Enhancing Model Poisoning Attacks to Byzantine-Robust Federated Learning via Critical Learning Periods

Gang Yan*
University of California Merced
Merced, CA, USA
gyan5@ucmerced.edu

Hao Wang
Stevens Institute of Technology
Hoboken, NJ, USA
hwang9@stevens.edu

Xu Yuan
University of Delaware
Newark, DE, USA
xyuan@udel.edu

Jian Li
Stony Brook University
Stony Brook, NY, USA
jian.li.3@stonybrook.edu

ABSTRACT

Most existing model poisoning attacks in federated learning (FL) control a set of malicious clients and share a fixed number of malicious gradients with the server in each FL training round, to achieve a desired tradeoff between the attack impact and the attack budget. In this paper, we show that such a tradeoff is not fundamental and an adaptive attack budget not only improves the impact of attack $\mathcal{A}$ but also makes it more resilient to defenses. However, adaptively determining the number of malicious clients that share malicious gradients with the central server in each FL training round has been less investigated. This is due to the fact that most existing model poisoning attacks mainly focus on FL optimization itself to maximize the damage to the global model, and largely ignore the impact of the underlying deep neural networks that are used to train FL models. Inspired by recent findings on critical learning periods (CLP), where small gradient errors have irrecoverable impact on model accuracy, we advocate CLP augmented model poisoning attacks $\mathcal{A}$-CLP in this paper. $\mathcal{A}$-CLP merely augments an existing model poisoning attack $\mathcal{A}$ with an adaptive attack budget scheme. Specifically, $\mathcal{A}$-CLP inspects the changes in federated gradient norms to identify CLP and adaptively adjusts the number of malicious clients that share their malicious gradients with the server in each round, leading to dramatically improved attack impact compared to $\mathcal{A}$ by up to 6.85×, with a smaller attack budget. This in turn improves the resilience of $\mathcal{A}$ by up to 2×. Since $\mathcal{A}$-CLP is orthogonal to the attack $\mathcal{A}$, it also crafts malicious gradients by solving a difficult optimization problem. To tackle this challenge and based on our understandings of $\mathcal{A}$-CLP, we further relax the inner attack subroutine $\mathcal{A}$ in $\mathcal{A}$-CLP and design GraSP, a lightweight CLP augmented similarity-based attack. We show that GraSP not only is more flexible but also achieves an improved attack impact compared to the strongest of existing model poisoning attacks.

*The work was done when G. Yan was a PhD student at Binghamton University.

RAID 2024, September 30-October 02, 2024, Padua, Italy

ACM ISBN 979-8-4007-0959-3/24/09
https://doi.org/10.1145/3678890.3678915

CCS CONCEPTS

• **Security and privacy**; • **Computing methodologies** → **Distributed algorithms**;

KEYWORDS

Federated Learning, Model Poisoning Attacks, Critical Learning Periods

ACM Reference Format:
Gang Yan, Hao Wang, Xu Yuan, and Jian Li. 2024. Enhancing Model Poisoning Attacks to Byzantine-Robust Federated Learning via Critical Learning Periods. In *The 27th International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2024), September 30-October 02, 2024, Padua, Italy*. Palazzo della Salute in Padua, Italy, 17 pages. https://doi.org/10.1145/3678890.3678915

1 INTRODUCTION

Federated learning (FL) [39] has emerged as an attractive distributed learning paradigm that leverages a large number of *untrusted clients* to collaboratively learn a global model, with training data on each client. A central server repeatedly coordinates clients and collects their local model updates computed using their local data, aggregates clients' updates using an *aggregation rule*, and finally uses aggregated updates to tune the global model, which is broadcast to a subset of clients at the beginning of each training round.

Unfortunately, FL is susceptible to *poisoning* by malicious clients compromised by an adversary [11, 14, 26, 30, 36, 49, 64], who hampers the global models' accuracy by instructing malicious clients to share malicious gradients with the server. Most existing untargeted model poisoning attacks, such as Fang [18], LIE [6], Min-Sum/Min-Max [48] and MPHM [50], control a set of malicious clients $\mathcal{M}$. In each training round, attack $\mathcal{A}$ crafts the gradients of a fixed number of malicious clients (i.e., a subset of $\mathcal{M}$), and shares their malicious gradients with the central server for global model update.

Tradeoff between the attack impact and the attack budget. However, choosing the number of malicious clients that share malicious gradients[1] with the central server in each FL training round presents a seemingly inherent tradeoff between the ***attack impact*** (measured by the reduction in model accuracy) and the ***attack***

[1]Unless otherwise specified, in the rest of this paper, we refer to "malicious clients" *only* as those that share malicious gradients with the central server for global model update in each FL training round. Such malicious clients are a subset of the total compromised clients controlled by the adversary.

budget (the average number of malicious clients per round). For example, when training a FL model using Multi-krum aggregation rule [10] on CIFAR-10 with AlexNet, Fang, LIE, Min-Sum and Min-Max with an attack budget of 25% of the total clients are 3.75×, 4.2×, 2.7× and 3.8× more impactful than those with an attack budget of 10% of the total clients [48]. However, such attack impact improvements are at the cost of sharing more malicious gradients in each FL training round, which in turn requires the adversary to invoke more malicious clients (see Section 3.4 for details).

This raises a fundamental question:

> *Is this observed tradeoff between the attack impact and the attack budget fundamental?*

In this paper, we show that such a tradeoff is *not* fundamental but a mere artifact of using a *fixed* attack budget throughout the FL training process. In other words, if the attack budget is adaptively tuned, i.e., the number of malicious clients is adaptively tuned over FL training rounds, then both the attack impact and the adversary's resilience can be significantly improved, compared against the case with a fixed number of malicious clients in each FL training round.

The gap between the literature and the practice. However, determining the number of malicious clients in *an adaptive manner* during the training process has been less investigated in the literature. This is due to the fact that most existing model poisoning attacks mainly focus on the FL optimization itself to maximize the distance between benign and malicious clients, and largely ignore *the impact of the underlying deep neural networks* (DNNs) that are used to train the FL models. As a result, existing model poisoning attacks implicitly assume that all FL training phases are equally important, and hence consistently craft gradients of a fixed number of malicious clients in each training round. Unfortunately, this assumption has recently been revealed to be invalid due to the existence of *critical learning periods* (CLP), i.e., the final quality of a DNN model is determined by the first few training rounds, in which deficits such as low quality or quantity of training data cause irreversible model degradation. Notably, this phenomenon has been revealed in the latest series of works in both centralized and federated settings [1, 19, 20, 28, 29, 62, 63]. Given this phenomenon, it is imperative that advanced poisoning attacks evolve to leverage these nuances. By focusing on exploiting the identified vulnerabilities during CLPs, we can unveil FL's susceptibilities as well as contribute to fortifying its defenses against adversarial attacks [3, 12, 16, 43, 64, 67]. This dual focus not only propels technical advancements but also underscores the ethical responsibility to ensure FL's secure deployment in real-world applications.

Our contributions. We build upon these aforementioned works and extend the notion of CLP to model poisoning attacks to Byzantine-robust FL.

1. $\mathcal{A}$-CLP: CLP Aware Model Poisoning Attacks. We attribute the power of adaptive attack budget to the CLP, and advocate *CLP aware model poisoning attacks* ($\mathcal{A}$-CLP), which *merely* augments a model poisoning attack $\mathcal{A}$ with an adaptive scheme for the attack budget (i.e., to determine the number of malicious clients) in each FL training round. Hence, $\mathcal{A}$-CLP is *orthogonal* to attack $\mathcal{A}$ since it does not change how attack $\mathcal{A}$ crafts malicious gradients. Specifically, $\mathcal{A}$-CLP first identifies CLP in an online manner using an easy-to-compute federated gradient norm metric, and then adaptively adjusts the number of malicious clients in each FL training round. We show that a larger attack budget is *only* required during CLP. As a result, $\mathcal{A}$-CLP significantly improves the impact of $\mathcal{A}$ attack itself while maintaining a smaller attack budget on average. This in turn improves the resilience of attack $\mathcal{A}$ and makes it less easier to be defeated by state-of-the-art defenses, such as FLTrust [12], SparseFed [43], cosDefense [16], FLAIR [3] and LeadFL [67].

Extensive experiments on two popular tasks (i.e., image classification and natural language processing) using five real-world datasets (i.e., CIFAR-10, CIFAR-100, MNIST, Fashion-MNIST, and the Shakespeare) across several representative models (i.e., AlexNet, VGG-11, ResNet-18, and LSTM) show that when augmenting the strongest state-of-the-art model poisoning attacks, e.g., Fang, LIE, Min-Sum/Min-Max and MPHM [50], our $\mathcal{A}$-CLP results in up to 6.85× more accuracy reduction compared to $\mathcal{A}$ itself (i.e., without being CLP aware). Moreover, when facing state-of-the-art defensive mechanisms, $\mathcal{A}$-CLP not only sustains efficacy but also enhances the resilience of $\mathcal{A}$ by up to 2×.

The limitation of $\mathcal{A}$-CLP: To achieve the above desired tradeoff, one needs to specify the inner attack subroutine $\mathcal{A}$ in $\mathcal{A}$-CLP. The goal of most existing model poisoning attacks $\mathcal{A}$ is to deviate the global model parameter *the most* towards the inverse of the direction along which the global model parameter would change without being attacked in each FL training round. However, optimizing such a global objective becomes difficult due to highly non-linear constraints, large state space of local models and non-IID local data at each client [37]. As a result, either sub-optimal approximation techniques [18] or a fixed perturbation to malicious gradients is assumed [48], with attack efficiency highly dependent on these artificial hyperparameters. Exacerbating the problem is the fact that full knowledge of the FL central server's aggregation rule is often required; however, the practice is often on the other side since FL platforms can conceal the details and/or the parameters of their Byzantine-robust aggregation rule to protect the security of the proprietary global model.

2. GraSP: CLP Aware Similarity-based Attack. To address the aforementioned limitation and based on our understandings on $\mathcal{A}$-CLP, we further relax the inner attack subroutine (i.e., $\mathcal{A}$) in $\mathcal{A}$-CLP so as to make it be better integrated with the existing of CLP in FL training process via a lightweight similarity-based poisoning attack. This results in *a CLP aware gradient-similarity-based poisoning attack*, dubbed as GraSP. Our key insight is that it is sufficient to *approximate* an inverse direction that deviates the gradient updates of malicious clients based on the proximities between the adversary's local updates, *but not necessarily* the most towards the inverse direction of the global model update as done in most existing model poisoning attacks.

To this end, we adopt a simple cosine similarity as a proximity between clients' gradients, and relax the adversary's goal to compromise a set of malicious clients such that the cosine similarity between after-attacked aggregated gradient and that of before-attack is beyond an *attack threshold* τ. Such a relaxation not only makes GraSP significantly computationally efficient compared to $\mathcal{A}$-CLP, but also ensures that GraSP achieves an improved attack impact by up to 1.4× compared to $\mathcal{A}$-CLP on CIFAR-10, CIFAR-100, MNIST, Fashion-MNIST and the Shakespeare datasets across several models.

A closer look reveals that the flexible attack threshold, rather than a maximal attack, makes $\mathcal{A}$ fit better with the existence of CLP, e.g., GraSP has a larger gradient magnitude than $\mathcal{A}$, especially during CLP. This contributes to the superior performance of GraSP.

We summarize our key contributions as follows:

• We advocate the CLP aware model poisoning attack $\mathcal{A}$-CLP that enables an existing attack $\mathcal{A}$ to adaptively determine the number of malicious clients in each FL training round by identifying CLP via an easy-to-compute federated gradient norm. $\mathcal{A}$-CLP increases $\mathcal{A}$'s effectiveness and resilience without sacrificing the attack budget (See Section 3).

• We further propose GraSP, a CLP aware similarity-based attack, which crafts malicious gradients based on an attack threshold, and hence is more flexible and easy to implement (See Section 4).

2 BACKGROUND

Federated Learning (FL) leverages a large set of clients, denoted as $\mathcal{N} = \{1, \cdots, N\}$, to collaboratively learn a model with decentralized data under the coordination of a central server. Formally, the goal of FL is to solve an optimization problem, which can be decomposed as: $\min_{\mathbf{w}\in\mathbb{R}^d} F(\mathbf{w}) := \sum_{i\in\mathcal{N}} p_i \cdot F_i(\mathbf{w})$, where $F_i(\mathbf{w}) = \frac{1}{|\mathcal{D}_i|}\sum_{\xi\in\mathcal{D}_i} \ell_i(\mathbf{w};\xi)$ is the local loss function associated with client i's dataset $\mathcal{D}_i$, and $p_i = |\mathcal{D}_i|/\sum_i |\mathcal{D}_i|$ is the relative sample size.

Critical Learning Periods. Recent works have revealed that the first few training epochs—known as critical learning periods (CLP)—determine the final quality of a DNN model in centralized learning. During CLP, deficits such as low quality or quantity of training data will cause irreversible model degradation, no matter how much additional training is performed after CLP. The existence of CLP in FL was recently discovered in [62]. However, studying CLP hinged on a costly information metric (e.g., eigenvalues of the Hessian) that emerges after the full training, limiting their practical benefits. We differ from most existing works by developing an easy-to-compute metric to identify CLP during the training process in an online manner, which can be easily leveraged to adaptively determine the attack budget for the adversary.

Poisoning Attacks on FL. FL is vulnerable to various poisoning attacks, which can be categorized into two classes based on the adversary's goal and capabilities. On one hand, an attack can be either *untargeted* [6, 17, 18, 38, 48, 60, 64], *targeted* [7, 15, 52], or *backdoor* [4, 5, 42, 53, 57, 58, 61, 68] based on the goal of the adversary, among which untargeted attacks can completely cripple the global model and hence pose more severe threats to FL. On the other hand, an attack can be either *model* [5, 7, 17, 18, 47, 48, 60, 64] or *data* [27, 40, 47] poisoning based on the capabilities of the adversary, among which model poisoning attacks often achieve higher attack impacts on FL. Therefore, our research focuses on untargeted model poisoning attacks in FL due to their broad and severe impact on model accuracy across all inputs, presenting a greater challenge than targeted attacks. Notably, advanced attacks such as Min-Max/Min-Sum [48] and MPHM [50] successfully bypassed Byzantine-robust algorithms such as Krum [10] and Trimmed-mean [59, 66], underscoring their effectiveness. The emergence of corresponding defenses such as FLTrust [12], SparseFed [43], and LeadFL [67] against these threats marks a pivotal shift in research towards tackling these security issues, highlighting the pressing need for effective countermeasures.

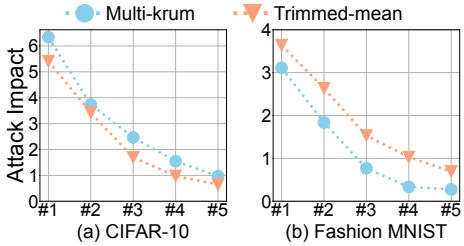

Figure 1: FL under model poisoning attacks exhibits CLP, where the Min-Max attack occurs in (#1) rounds 0-20; (#2) rounds 20-40; (#3) rounds 40-60; (#4) rounds 60-80; and (#5) rounds 80-100, respectively.

