions on neural networks and learning systems*, 31(10):4229–4238, 2019.

[9] Zheyi Chen, Weixian Liao, Kun Hua, Chao Lu, and Wei Yu. Towards asynchronous federated learning for heterogeneous edge-powered internet of things. *Digital Communications and Networks*, 7(3):317–326, 2021.

[10] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local model poisoning attacks to {Byzantine-Robust} federated learning. In *29th USENIX security symposium (USENIX Security 20)*, pages 1605–1622, 2020.

[11] Minghong Fang, Jia Liu, Neil Zhenqiang Gong, and Elizabeth S Bentley. Aflguard: Byzantine-robust asynchronous federated learning. In *Proceedings of the 38th Annual Computer Security Applications Conference*, pages 632–646, 2022.

[12] Rachid Guerraoui, Sébastien Rouault, et al. The hidden vulnerability of distributed learning in byzantium. In *International Conference on Machine Learning*, pages 3521–3530. PMLR, 2018.

[13] Prajjwal Gupta, Krishna Yadav, Brij B Gupta, Mamoun Alazab, and Thippa Reddy Gadekallu. A novel data poisoning attack in federated learning based on inverted loss function. *Computers & Security*, 130:103270, 2023.

[14] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

[15] Jiahui Hu, Zhibo Wang, Yongsheng Shen, Bohan Lin, Peng Sun, Xiaoyi Pang, Jian Liu, and Kui Ren. Shield against gradient leakage attacks: Adaptive privacy-preserving federated learning. *IEEE/ACM Transactions on Networking*, 2023. doi: 10.1109/TNET.2023.3317870.

[16] Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *2018 IEEE Symposium on Security and Privacy (S&P)*, pages 19–35. IEEE, 2018.

[17] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images (technical report). 2009. University of Toronto.

[18] Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. Data poisoning attacks on factorization-based collaborative filtering. *Advances in neural information processing systems*, 29:1893–1901, 2016.

[19] Liping Li, Wei Xu, Tianyi Chen, Georgios B Giannakis, and Qing Ling. Rsa: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 1544–1551, 2019.

[20] Youhuizi Li, Haitao Yu, Yan Zeng, and Qianqian Pan. Hfsa: A semi-asynchronous hierarchical federated recommendation system in smart city. *IEEE Internet of Things Journal*, 2023.

[21] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE communications surveys & tutorials*, 22(3):2031–2063, 2020.

[22] Qianpiao Ma, Yang Xu, Hongli Xu, Zhida Jiang, Liusheng Huang, and He Huang. Fedsa: A semi-asynchronous federated learning mechanism in heterogeneous edge computing. *IEEE Journal on Selected Areas in Communications*, 39(12):3654–3672, 2021.

[23] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[24] Yinbin Miao, Da Kuang, Xinghua Li, Shujiang Xu, Hongwei Li, Kim-Kwang Raymond Choo, and Robert H Deng. Privacy-preserving asynchronous federated learning under non-iid settings. *IEEE Transactions on Information Forensics and Security*, 2024.

[25] Yinbin Miao, Ziteng Liu, Xinghua Li, Meng Li, Hongwei Li, Kim-Kwang Raymond Choo, and Robert H Deng. Robust asynchronous federated learning with time-weighted and stale model aggregation. *IEEE Transactions on Dependable and Secure Computing*, 2023.

[26] Luis Muñoz-González, Kenneth T Co, and Emil C Lupu. Byzantine-robust federated machine learning through adaptive model averaging. *arXiv preprint arXiv:1909.05125*, 2019.

[27] John Nguyen, Kshitiz Malik, Hongyuan Zhan, Ashkan Yousefpour, Mike Rabbat, Mani Malek, and Dzmitry Huba. Federated learning with buffered asynchronous aggregation. In *International Conference on Artificial Intelligence and Statistics*, pages 3581–3607. PMLR, 2022.

[28] Xiaoyi Pang, Zhibo Wang, Zeqing He, Peng Sun, Meng Luo, Ju Ren, and Kui Ren. Towards class-balanced privacy preserving heterogeneous model aggregation. *IEEE Transactions on Dependable and Secure Computing*, 20(3):2421–2432, 2023.

