GRANULARITY BLIND SPOTS IN BYZANTINE-ROBUST FEDERATED LEARNING: SHAPLEY-GUIDED SUBSPACE POISONING OF WHOLE-UPDATE DEFENSES

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated learning (FL) exposes its optimization process to model poisoning by compromised clients. Byzantine-robust aggregators inspect complete updates using distances, coordinate statistics, or alignment with a reference. Because of the resulting mismatch in granularity, non-IID honest updates may appear anomalous, whereas malicious changes in a compact parameter subspace may evade detection. We demonstrate the systematic exploitability of this mismatch by introducing GRASP, a wrapper that identifies influential parameter blocks for each base attack and concentrates malicious perturbations in the corresponding subspace. GRASP ranks blocks in the target model using a scalable second-order Shapley approximation. This approximation captures individual contributions and cooperative effects within each layer without enumerating coalitions. It then injects the selected residual into the reference update and chooses the largest scale within an adaptive Euclidean radius. Across five poisoning strategies and nine Byzantine-robust defenses on MNIST, Fashion-MNIST, and CIFAR-10, GRASP exceeds the base attack in 85 of 95 settings (89.5%). It raises the average Attack Impact from 22.17 to 33.02 points. Under FLAME at a 40% malicious client fraction, it raises the malicious admission rate from 28.09% to 99.80%. These results expose a systematic blind spot in defenses that assess complete updates and motivate robustness mechanisms that operate at a finer granularity.

1 INTRODUCTION

Federated learning (FL) trains a shared model through client-side computation without centralizing raw data (McMahan et al., 2017; Bonawitz et al., 2019). Its update interface, however, gives compromised clients a direct path into global optimization: the server receives submissions but cannot generally verify how they were produced. Model poisoning attacks exploit this opacity by manipulating coordinate statistics, descent directions, or pairwise distances (Xie et al., 2020; Shejwalkar & Houmansadr, 2021), and Byzantine-robust aggregation counters them by suppressing suspicious updates via distances, ranks, or a trusted reference (Blanchard et al., 2017; Cao et al., 2021).

We argue that a structural mismatch underlies the vulnerability of current robust aggregation. Many defenses inspect submissions through *whole-update* geometry (cosine similarity to a reference, Euclidean distance to neighbors, or rank among updates) whereas the *effect* of a poisoning attack need not be distributed across the update. Under non-IID data, honest updates may appear geometrically anomalous despite reflecting legitimate local objectives (Karimireddy et al., 2022), while an adversary can confine malicious changes to a compact parameter subspace and still retain attack strength (Guo et al., 2026). FLAME, for instance, retains the majority cluster by whole-update cosine similarity (Nguyen et al., 2022), implicitly assuming that attack effect is proportional to whole-update geometric deviation. When this assumption fails, an attack can retain substantial effect in a few influential parameter blocks while limiting its whole-update geometric footprint. This is not a flaw of any single defense but a consequence of evaluating a localized phenomenon at the wrong granularity.

Recent defenses have begun to inspect updates more finely, confirming that the mismatch is recognized: LASA introduces layer-level filtering but does not identify malicious parameters within retained layers (Xu et al., 2025), and SkyMask learns parameter-level masks (Yan et al., 2025).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Both operate on the defense side. The complementary problem of selecting parameter blocks to maximize attack effect per unit of geometric deviation remains unformalized. Existing subspace poisoning makes the gap concrete. *Poisoning with a Pill* (Pill) confines iterative poisoning to a compact subnetwork by selecting a connected path and injecting it into an estimated honest update (Guo et al., 2026), but its search starts from random neurons and greedily follows connection weights, so selection depends on local topology, and its injection adjusts cosine similarity and distance to a coordinate-wise mean without a stated criterion for which blocks matter or how far the residual can be scaled before becoming conspicuous.

We formulate selective poisoning as a two-part decision problem and solve both parts. The first part asks which parameter blocks should carry the residual, given a target update; we answer it with a contribution score evaluated on the target model. This model defines the utility, and a second-order Shapley approximation (Chu et al., 2025) attributes to each structured block its individual and within-layer cooperative contribution. This replaces topology-driven path search with a utility-theoretic criterion and makes the selected blocks attack-specific rather than architecture-determined. The second part asks how far the residual can be scaled before becoming geometrically conspicuous; we answer it with a *majority-aware* construction that anchors the residual at the reference update nearest to the target update and sets its scale to the largest value retaining Euclidean support from a strict majority. This replaces heuristic cosine-and-distance adjustment with a one-dimensional feasibility problem that admits an exact interval-overlap solution.

The resulting method, Granularity-aware Residual Attack via Shapley Poisoning (GRASP), is a three-stage wrapper applied after a base attack produces its target. It is attack-agnostic in that it enhances any base attack, but not agnostic to the target update, since block scoring is conditioned on the model that target induces. All malicious clients submit the same adjusted update, which directly tests whether whole-update defenses remain effective under an honest majority. Our contributions are as follows. We identify the mismatch between whole-update defense inspection and localized attack effect as the central mechanism, casting selective poisoning as the joint problem of choosing which blocks carry the residual and how large it can be; this exposes the implicit assumptions of existing subspace attacks and yields an efficiency criterion under which selective poisoning provably dominates whole-update poisoning (Proposition 1). We instantiate this formulation with a second-order Shapley block score on the target model, a nearest-anchor residual, and a majority-support constraint solvable exactly by endpoint sweep, whose returned scale provably maximizes residual amplitude subject to majority support (Theorem 4). Finally, across five base attacks and nine defenses, GRASP exceeds the base attack in 85 of 95 FedSGD settings (89.5%) and raises the attack-averaged Multi-Krum admission rate from 52.9% to 75.1%, transferring across architectures, datasets, and from FedSGD to FedAvg, and persisting under a restricted-reference variant that assumes no honest-update knowledge beyond the base attack itself, locating the failure at the granularity of inspection rather than at any single aggregation rule.

2 Related Work

2.1 Model Poisoning in Federated Learning

Model poisoning attacks manipulate client-submitted updates through direct manipulation (Bhagoji et al., 2019), model replacement (Bagdasaryan et al., 2020), Sybil coordination (Fung et al., 2020), or optimization against robust aggregation (Zhang et al., 2025). A second line exploits aggregator statistics: ALIE hides perturbations within coordinate dispersion (Baruch et al., 2019), inner-product manipulation reverses the descent direction (Xie et al., 2020), and Min-Max and Min-Sum optimize malicious distances under feasibility constraints (Shejwalkar & Houmansadr, 2021). These attacks perturb the update globally and remain exposed to defenses that inspect whole-update geometry. Selective poisoning instead confines the attack to a subspace: *Poisoning with a Pill* selects a connected subnetwork by greedily following cumulative connection weights from random neurons, then injects it into an estimated honest update while heuristically adjusting cosine similarity and distance (Guo et al., 2026). This shows that subspace attacks can evade whole-update defenses, but leaves implicit the relationship between subspace selection, attack behavior, and update geometry: selection follows local topology rather than attack effect, and the injection scale lacks a stated criterion. Rather than proposing another attack heuristic, we formulate selective poisoning as the joint problem of *which* blocks carry the residual and *how large* it can be, and solve both with a stated criterion.

2.2 BYZANTINE-ROBUST AGGREGATION

Robust aggregation limits Byzantine influence without inspecting private data (Lamport et al., 1982; Alistarh et al., 2018). Classical rules operate on whole-update geometry: Krum and Bulyan select or average updates by pairwise distance (Blanchard et al., 2017; El-Mhamdi et al., 2018), refined by Multi-Krum, centered clipping, bucketing, and history-based optimization (Karimireddy et al., 2021; 2022). A parallel line relies on a reference or filter: FLTrust bootstraps trust from a server-held root set (Cao et al., 2021), FLAME clusters models by cosine distance (Nguyen et al., 2022), and FLDetector predicts updates from historical behavior (Zhang et al., 2022). Certified methods bound backdoor effects (Xie et al., 2021), and benchmarks have systematized the landscape (Zhang et al., 2025). A recent thread moves inspection to finer granularity: LASA combines sparsification with layer-wise filtering (Xu et al., 2025), and SkyMask learns parameter-level masks (Yan et al., 2025). Both operate on the defense side and remain agnostic to an attack's internal structure. Their emergence confirms that inspection granularity matters, but leaves open how an adversary can systematically exploit the gap between whole-update inspection and localized attack effect. Our efficiency analysis (Section B) characterizes when such exploitation is beneficial, and our method instantiates it.

2.3 SHAPLEY VALUES FOR MODEL PARAMETERS

Shapley (1953) introduced a value that attributes utility by averaging each player's marginal contribution across coalitions. Because exact computation is combinatorial, attribution work resorts to approximation: Ghorbani & Zou (2020) use sampled coalitions to identify responsible components. Chu et al. (2025) develop a path-integrated second-order approximation with blockwise Fisher information. These methods target *model explanation*, where utility is defined by the model's predictive behavior. We instead score blocks on the model *induced by a base attack*, so utility reflects the attack's realized target, and the ranking determines which blocks carry a poisoning residual. Applying the Shapley estimator to model poisoning is, to our knowledge, new.

2.4 THREAT MODEL

Adversary. The adversary controls $m < n/2$ participating clients and coordinates their submitted updates, including their local data and training. Following prior work on colluding model poisoning (Shejwalkar & Houmansadr, 2021; Xie et al., 2020), GRASP assumes by default an omniscient adversary that knows all client updates. To test dependence on this information, we also evaluate the attacker-local variant that constructs its reference geometry from compromised clients' pre-attack updates, adding no honest-update knowledge beyond the underlying base attack (Appendix D).

Aggregation server. The server follows the FL protocol and may inspect, filter, or reweight client submissions before applying the specified aggregation rule.

3 THE GRASP METHOD

GRASP is a wrapper applied after a base attack produces its target update. We use *base attack* to denote the poisoning method being enhanced (e.g., Sign Flipping, ALIE, or IPM), reserving *GRASP* for the enhancement itself. As shown in Figure 1, GRASP ranks structured blocks on the model induced by the base attack's target, injects the selected residual into an anchor, and selects the largest residual scale supported by a majority. All malicious clients then submit the resulting shared update.