Adversary's Goal is to craft malicious gradients such that the accuracy of the global model reduces indiscriminately, i.e., on any test input [8, 9, 27, 35, 46, 56, 65]. This is also known as untargeted model poisoning attack.

Adversary's Capability. The adversary has control over M out to N total clients, which is assumed to be less than 50%, i.e., $M/N < 50\%$ [18, 48]; otherwise, no Byzantine-robust aggregation rule will be able to defeat poisoning attacks. Following previous works [5–7, 18, 24, 48, 59], the adversary can access the global model parameters in each round and directly craft the gradients on malicious clients. In each FL training round, the adversary invokes M' out of M malicious clients, and the central server selects n out of N clients for model update, among which m are malicious.

Adversary's Knowledge can be characterized along two dimensions: aggregation rule and gradient updates shared by benign devices. In particular, many previous works [6, 18, 48, 59] assume full access to both knowledge, which has limited practical significance. For example, to protect the security of proprietary global models, FL platforms can conceal details and/or parameters of their robust aggregation rule, and hence the assumption of full knowledge of aggregation rule is not realistic. Instead, we consider a more practical and challenging setting where the adversary is *agnostic* to the aggregation rule and gradient updates shared by benign clients, i.e., the adversary only knows gradient updates on malicious clients. Since the adversary does not know the aggregation rule, we need to manipulate local updates for malicious clients based on a certain aggregation rule. To our best knowledge, only [48] considered a similar setting as ours.

3 $\mathcal{A}$-CLP: CLP AWARE MODEL POISONING ATTACKS

In this section, we advocate CLP aware model poisoning attack $\mathcal{A}$-CLP, which is *orthogonal* to attack $\mathcal{A}$. Unlike altering the method of generating malicious gradients by $\mathcal{A}$, $\mathcal{A}$-CLP enhances $\mathcal{A}$ with an adaptive approach to dynamically adjust the number of malicious clients $m(t)$ for each round t, moving away from the static approach of a constant $m(t) \equiv m$ for all rounds.

3.1 Model Poisoning Attacks to FL Exhibit CLP

We first show that FL under model poisoning attacks indeed exhibits CLP. For illustration, we leverage one of the strongest model

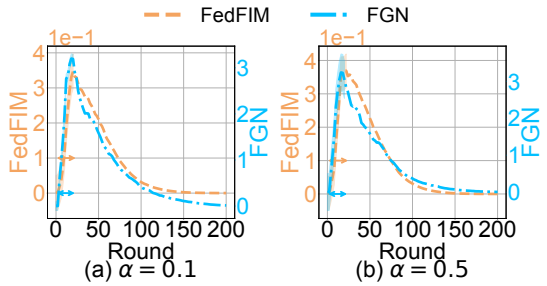

Figure 2: Detecting CLP via FGN and FedFIM, where the shade and double-arrows indicate identified CLP.

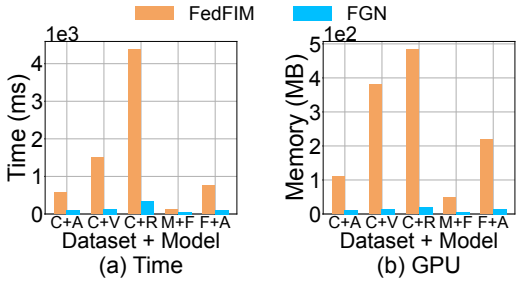

Figure 3: Computation time and memory consumption of FGN and FedFIM approach to detect CLP.

poisoning attacks, i.e., Min-Max when the underlying aggregation rules are Multi-krum [10] and Trimmed-mean [59, 66] using non-IID partitioned CIFAR-10 and Fashion-MNIST datasets. Inspired by [1, 62], we consider five cases that Min-Max attack *only* occurs in rounds (#1) 0-20; (#2) 20-40; (#3) 40-60; (#4) 60-80; and (#5) 80-100. We consider a system with $N = 128$ clients and the adversary controls $M = 32$ clients. In each round, the server randomly selects $n = 32$ clients to participate in global model update, and the adversary invokes $M' = 14$ malicious clients to guarantee that $m = 4$ malicious clients are selected among the $n = 32$ clients on average. Detailed parameter settings are discussed in Section 3.5. Figure 1 reports attack impacts of Min-Max affected by rounds where the attack occurs. All results consistently endorse that FL under model poisoning attacks exhibits CLP: if the attack does not occur in early training phases, its attack impact is significantly degraded. For instance, when the attack occurs in rounds 0-20 (i.e., #1), the attack impact is 5.5 under Trimmed-mean aggregation rule, while there is almost no attack impact if that occurs in round 80-100 (i.e., #5). Similar results hold for other attacks and hence are omitted here.

3.2 Identifying Critical Learning Periods

Prior works use the changes in eigenvalues of the Hessian or approximating Hessian using Fisher information [1, 29, 62] as an indicator to identify CLP. For example, [62] proposed the federated Fisher information (FedFIM) to identify CLP, which is computationally expensive (see Figure 3 with discussions blow). We deviate from these works by leveraging the computationally efficient federated gradient norm (FGN). Considering the difference in training loss for an individual data sample ξ, let $g(\mathbf{w};\xi) = \frac{\partial}{\partial w}\ell(\mathbf{w};\xi)$ denote the gradient of the loss function evaluated on ξ. After performing a step SGD on this sample, the training loss $\Delta\ell = \ell(\mathbf{w} - \eta g(\mathbf{w};\xi);\xi) - \ell(\mathbf{w};\xi)$ can be approximated by its gradient norm using Taylor expansion, i.e., $\Delta\ell \approx -\eta\|g(\mathbf{w};\xi)\|^2$. As a result, the overall training loss at round t, which we call it as FGN, can be approximated using the weighted average of training loss across all selected clients, i.e., $\text{FGN}(t) = \sum_{i\in\mathcal{N}(t)} \frac{|\mathcal{D}_i|}{\sum_{i\in\mathcal{N}(t)}|\mathcal{D}_i|}\Delta\ell_i(t)$. We then use a simple threshold-based rule to identify CLP: if $\frac{\text{FGN}(t)-\text{FGN}(t-1)}{\text{FGN}(t-1)} \geq \delta$, then the current training round t is in CLP, where δ is the threshold used to declare CLP.

We compare the CLP identified by FGN with that identified by using FedFIM in [62]. When training AlexNet on non-IID CIFAR-10, we observe that these two approaches yield similar results as shown in Figure 2, where the shade and double-arrows indicate identified CLP. However, the FGN approach is much more computationally efficient, i.e., being orders of magnitude faster to compute, as shown in Figure 3, where we implement our attacks in PyTorch [44] on Python 3 with three NVIDIA RTX A6000 GPUs, 48GB with 128GB RAM. Hence, the FGN approach can be easily leveraged for determining the number of malicious clients in each round during FL training process in an online manner.

Algorithm 1 $\mathcal{A}$-CLP: CLP Aware Model Poisoning Attacks

for $t = 0, 1, \cdots, T-1$ **do**
  if $\frac{\text{FGN}(t)-\text{FGN}(t-1)}{\text{FGN}(t-1)} \geq \delta$ **then**
    The adversary invokes a larger number of malicious clients to share malicious gradients (e.g., $2m$) with the central server //More malicious clients during CLP
  else
    A smaller number of malicious clients is invoked to share malicious gradients (e.g., $m/2$) with the central server //Fewer malicious clients after CLP
  end if
end for

3.3 The Design of $\mathcal{A}$-CLP

Per our discussions on CLP, the final model accuracy will be permanently impaired if not enough clients are involved in CLP no matter how much additional training is performed after CLP [62]. Therefore, $\mathcal{A}$-CLP automatically switches between *a larger* (e.g., $2m$) and *a smaller* (e.g., $m/2$) number of malicious clients that attack $\mathcal{A}$ shares their malicious gradients with the server in each round by identifying CLP in FL, given that attack $\mathcal{A}$ without being CLP aware always selects m malicious clients on average in each round throughout the FL training process. Therefore, once the CLP is identified, $\mathcal{A}$-CLP increases the number of malicious clients that $\mathcal{A}$ shares their malicious gradients with the server from m to $2m$, implying that more clients now are being activated to improve the attack impact on the global model during CLP. To save the attack budget, $\mathcal{A}$-CLP changes to share a smaller number (e.g., $m/2$) of malicious gradients after CLP. Algorithm 1 summarizes $\mathcal{A}$-CLP on top of any existing attack $\mathcal{A}$.

From a high-level perspective, $\mathcal{A}$-CLP exploits more malicious clients in the initial training phase than a fixed number of malicious clients for $\mathcal{A}$ itself in each training round, to promptly craft the

M'	Method	$n = 16$	**n=32**	$n = 48$
$m = 0.0625n$	Equation (1)	7	**7**	7
	Simulation	7	**7**	7
$m = 0.125n$	Equation (1)	14	**14**	14
	Simulation	14	**14**	14
$m = 0.25n$	Equation (1)	32	**32**	32
	Simulation	32	**32**	32

Table 1: The number of malicious clients M' invoked by the adversary so as to guarantee that on average m malicious clients are selected by the server.

global model with a higher attack impact since the initial learning phase plays a critical role in FL performance. However, the performance improvement comparison is unfair since more malicious clients are used during CLP. To address this, we decrease the number of malicious clients after CLP. Our empirical results show that this improves the attack budget without hurting the final attack impact. The key point is that more malicious clients should be involved in the global model update in the initial learning phase, and only a smaller number of malicious clients is needed after CLP.

Remark 1. *For simplicity and usability, we set the two numbers in Algorithm 1 to be $2m$ during CLP, and $m/2$ after CLP. However, our design can be generalized to other larger and smaller numbers. See the ablation study in Section 3.5.*

3.4 Feasibility Guarantee for $\mathcal{A}$-CLP

As aforementioned, the sever randomly selects a subset of n clients to participate in the global model update in each round. Note that the adversary cannot interfere the server's client selection. Instead, the adversary can invoke more clients out of the M malicious clients that it controlled in each round. A natural question is that *to ensure that m, $2m$ or $m/2$ malicious clients are among the n clients being selected by the central server, how many malicious clients out of M needed to be invoked by the adversary in each round?*

In the following, we provide a theoretical performance guarantee on the feasibility of $\mathcal{A}$-CLP. Specifically, let M' be the number of evoked malicious clients in each round by the adversary, satisfying $M' \leq M$. The expected number of malicious clients m selected by the server is directly influenced by M'. In the following, we characterize the relations between M' and m. Our key insight is that this problem can be transformed into a *hypergeometric distribution problem* [21, 22]. Specifically, let the number of malicious clients selected by the server at each round be a random variable X, which follows the hypergeometric distribution, i.e., $X \sim H(n, M', N - M + M')$. Our goal is to calculate the expected value of X, satisfying $\mathbb{E}[X] = m$. As $\frac{nM'}{N-M+M'} = m$, we obtain $M' = \frac{(N-M)m}{n-m}$. Since $M' \leq M$, we have $m \leq \frac{nM}{N}$. As M' and m are both integer, we have

$$M' = \left\lceil \frac{(N-M)m}{n-m} \right\rceil, \quad m \leq \left\lceil \frac{nM}{N} \right\rceil. \tag{1}$$

We numerically evaluate the performance of our proposed lightweight method in Equation (1). Let $N = 128$, $n = 32$ and $M = 32$. We consider four random seeds and report the results by averaging over $1,000$ independent runs in Table 1. When $m = 4$ malicious clients are selected, we have $M' = 14$, i.e., the adversary needs to invoke 14 malicious clients out of $M = 32$ clients that it controls. Likewise, when $2m = 8$ (resp. $m/2 = 2$), the adversary needs to invoke $M' = 32$ (resp. $M' = 7$) malicious clients out of $M = 32$ clients that it controls. All these cases are feasible since $M' \leq M = 32$. This validates the feasibility of our design on $\mathcal{A}$-CLP.

3.5 Evaluation of $\mathcal{A}$-CLP

3.5.1 *Experimental Setup.*

• **Datasets.** We consider two tasks: (i) image classification using CIFAR-10, CIFAR-100 [32], MNIST and Fashion-MNIST [34] datasets; and (ii) NLP for next-character prediction on the dataset of *The Complete Works of William Shakespeare* (Shakespeare) [39]. We simulate a heterogeneous partition into N clients by sampling $\boldsymbol{p}_i \sim \text{Dir}_N(\alpha)$, where α is the parameter of the Dirichlet distribution. We choose $\alpha = 0.5$ in our main experiments as done in [13, 18, 54, 55], and will numerically investigate its impact.

• **Models.** For image classification task, we consider four DNN models: AlexNet [33], VGG-11 [51], ResNet-18 [23] and a fully connected network (FC). In particular, we use AlexNet and VGG-11 as the global model architecture for CIFAR-10, ResNet-18 for CIFAR-100, FC for MNIST and AlexNet for Fashion-MNIST, respectively. For NLP task, we train a stacked character-level LSTM language model as in [31, 39]. Note that *our goal is not to achieve the largest attack impact or rates for considered datasets using DNN architectures, but rather to show that augmenting existing attacks $\mathcal{A}$ with CLP via our $\mathcal{A}$-CLP can significantly improve the attack impact of a state-of-the-art poisoning attack $\mathcal{A}$ of the learned DNN classifiers.*

• **Baselines.** We consider five strongest model poisoning attacks in the literature, i.e., Fang [18], LIE [6], Min-Sum/Min-Max [48] and MPHM [50]. As aforementioned, we consider a challenging case that all attacks do not know the benign gradients. Finally, despite data poisoning attack is relatively weaker compared to model poisoning attack (see Section 2), for the sake of completeness, we augment existing data poisoning attack named Label Flipping [18, 61] with CLP to illustrate the importance of CLP awareness.

• **Different CLP augmented schemes.** To illustrate the importance of being CLP augmented and take attack budget into account, we consider four schemes:

▷ **Tradition** (*The original/default scheme*): Attack $\mathcal{A}$ always shares m malicious gradients in all FL training rounds.

▷ **CL** (*The CLP augmented scheme*): As in Algorithm 1, attack $\mathcal{A}$ shares $2m$ malicious gradients with the server for model update in each round during CLP, and $m/2$ malicious gradients after CLP.

▷ **RCL** (*The reverse CLP augmented scheme*): In contrast to **CL**, attack $\mathcal{A}$ shares $m/2$ malicious gradients with the server for model update in each round during CLP, and $2m$ malicious gradients after CLP.

▷ **BC-RCL** (*The budget-constrained RCL scheme*): The total number of clients selected by attack $\mathcal{A}$ is the same as that of **Tradition** throughout the whole FL training process.