[29] Jungwuk Park, Dong-Jun Han, Minseok Choi, and Jaekyun Moon. Sageflow: Robust federated learning against both stragglers and adversaries. *Advances in neural information processing systems*, 34:840–851, 2021.

[30] Yumeng Shao, Jun Li, Long Shi, Kang Wei, Ming Ding, Qianmu Li, Zengxiang Li, Wen Chen, and Shi Jin. Robust model aggregation for heterogeneous federated learning: Analysis and optimizations. *arXiv preprint arXiv:2405.06993*, 2024.

[31] Virat Shejwalkar and Amir Houmansadr. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning. In *Network and Distributed Systems Security (NDSS) Symposium*, 2021.

[32] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *The 2011 international joint conference on neural networks*, pages 1453–1460. IEEE, 2011.

[33] Dimitris Stripelis, Paul M Thompson, and José Luis Ambite. Semi-synchronous federated learning for energy-efficient training and accelerated convergence in cross-silo settings. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(5):1–29, 2022.

[34] Peng Sun, Haoxuan Che, Zhibo Wang, Yuwei Wang, Tao Wang, Liantao Wu, and Huajie Shao. Pain-fl: Personalized privacy-preserving incentive for federated learning. *IEEE Journal on Selected Areas in Communications*, 39(12):3805–3820, 2021.

[35] Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*, 2019.

[36] Vale Tolpegin, Stacey Truex, Mehmet Emre Gursoy, and Ling Liu. Data poisoning attacks against federated learning systems. In *European Symposium on Research in Computer Security*, pages 480–501. Springer, 2020.

[37] Hongyi Wang, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy-yong Sohn, Kangwook Lee, and Dimitris Papailiopoulos. Attack of the tails: Yes, you really can backdoor federated learning. *Advances in Neural Information Processing Systems*, 33:16070–16084, 2020.

[38] Ning Wang, Yang Xiao, Yimin Chen, Yang Hu, Wenjing Lou, and Y Thomas Hou. Flare: defending federated learning against model poisoning attacks via latent space representations. In *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, pages 946–958, 2022.

[39] Qiyuan Wang, Qianqian Yang, Shibo He, Zhiguo Shi, and Jiming Chen. Asyncfeded: Asynchronous federated learning with euclidean distance based adaptive weight aggregation. *arXiv preprint arXiv:2205.13797*, 2022.

[40] Zhibo Wang, Kaixin Liu, Jiahui Hu, Ju Ren, Hengchang Guo, and Wei Yuan. Attrleaks on the edge: Exploiting information leakage from privacy-preserving co-inference. *Chinese Journal of Electronics*, 32(1):1–12, 2023.

[41] Zhongyu Wang, Zhaoyang Zhang, Yuqing Tian, Qianqian Yang, Hangguan Shan, Wei Wang, and Tony QS Quek. Asynchronous federated learning over wireless communication networks. *IEEE Transactions on Wireless Communications*, 21(9):6961–6978, 2022.

[42] Wentai Wu, Ligang He, Weiwei Lin, Rui Mao, Carsten Maple, and Stephen Jarvis. Safa: A semi-asynchronous protocol for fast federated learning with low overhead. *IEEE Transactions on Computers*, 70(5):655–668, 2020.

[43] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[44] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. Generalized byzantine-tolerant sgd. *arXiv preprint arXiv:1802.10116*, 2018.

[45] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance. In *International Conference on Machine Learning*, pages 6893–6901. PMLR, 2019.

[46] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Zeno++: Robust fully asynchronous sgd. In *International Conference on Machine Learning*, pages 10495–10503. PMLR, 2020.

[47] Jiarong Yang, Yuan Liu, Fangjiong Chen, Wen Chen, and Changle Li. Asynchronous wireless federated learning with probabilistic client selection. *IEEE Transactions on Wireless Communications*, pages 1–1, 2023.