3.1 PROBLEM FORMULATION

We consider FedSGD. At round t, let $w_t \in \mathbb{R}^d$ be the current global parameter vector. Each client submits an additive update in $\mathbb{R}^d$, which the server adds to the current model. A base attack produces either a single target update v shared by all malicious clients, or client-specific candidates $\{v_j^{\mathrm{src}}\}_{j\in\mathcal{M}}$; in the latter case we average the candidates, $v = \frac{1}{m}\sum_{j\in\mathcal{M}} v_j^{\mathrm{src}}$, so that all malicious clients submit a common update. We call $\vartheta_v = w_t + v$ the *target model*, since it is the model that would result from applying the target update to the current global model. Block scoring is evaluated on ϑ_v, whereas reference selection and residual injection operate directly on update vectors in $\mathbb{R}^d$.

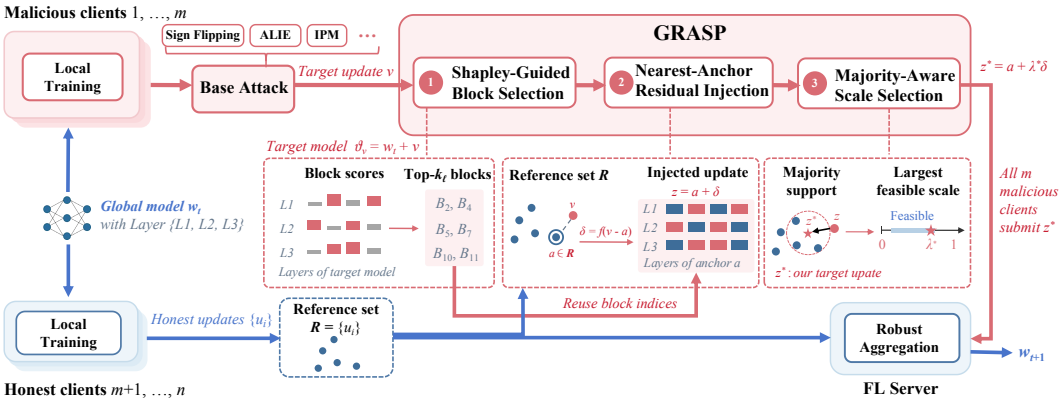


Figure 1: Overview of the GRASP pipeline. Given a base attack's target update, GRASP selects Shapley-ranked blocks per layer and injects their target-to-anchor residual into the nearest reference update. It then chooses the largest residual scale with majority support from the reference set, yielding a target update for all malicious clients.

We call R the *reference set*: the collection of observed updates used to choose an anchor and to calibrate the geometric envelope for residual scaling. Under the default omniscient threat model, R contains all $n - m$ honest updates. The GRASP attacker-local variant instead uses compromised clients' pre-attack updates; its calibration is given in Appendix D.1. We partition each layer into *structured blocks*: groups of output channels or weight rows, together with any associated parameters, defined precisely in Section 3.2. Let $\mathcal{B}$ denote the set of all structured blocks and $\mathcal{B}_\ell \subseteq \mathcal{B}$ those in layer ℓ. Each layer is assigned a selection budget k_ℓ, the number of blocks that may receive the injected residual; the default is 10% of $|\mathcal{B}_\ell|$.

GRASP makes two decisions. The first selects the blocks that carry the residual, maximizing the total block score subject to the per-layer budget:

$$\mathcal{S}^* \in \arg\max_{\mathcal{S}\subseteq\mathcal{B}} \sum_{B\in\mathcal{S}} s_B(v) \quad \text{s.t.} \quad |\mathcal{S}\cap\mathcal{B}_\ell| = k_\ell \quad \forall \ell, \tag{1}$$

where the score $s_B(v)$ is defined in Section 3.2. Given $\mathcal{S}^*$, Section 3.3 constructs a residual supported only on those blocks and an anchor-centered ray along which the residual is scaled. The second decision selects the scale $\lambda \geq 0$ of that ray. It is determined by an adaptive radius τ, derived from the pairwise distances within R, and a support threshold q; both are defined in Section 3.4. GRASP chooses the largest scale whose reference-neighbor count at radius τ is at least q, thereby maximizing residual amplitude subject to geometric support from the reference population.

3.2 Shapley-Guided Block Selection

Structured blocks. Within each layer, we group output channels or weight rows into structured blocks. A convolutional block contains the grouped filters and, when present, the affine parameters of the immediately following batch-normalization layer; a linear block contains the grouped weight rows and biases. Grouping details and per-architecture block counts appear in Appendix C.3. Parameters outside these blocks are excluded from selection and retain their anchor values during injection. The set $\mathcal{B} = \{B_1, \ldots, B_K\}$ indexes all blocks available for scoring and residual injection.

Scoring and ranking. We score blocks on the target model ϑ_v using labeled examples $\mathcal{D}_s = \{(x_e, y_e)\}_{e=1}^{P}$. Each compromised client reserves a label-stratified scoring holdout, disjoint from its local attack-training subset. Before training, these holdouts are pooled into a fixed panel $\mathcal{D}_s$, approximately balanced across compromised clients. Every score refresh reuses this panel, which contains no examples from honest clients. Sampling and refresh details appear in Appendix C.3. Evaluating both model parameters and loss gradients at ϑ_v makes the ranking depend on the realized target, so selection can vary across base attacks and rounds.

We use the second-order Model Shapley estimator of Chu et al. (2025) and instantiate its curvature term with a layerwise empirical Fisher. For layer ℓ, let $\mathcal{I}_\ell = \{i : B_i \in \mathcal{B}_\ell\}$ denote its block-index set, with $K_\ell = |\mathcal{I}_\ell|$. For each $i \in \mathcal{I}_\ell$, let θ_i vectorize the trainable target-model parameters in B_i,

and define $\theta_\ell = \mathrm{col}_{i\in\mathcal{I}_\ell}(\theta_i)$ in ascending block-index order. We then compute

$$q_\ell^{(e)} = \nabla_{\theta_\ell}\mathcal{L}(\vartheta_v; x_e, y_e), \qquad g_\ell = \frac{1}{P}\sum_{e=1}^{P} q_\ell^{(e)}, \qquad \widehat{F}_\ell = \frac{1}{P}\sum_{e=1}^{P} q_\ell^{(e)} q_\ell^{(e)\top}. \tag{2}$$

Here $\mathcal{L}$ denotes the per-example loss, and g_i is the slice of g_ℓ corresponding to B_i. With path weights set to one, the block-level estimator and the ranking score $s_{B_i}(v) \equiv s_i(v)$ are

$$\widetilde{\phi}_i = -\theta_i^\top g_i - \frac{1}{2}\theta_i^\top (\widehat{F}_\ell\theta_\ell)_i, \qquad s_i(v) = \frac{\left|\alpha_s\, \theta_i^\top g_i + \beta_s\, \theta_i^\top (\widehat{F}_\ell\theta_\ell)_i\right|}{\sqrt{|B_i|}}, \tag{3}$$

where $(\widehat{F}_\ell\theta_\ell)_i$ is the slice corresponding to B_i. The first term of $\widetilde{\phi}_i$ measures individual contribution, while the second aggregates self- and cross-block interactions within the layer. We evaluate $\widehat{F}_\ell\theta_\ell$ by accumulating $P^{-1}\sum_{e=1}^{P} q_\ell^{(e)}(q_\ell^{(e)\top}\theta_\ell)$, without materializing the Fisher matrix. The weights α_s and β_s control the individual and cooperative terms. With the default weights $\alpha_s = 1$ and $\beta_s = 0.5$, the score satisfies $s_i(v) = |\widetilde{\phi}_i|/\sqrt{|B_i|}$. We rank blocks by contribution magnitude because the target-to-anchor residual determines the injection direction, whereas the score determines which blocks to retain; taking the absolute value allows blocks with large positive or negative estimated contributions to be selected. The $\sqrt{|B_i|}$ factor makes unequal blocks comparable, and a per-layer budget prevents one layer from consuming the entire selection budget. Equation (3) thus serves as a block ranking score, with the top k_ℓ blocks in each layer forming $\mathcal{S}^*$ for anchor-based injection.

3.3 Nearest-Anchor Residual Injection

Given $\mathcal{S}^*$, residual injection combines the target update v with the reference set R. It selects the observed reference $a \in \arg\min_{u\in R}\|v-u\|_2$ nearest to v as the *anchor*. Unlike a coordinate-wise mean, this anchor is an actual reference update, and the nearest-anchor choice minimizes the norm of the target-to-anchor residual ($r = v - a$) over the available references. The injection residual δ retains this difference only on the selected blocks:

$$\delta[B] = \begin{cases} v[B] - a[B], & B \in \mathcal{S}^*, \\ 0, & B \notin \mathcal{S}^*, \end{cases} \qquad z(\lambda) = a + \lambda\delta, \quad \lambda \geq 0, \tag{4}$$

where δ is the injection residual. Thus every unselected block remains equal to its anchor value. At $\lambda = 1$, the injected update $z(1) = a + \delta$ copies the target coordinates on $\mathcal{S}^*$ and preserves the other coordinates from a; this is the unscaled block-injected update. The choice $\lambda = 0$ returns the anchor, whereas $\lambda > 1$ extrapolates beyond the selected coordinates of v relative to a along the same residual direction. A shared scalar λ preserves the direction and relative cross-block magnitudes of the selected residual. This reduces scale selection to a one-dimensional search, enabling the exact interval-overlap solver below.