Remark 2. *As an illustrative example, the average attack budget per round is shown in Figure 4, where we run AlexNet on non-IID partitioned CIFA-10 over 200 rounds. In **Tradition**, attack $\mathcal{A}$ always selects $m = 4$ malicious clients in each round. In **CL**, attack $\mathcal{A}$ selects $2m = 8$ malicious clients during CLP and $m/2 = 2$ malicious clients afterwards, resulting in an average attack budget of 2.88 malicious*

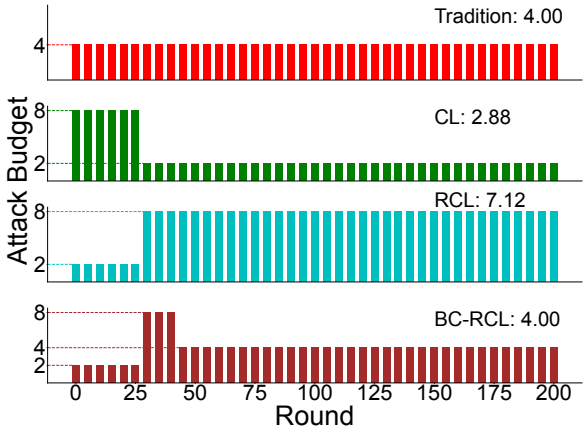

Figure 4: The attack budget: A fixed average attack budget of 4 per round.

*clients per round. Similarly, the average attack budgets under **RCL** and **BC-RCL** are 7.12 and 4, respectively. As we will see later, **CL** improves the attack impact of $\mathcal{A}$ compared to **Tradition** with a lower average attack budget. Likewise, though **RCL** has a higher average attack budget, it even degrades the attack impact of $\mathcal{A}$. This indicates the importance of not only being CLP augmented, but also being augmented in a proper manner.*

• **Parameter settings.** We implement our attacks in PyTorch on Python 3 with three NVIDIA RTX A6000 GPUs. We run each experiment using four different random seeds and report the average results. For ease of presentation, we omit the variances which are observed to be small in our experiments. By default, we consider a total number of $N = 128$ clients in our experiments and the server randomly selects $n = 32$ clients to participate in the global model update in each round. According to feasibility guarantee of $\mathcal{A}$-CLP (see Table 1), the adversary controls $M = 32$ clients, out of which $M' = 32$ are activated during CLP and $M' = 7$ after CLP so that $2m = 8$ and $m/2 = 2$ malicious clients can be chosen by the server in each round before and after CLP, respectively. For attack $\mathcal{A}$, it needs to evoke $M' = 14$ clients in each round so that $m = 4$ malicious clients are selected on average. Each client applies 2 epochs of the stochastic gradient descent to update its local model and the server aggregates local model updates from all selected clients. We set 200 rounds for all models on all datasets considered in this paper. The local learning rate η is initialized as 0.01 and decayed by a constant factor 0.95 after each communication round. The batch size is set to be 16. We set the weight decay to be 10^{-4} and the detection threshold $\delta = 0.01$ in all of our experiments. The Trimmed-mean aggregation rule prunes the largest and smallest β parameters, where $m \leq \beta \leq n/2$. By default, we consider $\beta = 2m$ as that is the default setting in [66].

3.5.2 *Significance of CLP Awareness.*

We evaluate $\mathcal{A}$-CLP in terms of attack impact, attack budget and resilience against defenses for all state-of-the-art attacks $\mathcal{A}$ considered in this paper using different models and aggregation rules. The impacts of $\mathcal{A}$ attack and its corresponding CLP aware attacks are summarized in Table 2. For ease of readability, we focus on showcasing the results of the four most effective model poisoning attacks

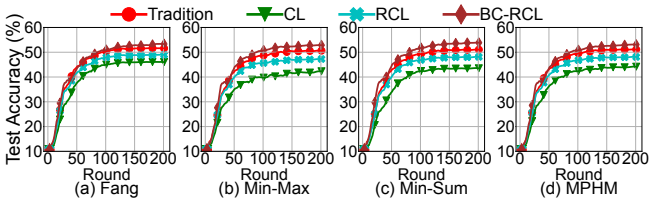

Figure 5: Comparisons of different CLP aware attacks to FL. All attacks *do not know the gradients on benign clients.*

using AlexNet on non-IID partitioned CIFAR-10 with Multi-krum aggregation rule in Figure 5.

• **Improved attack impact.** For any attack $\mathcal{A}$, when augmented by CLP (i.e., **CL** columns in Table 2 and **CL** curves in Figure 5), the attack impact is dramatically improved compared to $\mathcal{A}$ itself (i.e., **Trad.** columns in Table 2 and **Tradition** curves in Figure 5). $\mathcal{A}$-CLP attack is 1.25× to 6.85× more impactful than $\mathcal{A}$ itself. For example, when running AlexNet on non-IID CIFAR-10 with the underlying aggregation rule of Bulyan, Fang has an attack impact of 9.4, while augmenting it with CLP, the impact of Fang-CLP attack is 20.69, i.e., our $\mathcal{A}$-CLP for Fang (i.e., Fang-CLP) is 2.2× more effective than Fang itself. Take AlexNet on non-IID Fashion-MNIST with Multi-krum aggregation rule as another example, the impact of Min-Sum attack is 4.64, while the corresponding impact of Min-Sum-CLP is 12.1, i.e., Min-Sum-CLP is 2.6× impactful than Min-Sum itself without being augmented by CLP. We also notice the effectiveness of Label Flipping attack when augmented by CLP. To be consistent with the main theme of this paper and due to space constraint, we will only focus on model poisoning attacks in the rest of this paper.

• **Importance of properly leveraging CLP.** To further advocate the importance of being CLP aware *in a proper manner*, we consider two variants, i.e., **RCL** and **BC-RCL** in Figure 5. On one hand, **RCL** *only* exhibits a slightly better or even similar attack impact as **Tradition**. In other words, if $\mathcal{A}$ only shares malicious gradients from a smaller number of malicious clients during CLP, it will require $\mathcal{A}$ to share malicious gradients from a much larger number of malicious clients after CLP in order to achieve a slighter better or even similar attack impact as **Tradition**. This finding on the importance of being CLP aware in the FL training process and properly leveraging CLP to adaptively determine the number of malicious clients is consistent with recently reported observations that the initial learning phase plays a key role in determining the outcome of the training process [1, 62].

Further exacerbating the importance of properly leveraging CLP is the fact that **RCL** achieves similar attack performance as **Tradition** at the cost of a significant increase in the attack budget, i.e., a 78% attack budget increase (i.e., average 7.12 vs. 4 attack per round as shown in Figure 4). On the other hand, if we reduce the attack budget, i.e., keeping the total attack budget the same as that of **Tradition**, the impact of **BC-RCL** is significantly worse than **Tradition**. This coincides with the intuition and the functionality of CLP periods in FL training that if the learning models cannot be sufficiently crafted in early training phases, additional attacks cannot improve the attack impact.

• **Necessity of properly leveraging CLP.** Through the discussions presented, it becomes evident that the strategic application of CLP can enhance the efficacy of attacks while adhering to a

Dataset (Model)	Aggregation Rule	No Attack (Accuracy)	Fang		LIE		Min-Max		Min-Sum		MPHM		Label Flipping	
			Trad.	CL	Trad.	CL	Trad.	CL	Trad.	CL	Trad.	CL	Trad.	CL
CIFAR-10 (AlexNet)	Multi-krum [10]	57.57	10.40	**20.02**	5.73	**11.86**	12.03	**26.47**	11.32	**24.37**	11.19	**23.27**	3.23	**7.94**
	Bulyan [17]	56.34	9.40	**20.69**	7.50	**12.99**	7.98	**20.90**	6.53	**16.95**	8.15	**20.73**	4.89	**10.54**
	Trimmed-mean [59, 66]	57.33	10.32	**22.44**	7.36	**17.23**	9.50	**22.85**	8.35	**19.44**	8.91	**21.37**	6.19	**12.05**
	Median [59, 66]	55.46	11.73	**22.62**	10.89	**18.44**	9.10	**20.48**	7.91	**18.44**	9.03	**20.14**	6.95	**13.04**
	AFA [41]	57.89	6.99	**11.81**	2.98	**7.41**	9.27	**19.05**	7.73	**14.83**	8.81	**17.32**	2.01	**5.30**
CIFAR-10 (VGG-11)	Multi-krum [10]	62.63	9.13	**16.03**	6.24	**12.82**	9.94	**17.94**	9.50	**18.07**	9.72	**18.03**	3.12	**4.36**
	Bulyan [17]	63.37	15.16	**22.53**	13.46	**19.56**	14.91	**21.85**	14.54	**21.52**	14.88	**21.69**	7.13	**11.60**
	Trimmed-mean [59, 66]	62.90	11.62	**18.88**	11.20	**17.02**	13.14	**20.89**	10.09	**20.95**	12.53	**20.34**	5.54	**11.08**
	Median [59, 66]	60.13	15.23	**23.58**	12.80	**15.98**	15.05	**23.00**	14.38	**23.34**	14.49	**23.56**	5.44	**7.67**
	AFA [41]	62.75	7.21	**10.58**	6.26	**8.55**	8.54	**11.55**	7.87	**11.09**	8.19	**11.41**	4.43	**6.18**
CIFAR-100 (ResNet-18)	Multi-krum [10]	34.89	17.68	**25.62**	5.33	**11.09**	16.62	**25.53**	10.69	**20.23**	18.49	**25.22**	3.29	**6.02**
	Bulyan [17]	35.21	14.28	**16.61**	8.15	**11.67**	12.58	**19.11**	10.36	**14.93**	13.72	**18.62**	4.65	**8.75**
	Trimmed-mean [59, 66]	35.26	10.01	**18.49**	7.85	**9.41**	10.60	**18.20**	11.17	**19.62**	10.93	**19.34**	5.70	**8.19**
	Median [59, 66]	34.79	12.41	**23.59**	4.97	**9.71**	9.83	**21.18**	9.68	**17.93**	12.37	**23.90**	3.10	**6.84**
	AFA [41]	34.59	9.94	**11.85**	2.05	**6.33**	9.33	**13.70**	8.12	**13.38**	10.13	**13.93**	1.59	**3.67**
MNIST (FC)	Multi-krum [10]	97.02	1.59	**2.06**	0.26	**0.96**	1.51	**2.32**	1.47	**2.25**	1.49	**2.30**	0.04	**0.72**
	Bulyan [17]	97.21	1.36	**1.88**	0.84	**1.18**	1.32	**2.14**	1.23	**2.06**	1.28	**2.09**	0.34	**1.02**
	Trimmed-mean [59, 66]	97.24	1.49	**2.05**	0.24	**0.93**	1.35	**2.28**	1.35	**2.23**	1.32	**2.27**	0.08	**0.62**
	Median [59, 66]	96.93	1.51	**2.03**	0.31	**1.00**	1.31	**2.15**	1.25	**2.12**	1.27	**2.16**	0.08	**0.57**
	AFA [41]	97.20	1.27	**1.70**	0.13	**0.89**	1.28	**2.06**	1.28	**2.08**	1.29	**2.10**	0.02	**0.52**
Fashion MNIST (AlexNet)	Multi-krum [10]	83.24	5.97	**11.05**	3.51	**6.30**	5.06	**15.05**	4.64	**12.10**	5.80	**15.37**	2.08	**2.69**
	Bulyan [17]	83.12	7.79	**20.58**	3.95	**7.42**	6.80	**13.24**	5.51	**12.88**	7.95	**20.34**	1.62	**3.97**
	Trimmed-mean [59, 66]	83.53	6.10	**9.39**	4.46	**11.62**	5.21	**8.75**	4.93	**8.57**	6.02	**11.77**	2.66	**3.42**
	Median [59, 66]	81.81	5.34	**8.88**	5.84	**10.65**	4.27	**8.25**	4.14	**8.72**	5.49	**9.21**	1.23	**2.66**
	AFA [41]	83.97	4.04	**6.46**	2.96	**5.09**	4.91	**9.49**	3.62	**7.57**	4.86	**9.30**	2.26	**3.91**
Shakespeare (LSTM)	Multi-krum [10]	47.14	9.65	**11.94**	2.65	**4.73**	8.80	**11.75**	8.08	**11.07**	8.23	**11.29**	1.68	**3.34**
	Bulyan [17]	46.52	10.38	**13.71**	1.63	**3.48**	8.25	**12.14**	7.71	**11.50**	7.99	**11.60**	1.22	**2.69**
	Trimmed-mean [59, 66]	46.93	9.03	**12.18**	2.23	**3.98**	8.26	**11.12**	7.92	**10.76**	8.04	**10.98**	1.53	**3.26**
	Median [59, 66]	45.76	9.09	**11.53**	1.37	**3.16**	7.45	**10.38**	7.05	**9.96**	7.25	**9.52**	1.05	**2.44**
	AFA [41]	47.41	7.19	**10.14**	4.09	**5.50**	8.58	**10.98**	8.47	**9.91**	8.36	**9.68**	1.43	**2.97**

Table 2: The attack impact for state-of-the-art model poisoning attack $\mathcal{A}$ and the corresponding CLP aware attack $\mathcal{A}$-CLP under various threats using non-IID partitioned datasets when *benign gradients are unknown* to attack $\mathcal{A}$.

Dataset (Model)	Case	Multi-krum				Trimmed-mean			
		Trad.	CL	RCL	BC-RCL	Trad.	CL	RCL	BC-RCL
CIFAR-10 (AlexNet)	Fixed	12.03	**26.47**	18.89	8.23	9.50	**22.85**	14.51	5.60
	Random	12.03	11.48	**27.94**	11.83	9.50	8.92	**23.53**	8.95
	Missing	9.32	19.27	15.38	8.12	8.54	16.51	12.29	7.88
Fashion MNIST (AlexNet)	Fixed	5.06	**15.05**	12.24	4.89	5.21	**8.75**	6.44	5.03
	Random	5.06	4.91	**15.89**	5.57	5.21	4.72	**8.59**	4.88
	Missing	3.13	12.43	10.50	3.19	3.78	6.18	5.23	3.41
Shakespeare (LSTM)	Fixed	8.80	**11.75**	9.47	7.63	8.26	**11.12**	8.92	7.39
	Random	8.80	9.07	**11.54**	8.45	8.26	7.74	**11.43**	8.53
	Missing	6.23	9.81	8.87	5.62	5.69	9.03	8.19	5.05

Table 3: The necessity of properly leveraging CLP: The attack impact under Min-Max attack.

constrained budget. To elucidate the importance of employing CLP judiciously, we examine two scenarios, which underscore the effectiveness of employing fixed-scheme methodologies: (1) **"Random"**. Similar to the budget allocation in Figure 4, we maintain an fixed average attack budget of 4 per round. Different from Figure 4, we now "randomly" allocate the budget in each round for "CL", "RCL", and "BC-RCL" as shown in Figure 6. (2) **Missing**. In practice, the attacker may lose connectivity with the FL training process in some rounds. Specifically, we assume that with probability of 30%, the attacks may not be executed in each training round. As observed from Table 3, CLP-augmented method robustly outperforms the default scheme (without CLP awareness) across all scenarios with a notably high attach impact. This further affirm the validity of our

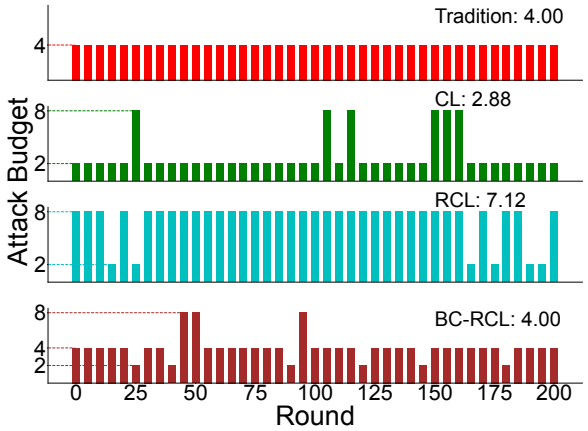

Figure 6: The attack budget is randomly allocated over training rounds.

proposed CLP aware attacks. Finally, we note that in scenarios characterized by randomness, the **RCL** approach emerges as the most effective, attributing to its utilization of the maximum attack budget. This underscores a critical insight: the improper leverage of CLP can significantly diminish performance, reinforcing the necessity for strategic CLP application to optimize outcomes.