[48] Ming Yang, Hang Cheng, Fei Chen, Ximeng Liu, Meiqing Wang, and Xibin Li. Model poisoning attack in differential privacy-based federated learning. *Information Sciences*, 630:158–172, 2023.

[49] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*, pages 5650–5659. PMLR, 2018.

[50] Linlin You, Sheng Liu, Yi Chang, and Chau Yuen. A triple-step asynchronous federated learning mechanism for client activation, interaction optimization, and aggregation enhancement. *IEEE Internet of Things Journal*, 9(23):24199–24211, 2022.

[51] Yu Zang, Zhe Xue, Shilong Ou, Lingyang Chu, Junping Du, and Yunfei Long. Efficient asynchronous federated learning with prospective momentum aggregation and fine-grained correction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 16642–16650, 2024.

[52] Yu Zhang, Morning Duan, Duo Liu, Li Li, Ao Ren, Xianzhang Chen, Yujuan Tan, and Chengliang Wang. Csafl: A clustered semi-asynchronous federated learning framework. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–10. IEEE, 2021.

[53] Yu Zhang, Duo Liu, Moming Duan, Li Li, Xianzhang Chen, Ao Ren, Yujuan Tan, and Chengliang Wang. Fedmds: An efficient model discrepancy-aware semi-asynchronous clustered federated learning framework. *IEEE Transactions on Parallel and Distributed Systems*, 34(3):1007–1019, 2023.

[54] Chendi Zhou, Hao Tian, Hong Zhang, Jin Zhang, Mianxiong Dong, and Juncheng Jia. Tea-fed: time-efficient asynchronous federated learning for edge computing. In *Proceedings of the 18th ACM International Conference on Computing Frontiers*, pages 30–37, 2021.

[55] Zihao Zhou, Yanan Li, Xuebin Ren, and Shusen Yang. Towards efficient and stable k-asynchronous federated learning with unbounded stale gradients on non-iid data. *IEEE Transactions on Parallel and Distributed Systems*, 33(12):3291–3305, 2022.

[56] Hongbin Zhu, Junqian Kuang, Miao Yang, and Hua Qian. Client selection with staleness compensation in asynchronous federated learning. *IEEE Transactions on Vehicular Technology*, 72(3):4124–4129, 2023.

[57] Hongbin Zhu, Miao Yang, Junqian Kuang, Hua Qian, and Yong Zhou. Client selection for asynchronous federated learning with fairness consideration. In *2022 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 800–805, 2022.

[58] Hongbin Zhu, Yong Zhou, Hua Qian, Yuanming Shi, Xu Chen, and Yang Yang. Online client selection for asynchronous federated learning with fairness consideration. *IEEE Transactions on Wireless Communications*, 22(4):2493–2506, 2022.

Appendices

A More Details about Experiments

In the following, we introduce more details about the experiments.

A.1 Model architecture

The details of architecture CNN-2 and CNN-STN are shown in Table 5 and Table. 6, respectively.

Table 5: Model architecture of CNN-2.

Layer Type	Parameters
Convolution + ReLU	3 x 3 x 128
Dropout	0.2
AvgPooling	2 x 2
Convolution + ReLU	3 x 3 x 256
Dropout	0.2
FC	2304 x 10

Table 6: Model architecture of CNN-STN.

	Layer Type	Parameters
STN Layer	Localization Network Output	10 * 4 * 4
	FC1	160 x 32
	FC2	32 x 6
CNN Layers	Convolution + BN + ReLU	3 x 5 x 100
	MaxPooling	2 x 2
	Convolution + BN + ReLU	100 x 3 x 150
	MaxPooling	2 x 2
	Convolution + BN + ReLU	150 x 3 x 250
	MaxPooling	2 x 2
	FC3	1000 x 350
	FC4	350 x 43