3.4 Majority-Aware Scale Selection

Given the anchor-centered ray $z(\lambda)$ from Eq. (4), majority-aware scale selection determines its final scale. Fixed scaling is brittle because update dispersion changes across datasets, rounds, and non-IID partitions. GRASP instead estimates a natural envelope using pairwise distances among reference updates. With n participating clients and m attackers, the proportion of the honest population needed to complement the malicious group to a majority is

$$\rho = \frac{\lfloor n/2\rfloor + 1 - m}{n - m}, \qquad q = \min\left\{|R|, \lceil \rho|R|\rceil\right\}, \qquad p = 100\rho. \tag{5}$$

Here q is the minimum number of references that must support a candidate for it to be treated as majority-backed, and p is the corresponding percentile. The Euclidean radius τ is the pth percentile of off-diagonal pairwise distances among reference updates. Under the default omniscient model, $|R| = n - m$, so $q = \lfloor n/2\rfloor + 1 - m$. The reference-neighbor count of a candidate x is the number of references $u \in R$ within distance τ of x, i.e., $N_R(x;\tau) = \sum_{u\in R}\mathbf{1}\{\|x-u\|_2 \leq \tau\}$, and the feasible set is $\mathcal{F}_q = \{\lambda \geq 0 : N_R(z(\lambda);\tau) \geq q\}$. For $\delta \neq 0$ and $\mathcal{F}_q \neq \varnothing$, GRASP selects the largest supported scale,

$$\lambda^* = \max\left\{\lambda \geq 0 : N_R\big(z(\lambda);\tau\big) \geq q\right\}. \tag{6}$$

For $\delta \neq 0$, each reference u induces $I_u = \{\lambda \geq 0 : \|a + \lambda\delta - u\|_2 \leq \tau\}$. Each nonempty I_u is a closed, bounded interval because squared distance is a strictly convex quadratic in λ. GRASP sorts the endpoints and selects the rightmost point with at least q-fold coverage, treating the intervals as closed. Because $\|z(\lambda) - a\|_2 = \lambda\|\delta\|_2$ for $\lambda \geq 0$, this point maximizes residual amplitude under the support constraint in Eq. (6). The interval endpoints and the solver are derived in Appendix B.3.

If $\mathcal{F}_q$ is empty, the constrained scale problem has no solution. We then return the anchor by setting the output scale to zero and record infeasibility. The anchor-copy fallback instantiates Mimic: all malicious clients copy the same observed honest update (Karimireddy et al., 2022).

4 Complexity Analysis

Let d denote the update dimension, d_s the number of scored parameters, m the number of compromised clients, P the scoring sample size, and K_ℓ the block count in layer ℓ.

Time complexity. Per-example gradients cost $C_{\mathrm{pg}}(P) = O(Pd)$ under Model Shapley's parameter-count proxy (Chu et al., 2025). Accumulating g_ℓ and the matrix-free product $\widehat{F}_\ell\theta_\ell$ costs $O(Pd_s)$, and block ranking costs $O(\sum_\ell K_\ell \log K_\ell)$. Block construction and replicated submissions cost $O(d)$ and $O(md)$, while reference geometry and the interval sweep cost $O(|R|^2 d + |R| \log |R|)$. A score-refresh round therefore costs $O(C_{\mathrm{pg}}(P) + Pd_s + \sum_\ell K_\ell \log K_\ell + |R|^2 d + |R| \log |R| + md)$. Cached masks omit the scoring terms between refreshes.

Space complexity. For a scoring mini-batch of size $b \leq P$, matrix-free Fisher accumulation avoids materializing $\widehat{F}_\ell$. Batched per-example gradients, accumulators, block scores, and pairwise distances require auxiliary space $O(d + bd_s + |R|^2 + \sum_\ell K_\ell)$. Including stored references and materialized malicious submissions, total space is $O((|R| + m + 1)d + bd_s + |R|^2 + \sum_\ell K_\ell)$.

Remark. These bounds cover one shared update construction and m replicated submissions, rather than independent construction by every compromised client. In a colluding attack, an attacker-side compute node with sufficient computing resources can construct the shared update and distribute it to the compromised clients. For the enhancement step, these clients only receive and submit the update, without repeating block scoring or scale selection. The complexity bounds suggest that GRASP is computationally feasible when the scoring sample and reference set are kept small. Reusing cached masks reduces repeated scoring, while computing Fisher products without storing the matrix limits memory use. Pairwise distance computation can become costly for large reference sets.

5 Evaluation

5.1 Experiment Setup

We evaluate five representative base attacks: Sign Flipping, A Little Is Enough (ALIE), Inner Product Manipulation (IPM), Min-Max, and Min-Sum (Baruch et al., 2019; Xie et al., 2020; Shejwalkar & Houmansadr, 2021). We compare their performance without enhancement (labeled *Base* in tables and figures) with that of their Pill-enhanced (Guo et al., 2026) and GRASP-enhanced variants. The evaluation covers nine representative Byzantine-robust defenses (Zhang et al., 2025), as shown in the Table 2, and three model–dataset configurations: LeNet on MNIST (LeCun et al., 1998), a two-convolutional-layer CNN on Fashion-MNIST (Xiao et al., 2017), and ResNet-18 on CIFAR-10 (Krizhevsky, 2009). Table 1 summarizes the model parameter and block counts and the principal training and federation settings. We provide experimental and implementation details in Appendix C.

Table 1: Main model configurations, parameter/block counts, and training settings.

Dataset	Model	# Trainable parameters	# Blocks	# Rounds	# Clients / # Byzantine ones	Batch size	Learning rate
MNIST	LeNet	61,706	236	100	50 / 10	64	0.01
Fashion-MNIST	CNN	266,060	290	100	50 / 10	64	0.01
CIFAR-10	ResNet-18	11,173,962	972	200	20 / 4	64	0.05

Blocks follow architecture-aware groups of convolutional output filters, corresponding affine batch-normalization channels, and linear rows. For GRASP, each compromised client reserves a label-

stratified 10% of its local training data as a scoring holdout, which is excluded from that client's attack training. The holdouts from all compromised clients are pooled, and a single fixed panel of at most 128 examples is drawn from this pool and reused for all block scoring throughout the run. We use a 10% per-layer block-count budget, set $\alpha_s = 1$ and $\beta_s = 0.5$, and refresh scores every ten rounds using the same panel. The system is implemented in Python using PyTorch (Paszke et al., 2019). Experiments use an Intel i9-14900K CPU and an NVIDIA RTX 4090 GPU. The main attack-defense comparisons (Tables 2, 4, and 5; and Figure 3) average five runs.

Evaluation metric. To evaluate attack effectiveness and stealth, we use Attack Impact and the malicious admission rate (MAR), respectively. Attack Impact is the clean accuracy (obtained in the corresponding no-attack run) minus the attacked accuracy at the final round, reported in percentage points. MAR is the percentage of malicious submissions a defense retains, averaged over rounds.

5.2 Attack Impact and stealth

In this section, we evaluate how GRASP enhances poisoning attacks across nine defenses and whether its malicious updates evade detection under an honest client majority.

Attack Impact. We use MNIST with FedSGD for 100 rounds and a Dirichlet non-IID partition with $\alpha = 0.1$. Each round includes 50 clients, of which 10 are Byzantine. Table 2 compares each base attack before and after enhancement by Pill or GRASP.

Table 2: Attack Impact across nine Byzantine-robust defenses. The clean accuracy row reports the corresponding no-attack run. Standard deviations are computed across five runs.

Attack \ Defense	FLTrust	FLAME	Multi-Krum	Bulyan	Sign Guard	DnC	Centered Clipping	FL Detector	SkyMask
Clean accuracy	77.22 ± 10.18	89.04 ± 1.72	89.51 ± 2.01	91.16 ± 0.15	81.12 ± 3.04	91.10 ± 0.11	91.09 ± 0.12	81.81 ± 1.90	86.95 ± 3.48
IPM	13.43 ± 1.06	7.42 ± 1.18	**72.29** ± 2.39	8.22 ± 0.91	1.10 ± 3.18	8.04 ± 1.05	1.93 ± 0.42	**46.59** ± 1.47	17.28 ± 3.84
+Pill	14.82 ± 2.66	4.29 ± 4.00	64.53 ± 6.66	40.98 ± 1.80	10.10 ± 2.93	12.78 ± 1.01	**22.50** ± 9.34	31.08 ± 1.28	11.98 ± 1.35
+GRASP	**19.59** ± 0.75	**8.89** ± 1.86	63.06 ± 3.07	**81.53** ± 8.10	**24.56** ± 2.90	**32.28** ± 5.17	16.82 ± 6.05	43.13 ± 0.60	**22.55** ± 1.47
Min-Max	13.77 ± 1.06	**72.62** ± 3.53	63.13 ± 0.93	68.08 ± 0.33	14.34 ± 3.12	11.91 ± 0.33	80.82 ± 1.09	65.26 ± 2.20	17.97 ± 3.73
+Pill	15.07 ± 2.57	62.70 ± 4.00	54.53 ± 4.16	71.50 ± 1.18	6.26 ± 3.27	12.38 ± 1.21	69.63 ± 1.49	61.76 ± 0.34	15.70 ± 3.13
+GRASP	**18.98** ± 0.55	63.10 ± 2.76	**63.16** ± 0.77	**81.17** ± 4.04	**15.83** ± 1.40	**20.12** ± 2.10	**81.43** ± 2.44	**66.86** ± 2.55	**23.26** ± 2.97
Sign Flipping	15.46 ± 1.04	13.76 ± 4.62	9.66 ± 5.69	50.95 ± 8.65	14.33 ± 3.47	7.68 ± 3.40	69.92 ± 1.08	71.28 ± 1.17	**31.53** ± 5.77
+Pill	14.73 ± 2.04	34.29 ± 4.00	34.53 ± 8.66	35.49 ± 1.65	22.66 ± 3.10	10.35 ± 0.28	14.10 ± 11.60	74.13 ± 0.83	25.79 ± 6.37
+GRASP	**17.99** ± 7.48	**64.44** ± 7.05	**73.03** ± 3.11	**67.68** ± 1.57	**32.82** ± 2.95	**20.94** ± 4.04	**75.10** ± 0.92	**77.68** ± 4.13	23.81 ± 6.27
ALIE	24.12 ± 5.13	4.13 ± 0.25	23.30 ± 1.18	71.24 ± 2.88	3.16 ± 0.72	6.49 ± 0.56	5.78 ± 1.05	2.06 ± 0.76	10.56 ± 9.68
+Pill	24.88 ± 2.77	7.29 ± 1.00	29.53 ± 3.66	67.20 ± 1.48	**18.79** ± 3.09	8.16 ± 0.82	53.35 ± 3.40	7.07 ± 4.05	10.45 ± 3.33
+GRASP	**35.70** ± 1.66	**9.60** ± 2.50	**63.93** ± 0.17	**75.99** ± 2.38	13.22 ± 4.94	**12.19** ± 3.34	**59.48** ± 4.98	**7.49** ± 4.31	**11.67** ± 6.51
Min-Sum	13.43 ± 1.06	8.65 ± 3.93	10.07 ± 2.48	**78.84** ± 1.17	15.08 ± 0.98	7.36 ± 1.67	20.82 ± 0.88	**71.25** ± 1.55	3.19 ± 1.30
+Pill	15.28 ± 2.69	14.29 ± 4.00	34.53 ± 0.66	45.61 ± 2.06	16.98 ± 1.81	2.58 ± 1.01	19.80 ± 5.01	65.08 ± 0.37	11.61 ± 3.02
+GRASP	**18.76** ± 0.50	**36.68** ± 9.36	**50.55** ± 3.91	60.99 ± 1.69	**22.66** ± 2.64	**8.81** ± 2.34	**26.64** ± 4.31	68.15 ± 4.55	**19.46** ± 3.60

* Bold indicates the strongest variant for each attack-defense setting. The same notation applies to following tables.