Dataset (Model)	Defense	Fang		Min-Max		Min-Sum		MPHM	
		Trad.	CL	Trad.	CL	Trad.	CL	Trad.	CL
CIFAR-10 (AlexNet)	FLTrust	4.74	**10.35**	5.21	**12.79**	5.57	**11.76**	6.13	**13.26**
	SparseFed	6.84	**11.73**	6.32	**12.61**	6.22	**12.46**	7.03	**12.68**
	cosDefense	5.71	**11.40**	6.35	**13.47**	5.96	**12.56**	6.15	**13.22**
	FLAIR	6.57	**12.53**	7.27	**13.81**	8.03	**14.13**	7.53	**14.01**
	LeadFL	6.31	**10.06**	5.12	**9.96**	6.20	**11.49**	6.36	**11.60**
CIFAR-10 (VGG-11)	FLTrust	1.57	**2.85**	1.30	**3.23**	1.74	**2.25**	1.57	**2.77**
	SparseFed	2.71	**4.63**	2.60	**4.42**	2.57	**4.02**	3.01	**4.88**
	cosDefense	3.07	**4.52**	2.42	**4.64**	3.41	**4.55**	3.43	**4.74**
	FLAIR	2.86	**4.25**	3.05	**4.96**	4.01	**5.04**	3.76	**4.79**
	LeadFL	1.31	**2.43**	0.88	**2.65**	0.73	**2.15**	1.18	**2.86**
CIFAR-100 (ResNet-18)	FLTrust	3.10	**4.49**	2.45	**5.19**	2.43	**5.47**	2.99	**5.73**
	SparseFed	3.32	**6.25**	2.64	**5.01**	2.95	**5.39**	3.21	**5.57**
	cosDefense	3.39	**5.42**	4.51	**5.56**	3.62	**5.85**	5.39	**6.45**
	FLAIR	3.38	**4.97**	4.94	**6.25**	5.20	**5.96**	6.25	**7.10**
	LeadFL	1.85	**3.55**	0.97	**3.95**	0.82	**3.72**	2.19	**4.34**
MNIST (FC)	FLTrust	1.33	**1.66**	1.37	**2.10**	1.31	**2.03**	1.39	**2.16**
	SparseFed	1.22	**1.65**	1.12	**1.81**	1.53	**1.79**	1.60	**1.84**
	cosDefense	1.09	**1.78**	1.48	**2.05**	1.20	**1.94**	1.37	**2.00**
	FLAIR	1.28	**1.58**	1.01	**1.93**	1.24	**2.07**	1.18	**2.05**
	LeadFL	1.23	**1.63**	1.07	**1.99**	1.19	**2.08**	1.30	**2.03**
Fashion MNIST (AlexNet)	FLTrust	2.93	**5.80**	3.28	**6.82**	3.82	**7.63**	3.60	**7.21**
	SparseFed	3.25	**5.35**	2.56	**4.41**	2.51	**5.39**	3.06	**4.76**
	cosDefense	3.12	**7.45**	3.36	**7.85**	2.99	**6.77**	3.61	**8.09**
	FLAIR	3.48	**7.32**	3.72	**7.58**	3.92	**7.86**	4.17	**8.13**
	LeadFL	3.34	**5.50**	3.41	**5.85**	2.89	**5.12**	3.34	**5.91**
Shakespeare (LSTM)	FLTrust	4.43	**5.58**	5.41	**7.24**	5.74	**7.05**	5.64	**7.38**
	SparseFed	5.20	**7.25**	6.04	**8.22**	6.38	**8.54**	6.34	**8.81**
	cosDefense	5.31	**6.94**	5.35	**7.78**	6.05	**7.58**	5.78	**7.74**
	FLAIR	6.08	**7.20**	5.97	**6.96**	6.94	**7.75**	5.87	**7.09**
	LeadFL	4.05	**6.57**	5.12	**7.89**	4.39	**7.41**	4.78	**8.16**

Table 4: Attack impacts of $\mathcal{A}$ and $\mathcal{A}$-CLP defended by FLTrust, SparseFed, cosDefense, FLAIR and LeadFL.

• **Improved resilience against defenses.** The benefit of being CLP aware for improved attack impact is reflected in the attack budget as shown in Figure 4. It is clear that $\mathcal{A}$-CLP is more impactful than $\mathcal{A}$ itself with even a smaller number of attack budgets, i.e., $\mathcal{A}$-CLP achieves a higher attack impact and a smaller attack budget at the same time. This in turn improves the resilience of $\mathcal{A}$. For example, FLTrust [12], SparseFed [43], cosDefense [16], FLAIR [3] and LeadFL [67] are five state-of-the-art defenses against model poisoning attacks. We present the attack impact of Fang, Min-Sum, Min-Max and MPHM when they are defended by these defenses under different cases in Table 4. It is clear that being CLP aware significantly improves the attack effectiveness, e.g., in CIFAR-10 using AlexNet, the impact of Min-Sum attack is 5.57% when defended by FLTrust, while the impact of Min-Sum-CLP attack is 11.76%, i.e., the CLP makes Min-Sum 2.11× impactful.

Takeaway 1. *We show that attack $\mathcal{A}$ should be aware of the CLP to determine the number of malicious clients to share malicious gradients with the central server for global model update in each FL training round to avoid the tradeoff between attack impact and attack budget, and hence the vulnerability to defenses. In other words, $\mathcal{A}$-CLP dramatically improves both the attack impact and the vulnerability of $\mathcal{A}$. In addition, CLP should be leveraged in a proper manner, i.e., more malicious clients are only needed during the CLP.*

3.5.3 *Ablation Study.*

• **Sensitivity of CLP detection threshold.** As aforementioned, CLP can be efficiently identified using FGN via a threshold-type rule. We now evaluate the sensitivity of threshold value δ to declare CLP. Figure 7 illustrates the final model accuracy of Fang-CLP, Min-Max-CLP, Min-Sum-CLP and MPHM-CLP attacks to FL with Multi-krum and Trimmed-mean using AlexNet on non-IID partitioned CIFAR-10. It is clear that the CLP declaration determined by δ has an observable effect on the impact of $\mathcal{A}$-CLP. This is because as δ becomes larger, fewer rounds in the initial training phases are declared as CLP. As a result, $\mathcal{A}$-CLP only uses a larger number of malicious clients to participate in the global model update in fewer rounds according to Algorithm 1, and hence the effect of being CLP aware on the attack impact is diminished. However, we observe that even using a large threshold, e.g., $\delta = 0.5$, $\mathcal{A}$-CLP is still more impactful than $\mathcal{A}$ itself. For example, when $\delta = 0.5$ with Trimmed-mean aggregation rule, the impact of Min-Sum-CLP is 14.51% on CIFAR-10, while that of Min-Sum is 8.35%. For ease of simplicity and from Figure 7, we set $\delta = 0.01$ in all of our experiments.

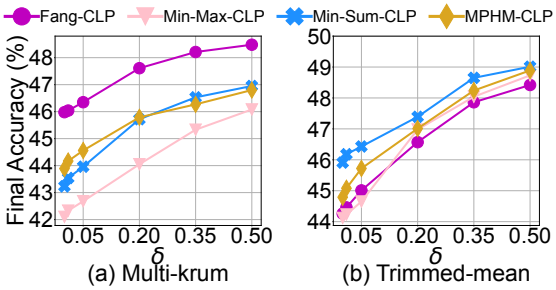

Figure 7: CLP detection threshold δ.

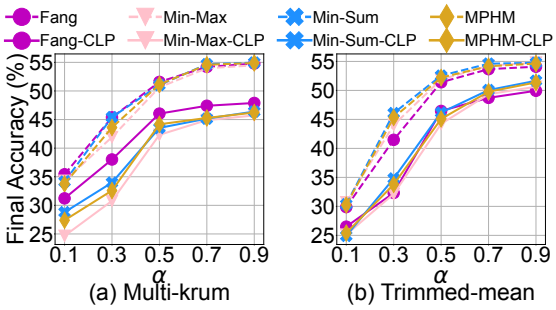

Figure 8: Non-IID degree.

• **Non-IID degrees of data distribution.** We simulate a heterogeneous data partition into N clients using the Dirichlet distribution with parameter α. As observed in Figure 8 using AlexNet on CIFAR10, when the non-IID degree increases, the impacts of Fang-CLP, Min-Max-CLP, Min-Sum-CLP and MPHM-CLP increase. This is quite intuitive since a higher degree of non-IID data makes the Byzantine-robust aggregation rule harder to detect and remove malicious gradients. As a result, the attack crafts more malicious gradients without being detected and hence improves their attack impacts. In addition, we observe that our $\mathcal{A}$-CLP consistently outperforms its counterpart $\mathcal{A}$ across all settings. Without loss of generality, we set $\alpha = 0.5$ in all of our experiments.

• **Participated malicious client number.** Our $\mathcal{A}$-CLP automatically switches between a larger and a smaller number of malicious clients that attack $\mathcal{A}$ shares their malicious gradients with the central server during training. For simplicity, we set the number to be $2m$ and $m/2$ in Algorithm 1 given that $\mathcal{A}$ without being CLP aware always selects m malicious clients in each round. We now vary the larger (resp. smaller) number to be $m, 2m, 3m, 4m$ (resp.

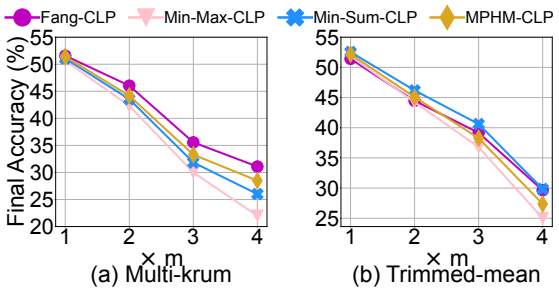

Figure 9: Participated malicious client number.

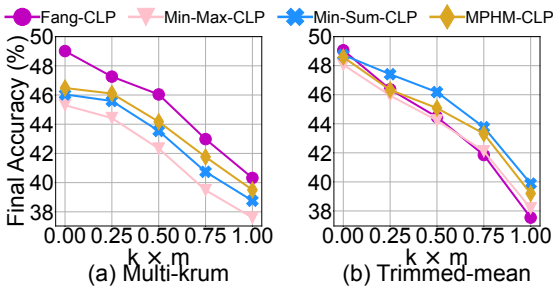

Figure 10: Participated malicious client number after CLP.

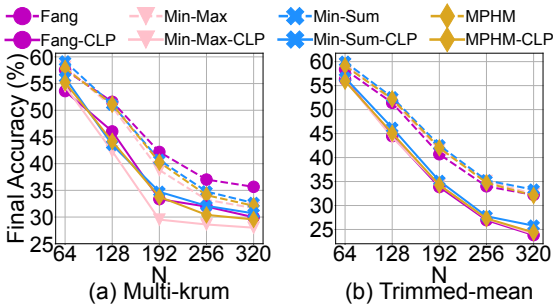

Figure 11: Total number of clients N.

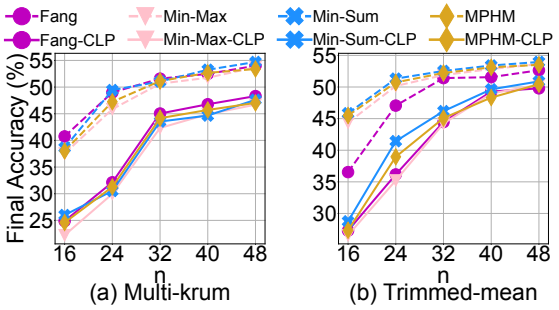

Figure 12: Total number of participants n.

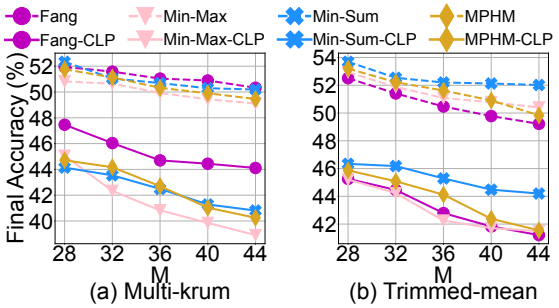

Figure 13: Total number of controlled clients M.

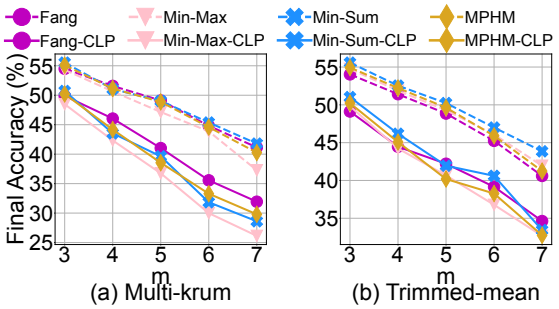

Figure 14: Total number of malicious participants m.

$m, m/2, m/3, m/4$). As shown in Figure 9, as a larger number of malicious clients is selected during CLP, the final accuracy is more severely degraded, which is consistent with above observations (see Takeaway 1). However, as more malicious clients are involved during CLP, the average attack budget is also increased (see Figure 4). To balance the tradeoff between attack impact and attack budget, and for simplicity, we choose $2m$ and $m/2$ in our experiments. To substantiate our selection of $m/2$, we embark on a series of additional experiments. Specifically, we maintain $2m$ as a constant during the CLP, subsequently adjusting the value of $k \cdot m$ within the range from $0 \cdot m$ to $1 \cdot m$. The outcomes of these experiments are illustrated in Figure 10. We observe that as k escalates, the impact of the attack intensifies, albeit at the cost of an increased attack budget. Aiming to strike a balance, we opt for $k = 0.5$ for our experiments.