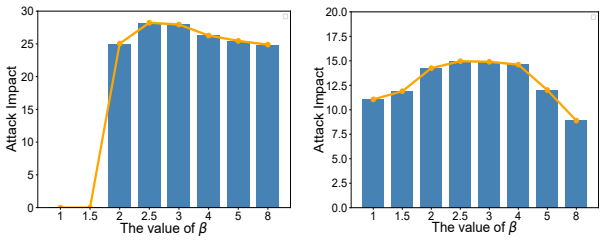
A.2 Effect of the hyperparameter β

β ($\beta \geq 1$) relaxes the loss constraint for malicious model crafting in Eq. (5). Figure. A.2 shows PoiSAFL's attack impact under Sageflow defense when changing β from 1 to 8. We can observe the initial increase and subsequent decrease in the attack impact as the value of β increases from 1 to 8. This can be explained as follows. When the β value is small, the loss constraint for malicious models becomes strict, i.e., the loss of malicious models should be small, leading to a low misclassification probability. In this case, malicious models may demonstrate weak poisonousness and cannot significantly degrade the global model performance. On the contrary, when the value of β is large, the loss constraint becomes loose, leading to poor performance of malicious models. Then, malicious models become easily detectable for performance-based defenses and cannot demonstrate a significant negative impact on the global model as expected. In summary, when β is too small or too large, i.e., the loss constraint is too tight or too loose, it becomes challenging to train malicious model to find a balance point between the negative impact and undetectability within the maximum training rounds. Then, based on our experimental results in Figure. A.2, we set $\beta = 2.5$ in our experiments to achieve the trade-off between the attack effectiveness and stealthiness.

A.3 Compared attacks

The details of the compared attacks are as follows.

- ***LabelFlip*** [36] is a typical data poisoning attack, which flips the label of malicious clients' local training data to construct poisoned data.
- ***SignFlip*** [19] is a typical model poisoning attack that



(a) Fashion MNIST (IID) (b) GTSRB (Non-IID)

Figure 8: Effect of β on PoiSAFL's attack impact when the server deploys Sageflow defense.

flips the signs and enlarges the magnitudes of the parameters of all malicious clients' local models.

- ***LIE*** [2] is a typical model poisoning attack, which adds noise to the average of benign gradients to construct malicious local updates.
- ***LA*** [10] seeks to create malicious model updates that maximize the discrepancy between aggregated model updates before and after the attack under various statistical similarity-based defenses.
- ***MinSum*** [31] crafts malicious local updates to ensure that the sum of distances between the malicious model update and all benign model updates is no greater than the sum of distances between any benign model update and the other benign updates, thus evading statistical similarity-based detection.
- ***Grad*** [31] offers a general FL poisoning framework by crafting and sending optimal malicious gradients to compromise the global model while bypassing the detection of some similarity-based defenses.

A.4 Defenses

The details of tested defenses are as follows.

- ***Median*** [44, 49] sorts values of each dimension in buffered local model parameters and uses the median as the aggregation result of this dimension.
- ***TriMean*** [44, 49] sorts all the values of each dimension in buffered local models parameters, removes the largest and smallest r among them, and then computes the average of the remaining values as the aggregated result of this dimension. By default, $r = C$.
- ***NormClip*** [35] first clips each buffered local model to a predetermined norm and then computes the average of the clipped models as the aggregated result. The predefined norm is set as the average norm of the buffered model updates in each global round.
- ***FLARE*** [38] measures the discrepancies between penultimate layer representations (PLRs) of clients' local models using Maximum Mean Discrepancy. It assigns a root score to each client based on the divergence of its PLR from the aggregated PLR of other clients and performs the weighted aggregation based on the root scores.
- ***AFLGuard*** [11] first calculates a reference model update based on a public trusted dataset, and rejects updates whose deviation in direction and magnitude from the reference update is beyond a predefined threshold when performing the global aggregation.
- ***LFR*** [10] removes local models that have large adverse impact on the global model's loss on a public validation dataset before executing the global aggregation.
- ***Efliter*** [29] is the entropy-based model filtering method. It filters out local models with high prediction entropy and aggregates the rest.
- ***Sageflow*** [29] considers both performance and entropy to detect malicious local models. Relying on a trusted public dataset, it first removes local models with high prediction entropy. Then it performs weighted aggregation among the rest of the local models based on their loss. It is a combination of performance-based and entropy-based detection.