Across 45 attack-defense settings, GRASP exceeds the base attack in 39 cases (86.7%), Pill in 41 cases (91.1%), and both in 37 cases (82.2%). When averaged across the five attacks, GRASP achieves the highest Attack Impact under every defense, showing that the gains are not concentrated in a single base attack or defense.

Stealth. Multi-Krum assigns each client update a distance-based score relative to its nearest neighbors and favors updates with lower scores. To evaluate stealth, we compare the average distance scores of malicious clients, with and without GRASP, with those of honest clients for each base attack. Specifically, we report the ratio of the average malicious-client score to the average benign-client score, together with MAR in Table 3. As summarized in the Average column, GRASP raises the Multi-Krum MAR from 52.9% to 75.1% and lowers the score ratio from 0.811 to 0.779. For ALIE under Multi-Krum, GRASP raises Attack Impact from 23.30 to 63.93, while the score ratio increases from 0.537 to 0.761 and MAR remains high at 93.9%.

Table 3: Multi-Krum stealth with 20% malicious clients.

Diagnostic	Method	IPM	Min-Max	S.F.	ALIE	Min-Sum	Average
Score ratio	Base	0.562	0.842	1.056	0.537	1.060	0.811
	GRASP	0.778	0.785	0.777	0.761	0.796	0.779
MAR (%)	Base	100.0	59.9	4.6	100.0	0.0	52.9
	GRASP	82.0	78.2	70.5	93.9	50.7	75.1

* S.F. denotes Sign Flipping.

FLAME provides a complementary cluster-based diagnostic. It treats a cluster containing more than half the clients as benign (Nguyen et al., 2022); TNR denotes the fraction of honest updates retained in that cluster. We vary the malicious client fraction from 10% to 40%. Figure 2 summarizes the Attack Impact, MAR, and TNR under FLAME, averaged across the five base attacks. Across all malicious client fractions, GRASP increases Attack Impact and MAR, while TNR declines as more malicious updates enter FLAME's benign cluster. At 40%, MAR reaches 99.80% and TNR falls to 41.40%. An honest client majority does not prevent malicious admission under FLAME in these evaluated settings. Table 11 in the appendix reports the attack-level results.

These results indicate a favorable attack-strength-stealth trade-off: GRASP can substantially increase Attack Impact while preserving, and in some cases improving, malicious admission.

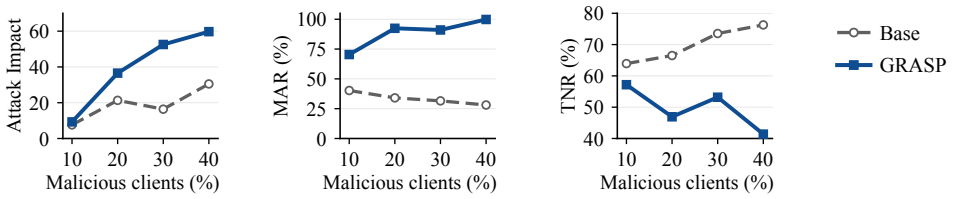


Figure 2: Effect of the malicious client fraction under FLAME on MNIST.

5.3 GENERALITY

Table 4: Attack Impact of the base attacks and their Pill- and GRASP-enhanced variants on Fashion-MNIST and CIFAR-10.

Attack \ Defense	Fashion-MNIST (non-IID)					CIFAR-10 (IID)				
	FLAME	**Multi Krum**	**Bulyan**	**Sign Guard**	**DnC**	**FLAME**	**Multi Krum**	**Bulyan**	**Sign Guard**	**DnC**
Clean accuracy	71.12 ± 0.15	77.69 ± 0.32	81.76 ± 0.07	76.11 ± 2.57	81.76 ± 0.07	67.55 ± 2.16	69.77 ± 0.28	75.97 ± 1.20	63.28 ± 3.16	76.96 ± 2.12
IPM	24.49 ± 8.76	25.50 ± 9.40	55.50 ± 1.00	5.87 ± 1.90	11.48 ± 1.49	23.85 ± 3.24	15.50 ± 2.28	28.76 ± 1.12	19.03 ± 3.16	13.77 ± 1.08
+Pill	25.66 ± 3.25	22.43 ± 2.72	47.83 ± 1.93	14.79 ± 3.23	11.81 ± 0.72	22.42 ± 1.18	11.75 ± 4.11	16.36 ± 2.23	15.80 ± 3.27	14.06 ± 1.11
+GRASP	**27.63** ± 2.45	**44.64** ± 2.66	**57.60** ± 5.39	**37.40** ± 5.04	**15.32** ± 3.05	**25.68** ± 3.54	**26.12** ± 1.35	**30.79** ± 3.15	**20.01** ± 4.31	**18.53** ± 6.39
Min-Max	23.73 ± 1.02	27.65 ± 0.15	24.28 ± 0.38	16.10 ± 1.81	6.45 ± 0.77	12.14 ± 2.95	14.93 ± 1.47	22.40 ± 1.84	7.48 ± 0.56	14.03 ± 1.16
+Pill	25.19 ± 6.25	36.85 ± 0.25	31.29 ± 2.93	18.83 ± 3.68	11.81 ± 0.72	11.05 ± 2.61	**21.19** ± 2.98	**25.88** ± 1.64	15.37 ± 1.24	12.99 ± 2.15
+GRASP	**37.61** ± 4.92	**52.53** ± 9.48	**34.94** ± 2.41	**21.26** ± 1.30	**27.00** ± 1.00	**18.11** ± 2.23	19.87 ± 3.47	24.56 ± 3.20	**17.34** ± 2.01	**16.78** ± 1.12
Sign Flipping	19.31 ± 1.60	11.39 ± 3.30	**43.43** ± 3.83	8.53 ± 5.55	6.84 ± 1.10	11.91 ± 1.84	23.07 ± 1.21	21.89 ± 2.12	5.57 ± 2.76	13.30 ± 1.33
+Pill	24.37 ± 0.25	16.85 ± 0.25	30.13 ± 0.46	**17.64** ± 4.76	6.81 ± 0.72	15.89 ± 1.15	20.36 ± 1.06	25.38 ± 2.18	**24.84** ± 4.15	14.08 ± 1.25
+GRASP	**46.24** ± 9.05	**48.12** ± 1.26	37.48 ± 3.13	14.84 ± 4.12	**8.10** ± 1.03	**24.31** ± 3.99	**26.12** ± 4.63	**32.15** ± 4.50	22.10 ± 3.21	**28.53** ± 5.45
ALIE	**6.99** ± 1.44	12.74 ± 2.53	52.96 ± 1.84	14.70 ± 1.46	4.83 ± 1.96	11.46 ± 2.57	14.83 ± 5.11	15.88 ± 1.98	**15.46** ± 4.03	12.33 ± 2.86
+Pill	5.72 ± 0.25	15.82 ± 2.41	20.60 ± 0.34	7.42 ± 3.07	1.87 ± 0.72	10.69 ± 1.24	19.55 ± 0.86	15.36 ± 1.97	7.11 ± 1.95	**13.56** ± 3.90
+GRASP	5.27 ± 0.75	**42.85** ± 1.94	**53.19** ± 7.73	**19.55** ± 3.37	**5.20** ± 1.54	**18.54** ± 2.98	**22.77** ± 2.39	**19.73** ± 5.73	10.29 ± 1.20	12.19 ± 3.59
Min-Sum	7.69 ± 0.21	18.57 ± 1.29	56.36 ± 1.74	11.36 ± 3.50	2.78 ± 1.12	10.23 ± 2.10	2.16 ± 1.84	14.87 ± 1.19	9.40 ± 3.62	3.64 ± 2.32
+Pill	6.81 ± 2.25	26.85 ± 0.25	28.27 ± 3.91	**17.36** ± 2.22	1.81 ± 0.72	12.97 ± 4.33	10.08 ± 5.04	16.15 ± 2.13	4.25 ± 3.12	5.45 ± 1.92
+GRASP	**54.22** ± 1.72	**46.71** ± 3.80	**63.15** ± 11.18	14.88 ± 2.89	**15.66** ± 3.34	**14.91** ± 1.35	**11.95** ± 4.14	**21.77** ± 2.76	**13.22** ± 4.15	**6.91** ± 1.72

To further assess the generality of GRASP, we examine whether the attack enhancement transfers across datasets, model architectures, and training.

Architectures and datasets. Fashion-MNIST uses FedSGD with a Dirichlet non-IID partition ($\alpha = 0.1$), whereas CIFAR-10 uses IID data. Table 4 compares the base attacks and their Pill- and GRASP-enhanced variants for both datasets across five defenses.

On Fashion-MNIST, GRASP exceeds the base attack in 23 of 25 settings (92.0%) and Pill in 22 (88.0%), leading both in 21 (84.0%). Its average Attack Impact is 33.26, compared with 19.98 for the base attacks and 18.99 for Pill. On CIFAR-10, the corresponding counts are 23 (92.0%), 21 (84.0%), and 20 (80.0%), with an average Attack Impact of 20.13, compared with 14.32 and 15.30, respectively. The gains extend beyond LeNet on MNIST to the evaluated CNN and residual-network configurations, supporting transfer across the tested dataset-architecture combinations.

Table 5: Attack Impact under FedAvg using LeNet on MNIST with non-IID data.