• **Number of total clients and selected clients.** In the comprehensive series of experiments, we examine a FL configuration involving $N = 128$ total clients, with $n = 32$ being selected for participation. Expanding upon the experimental framework outlined in Section 3.5.1, we explore variations in N and n within a system employing Multi-krum aggregation rules, utilizing AlexNet on a non-IID partitioned CIFAR-10 dataset. Adhering to the established principles for ensuring the practicality of CLP discussed in Section 3.4, as illustrated in Figures 11 and 12, we observe consistent enhancements in performance with CLP-augmented strategies (namely, `Fang-CLP`, `Min-Max-CLP`, `Min-Sum-CLP` and `MPHM-CLP`), which significantly surpass their non-augmented counterparts across varying client populations. Without loss of generality, we set $N = 128$ and $n = 32$, respectively, in our experiments.

• **Number of malicious clients.** Additional experiments are conducted to evaluate the impact of varying M (total controlled clients) and m (selected malicious clients). The results, as depicted in Figures 13 and 14, illustrate a clear trend: as M increases, the effectiveness of the CLP-augmented attacks gradually improves, while Figure 14 shows that the success of CLP-augmented attacks remains consistent. These observations further corroborate the superior performance of CLP-augmented attacks.

4 CLP AWARE SIMILARITY-BASED ATTACK

As discussed in Section 1, the inner attack subroutine (i.e., $\mathcal{A}$) in $\mathcal{A}$-CLP crafts malicious gradients via solving a difficult optimization problem to deviate the global model parameter *the most* towards the inverse of the direction along which the global model parameter would change before-attack. To address this limitation of $\mathcal{A}$-CLP and avoid solving a complex optimization, we relax the inner attack subroutine and propose GraSP, a lightweight CLP aware gradient-similarity-based poisoning attack. For ease of presentation, we present GraSP with all benign gradients known in the design (Section 4.1), which can be easily generalized to the full agnostic case (Remark 3).

4.1 The Design of GraSP

4.1.1 ***Background.*** In the general setting of model poisoning attacks to FL, there is one global optimization goal, which is to maximize the damage to the global model [18, 48]. Specifically, let $s_j(t)$ be the changing direction of the j-th global model parameter in round t when there is no attack, where $s_j(t) = 1$ (resp. $s_j(t) = 1$) means that the j-th global model parameter increases (resp. decreases) upon the previous round. Denote $\mathbf{s}(t) = (s_j(t))_{j=1,\cdots,d}$. Suppose in round t, $\mathbf{w}_i(t)$ (resp. $\mathbf{g}_i(t)$) is the local model (resp. gradient) update that client i tends to send to the central server when there is no attack, and $\tilde{\mathbf{w}}_i(t)$ (resp. $\tilde{\mathbf{g}}_i(t)$) is the local model (resp. gradient) update if client i is compromised. Like most of the existing attacks, e.g., [18, 48], we restrict ourselves to

$$\tilde{\mathbf{w}}_i(t) \coloneqq \mathbf{w}_i(t) - \eta\lambda_i \mathbf{s}_t, \tag{2}$$

which models the deviation between the crafted local model $\tilde{\mathbf{w}}_i(t)$ and the before-attack local model $\mathbf{w}_i(t)$, with $\lambda_i > 0$. Since $\mathbf{w}_i(t) = \mathbf{w}_i(t-1) - \eta\mathbf{g}_i(t)$ and $\tilde{\mathbf{w}}_i(t) = \mathbf{w}_i(t-1) - \eta\tilde{\mathbf{g}}_i(t)$,where $\mathbf{w}_i(t-1)$ is the received latest global model at the beginning of round t, from (2), we have $\tilde{\mathbf{g}}_i(t) = \mathbf{g}_i(t) + \lambda_i \mathbf{s}_t$. The adversary's goal is then to derivate the global model parameter *the most* towards the *inverse of the direction* along which the global model parameter would change without attacks at each round t, i.e., for any aggregation rule $\mathcal{H}(\cdot)$,

$$\max_{\tilde{\mathbf{g}}_1(t),\cdots,\tilde{\mathbf{g}}_m(t)} \mathbf{s}_t^{\intercal}(\mathbf{g}_t - \tilde{\mathbf{g}}_t), \tag{3}$$

$$\text{s.t. } \mathbf{g}(t) = \mathcal{H}(\mathbf{g}_1(t), \cdots, \mathbf{g}_m(t), \mathbf{g}_{m+1}(t), \cdots, \mathbf{g}_n(t)), \tag{4}$$

$$\tilde{\mathbf{g}}(t) = \mathcal{H}(\tilde{\mathbf{g}}_1(t), \cdots, \tilde{\mathbf{g}}_m(t), \mathbf{g}_{m+1}(t), \cdots, \mathbf{g}_n(t)). \tag{5}$$

Given (2), Fang et al. [18] showed that the above optimization problem can be transformed to one with the objective function of $(\lambda_i)_{i=1,\cdots,m}$. However, optimizing such a global objective for $(\lambda_i)_{i=1,\cdots,m}$ becomes difficult due to highly non-linear constraints, large state space of local models and non-IID local data distributions at each client [37]. Below, we first provide intuition behind our attack and then propose a CLP aware similarity-based poisoning attack, GraSP to compromise malicious clients in FL.

4.1.2 ***Intuition.*** Most Byzantine-robust FL aggregation rules are distance-based, i.e., removing gradients that lie outside of the clique formed by benign gradients. In particular, the distances could be from benign gradients [2], or difference between ℓ_p-norms of benign and malicious clients [52], or distributional differences with benign gradients [7]. A natural idea to maximize the performance of the adversary is to ensure that malicious gradients lie within the clique of benign gradients. However, to guarantee such a similarity is *far from* trivial. As discussed earlier, optimizing a complex global objective is often difficult [18]. Instead of solving a complex global optimization problem to determine the changing directions, *why not simply craft the malicious clients' gradients based on the proximities between their local models?*

4.1.3 **GraSP.** Our key insight is that it is sufficient to *approximate* an inverse direction that deviates the malicious gradient updates based on proximities between the adversary's local updates, *but not necessarily* the most towards the inverse direction of global model update as in existing attacks $\mathcal{A}$. The number of such malicious clients in each round is determined by the identified CLP as in Algorithm 1. This naturally leads to two questions: *(i) how to measure the proximity or distance?* and *(ii) how to determine the attack goal of the adversary in each communication round?*

For the choice of measure, the ℓ_p distance has been used as a heuristic between models [48]. However, this often suffers from huge computational overheads due to the large state space of local models. The cosine similarity between gradients calculated by updates of model parameters is an alternative lightweight measure. Specifically, the cosine similarity between gradient updates of any two clients i and i' satisfies

$$\mathcal{F}(\mathbf{g}_i(t), \mathbf{g}_{i'}(t)) \coloneqq \frac{\langle \mathbf{g}_i(t), \mathbf{g}_{i'}(t) \rangle}{\|\mathbf{g}_i(t)\| \cdot \|\mathbf{g}_{i'}(t)\|}. \tag{6}$$

The expectation of $\mathcal{F}(\cdot,\cdot)$ remains asymptotically constant as dimensionality increases [45].

Using this similarity measure, the goal of the adversary boils down to craft the gradients of m^{CLP} malicious clients such that the cosine similarity between the after-attacked aggregated gradient computed by the adversary and that of before-attack is $\tau \in [-1, 1]$, where τ is a system-wide control knob, which the adversary can set to tradeoff between the severity of attacks and possibility to be defensed. The number of malicious clients is $m^{\text{CLP}} = 2m$ if the current round is in CLP, and otherwise $m/2$ as in $\mathcal{A}$-CLP. We call τ as the *attack threshold*. The choice of τ is very much dependent on the adversary, and having τ as an adversary input adds to the "flexibility" of the overall attacking framework and ultimately, shows the wide applicability of our GraSP.

Therefore, for a given attack threshold τ, *the goal of the adversary* is to find changing directions via $\lambda_i, \forall i$ to craft gradients of each of m^{CLP} malicious clients by solving (7),

$$\mathcal{F}(\mathbf{g}(t), \tilde{\mathbf{g}}(t)) = \tau, \tag{7}$$

where $\mathbf{g}(t)$ and $\tilde{\mathbf{g}}(t)$ are given in (4) and (5), and $\tilde{\mathbf{g}}_i(t) = \mathbf{g}_i(t) + \lambda_i \mathbf{s}_t$, $\forall i = 1, \cdots, m^{\text{CLP}}$. One challenge to solve (7) is that the adversary does not know the aggregation rule. To this end, we make one approximation. The attack threshold provides a "flexibility" to the adversary since it does not need to attack towards the most inverse direction by solving a complex optimization problem, and hence our approximation can be treated as part of such a flexibility. As we will demonstrate in experiments, GraSP using such an approximation for all considered aggregation rules can substantially increase the attack impact compared to the strongest state of the arts.

Specifically, we assume that the adversary adopts an "average rule" to approximate the aggregation rule of the server, i.e., $\mathbf{g}(t) \approx$

$\frac{1}{n}\sum_{i=1}^{n}\mathbf{g}_i(t), \tilde{\mathbf{g}}(t) \approx \frac{1}{n}\sum_{i=1}^{n}\tilde{\mathbf{g}}_i(t)$, and $\lambda \triangleq \sum_{i=1}^{m^{\text{CLP}}}\lambda_i$. Then $\tilde{\mathbf{g}}(t) \approx \mathbf{g}(t) + \lambda\mathbf{s}(t)$, from which we can easily solve a so-called "global" λ that is common for all m^{CLP} malicious clients. Formally, we have the following proposition.

Proposition 1. *Suppose that λ is the changing direction to craft gradients of m^{CLP} malicious clients based on the cosine similarity. For any given attack threshold τ, the value of λ is*

$$\lambda = \frac{-z - \sqrt{z^2 - 4xy}}{2x}, \tag{8}$$

where $x = (\boldsymbol{g}(t)^\intercal \boldsymbol{s}(t))^2 - \tau^2\|\boldsymbol{g}(t)\|^2 \cdot \|\boldsymbol{s}(t)\|^2$, $y = (1-\tau^2)\cdot\|\boldsymbol{g}(t)\|^4$, and $z = 2(\tau^2-1)\|\boldsymbol{g}(t)\|^2 \cdot \boldsymbol{g}(t)^\intercal\boldsymbol{s}(t)$.

Given the value of λ in Proposition 1, the adversary determines the changing directions for the aggregated gradient $\tilde{\mathbf{g}}(t)$ of all malicious clients at round t. Since in practical FL systems, clients often have heterogeneous data distributions and system capabilities [11, 25, 30], and hence a heterogeneous changing direction determined by $\lambda_i, \forall i$ is more preferable than a "global" one λ across all malicious clients. To achieve this goal, we first leverage the definition of cosine similarity to obtain $\mathcal{F}(\mathbf{g}(t), \tilde{\mathbf{g}}(t)) = \frac{1}{m^{\text{CLP}}}\frac{\sum_{i=1}^{m^{\text{CLP}}}\langle \mathbf{g}(t), \tilde{\mathbf{g}}_i(t)\rangle}{\|\mathbf{g}(t)\|\cdot\|\tilde{\mathbf{g}}(t)\|}$. Then the changing direction to craft each malicious gradient $\forall i$ can be approximated by $\mathcal{F}(\mathbf{g}(t), \tilde{\mathbf{g}}_i(t)) = \frac{\langle \mathbf{g}(t), \tilde{\mathbf{g}}_i(t)\rangle}{\|\mathbf{g}(t)\|\cdot\|\tilde{\mathbf{g}}(t)\|} \approx \tau$. When combined with $\tilde{\mathbf{g}}_i(t) = \mathbf{g}_i(t) + \lambda_i\mathbf{s}_t$, we can determine $\lambda_i, \forall i$, which is summarized in the following corollary.

Lemma 1. *Suppose that λ_i is the changing direction to craft malicious gradient of the malicious client i, $\forall i = 1, \cdots, m^{CLP}$. Then for any given attack threshold τ, the value of λ_i satisfies*

$$\lambda_i = \frac{\left\langle \boldsymbol{g}(t), \boldsymbol{g}_i(t)\right\rangle - \tau\|\boldsymbol{g}(t)\|\|\tilde{\boldsymbol{g}}(t)\|}{\boldsymbol{g}(t)^\intercal\boldsymbol{s}(t)}, \ \forall i = 1, \cdots, m^{CLP}. \tag{9}$$

Remark 3. *Our* GraSP *can be easily generalized when benign gradients are unknown to the adversary. Since the adversary does not have benign gradients, the changing directions $\mathbf{s}(t), \forall t$ are not known and hence we cannot directly solve for λ_i using (9). However, the before-attack local models on malicious clients are known to the adversary. Similar to [18, 48], we estimate changing directions using the mean before-attack local model of malicious clients. In other words, if the mean of the j-parameter is larger than the j-th global model parameter received from the server in the current round, then $s_j(t)$ is approximated to be* 1*, and otherwise* -1*. Using this approximation, we can obtain the changing directions, which we denote as $\tilde{\mathbf{s}}$, and hence the $\tilde{\lambda}_i, \forall i$ using (9).*

4.2 Evaluation of GraSP

We compare GraSP with state-of-the-art CLP aware attacks (Section 3) when benign gradients are *unknown* (Table 5) to the adversary. We consider the same experimental setup as in Section 3.5. Similar observations hold when benign gradients *known* to the adversary, and hence are omitted here due to space constraint.