Attack \ Defense	FLAME	Multi Krum	Bulyan	Sign Guard	DnC	Average
Clean accuracy	88.03	84.55	93.99	93.37	94.01	90.79
ALIE	3.63	1.35	11.05	5.73	0.54	4.46
+GRASP	**4.84**	**74.43**	**20.53**	**10.71**	**1.09**	**22.32**
Min-Max	**78.45**	1.34	14.38	**20.35**	0.27	22.96
+GRASP	70.26	**39.75**	**18.55**	15.39	**3.48**	**29.49**
Sign Flipping	4.70	0.70	14.22	4.55	**9.65**	6.76
+GRASP	**16.81**	**50.17**	**25.71**	**11.25**	3.77	**21.54**

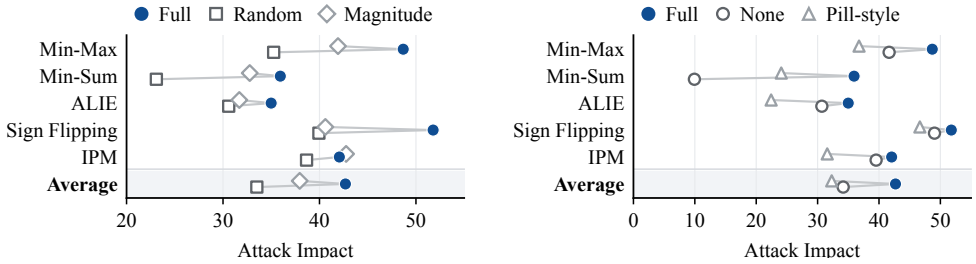
Training protocol. We also test enhancement transfer from one-step FedSGD to FedAvg. Table 5 compares the base attacks and their GRASP-enhanced variants. Across the fifteen attack-defense settings, GRASP increases Attack Impact in twelve and raises the overall average from 11.39 to 24.45 points. Under Multi-Krum, the attack-averaged impact rises from 1.13 to 54.78 points. This pronounced gain is consistent with Multi-Krum's distance-based scoring: by adjusting malicious updates according to reference-set distances, GRASP directly targets the geometric information used by Multi-Krum to rank clients.

Taken together, these experiments show that GRASP's attack-enhancement benefits extend across the tested dataset-architecture combinations and from one-step FedSGD to multi-step FedAvg.

5.4 Component ablations

To understand the contribution of each component, we examine Shapley-guided block selection and majority-aware scale selection separately, varying one while keeping the other fixed.

Figure 3 reports results on MNIST with non-IID data, with Attack Impact averaged across Multi-Krum, Bulyan, SignGuard, DnC, and FLAME. Figure 3(a) varies block selection while keeping anchor-based injection and majority-aware scale selection fixed. Random samples blocks uniformly within each layer; Magnitude ranks blocks by the ℓ_1 norm of their parameters. Both use the full method's per-layer budget. Figure 3(b) fixes the Shapley-selected blocks and injection ray and varies only the scale. None uses $\lambda = 1$; Pill-style adapts Pill's distance-based adjustment (Guo et al., 2026) by scaling the residual along the fixed injection ray so that the update's distance from the anchor does not exceed the maximum distance between the anchor and any reference update.



(a) Block selection (b) Majority-aware scale selection

Figure 3: Component ablations on LeNet with non-IID MNIST.

Figure 3(a) shows that the full method leads on four of five attacks; only IPM favors Magnitude (42.76 versus 42.06 points). Random and Magnitude lower the average from 42.69 to 33.51 and 37.95. In Figure 3(b), the full method leads on all attacks, scoring 42.69 versus 34.17 for None and 32.30 for Pill-style. The combined score also beats either individual or cooperative scoring alone on all three attacks (Figure 4). Together, these ablations support Shapley-guided selection, majority-aware scaling, and the inclusion of both individual and cooperative terms in the block score.

6 Conclusion

We presented GRASP, an attack-agnostic wrapper that strengthens poisoning under an honest client majority through Shapley-guided block selection, residual injection, and majority-aware scale selection. Across base attacks, defenses, datasets, and architectures, the wrapper generally increased Attack Impact while modifying only a structured parameter subset. These findings motivate defenses that inspect influential blocks rather than relying only on whole-update statistics.

AI USE STATEMENT

Generative AI tools were used to assist with data processing, language editing, and formatting-related changes to the LaTeX source. The authors reviewed all AI-assisted work, including the data-processing outputs, and take responsibility for the final content, claims, and artifacts of this work.

REFERENCES

Dan Alistarh, Zeyuan Allen-Zhu, and Jerry Li. Byzantine stochastic gradient descent. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, pp. 2938–2948. PMLR, 2020.

Gilad Baruch, Moran Baruch, and Yoav Goldberg. A little is enough: Circumventing defenses for distributed learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 634–643. PMLR, 2019.

Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Gunnar Blom. *Statistical Estimates and Transformed Beta-Variables*. Wiley, 1958.

Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé Kiddon, Jakub Konečný, Stefano Mazzocchi, Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards federated learning at scale: System design. *Proceedings of Machine Learning and Systems*, 1:374–388, 2019. URL https://proceedings.mlsys.org/paper_files/paper/2019/hash/7b770da633baf74895be22a8807f1a8f-Abstract.html.

Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. FLTrust: Byzantine-robust federated learning via trust bootstrapping. In *Network and Distributed System Security Symposium*, 2021.

Xu Chu, Xinke Jiang, Rihong Qiu, Jiaran Gao, and Junfeng Zhao. Model Shapley: Find your ideal parameter player via one gradient backpropagation. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://openreview.net/forum?id=9ccmoYhZue.

El-Mahdi El-Mhamdi, Rachid Guerraoui, and Sébastien Rouault. The hidden vulnerability of distributed learning in byzantium. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 3521–3530, 2018.

Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Local model poisoning attacks to byzantine-robust federated learning. In *29th USENIX Security Symposium*, pp. 1605–1622, 2020.

Clement Fung, Chris J. M. Yoon, and Ivan Beschastnikh. Mitigating sybils in federated learning poisoning, 2020.

Amirata Ghorbani and James Zou. Neuron shapley: Discovering the responsible neurons. In *Advances in Neural Information Processing Systems*, volume 33, pp. 5922–5932, 2020.

Hanxi Guo, Hao Wang, Tao Song, Tianhang Zheng, Yang Hua, Haibing Guan, and Xiangyu Zhang. Poisoning with a pill: Circumventing detection in federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 21432–21440, 2026. doi: 10.1609/aaai.v40i26.39290. URL https://arxiv.org/abs/2407.15389.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Sai Praneeth Karimireddy, Lie He, and Martin Jaggi. Learning from history for byzantine robust optimization. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 5311–5319, 2021. URL https://proceedings.mlr.press/v139/karimireddy21a.html.

Sai Praneeth Karimireddy, Lie He, and Martin Jaggi. Byzantine-robust learning on heterogeneous datasets via bucketing. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=jXKKDEi5vJt.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pp. 1273–1282, 2017.

Thien Duc Nguyen, Phillip Rieger, Huili Chen, Hossein Yalame, Helen Möllering, Hossein Fereidooni, Samuel Marchal, Markus Miettinen, Azalia Mirhoseini, Shaza Zeitouni, Farinaz Koushanfar, Ahmad-Reza Sadeghi, and Thomas Schneider. FLAME: Taming backdoors in federated learning. In *31st USENIX Security Symposium*, pp. 1415–1432, 2022. URL https://www.usenix.org/conference/usenixsecurity22/presentation/nguyen.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, pp. 8024–8035, 2019. URL https://papers.nips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.

Lloyd S. Shapley. A value for n-person games. In *Contributions to the Theory of Games II*, pp. 307–317. Princeton University Press, 1953.

Virat Shejwalkar and Amir Houmansadr. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning. In *Network and Distributed System Security Symposium*, 2021.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms, 2017. URL https://arxiv.org/abs/1708.07747.

Chulin Xie, Minghao Chen, Pin-Yu Chen, and Bo Li. CRFL: Certifiably robust federated learning against backdoor attacks. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 11372–11382. PMLR, 2021.

Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. Fall of empires: Breaking byzantine-tolerant SGD by inner product manipulation. In *Proceedings of the 35th Conference on Uncertainty in Artificial Intelligence*, volume 115 of *Proceedings of Machine Learning Research*, pp. 261–270. PMLR, 2020. URL https://proceedings.mlr.press/v115/xie20a.html.

Jiahao Xu, Zikai Zhang, and Rui Hu. Achieving byzantine-resilient federated learning via layer-adaptive sparsified model aggregation. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, pp. 1508–1517, 2025. URL https://openaccess.thecvf.com/content/WACV2025/html/Xu_Achieving_Byzantine-Resilient_Federated_Learning_via_Layer-Adaptive_Sparsified_Model_Aggregation_WACV_2025_paper.html.

Jian Xu, Shao-Lun Huang, Linqi Song, and Tian Lan. Byzantine-robust federated learning through collaborative malicious gradient filtering. In *2022 IEEE 42nd International Conference on Distributed Computing Systems*, pp. 1223–1235, 2022. doi: 10.1109/ICDCS54860.2022.00120. URL https://arxiv.org/abs/2109.05872.

Peishen Yan, Hao Wang, Tao Song, Yang Hua, Ruhui Ma, Ningxin Hu, Mohammad Reza Haghighat, and Haibing Guan. SkyMask: Attack-agnostic robust federated learning with fine-grained learnable masks. In *Computer Vision – ECCV 2024*, volume 15077 of *Lecture Notes in Computer Science*, pp. 291–308. Springer, 2025. doi: 10.1007/978-3-031-72655-2_17. URL https://link.springer.com/chapter/10.1007/978-3-031-72655-2_17.

Heyi Zhang, Yule Liu, Xinlei He, Jun Wu, Tianshuo Cong, and Xinyi Huang. SoK: Benchmarking poisoning attacks and defenses in federated learning, 2025. URL https://arxiv.org/abs/2502.03801.

Zaixi Zhang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. FLDetector: Defending federated learning against model poisoning attacks via detecting malicious clients. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2545–2555, 2022. doi: 10.1145/3534678.3539231.

A Notation & Algorithm

Table 6 lists the core notation and sensitivity parameters in this paper. Algorithm 1 summarizes the complete pipeline of GRASP.

Table 6: Core notation and sensitivity parameters used by GRASP.

Symbol	Meaning
$w_t,\ v,\ \vartheta_v$	Global model, target update in the algorithm's submission space, and induced target model. For FedSGD, $\vartheta_v = w_t + v$; for FedAvg, $\vartheta_v = v$.
R	Reference set in the algorithm's submission space, used for anchor selection and geometric calibration.
$\mathcal{B},\ \mathcal{S}$	Structured-block set and selected subset for target injection.
$\mathcal{I}_\ell,\ K_\ell,\ P$	Block-index set and block count for layer ℓ, and scoring sample size.
r_{grp}	Maximum number of consecutive output channels or rows grouped into one structured block.
$s_B(v)$	Ranking score for block B, evaluated on the target model.
$\alpha_s,\ \beta_s$	Weights of the individual and cooperative scoring terms.
a	Target-nearest anchor.
$z(\lambda),\ \lambda^*$	Adjusted shared submission and selected residual scale.
$N_R(x;\tau)$	Number of references within radius τ of update x.
$\rho,\ q,\ \tau$	Majority-support proportion, reference-support target, and adaptive radius.
$h,\ \delta,\ \mathcal{F}_q$	Majority threshold, selected-block residual, and feasible-scale set.
$\widehat{\tau},\ \gamma_{\mathrm{B}},\ \tau^{\mathrm{local}}$	Observed local radius, Blom scaling factor, and calibrated attacker-local radius.
m/n	Malicious client fraction varied in the sensitivity analysis.
$r_{\mathrm{score}},\ B_{\max}$	Attacker-local scoring fraction and maximum size of the fixed scoring panel.

B Theoretical Analysis

This section first characterizes when selective residual concentration improves attack effect per unit geometric deviation under a linear utility approximation. It then relates the reference-support constraint to majority support, proves that the interval solver returns the largest supported scale, and characterizes the infeasible case. It also quantifies how block injection and majority-aware scaling retain or modify the target update. The majority-support statements apply to the default omniscient threat model. The interval and feasibility results apply to any fixed reference set.

Algorithm 1 GRASP: granularity-aware residual attack via Shapley poisoning

Require: global model w_t; target update v; reference set R; numbers of participating and compromised clients n, m; per-layer budgets $\{k_\ell\}$; scoring weights α_s, β_s; fixed scoring panel $\mathcal{D}_s$
Ensure: shared malicious update z^*

```
Partition the model into structured blocks B
if score refresh then
    Set ϑ_v ← w_t + v; reuse D_s
    Compute per-example gradients and g_ℓ via Eq. (2)
    Accumulate F̂_ℓ θ_ℓ without materializing F̂_ℓ
    Score each block via Eq. (3)
    Select top k_ℓ blocks per layer → S*
else
    Reuse cached S*
end if
a ← arg min_{u∈R} ‖v − u‖_2
δ ← residual of v − a on S*, zero elsewhere
if δ = 0 then
    return z* = a
end if
Compute ρ, q, p via Eq. (5)
τ ← pth percentile of off-diagonal pairwise distances in R
Construct the nonempty closed intervals I_u via Eq. (17)
if ∃ λ ≥ 0 with N_R(a + λδ; τ) ≥ q then
    Sweep interval endpoints, counting both starts and ends at each shared endpoint
    λ* ← rightmost endpoint covered by at least q closed intervals
else
    λ* ← 0; record infeasibility fallback
end if
return z* = a + λ*δ
```

B.1 Granularity Mismatch

We analyze the granularity mismatch using a linear approximation of attack utility at the anchor. Fix the target update v, anchor a, and selected structured blocks $\mathcal{S}^*$. Write $r = v - a$ for the full target-to-anchor residual and $\delta = P_{\mathcal{S}^*} r$ for the injection residual in Eq. (4). The coordinate projection $P_{\mathcal{S}^*}$ keeps the selected blocks and zeros all other coordinates. Let J_v be a differentiable attack-utility surrogate for the target update v, defined near a in submission space. This is an analytical surrogate, not an objective assumed to be optimized by every base attack. Define $g_J = \nabla J_v(a)$ and the first-order change $A_J(x) = g_J^\top x$, so $J_v(a + x) - J_v(a) = A_J(x) + o(\|x\|_2)$ as $x \to 0$. Here g_J differs from the scoring gradients g_i evaluated at ϑ_v in Eq. (3).

Let Q be symmetric positive definite on the considered update subspace, and write $\|x\|_Q = (x^\top Q x)^{1/2}$. This quadratic form is an analytical surrogate for aggregate whole-update geometric deviation near the anchor, not an exact model of each defense. For $r, \delta \neq 0$ and $A_J(r) \neq 0$, define attack-effect retention, geometric-energy retention, and first-order attack efficiency by

$$\alpha_{\mathcal{S}^*} = \frac{|A_J(\delta)|}{|A_J(r)|}, \qquad \beta_{\mathcal{S}^*} = \frac{\|\delta\|_Q^2}{\|r\|_Q^2}, \qquad \eta_Q(x) = \frac{|A_J(x)|}{\|x\|_Q} \quad (x \neq 0). \tag{7}$$

These retention ratios need not lie in $[0, 1]$: effect cancellation can make $\alpha_{\mathcal{S}^*} > 1$, and coordinate projection need not contract a general Q-norm. Absolute values compare effect magnitudes; interpreting them as increased attack utility additionally requires $A_J(r), A_J(\delta) > 0$.

Proposition 1 (Granularity-mismatch efficiency)**.** *Under the preceding assumptions,*

$$\frac{\eta_Q(\delta)}{\eta_Q(r)} = \frac{\alpha_{\mathcal{S}^*}}{\sqrt{\beta_{\mathcal{S}^*}}}. \tag{8}$$

Consequently, $\eta_Q(\delta) > \eta_Q(r)$ if and only if $\alpha_{\mathcal{S}^*} > \sqrt{\beta_{\mathcal{S}^*}}$, equivalently $\alpha_{\mathcal{S}^*}^2 > \beta_{\mathcal{S}^*}$.

Proof. Substitution gives $\eta_Q(\delta)/\eta_Q(r) = (|A_J(\delta)|/|A_J(r)|)(\|r\|_Q/\|\delta\|_Q) = \alpha_{\mathcal{S}^*}/\sqrt{\beta_{\mathcal{S}^*}}$. The inequality follows because $\eta_Q(r)$ and $\beta_{\mathcal{S}^*}$ are positive. □

Corollary 2 (Equal geometric budget)**.** *Under the same assumptions, fix a geometric deviation budget $\xi > 0$ and set $\widetilde{r} = \xi r/\|r\|_Q$ and $\widetilde{\delta} = \xi\delta/\|\delta\|_Q$. Both have Q-norm ξ, and*

$$\frac{|A_J(\widetilde{\delta})|}{|A_J(\widetilde{r})|} = \frac{\alpha_{\mathcal{S}^*}}{\sqrt{\beta_{\mathcal{S}^*}}}. \tag{9}$$

Thus, $\alpha_{\mathcal{S}^*} > \sqrt{\beta_{\mathcal{S}^*}}$ gives a larger first-order attack-effect magnitude at the same whole-update geometric budget.

Proof. Linearity yields $|A_J(\xi x/\|x\|_Q)| = \xi\eta_Q(x)$; apply Proposition 1. □

Selective poisoning can therefore remove low-utility geometric mass while retaining high-utility components of the full residual r when the condition holds. The fixed-budget comparison is exact for the linear model; its interpretation for J_v applies near a for sufficiently small ξ. If $A_J(r) = 0$, the relative effect comparison is undefined. If $\delta = 0$, its normalized direction is undefined; if $\delta \neq 0$ but $A_J(\delta) = 0$, its efficiency is zero.

Euclidean reference geometry. For $Q = I$, $\beta_{\mathcal{S}^*} = \|\delta\|_2^2/\|r\|_2^2 \leq 1$, and the improvement condition becomes

$$\frac{|g_J^\top \delta|}{|g_J^\top r|} > \frac{\|\delta\|_2}{\|r\|_2}. \tag{10}$$

The selected blocks then preserve first-order attack effect more strongly than Euclidean residual magnitude. This geometry matches the Euclidean reference support in Section 3.4, but equal anchor-distance budgets need not yield equal reference support or malicious admission.

GRASP uses the second-order Shapley score in Eq. (3) to identify candidate blocks through individual and within-layer cooperative contributions at ϑ_v, including interactions omitted by the linear surrogate. The proposition explains when the selected residual improves efficiency. It neither identifies the score with $g_J^\top \delta$ nor guarantees that the ranking maximizes $\alpha_{\mathcal{S}^*}/\sqrt{\beta_{\mathcal{S}^*}}$.

The analysis characterizes the tradeoff between attack effect and whole-update geometric deviation under a linear utility approximation at the anchor. The experiments evaluate GRASP across defenses with different acceptance rules.

B.2 Majority support

Let $h = \lfloor n/2 \rfloor + 1$ be the number of current-round client submissions required for a majority, that is, strictly more than half of all participating clients. All m malicious clients submit the same adjusted update $z(\lambda)$. To compare candidate scales, let $\mathcal{X}_t(\lambda) = (x_1(\lambda), \ldots, x_n(\lambda))$ denote the client submissions that would result if all m malicious clients submitted $z(\lambda)$, with the honest submissions held fixed. Define

$$M_\tau(\lambda) = \sum_{i=1}^{n} \mathbf{1}\{\|x_i(\lambda) - z(\lambda)\|_2 \leq \tau\}. \tag{11}$$

The sum counts clients rather than distinct update vectors, so identical submissions are counted separately.

Definition. We say that $z(\lambda)$ has majority support within radius τ if

$$M_\tau(\lambda) \geq h. \tag{12}$$

This definition is purely geometric: it states how many submissions are close to $z(\lambda)$ in Euclidean distance.

Proposition 3 (Reference support and majority support)**.** *Assume the default omniscient threat model, in which $m < n/2$ and R contains all $n - m$ honest current-round updates. Let $\rho = (h - m)/(n - m)$ and $q = \min\{|R|, \lceil \rho|R| \rceil\}$. For any $\tau \geq 0$ and $\lambda \geq 0$,*

$$N_R(z(\lambda); \tau) \geq q \quad \Longleftrightarrow \quad M_\tau(\lambda) \geq h. \tag{13}$$

Thus, the reference-support constraint is equivalent to requiring a majority of the current-round submissions within radius τ of the shared malicious update.

Proof. Because $|R| = n - m$, we have $\rho|R| = h - m$. The quantity $h - m$ is an integer between one and $|R|$ under $m < n/2$, so $q = h - m$. Under the default omniscient threat model, all $n - m$ honest submissions are represented in R, while the m malicious submissions equal $z(\lambda)$. Hence $M_\tau(\lambda) = m + N_R(z(\lambda); \tau)$, and

$$N_R(z(\lambda); \tau) \geq q \quad \iff \quad m + N_R(z(\lambda); \tau) \geq m + q = h.$$

□

The equivalence holds for any τ; the reference-distance percentile is the data-dependent rule used to calibrate that radius. For the attacker-local reference variant, N_R counts compromised clients' pre-attack updates rather than honest current-round submissions. Its constraint therefore provides reference support but is not equivalent to Eq. (12).