• **Attack impacts.** When benign gradients are unknown to the adversary, the impacts of GraSP attack and that of the best among LIE-CLP, Fang-CLP, Min-Max-CLP, Min-Sum-CLP and MPHM-CLP attacks (Table 2), denoted as $\mathcal{A}^*$-CLP, are reported in Table 5. GraSP

Dataset (Model)	Aggregation Rule	$\mathcal{A}^*$-CLP			GraSP		
		Attack Impact	Time (ms)	Memory (MB)	Attack Impact	Time (ms)	Memory (MB)
CIFAR-10 (AlexNet)	Multi-krum	26.47	283.9	95.7	28.55	54.2	94.5
	Bulyan	20.90	264.9	94.3	22.11	52.1	93.2
	Trimmed-mean	22.85	266.0	91.9	24.31	52.7	90.9
	Median	22.62	537.4	107.6	23.60	50.5	98.3
	AFA	19.05	244.0	95.1	20.27	49.9	96.2
CIFAR-10 (VGG)	Multi-krum	18.07	986.7	382.5	20.13	110.7	403.1
	Bulyan	22.53	2134.9	430.5	24.03	118.8	410.4
	Trimmed-mean	20.95	842.4	371.4	22.62	108.1	398.9
	Median	23.58	2203.5	405.7	24.21	113.3	407.2
	AFA	11.55	901.3	392.4	13.27	107.1	415.3
CIFAR-100 (ResNet)	Multi-krum	25.62	1440.2	498.1	27.13	217.7	449.3
	Bulyan	19.11	1099.9	423.5	20.30	219.6	431.3
	Trimmed-mean	19.62	1121.7	419.3	21.35	220.4	428.7
	Median	23.90	1414.6	472.4	24.30	216.8	455.6
	AFA	13.93	1103.8	420.2	14.01	215.8	412.9
MNIST (FC)	Multi-krum	2.32	42.2	15.6	2.86	13.5	17.2
	Bulyan	2.14	43.8	16.2	3.02	13.4	15.8
	Trimmed-mean	2.28	42.1	14.3	2.85	14.1	14.1
	Median	2.16	43.0	17.1	2.72	13.6	16.9
	AFA	2.10	41.8	13.9	2.55	12.6	15.3
Fashion MNIST (AlexNet)	Multi-krum	15.37	212.8	106.9	16.37	43.9	107.8
	Bulyan	20.58	390.2	116.8	21.81	56.6	103.2
	Trimmed-mean	11.77	126.6	76.5	12.41	58.0	98.0
	Median	10.65	116.3	82.4	11.35	50.6	104.3
	AFA	9.49	201.3	98.6	10.68	45.1	102.7
Shakespeare (LSTM)	Multi-krum	11.94	340.6	63.8	12.85	43.6	56.8
	Bulyan	13.71	327.9	59.4	14.46	42.5	53.4
	Trimmed-mean	12.18	329.6	67.2	13.05	43.0	58.1
	Median	11.53	316.4	56.8	12.21	42.6	52.9
	AFA	10.98	256.2	50.5	11.40	43.6	54.7

Table 5: Comparisons of GraSP and $\mathcal{A}^*$-CLP in terms of attack impact and computational complexity under various threat models using non-IID partitioned datasets, when *the benign gradients are unknown.*

is consistently more impactful than the strongest of existing poisoning attacks. For example, GraSP is 1.1× more impactful than $\mathcal{A}^*$-CLP attack for AlexNet and VGG-11 models on non-IID partitioned CIFAR-10. Combined with results in Table 2, GraSP is up to 2.9× more impactful than the strongest state-of-the-art LIE, MIN-Sum and Min-Max attacks. Similarly, GraSP is 1.4× and 1.1× more impactful than $\mathcal{A}^*$-CLP attack for FC model on non-IID MNIST and AlexNet model on non-IID Fashion-MNIST, respectively, and hence is 9.6× and 3.3× more impactful than the strongest state-of-the-art attacks, respectively.

• **Gradient magnitude.** Given the above results, a natural question is why GraSP is much more impactful than $\mathcal{A}^*$-CLP? Recall that existing attacks $\mathcal{A}$ craft malicious gradients via solving a difficult optimization problem to deviate the global model parameter *the most* towards the inverse of the direction along which the global model parameter would change before-attack. This is a two-edged sword. On one hand, it is successful, then it brings the largest attack impact to the global model. On the other hand, this makes it vulnerable and easier to be detected by existing Byzantine-robust aggregation rule, which in turn diminish its attack impacts. In contrast, GraSP relaxes the hard optimization rather than deviating the most inverse directions (see Section 4.1), which becomes harder to be detected and in turn improves the attack impact.

Another reason that leads to the higher attack impact of GraSP is due to its larger gradient magnitude. More precisely, let $\mathbf{g}_i(t)$ be the gradient of i-th client at t-th training round without being attacked, and $\mathbf{g}'_i(t)$ be the modified gradient of i-th client after being attacked.

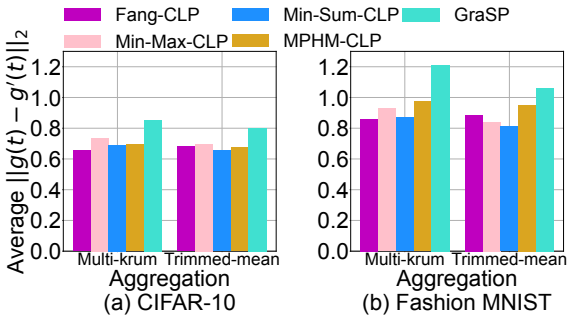

Figure 15: The average ℓ_2-norm of gradient magnitude different of CLP augmented attacks when *benign gradients are unknown* to the adversary.

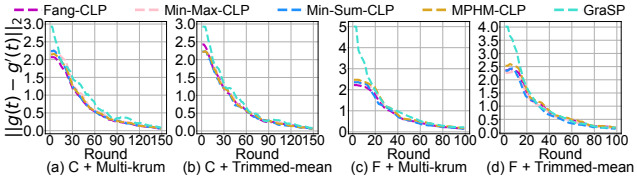

Figure 16: The ℓ_2-norm of gradient magnitude different of CLP augmented attacks when *benign gradients are unknown* to the adversary, where "C" and "F" stands for CIFAR-10 and Fashion-MNIST datasets, respectively.

Consider the ℓ_2-norm of *gradient magnitude difference* between $\mathbf{g}_i(t)$ and $\mathbf{g}'_i(t)$ at round t as $||\mathbf{g}(t) - \mathbf{g}'(t)||_2 = \frac{1}{m}\sum_{i=1}^{m} ||\mathbf{g}_i(t) - \mathbf{g}'_i(t)||_2$. For simplicity, we denote "Average$||\mathbf{g}(t) - \mathbf{g}'(t)||_2$" as the average value over T training rounds. We present the results in Figure 15 and the real-time performance in Figure 16. We observe from Figure 15 that GraSP has a larger gradient magnitude than considered baselines, and more intuitively, these benefits mainly come from the early training phases (CLP) as shown in Figure 16.

• **Computational complexity.** The improved attack impact is further pronounced when we compare the computational complexity of our GraSP and $\mathcal{A}^*$-CLP, which are implemented in PyTorch on Python 3 with three NVIDIA RTX A6000 GPUs, 48GB with 128GB RAM. From Table 5, it is obvious that our GraSP is orders of magnitude faster to compute compared to $\mathcal{A}^*$-CLP with modest memory cost (CPU & GPU). This further validates the limitations of most existing attacks $\mathcal{A}$ and the design motivation of our GraSP based on our understanding of $\mathcal{A}$-CLP. Furthermore, we observe that the computational time is longer when benign gradients are known. This is due to the fact that when benign gradients are known, the adversary needs to deal with more model parameters.

• **Resilience against defenses.** We further compare the resilience of GraSP and $\mathcal{A}^*$-CLP. Table 6 illustrates the attack impact of GraSP and $\mathcal{A}^*$-CLP when defended by FLTrust, SparseFed, cosDefense, FLAIR and LeadFL, respectively. We observe that GraSP is more resilient than $\mathcal{A}^*$-CLP in all settings. For example, GraSP makes FLTrust 1.45× less effective to be defeated than $\mathcal{A}^*$-CLP using VGG-11 on CIFAR-10 with unknown benign gradients.

• **Impact of attack threshold τ.** Our GraSP contains a control knob called the attack threshold τ, which is much dependent on the adversary. We now evaluate the impact of τ on the attack impact of GraSP. For illustration and due to space constraints, we only show results using CIFAR-10 and Fashion-MNIST datasets with the underlying aggregation rules being Multi-krum and Trimmed-mean. As shown in Figure 17, as τ increases, the attack impact decreases. This is quite intuitive and consistent with our design. This is due to the fact that as τ increases, the similarity between after-attacked aggregated gradients and that of before-attack increases, which in turn diminishes the impact of the attack as shown in Lemma 1. Without loss of generality, we set $\tau = 0.1$ in all of our experiments.

Dataset (Model)	Attack	FLTrust	SparseFed	cosDefense	FLAIR	LeadFL
CIFAR-10 (AlexNet)	Best $\mathcal{A}^*$-CLP	13.26	12.68	13.47	14.13	11.60
	GraSP	**13.98**	**14.21**	**14.98**	**15.14**	**13.33**
CIFAR-10 (VGG-11)	Best $\mathcal{A}^*$-CLP	3.23	4.88	4.74	5.04	2.86
	GraSP	**4.68**	**5.60**	**6.42**	**5.89**	**3.98**
CIFAR-100 (ResNet-18)	Best $\mathcal{A}^*$-CLP	5.73	6.25	6.45	7.10	4.34
	GraSP	**6.83**	**7.33**	**8.67**	**7.10**	**6.61**
MNIST (FC)	Best $\mathcal{A}^*$-CLP	2.16	1.84	2.05	2.07	2.08
	GraSP	**2.50**	**2.35**	**3.34**	**2.88**	**2.66**
F. MNIST (AlexNet)	Best $\mathcal{A}^*$-CLP	7.63	5.39	8.09	8.13	5.91
	GraSP	**7.95**	**6.74**	**9.42**	**8.57**	**7.19**
Shakespeare (LSTM)	Best $\mathcal{A}^*$-CLP	7.38	8.81	7.78	7.75	8.16
	GraSP	**8.23**	**10.23**	**9.84**	**8.59**	**9.40**

Table 6: Attack impacts of GraSP and $\mathcal{A}^*$-CLP when defended by FLTrust, SparseFed, cosDefense, FLAIR and LeadFL under various threat models using non-IID partitioned datasets.

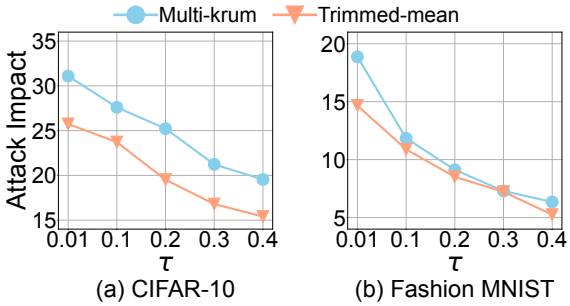

Figure 17: Attack threshold τ.

• **Discussion of defense against proposed methods.** To defend against the proposed attack algorithms, we implement differentiated defense strategies based on CLP detection. This work introduces the FGN metric to detect CLP, which is computationally efficient, as illustrated in Figure 3. The FGN metric can be executed by the server throughout the training process without relying on the client's private data, requiring only the gradient norms collected locally during client training. This approach minimizes communication overhead and enhances privacy protection.

To counter the $\mathcal{A}$-CLP attack, the most straightforward defense is to increase the number of participating clients during CLP, thereby reducing the proportion of malicious clients and mitigating the impact of their updates. However, this method leads to high communication costs and increased attack budgets. For the GraSP attack, defense strategies must consider the attack pattern by employing a layer-based method to calculate the similarity of all client updates at a specific layer, subsequently detecting anomalies. Similar methods have been used in AFA [41] and cosDefense [16]. Based on the detection results at each layer, potential malicious clients are identified. During the detected CLP, a strict strategy is adopted, excluding any potentially risky updates from the model aggregation.

Dataset (Model)	Aggregation Rule	Unknown		Known	
		$\mathcal{A}^*$	GraSP*	$\mathcal{A}^*$	GraSP*
CIFAR-10 (AlexNet)	Multi-krum	12.03	**13.82**	19.88	**21.76**
	Bulyan	9.40	**10.58**	13.61	**15.01**
	Trimmed-mean	10.32	**11.76**	16.43	**17.36**
	Median	11.73	**12.69**	18.31	**19.12**
	AFA	9.27	**10.42**	15.11	**16.88**
CIFAR-10 (VGG)	Multi-krum	9.94	**11.18**	19.43	**21.04**
	Bulyan	15.16	**15.98**	21.00	**22.25**
	Trimmed-mean	13.14	**14.68**	22.24	**23.88**
	Median	15.23	**16.08**	20.02	**21.43**
	AFA	8.54	**9.03**	12.87	**13.66**
CIFAR-100 (ResNet-18)	Multi-krum	18.49	**19.08**	23.61	**24.88**
	Bulyan	14.28	**15.01**	22.63	**24.51**
	Trimmed-mean	11.17	**12.31**	14.49	**16.74**
	Median	12.41	**13.12**	18.45	**19.82**
	AFA	10.13	**11.06**	12.83	**14.43**
MNIST (FC)	Multi-krum	1.59	**1.81**	2.12	**2.31**
	Bulyan	1.36	**1.73**	1.71	**1.95**
	Trimmed-mean	1.49	**1.97**	1.76	**2.13**
	Median	1.51	**1.75**	1.85	**1.97**
	AFA	1.29	**1.51**	1.80	**1.96**
Fashion MNIST (AlexNet)	Multi-krum	5.97	**6.90**	9.43	**10.39**
	Bulyan	7.95	**8.59**	12.59	**13.20**
	Trimmed-mean	6.10	**7.42**	6.74	**7.69**
	Median	5.84	**6.74**	9.47	**10.75**
	AFA	4.91	**5.58**	7.76	**8.59**
Shakespeare (LSTM)	Multi-krum	9.65	**10.58**	13.34	**13.93**
	Bulyan	10.38	**10.89**	12.23	**12.95**
	Trimmed-mean	9.03	**9.92**	10.18	**10.65**
	Median	9.09	**9.64**	10.72	**11.32**
	AFA	8.58	**9.21**	10.10	**10.52**

Table 7: Comparisons of GraSP* (i.e., GraSP without CLP) and $\mathcal{A}^*$ in terms of attack impact under various threat models using non-IID partitioned datasets, where $\mathcal{A}^*$ is the best among baseline attacks reported in Table 2. "Unknown" ("Known") means that benign gradients are unknown (known) to the adversary.

• **Without CLP awareness.** In this paper, we focus on advocating CLP aware attacks, and directly present the results of GraSP. We also evaluate the performance of GraSP without CLP awareness and denote the corresponding method as GraSP*. As shown in Table 7, GraSP* improves the performance of $\mathcal{A}$, however, its benefits are further pronounced when being CLP aware, i.e., GraSP brings much more benefits than GraSP* as shown in Tables 5.

Takeaway 2. *There are two fundamental differences between* GraSP *and existing attacks that contribute to the superior performance of* GraSP. *First, rather than solving a complex optimization problem to maximize the difference in the direction between malicious and benign gradients, a key insight in the design of* GraSP *is that it is sufficient to approximate the largest derivation via an attack threshold. This flexible control knob relaxes the assumptions (see Section 1) needed in state of the arts [18, 48], whose performance largely depend on these hyperparameters. In addition,* GraSP *carefully crafts the gradient of each malicious client (i.e., using different λ_i) due to the practical heterogeneity among FL clients, rather than a single attack across all malicious clients, e.g., [18].*

Second, our GraSP *leverages CLP to adaptively determine the number of malicious clients in each round while existing attacks are agnostic to CLP. Though being CLP aware also significantly improves the impact of these attacks (see Section 3.5),* GraSP *is superior to their $\mathcal{A}$-CLP counterparts. We conjecture that a flexible attack threshold, rather than a maximal attack, fits better with CLP, e.g.,* GraSP *has a larger gradient magnitude especially during CLP (see Figure 15), which contributes to the its superior performance. Building a better theoretical understanding of* GraSP *is our future work.*

5 CONCLUSIONS

In this paper, we advocated CLP aware model poisoning attacks, dubbed as $\mathcal{A}$-CLP. We demonstrated that by augmenting an existing state-of-the-art model poisoning attack $\mathcal{A}$ with the CLP, the attack impact and the resilience can be significantly improved. We further proposed a lightweight CLP aware similarity-based attack GraSP that outperforms the strongest of existing model poisoning attacks by large margins. In the future work, it would be interesting to study CLP aware targeted and backdoor attacks, and to design new CLP aware defenses against CLP aware attacks.

ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation (NSF) grants 2315612, 2315613, 2315614, 2327480 and 2348452. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

REFERENCES

[1] Alessandro Achille, Matteo Rovere, and Stefano Soatto. 2019. Critical Learning Periods in Deep Networks. In *Proc. of ICLR.*
[2] Dan Alistarh, Zeyuan Allen-Zhu, and Jerry Li. 2018. Byzantine stochastic gradient descent. In *Proc. of NeurIPS.*
[3] Joshua Zhao Qiang Qiu Saurabh Bagchi Atul Sharma, Wei Chen and Somali Chaterji. 2023. FLAIR: Defense against Model Poisoning Attack in Federated Learning. In *Proc. of ASIA CCS.*
[4] Eugene Bagdasaryan and Vitaly Shmatikov. 2021. Blind backdoors in deep learning models. In *Proc. of USENIX Security Symposium.*
[5] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. 2020. How to backdoor federated learning. In *Proc. of AISTATS.*
[6] Gilad Baruch, Moran Baruch, and Yoav Goldberg. 2019. A little is enough: Circumventing defenses for distributed learning. In *Proc. of NeurIPS.*
[7] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. 2019. Analyzing federated learning through an adversarial lens. In *Proc. of ICML.*
[8] Battista Biggio, Luca Didaci, Giorgio Fumera, and Fabio Roli. 2013. Poisoning attacks to compromise face templates. In *Proc. of IEEE International Conference on Biometrics (ICB).*
[9] Battista Biggio, Blaine Nelson, and Pavel Laskov. 2012. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389* (2012).
[10] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. 2017. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Proc. of NeurIPS.*
[11] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečnỳ, Stefano Mazzocchi, Brendan McMahan, et al. 2019. Towards federated learning at scale: System design. In *Proc. of MLSys.*
[12] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. 2021. FLTrust: Byzantine-robust Federated Learning via Trust Bootstrapping. In *Proc. of NDSS.*
[13] Xiaoyu Cao and Neil Zhenqiang Gong. 2022. MPAF: Model Poisoning Attacks to Federated Learning based on Fake Clients. *arXiv preprint arXiv:2203.08669* (2022).
[14] Xiaoyu Cao, Jinyuan Jia, Zaixi Zhang, and Neil Zhenqiang Gong. 2023. FedRecover: Recovering from Poisoning Attacks in Federated Learning using Historical Information. In *Proc. of IEEE S&P.*

[15] Zheyi Chen, Pu Tian, Weixian Liao, and Wei Yu. 2021. Towards multi-party targeted model poisoning attacks against federated learning systems. *High-Confidence Computing* (2021).
[16] Tuo Zhang Duygu Nur Yaldiz and Salman Avestimehr. 2023. Secure Federated Learning against Model Poisoning Attacks via Client Filtering. In *Proc of ICLR Workshop*.
[17] Mahdi El El Mhamdi, Rachid Guerraoui, and Sébastien Rouault. 2018. The hidden vulnerability of distributed learning in byzantium. In *Proc. of ICML*.
[18] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. 2020. Local Model Poisoning Attacks to Byzantine-Robust Federated Learning. In *Proc. of USENIX Security*.
[19] Jonathan Frankle, David J Schwab, and Ari S Morcos. 2020. The Early Phase of Neural Network Training. In *Proc. of ICLR*.
[20] Aditya Sharad Golatkar, Alessandro Achille, and Stefano Soatto. 2019. Time Matters in Regularizing Deep Networks: Weight Decay and Data Augmentation Affect Early Learning Dynamics, Matter Little Near Convergence. *Proc. of NeurIPS* (2019).
[21] William C Guenther. 1978. Some remarks on the runs test and the use of the hypergeometric distribution. *The American Statistician* 32, 2 (1978), 71–73.
[22] William L Harkness. 1965. Properties of the extended hypergeometric distribution. *The Annals of Mathematical Statistics* 36, 3 (1965), 938–945.
[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Proc. of IEEE CVPR*.
[24] Lie He, Sai Praneeth Karimireddy, and Martin Jaggi. 2020. Byzantine-robust learning on heterogeneous datasets via resampling. (2020).
[25] Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip Gibbons. 2020. The non-iid data quagmire of decentralized machine learning. In *Proc. of ICML*.
[26] Ahmed Imteaj, Khandaker Mamun Ahmed, Urmish Thakker, Shiqiang Wang, Jian Li, and M Hadi Amini. 2022. Federated learning for resource-constrained iot devices: Panoramas and state of the art. *Federated and Transfer Learning* (2022), 7–27.
[27] Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. 2018. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *Proc. of IEEE S&P*.
[28] Stanislaw Jastrzebski, Devansh Arpit, Oliver Astrand, Giancarlo B Kerg, Huan Wang, Caiming Xiong, Richard Socher, Kyunghyun Cho, and Krzysztof J Geras. 2021. Catastrophic Fisher Explosion: Early Phase Fisher Matrix Impacts Generalization. In *Proc. of ICML*.
[29] Stanislaw Jastrzebski, Zachary Kenton, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos J Storkey. 2019. On the Relation Between the Sharpest Directions of DNN Loss and the SGD Step Length. In *Proc. of ICLR*.
[30] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. 2019. Advances and Open Problems in Federated Learning. *arXiv preprint arXiv:1912.04977* (2019).
[31] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *Proc. of AAAI*.
[32] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning Multiple Layers of Features from Tiny Images. (2009).
[33] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet Classification with Deep Convolutional Neural Networks. *Proc. of NIPS* (2012).
[34] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-Based Learning Applied to Document Recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.
[35] Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. 2016. Data poisoning attacks on factorization-based collaborative filtering. In *Proc. of NIPS*.
[36] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated Optimization in Heterogeneous Networks. In *Proc. of MLSys*.
[37] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2020. On the Convergence of FedAvg on Non-IID Data. In *Proc. of ICLR*.
[38] Saeed Mahloujifar, Mohammad Mahmoody, and Ameer Mohammed. 2019. Universal multi-party poisoning attacks. In *Proc. of ICML*.
[39] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proc. of AISTATS*.
[40] Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C Lupu, and Fabio Roli. 2017. Towards poisoning of deep learning algorithms with back-gradient optimization. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. 27–38.
[41] Luis Muñoz-González, Kenneth T Co, and Emil C Lupu. 2019. Byzantine-robust federated machine learning through adaptive model averaging. *arXiv preprint arXiv:1909.05125* (2019).
[42] Thien Duc Nguyen, Phillip Rieger, Huili Chen, Hossein Yalame, Helen Möllering, and Hossein Fereidooni. 2022. FLAME: Taming backdoors in federated learning. In *Proc. of USENIX Security Symposium*.
[43] Ashwinee Panda and Saeed Mahloujifar. 2022. SparseFed: Mitigating Model Poisoning Attacks in Federated Learning with Sparsification. In *Proc. of AISTATS*.
[44] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS-W*.
[45] Milos Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović. 2010. On the existence of obstinate results in vector space models. In *Proc. of ACM SIGIR*.
[46] Benjamin IP Rubinstein, Blaine Nelson, Ling Huang, Anthony D Joseph, Shing-hon Lau, Satish Rao, Nina Taft, and J Doug Tygar. 2009. Antidote: understanding and defending against poisoning of anomaly detectors. In *Proc. of ACM IMC*.
[47] Subhash Sagar, Chang-Tsun Li, Seng W. Loke, and Jinho Choi. 2023. Poisoning Attacks and Defenses in Federated Learning: A Survey. *arXiv preprint arXiv:2301.05795* (2023).
[48] Virat Shejwalkar and Amir Houmansadr. 2021. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning. In *Proc. of NDSS*.
[49] Virat Shejwalkar, Amir Houmansadr, Peter Kairouz, and Daniel Ramage. 2022. Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning. In *Proc. of IEEE S&P*.
[50] Lei Shi, Zhen Chen, Yucheng Shi, Lin Wei, Yongcai Tao, Mengyang He, Qingxian Wang, Yuan Zhou, and Yufei Gao. 2023. MPHM: Model poisoning attacks on federal learning using historical information momentum. *Security and Safety* (2023).
[51] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-scale Image Recognition. In *Proc. of ICLR*.
[52] Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. 2019. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963* (2019).
[53] Hongyi Wang, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy-yong Sohn, Kangwook Lee, and Dimitris Papailiopoulos. 2020. Attack of the tails: Yes, you really can backdoor federated learning. In *Proc. of NeurIPS*.
[54] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. 2020. Federated Learning with Matched Averaging. In *Proc. of ICLR*.
[55] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. 2020. Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization. *Proc. of NeurIPS* (2020).
[56] Huang Xiao, Battista Biggio, Gavin Brown, Giorgio Fumera, Claudia Eckert, and Fabio Roli. 2015. Is feature selection secure against training data poisoning?. In *Proc. of ICML*.
[57] Chulin Xie, Minghao Chen, Pin-Yu Chen, and Bo Li. 2021. CRFL:Certifiably Robust Federated Learning against Backdoor Attacks. In *Proc. of ICML*.
[58] Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li. 2019. Dba: Distributed backdoor attacks against federated learning. In *Proc. of ICLR*.
[59] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. 2018. Generalized byzantine-tolerant sgd. *arXiv preprint arXiv:1802.10116* (2018).
[60] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. 2020. Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation. In *Proc. of UAI*.
[61] Chulin Xie, Yunhui Long, Pin-Yu Chen, Qinbin Li, Sanmi Koyejo, and Bo Li. 2023. Unraveling the Connections between Privacy and Certified Robustness in Federated Learning Against Poisoning Attacks. In *Proc. of ACM CCS*.
[62] Gang Yan, Hao Wang, and Jian Li. 2022. Seizing Critical Learning Periods in Federated Learning. In *Proc. of AAAI*.
[63] Gang Yan, Hao Wang, Xu Yuan, and Jian Li. 2023. Criticalfl: A critical learning periods augmented client selection framework for efficient federated learning. In *Proc. of ACM SIGKDD*.
[64] Gang Yan, Hao Wang, Xu Yuan, and Jian Li. 2023. Defl: Defending against model poisoning attacks in federated learning via critical learning periods awareness. In *Proc. of AAAI*.
[65] Guolei Yang, Neil Zhenqiang Gong, and Ying Cai. 2017. Fake Co-visitation Injection Attacks to Recommender Systems.. In *Proc. of NDSS*.
[66] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. 2018. Byzantine-robust distributed learning: Towards optimal statistical rates. In *Proc. of ICML*.
[67] Chaoyi Zhu, Stefanie Roos, and Lydia Y. Chen. 2023. LeadFL: Client Self-Defense against Model Poisoning in Federated Learning. In *Proc. of ICML*.
[68] Haomin Zhuang, Mingxian Yu, Hao Wang, Yang Hua, Jian Li, and Xu Yuan. 2023. Backdoor Federated Learning by Poisoning Backdoor-Critical Layers. *arXiv preprint arXiv:2308.04466* (2023).

A APPENDIX: DATASETS AND MODELS

In this study, our exploration spans two pivotal tasks using a variety of datasets and models, as we aim to dissect the intricacies of both image classification and natural language processing (NLP).

For image classification, we engage with the CIFAR-10 and CIFAR-100 datasets [32], each comprising 60,000 color images of

Parameter	Shape	Layer hyper-parameter
layer1.conv1.weight	$3 \times 3, 64$	stride:1; padding: 1
layer1.conv1.bias	64	N/A
batchnorm2d	64	N/A
layer2.conv2	$\begin{bmatrix} 3 \times 3, & 64 \\ 3 \times 3, & 64 \end{bmatrix} \times 2$	stride:1; padding: 1
layer3.conv3	$\begin{bmatrix} 3 \times 3, & 128 \\ 3 \times 3, & 128 \end{bmatrix} \times 2$	stride:1; padding: 1
layer4.conv4	$\begin{bmatrix} 3 \times 3, & 256 \\ 3 \times 3, & 256 \end{bmatrix} \times 2$	stride:1; padding: 1
layer5.conv5	$\begin{bmatrix} 3 \times 3, & 512 \\ 3 \times 3, & 512 \end{bmatrix} \times 2$	stride:1; padding: 1
pooling.avg	N/A	N/A
layer6.fc6.weight	512×100	N/A
layer6.fc6.bias	100	N/A

Table 8: Detailed information of the ResNet-18 architecture used in our experiments. All non-linear activation functions in this architecture are ReLU. The shapes for convolution layers follow (C_{in}, C_{out}, c, c).

Parameter	Shape	Layer hyper-parameter
layer1.conv1.weight	$3 \times 64 \times 3 \times 3$	stride:1; padding: 1
layer1.conv1.bias	64	N/A
pooling.max	N/A	kernel size:2; stride: 2
layer2.conv2.weight	$64 \times 128 \times 3 \times 3$	stride:1; padding: 1
layer2.conv2.bias	128	N/A
pooling.max	N/A	kernel size:2; stride: 2
layer3.conv3.weight	$128 \times 256 \times 3 \times 3$	stride:1; padding: 1
layer3.conv3.bias	256	N/A
layer4.conv4.weight	$256 \times 256 \times 3 \times 3$	stride:1; padding: 1
layer4.conv4.bias	256	N/A
pooling.max	N/A	kernel size:2; stride: 2
layer5.conv5.weight	$256 \times 512 \times 3 \times 3$	stride:1; padding: 1
layer5.conv5.bias	512	N/A
layer6.conv6.weight	$512 \times 512 \times 3 \times 3$	stride:1; padding: 1
layer6.conv6.bias	512	N/A
pooling.max	N/A	kernel size:2; stride: 2
layer7.conv7.weight	$512 \times 512 \times 3 \times 3$	stride:1; padding: 1
layer7.conv7.bias	512	N/A
layer8.conv8.weight	$512 \times 512 \times 3 \times 3$	stride:1; padding: 1
layer8.conv8.bias	512	N/A
pooling.max	N/A	kernel size:2; stride: 2
dropout	N/A	p=20%
layer9.fc9.weight	4096×512	N/A
layer9.fc9.bias	512	N/A
layer10.fc10.weight	512×512	N/A
layer10.fc10.bias	512	N/A
dropout	N/A	p=20%
layer11.fc11.weight	512×10	N/A
layer11.fc11.bias	10	N/A

Table 9: Detailed information of the VGG-11 architecture used in our experiments. All non-linear activation functions in this architecture are ReLU. The shapes for convolution layers follow (C_{in}, C_{out}, c, c).

32×32 pixels across 10 and 100 classes respectively, segmented into

Parameter	Shape	Layer hyper-parameter
layer1.conv1.weight	$3 \times 64 \times 3 \times 3$	stride:2; padding: 1
layer1.conv1.bias	32	N/A
pooling.max	N/A	kernel size:2; stride: 2
layer2.conv2.weight	$64 \times 192 \times 3 \times 3$	stride:1; padding: 1
layer2.conv2.bias	64	N/A
pooling.max	N/A	kernel size:2; stride: 2
layer3.conv3.weight	$192 \times 384 \times 3 \times 3$	stride:1; padding: 1
layer3.conv3.bias	128	N/A
layer4.conv4.weight	$384 \times 256 \times 3 \times 3$	stride:1; padding: 1
layer4.conv4.bias	128	N/A
layer5.conv5.weight	$256 \times 256 \times 3 \times 3$	stride:1; padding: 1
layer5.conv5.bias	256	N/A
pooling.max	N/A	kernel size:2; stride: 2
dropout	N/A	p=5%
layer6.fc6.weight	1024×4096	N/A
layer6.fc6.bias	512	N/A
dropout	N/A	p=5%
layer7.fc7.weight	4096×4096	N/A
layer7.fc7.bias	512	N/A
layer8.fc8.weight	4096×10	N/A
layer8.fc8.bias	10	N/A

Table 10: Detailed information of the AlexNet architecture used in our experiments. All non-linear activation functions in this architecture are ReLU. The shapes for convolution layers follow (C_{in}, C_{out}, c, c).

Parameter	Shape	Layer hyper-parameter
layer1.embeding	80×256	N/A
layer2.lstm	256×512	num_layers=2, batch_first=True
dropout	N/A	p=5%
layer3.fc.weight	512×80	N/A
layer3.fc.bias	80	N/A

Table 11: Detailed information of the LSTM architecture used in our experiments.

Parameter	Shape	Layer hyper-parameter
layer1.fc1.weight	1024×256	N/A
layer1.fc1.bias	256	N/A
layer2.fc2.weight	256×256	N/A
layer2.fc2.bias	256	N/A
layer3.fc3.weight	256×10	N/A
layer3.fc3.bias	10	N/A

Table 12: Detailed information of the FC architecture used in our experiments. All non-linear activation functions in this architecture are ReLU. The shapes for convolution layers follow (C_{in}, C_{out}, c, c).

sets of 50,000 images for training and 10,000 for testing. Additionally, we utilize the Fashion-MNIST and MNIST datasets [34], each containing 60,000 28×28 grayscale training images and 10,000 test images, distributed among 10 classes.

In the domain of NLP, our focus shifts to next-character prediction, employing "The Complete Works of William Shakespeare" dataset [39], which encompasses 734,057 training data points and 70,657 test data points, spread over 74 characters. Through this diverse dataset and model utilization, our work seeks to provide

nuanced insights into the methodologies' effectiveness and adaptability across different challenges.

For classification tasks, we deploy several models accordingly: ResNet-18 [23], VGG-11 [51], and AlexNet [33]. Detailed specifications of these models can be found in Tables 8, 10 and 9. For NLP's next-character prediction, use an LSTM language model, as detailed in Table 11, following the configuration in [31]. The FC model used for the MNIST dataset is described in Table 12. This diverse array of datasets and models provides a robust platform for evaluating our federated learning strategies.

B APPENDIX: DESIGN DETAILS

B.1 Feasibility Guarantee for $\mathcal{A}^*$-CLP

As aforementioned, the central server randomly selects a subset of n out of N clients to participate in the global model update in each round. In our $\mathcal{A}$-CLP framework, m out of n clients should be malicious clients. Then a natural question is that *how many clients in total (denoted as M) should the adversary $\mathcal{A}$ control so that our $\mathcal{A}$-CLP framework is feasible?* In the following, we provide a theoretical performance guarantee on the feasibility of $\mathcal{A}$-CLP. In other words, we determine the so-called control rate M of attack $\mathcal{A}$ such that the event that at least m malicious clients are selected in each round t and hence contribute to the global model update, occurs with a probability p_0, i.e.,

$$\frac{1}{\binom{N}{n}} \sum_{i=m}^{\min(M,n)} \binom{N-M}{n-i}\binom{M}{i} \geq p_0. \tag{10}$$

Unfortunately, (10) is hard to be solved directly due to the computational complexity, especially when N is large. Our key insight is that this problem can be equivalently transformed into a *hypergeometric distribution problem* [21, 22]. Specifically, denote X as a random variable indicating the number of malicious clients selected by the central server at each round, which follows the hypergeometric distribution, i.e., $X \sim H(n, M, N)$, with its mean $\tilde{\mu} = \frac{nM}{N}$ and variance $\tilde{\sigma}^2 = \frac{nM}{N}(1 - \frac{M}{N})\frac{N-n}{N-1}$. When the total number of clients N is large, the hypergeometric distribution can be approximated by the binomial distribution and hence X approximately follows the normal distribution $\psi(\tilde{\mu}, \tilde{\sigma}^2)$ due to the central limit theorem. As a result, the number of selected malicious clients X satisfies

$$\mathbb{P}(X \geq m) = \mathbb{P}\left(\frac{X - \tilde{\mu}}{\tilde{\sigma}} \geq \frac{m - \tilde{\mu}}{\tilde{\sigma}}\right) \geq p_0, \tag{11}$$

where $\frac{X-\tilde{\mu}}{\tilde{\sigma}} \sim \psi(0, 1)$. Therefore, we can obtain M by solving (11), which satisfies

$$M \geq \frac{-b + \sqrt{b^2 - 4ac}}{2a}, \tag{12}$$

where $a = \frac{2n}{N} + (Q(p_0))^2 \frac{(N-n)n}{N^2(N-1)}$, $b = -2\frac{nm}{N} - (Q(p_0))^2 \frac{(N-n)n}{N(N-1)}$, $c = m^2$, and $Q(p_0) = \frac{m-\tilde{\mu}}{\tilde{\sigma}}$ is the quantile of normal distribution. We remark that (12) can be easily solved for any given n, m, N, p_0.

We now numerically evaluate the performance of our proposed lightweight approximated method in Equation (12) to determine the control rate, i.e., M of attack $\mathcal{A}$ for any given n, m, N, p_0. We compare it with the exact results computed from Equation (10), which is order of magnitude complex than our method in Equation (12).

$p_0 = 0.55$	Method	$n = 16$	$n = 24$	$n = 32$
$m = 0.125n$	Equation (10)	15	15	16
	Equation (12)	17	17	17
$m = 0.25n$	Equation (10)	31	31	32
	Equation (12)	33	33	33
$m = 0.375n$	Equation (10)	47	47	48
	Equation (12)	50	49	49

Table 13: Comparison of the control rate M computed by Equation (10) and Equation (12) when $N = 128$ and $p_0 = 0.55$.

$p_0 = 0.55$	Method	$n = 32$	$n = 48$	$n = 64$
$m = 0.125n$	Equation (10)	31	32	32
	Equation (12)	34	33	33
$m = 0.25n$	Equation (10)	63	64	64
	Equation (12)	66	66	65
$m = 0.375n$	Equation (10)	95	96	96
	Equation (12)	98	98	98

Table 14: Comparison of the control rate M computed by Equation (10) and Equation (12) when $N = 256$ and $p_0 = 0.55$.

$p_0 = 0.9$	Method	$n = 16$	$n = 24$	$n = 32$
$m = 0.125n$	Equation (10)	28	26	24
	Equation (12)	31	27	25
$m = 0.25n$	Equation (10)	47	44	42
	Equation (12)	49	45	43
$m = 0.375n$	Equation (10)	64	61	59
	Equation (12)	65	62	59

Table 15: Comparison of the control rate M computed by Equation (10) and Equation (12) when $N = 128$ and $p_0 = 0.9$.

$p_0 = 0.9$	Method	$n = 32$	$n = 48$	$n = 64$
$m = 0.125n$	Equation (10)	49	46	44
	Equation (12)	52	47	44
$m = 0.25n$	Equation (10)	86	82	79
	Equation (12)	87	82	79
$m = 0.375n$	Equation (10)	119	115	112
	Equation (12)	120	115	112

Table 16: Comparison of the control rate M computed by Equation (10) and Equation (12) when $N = 256$ and $p_0 = 0.9$.

For ease of complexity (mainly for computing Equation (10)), we consider two cases: (i) $N = 128$, and the FL central server selects $n = 16$, 24 or 32 clients for global model update in each round; and (ii) $N = 256$, and the FL central server selects $n = 32$, 48 or 64 clients for global model update in each round.

The adversary needs to guarantee m malicious clients are selected with probability p_0. Specifically, we consider the following cases with $m = 0.125n$ to $m = 0.375n$ and $p_0 = 0.55, 0.9$. The number of malicious clients M that the adversary need to control computed by Equation (10) and Equation (12) for the above cases are presented in Tables 13 and 15, and Tables 14 and 16, respectively. It is clear that the results computed by these two methods are quite close to each other, especially when n and m become larger. Hence we use our lightweight method in Equation (12) to determine the control rate M for the adversary in our experiments.

Dataset (Model)	Aggregation	No Attack	Fang		LIE		Min-Max		Min-Sum	
			FedFIM	FGN	FedFIM	FGN	FedFIM	FGN	FedFIM	FGN
CIFAR-10 (AlexNet)	Multi-krum	57.57	20.85	20.02	12.45	11.86	27.13	26.47	24.47	24.37
	Bulyan	56.34	20.72	20.69	13.53	12.99	21.24	20.9	17.04	16.95
	Trimmed-mean	57.33	22.56	22.44	17.37	17.23	23.61	22.85	18.99	19.44
	Median	55.46	23.04	22.62	17.89	18.44	20.25	20.48	17.72	18.44
	AFA	57.89	11.91	11.81	7.56	7.41	18.85	19.05	15.07	14.83
Fashion MNIST (AlexNet)	Multi-krum	83.24	12.40	11.05	5.9	6.3	14.7	15.05	12.38	12.1
	Bulyan	83.12	20.59	20.58	7.68	7.42	13.06	13.24	12.55	12.88
	Trimmed-mean	83.53	9.35	9.39	12.11	11.62	8.77	8.75	9.09	8.57
	Median	81.81	9.33	8.88	10.67	10.65	8.82	8.25	8.54	8.72
	AFA	83.97	6.75	6.46	5.05	5.09	9.13	9.49	7.84	7.57
Shakespeare (LSTM)	Multi-krum	47.14	12.25	11.94	4.93	4.73	12.22	11.75	11.31	11.07
	Bulyan	46.52	13.92	13.71	3.89	3.48	12.53	12.14	12.49	11.50
	Trimmed-mean	46.93	13.55	12.18	3.55	3.98	10.91	11.12	10.84	10.76
	Median	45.76	12.80	11.53	3.73	3.16	9.93	10.38	9.43	9.96
	AFA	47.41	9.96	10.14	5.74	5.50	10.54	10.98	9.47	9.91

Table 17: Attack Impact of $\mathcal{A}$-CLP when leveraging CLP identified by using FedFIM and FGN approaches.

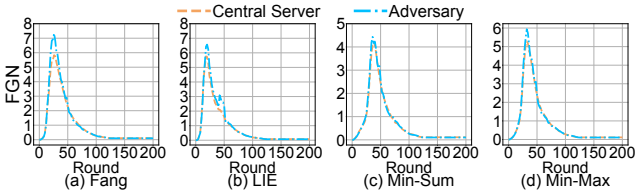

Figure 18: Comparisons of detecting CLP from the perspectives of FL central server and adversary $\mathcal{A}$ using AlexNet on Fashion-MNIST with the Multi-krum aggregation rule.

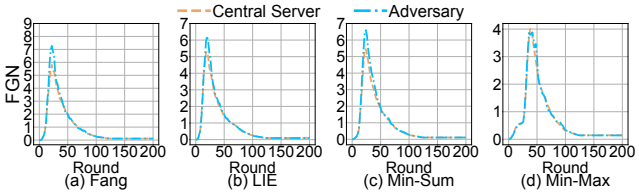

Figure 19: Comparisons of detecting CLP from the perspectives of FL central server and adversary $\mathcal{A}$ using AlexNet on Fashion-MNIST with the Trimmed-mean aggregation rule.

B.2 Robustness of Identifying CLP

We propose a lightweight FGN metric to identify CLP in federated settings in Section 3.3. Our numerical results show that our FGN approach yields similar results on identifying CLP as that using the state-of-the-art FedFIM approach [62] as shown in Figure 2, however, our FGN approach is much more computationally efficient than FedFIM approach as shown in Figure 3. Therefore, we leverage the CLP identified by our FGN approach in the design of $\mathcal{A}$-CLP as in Algorithm 1. In Table 17, we also report the attack impact when $\mathcal{A}$-CLP leverages the CLP identified by the FedFIM approach using CIFAR-10 and Fashion-MNIST datasets. As expected, the attack impacts are similar since the CLP identified by these two approaches are similar.

Note that in Figure 2, we compute the FGN from the perspective of the FL central server, which controls a total of N clients and randomly selects n clients for model update in each FL training round. This is the same setting as in [62] for comparison.

Once the central server identifies the CLP, it may broadcast the information to all clients along with the updated global model at the beginning of each round. For the defense purpose, the central server may not want to share such information with the adversary $\mathcal{A}$, as our $\mathcal{A}$-CLP framework shows that the attack $\mathcal{A}$ can leverage the

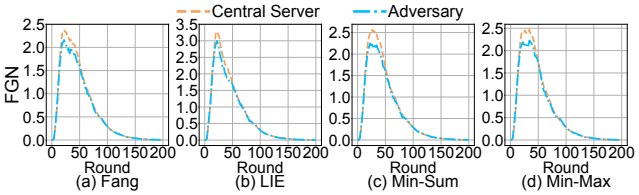

Figure 20: Comparisons of detecting CLP from the perspectives of FL central server and adversary $\mathcal{A}$ using AlexNet on CIFAR-10 with the Multi-krum aggregation rule.

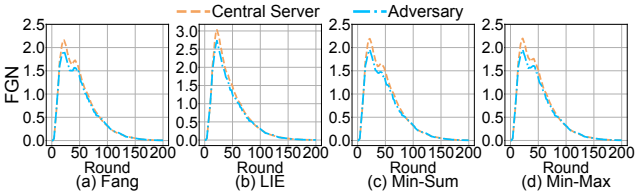

Figure 21: Comparisons of detecting CLP from the perspectives of FL central server and adversary $\mathcal{A}$ using AlexNet on CIFAR-10 with the Trimmed-mean aggregation rule.

CLP information to significantly improve its attack impact. Since the adversary A controls a set of M clients, and shares m malicious gradients with the central server for model update in each round, a natural question is that can the adversary $\mathcal{A}$ also detect the CLP by itself? If so, is the identified CLP the same as that identified by the central server?

Here, we provide affirmative answers to these questions. Specifically, the adversary $\mathcal{A}$ computes the FGN using the information from the set of M clients it controls. For ease of readability, we consider to train AlexNet on Fashion-MNIST dataset where the server uses the Multi-krum or Trimmed-mean aggregation rules.

We evaluate the CLP identified by four attacks considered in this paper. As shown in Figures 18, 19, 20 and 21, where we compute the FGN from both the perspectives of the central server and the adversary $\mathcal{A}$, we observe that the CLPs identified by the central server and the adversary are very similar. Hence, our proposed FGN approach is robust and can be applied to both the FL central server and the adversary.