B.3 Largest supported scale and feasibility

Let $\delta = P_{\mathcal{S}^*}(v - a)$ be the injection residual, so the adjusted update follows $z(\lambda) = a + \lambda\delta$ for $\lambda \geq 0$. For each selected block $B \in \mathcal{S}^*$,

$$z(\lambda)[B] = a[B] + \lambda(v[B] - a[B]), \qquad \|z(\lambda)[B] - v[B]\|_2 = |1 - \lambda| \, \|v[B] - a[B]\|_2. \tag{14}$$

Thus, $\lambda = 1$ recovers the selected coordinates of the target update, whereas $\lambda < 1$ contracts and $\lambda > 1$ extrapolates the selected residual. For a fixed reference $u \in R$, define

$$A = \|\delta\|_2^2, \qquad B_u = \delta^\top (a - u), \qquad C_u = \|a - u\|_2^2 - \tau^2. \tag{15}$$

The neighbor condition is equivalent to

$$\|a + \lambda\delta - u\|_2^2 \leq \tau^2 \quad \iff \quad A\lambda^2 + 2B_u\lambda + C_u \leq 0. \tag{16}$$

When $A > 0$, let $\Delta_u = B_u^2 - AC_u$. The feasible scales contributed by reference u are

$$I_u = \left[\frac{-B_u - \sqrt{\Delta_u}}{A}, \frac{-B_u + \sqrt{\Delta_u}}{A} \right] \cap [0, \infty) \tag{17}$$

when $\Delta_u \geq 0$ and the clipped interval is nonempty; otherwise $I_u = \varnothing$. Define the feasible set

$$\mathcal{F}_q = \left\{ \lambda \geq 0 : \sum_{u \in R} \mathbf{1}\{\lambda \in I_u\} \geq q \right\}. \tag{18}$$

Feasibility. For $\delta \neq 0$, the neighbor count is exactly the overlap depth of the intervals $\{I_u : u \in R\}$. Hence, $\mathcal{F}_q$ is nonempty if and only if $\max_{\lambda \geq 0} N_R(a + \lambda\delta; \tau) \geq q$. In particular, $N_R(a; \tau) \geq q$ or $N_R(a + \delta; \tau) \geq q$ guarantees feasibility at $\lambda = 0$ or $\lambda = 1$, respectively.

Theorem 4 (Largest scale with majority support within radius τ)**.** *Assume the default omniscient threat model, with h and q defined above, $\delta \neq 0$, and $\tau \geq 0$. If $\mathcal{F}_q$ is nonempty, an endpoint sweep that treats the intervals as closed returns $\lambda^* = \max \mathcal{F}_q$. The returned update satisfies*

$$M_\tau(\lambda^*) \geq h, \tag{19}$$

whereas no larger scale has majority support:

$$\lambda > \lambda^* \quad \implies \quad M_\tau(\lambda) \leq \left\lfloor \frac{n}{2} \right\rfloor. \tag{20}$$

Moreover, λ^* maximizes the displacement from the anchor along the injection direction, measured by $\|z(\lambda) - a\|_2 = \lambda\|\delta\|_2$, subject to the majority-support constraint:

$$\lambda^* \in \arg\max_{\lambda \geq 0} \ \|z(\lambda) - a\|_2 \quad \text{s.t.} \quad M_\tau(\lambda) \geq h. \tag{21}$$

Proof. Since $\delta \neq 0$, $A > 0$. Equation (16) is a strictly convex quadratic in λ, so every nonempty I_u in Eq. (17) is closed and bounded. The coverage count can change only at one of the finitely many interval endpoints. Equivalently, $\mathcal{F}_q$ is a finite union of intersections of closed, bounded intervals and is therefore closed and bounded whenever it is nonempty. Hence $\max \mathcal{F}_q$ exists and is a right endpoint of at least one interval. Sweeping the endpoints while counting closed-interval coverage returns this rightmost feasible point.

By Proposition 3, feasibility of λ^* gives Eq. (19). For any $\lambda > \lambda^*$, $\lambda \notin \mathcal{F}_q$, so $N_R(z(\lambda); \tau) < q$ and $M_\tau(\lambda) < h$, which is equivalent to Eq. (20). Finally, $\lambda \mapsto \lambda \|\delta\|_2$ is strictly increasing on $[0, \infty)$, so the rightmost feasible scale globally maximizes the displacement from the anchor along the injection direction under the majority-support constraint. □

The endpoint argument itself applies to any fixed reference set; only the majority-support interpretation requires the default omniscient threat model. The endpoint sweep tracks changes in neighbor count through interval coverage along the injection ray. Constructing the coefficients costs $O(|R|d)$, and sorting and sweeping the at most $2|R|$ endpoints costs $O(|R| \log |R|)$.

Mimic fallback. If $\mathcal{F}_q$ is empty, the constrained problem has no solution. Under the default omniscient threat model, Proposition 3 then gives $M_\tau(\lambda) \leq \lfloor n/2 \rfloor$ for every $\lambda \geq 0$ along the fixed ray. The implementation returns the anchor a through the Mimic fallback. A feasible optimum can also occur at $\lambda^* = 0$, so a zero-scale output does not by itself imply infeasibility.

Corollary 5 (Neighbor-distance bound)**.** *Under the conditions of Theorem 4, let $z^* = z(\lambda^*)$, and let $D_{(k)}(z^*)$ be the kth smallest distance from one malicious submission to the other $n-1$ client submissions, counting identical updates separately. Then $D_{(\lfloor n/2 \rfloor)}(z^*) \leq \tau$.*

Proof. Besides the selected malicious submission itself, the radius-τ ball contains the other $m-1$ malicious submissions and at least q honest submissions. Their total number is $(m-1)+q = h-1 = \lfloor n/2 \rfloor$. □

The same count gives an index set $I^* \subseteq \{1, \ldots, n\}$ with $|I^*| = h$ such that $\|x_i(\lambda^*) - z^*\|_2 \leq \tau$ for every $i \in I^*$. The triangle inequality therefore gives $\operatorname{diam}\{x_i(\lambda^*) : i \in I^*\} \leq 2\tau$.

C More Details of Evaluation

C.1 Base Attacks

Let $\mathbf{g}_i$ and $\mathbf{g}_{\text{mal}}$ denote a client update and a forged update, respectively. The coordinate-wise mean $\boldsymbol{\mu}$ and standard deviation $\boldsymbol{\sigma}$ use each attack's current-round pre-attack references.

- *Sign Flipping* Guo et al. (2026). Each malicious client reverses its local model delta, with the scale fixed to one throughout the experiments.
- *A Little Is Enough (ALIE)* Baruch et al. (2019). Attackers submit $\mathbf{g}_{\text{mal}} = \boldsymbol{\mu} + z_{\max}\boldsymbol{\sigma}$, with operations applied coordinate-wise. Both statistics use the m malicious clients' local pre-attack updates. For n participating clients and m attackers, we set

$$s = \lfloor n/2 \rfloor + 1 - m, \qquad z_{\max} = \Phi^{-1}\left(\frac{n-m-s}{n-m}\right),$$

  where Φ^{-1} is the standard-normal quantile function.
- *Inner Product Manipulation (IPM)* Xie et al. (2020). Attackers submit $\mathbf{g}_{\text{mal}} = -\epsilon\boldsymbol{\mu}$ with $\epsilon = 0.1$ across all three combinations of models and datasets. The mean uses the $n-m$ honest pre-attack updates under the default omniscient threat model.
- *Min-Max* Shejwalkar & Houmansadr (2021). The forged update is $\boldsymbol{\mu} + \lambda\mathbf{d}$, with $\mathbf{d} = -\boldsymbol{\mu}/\|\boldsymbol{\mu}\|_2$ and honest pre-attack references. The search maximizes λ subject to

$$\max_i \|\boldsymbol{\mu} + \lambda\mathbf{d} - \mathbf{g}_i\|_2^2 \leq \max_{i,j} \|\mathbf{g}_i - \mathbf{g}_j\|_2^2 . \tag{22}$$

The trial scale starts at λ_0, the step at $\lambda_0/2$, and the best feasible scale at zero. Each trial increases or decreases the scale according to feasibility and halves the step. The search stops when the trial and best feasible scales differ by at most 10^{-5}, without a separate iteration cap. We use $\lambda_0 = 60$ for MNIST and Fashion-MNIST and $\lambda_0 = 10$ for CIFAR-10.

- *Min-Sum* Shejwalkar & Houmansadr (2021). Min-Sum uses the same references, direction, search procedure, and initial values as Min-Max, with the constraint

$$\sum_i \|\boldsymbol{\mu} + \lambda \mathbf{d} - \mathbf{g}_i\|_2^2 \leq \max_i \sum_j \|\mathbf{g}_i - \mathbf{g}_j\|_2^2 . \tag{23}$$

The base attacks and their Pill- and GRASP-enhanced variants share the same compromised clients, training configuration, and base-attack parameters.

C.2 Defenses

The nine evaluated defenses use the following configurations:

- *FLTrust* Cao et al. (2021). The clean server-held root set samples $\lfloor 100 N_c/N \rfloor$ examples per class without replacement. Its realized size is 94 for MNIST and 100 for Fashion-MNIST and CIFAR-10. The root client shares the participating clients' optimizer and per-round update configuration. Aggregation uses nonnegative cosine trust scores and normalizes client-update norms to the root norm.
- *FLAME* Nguyen et al. (2022). HDBSCAN clusters full client-update vectors by cosine distance, with minimum cluster size $\lfloor n/2 \rfloor + 1$, `min_samples` set to one, and a single cluster allowed. Retained updates are clipped at their median norm. Gaussian-noise standard deviation is 1.2×10^{-5} times the round-specific median update norm.
- *Multi-Krum (M.K)* Blanchard et al. (2017). Scores based on Euclidean neighbor distances select the lowest-scoring $\lfloor 0.2n \rfloor$ clients for averaging. The assumed Byzantine count is the configured attacker count m.
- *Bulyan* El-Mhamdi et al. (2018). Repeated Krum selection builds a candidate set of size $n - 2m$. Each coordinate averages the $n - 4m$ candidate values closest to their median.
- *SignGuard (S.G)* Xu et al. (2022). Norm filtering retains updates within $[0.1, 3]$ times the median norm. Sign filtering applies MeanShift to statistics from 10% of coordinates; aggregation retains the intersection of both filters. The global run seed controls the randomized coordinate window. MeanShift uses bandwidth quantile 0.5, at most 50 samples, bin seeding, and leaves residual noise points unassigned.
- *Divide-and-Conquer (DnC)* Shejwalkar & Houmansadr (2021). Each of five iterations samples 20% of coordinates, centers the updates, and retains $n - m$ clients using scores along the leading right singular vector. Aggregation uses the intersection of the five retained sets.
- *Centered Clipping (C.C)* Karimireddy et al. (2021). One iteration clips update residuals around a momentum center at an ℓ_2 threshold of 100 before averaging. The center starts at zero and persists across rounds.
- *FLDetector (FLD)* Zhang et al. (2022). Limited-memory BFGS predicts client updates using a ten-round history window. With start round 50, prediction-error scoring begins after round 60. The gap statistic uses 20 reference samples and tests up to ten clusters; final two-cluster KMeans uses ten initializations. The higher-error cluster is removed once sufficient scores accumulate; beforehand, all client updates are averaged.
- *SkyMask* Yan et al. (2025). Parameter-level learnable masks are applied to frozen client model updates and trained on a small clean root dataset to distinguish honest and malicious updates.

C.3 Datasets, Models, and Block Construction

We use three image-classification benchmarks with increasing input and label complexity:

- *MNIST* LeCun et al. (1998) contains 60,000 training and 10,000 test images. Each example is a 28×28 grayscale digit from one of ten classes.

- *CIFAR-10* Krizhevsky (2009) contains 50,000 training and 10,000 test images. Each example is a 32×32 RGB image from one of ten classes.
- *Fashion-MNIST* Xiao et al. (2017) contains 60,000 training and 10,000 test images. Each example is a 28×28 grayscale image of a fashion item from one of ten classes.

LeNet follows the two-convolution and three-linear-layer architecture LeCun et al. (1998). Our ResNet-18 uses a 3×3, stride-one input convolution and omits the ImageNet stem pooling layer He et al. (2016). Our Fashion-MNIST CNN follows the architecture used by Fang et al. (2020): two unpadded 3×3 convolutions with 30 and 50 output channels, each followed by ReLU and 2×2 max pooling, and two linear layers with 200 and ten outputs, respectively.

Within each layer, each structured block groups up to r_{grp} consecutive output channels or weight rows. We set $r_{\mathrm{grp}} = 1$ for LeNet and the Fashion-MNIST CNN, and $r_{\mathrm{grp}} = 5$ for ResNet-18. Table 7 summarizes the block construction and parameter coverage. Within each layer ℓ, the default budget selects $\max(1, \lceil 0.1|\mathcal{B}_\ell| \rceil)$ blocks. Parameter fractions use all trainable model parameters as the denominator. The ResNet-18 coverage is an attainable range because its final channel group in each convolution can contain fewer than r_{grp} channels.

Table 7: Structured blocks and parameter coverage under the default 10% per-layer block budget.

	LeNet	**CNN**	**ResNet-18**
Dataset	MNIST	Fashion-MNIST	CIFAR-10
Selection groups (layers)	5	4	21
r_{grp}	1	1	5
Convolutional / linear blocks	22 / 214	80 / 210	970 / 2
Total blocks	236	290	972
Trainable parameters	61,706	266,060	11,173,962
Excluded convolutional biases	22	80	0
Non-trainable BN buffers	0	0	9,620
Selected blocks (10% BSR)	25	29	111
Parameters in selected blocks	6,311	26,598	1,142,752–1,235,295
Fraction of trainable parameters	10.23%	10.00%	10.23–11.06%

For ResNet-18, each main-path convolution, shortcut convolution, and final linear transformation is treated as a separate layer. Thus, ℓ indexes an individual parameterized transformation rather than an entire residual block or network stage. Convolutional biases in LeNet and the Fashion-MNIST CNN and non-trainable ResNet-18 batch-normalization buffers are excluded from selection and retain anchor values during injection.

C.4 Experiment Setup

Table 1 summarizes the main training and federation settings for the three combinations of models and datasets. Here we detail data preparation, client training, attack implementation, and the controls used to match experimental runs.

Data preparation. For IID CIFAR-10, we shuffle indices within each class and divide that class nearly evenly among clients, assigning remainder examples to the lowest-indexed clients. For non-IID MNIST and Fashion-MNIST, we draw client proportions $\boldsymbol{\pi}_c \sim \mathrm{Dirichlet}(\alpha \mathbf{1}_n)$ for each class c and split that class using floored cumulative proportions. We impose no minimum client-size constraint and do not rebalance or resample the resulting partitions.

MNIST images are resized to 32×32, whereas Fashion-MNIST retains its native 28×28 resolution. We normalize MNIST with mean 0.1307 and standard deviation 0.3081, and Fashion-MNIST with mean 0.2859 and standard deviation 0.3530. CIFAR-10 is normalized channel-wise with mean $(0.4914, 0.4822, 0.4465)$ and standard deviation $(0.2470, 0.2430, 0.2610)$.

FedSGD training. In the three main FedSGD settings, each participating client performs exactly one local optimizer step on one shuffled mini-batch per round. The client then submits the additive model delta to the server. Training uses cross-entropy loss and SGD with momentum 0.9 and weight

decay 5×10^{-4}. CIFAR-10 uses a MultiStep schedule that multiplies the learning rate by 0.1 after rounds 100 and 160. The reduced learning rates therefore first apply in rounds 101 and 161.

FedAvg training. The separate FedAvg experiment uses LeNet on non-IID MNIST with $\alpha = 0.1$ and 50 clients, including ten Byzantine clients. Training runs for 50 communication rounds with a batch size of 64 and a client learning rate of 0.01. Each client performs five local optimizer steps per round on successive shuffled mini-batches and submits the resulting full local-model vector.

For FedAvg, let $x_i \in \mathbb{R}^d$ denote the full local-model vector submitted by client i. The base attack produces full-model candidates $\{v_j^{\mathrm{src}}\}_{j\in\mathcal{M}}$, whose mean defines the shared target $v = m^{-1}\sum_{j\in\mathcal{M}} v_j^{\mathrm{src}}$. The induced target model is therefore $\vartheta_v = v$, whereas FedSGD uses $\vartheta_v = w_t + v$. The reference set R, target-nearest anchor a, and injection ray $z(\lambda)$ in Eq. (4) are all defined in full-model submission space. When a defense requires delta-form inputs, it centers each submission as $x_i - w_t$ and restores w_t after aggregation. This common translation preserves pairwise Euclidean distances, the nearest anchor, the residual $v - a$, and the majority-support geometry.

Block scoring and refresh. Scores and the selected-block mask are first computed on the first attacked round. They are refreshed once at least ten rounds have elapsed since the previous refresh; intervening attacked rounds reuse the selected block names. Equal scores follow the block-construction order, with lower-indexed blocks selected first within each layer.

Before training in the main experiments, label-stratified 10% scoring holdouts disjoint from compromised clients' local attack-training subsets are pooled into one fixed panel of $P \le 128$ labeled examples, approximately balanced across compromised clients. Each refresh recomputes the scores and selected-block mask on this same panel. The examples are processed with the configured mini-batch size and without shuffling. Scoring loads ϑ_v into an isolated model copy in evaluation mode while retaining automatic differentiation. For each example e, we compute the layerwise cross-entropy gradient $q_\ell^{(e)}$. We average these gradients to obtain g_ℓ and accumulate $P^{-1}\sum_{e=1}^{P} q_\ell^{(e)}(q_\ell^{(e)\top}\theta_\ell)$ to evaluate $\widehat{F}_\ell\theta_\ell$. Non-trainable batch-normalization buffers retain target-model values during scoring and anchor values during injection.

Reproducibility. We use five run seeds, $\{0, 1, 2, 3, 4\}$, each controlling both NumPy and PyTorch randomness. For each seed, the clean run, base attack, Pill enhancement, and GRASP enhancement share the same client partition and model initialization. All attacks are active from the first communication round of each run. The software environment uses Python 3.10.16, PyTorch 2.6.0, torchvision 0.21.0, CUDA 12.4, NumPy 2.2.2, SciPy 1.15.1, scikit-learn 1.6.1, and hdbscan 0.8.40. The reported runs used Ubuntu 22.04 LTS with 24 GB of GPU memory.

C.5 Empirical Rationale for Shapley Block Selection

We examine whether conditioning the scores on the target update improves the resulting attack and whether the selected blocks are attack-specific. The controlled component ablation in Figure 3(a) provides the main selection-rule comparison: under a fixed block budget, injection ray, and scale adjustment, Shapley-guided selection has the highest mean Attack Impact among the evaluated rules. Here we further focus on scoring location and selected-block overlap.

Effect of scoring location. To isolate how scoring location changes block selection at a fixed state, we compare Global (w_t), Anchor ($w_t + a$), and Target ($w_t + v$) scoring models while keeping the formula and weights in Eq. (3) fixed. We use the training configuration from Section 5.2 with fixed residual scale $\lambda = 1$, without majority-aware scaling. We evaluate Min-Max, ALIE, and Sign Flipping against FLAME, Multi-Krum, DnC, Bulyan, and SignGuard, giving 45 runs across the three scoring locations.

At each refresh within each training trajectory, we additionally compute all three masks using the same model state, target update, anchor, and scoring sample. These diagnostic computations preserve the training random-number state and do not change the active scoring strategy. We average pairwise Jaccard overlap over ten refreshes and 15 combinations of attacks and defenses within each trajectory, then give the three trajectories equal weight. Final Attack Impact compares the complete training runs produced by the three scoring strategies.

Table 8: Effect of scoring location under the training configuration in Section 5.2 at fixed $\lambda = 1$. Attack Impact is averaged over five defenses. Jaccard compares selected block sets at the same state and is averaged equally across the three training trajectories. S.F denotes Sign Flipping.

(a) Attack Impact.

Scoring	Min-Max	ALIE	S.F	Average
Global	40.79	29.90	45.30	38.66
Anchor	40.20	30.33	44.01	38.18
Target	**41.65**	**30.69**	**49.04**	**40.46**

(b) Average same-state Jaccard.

Scoring model	Global	Anchor	Target
Global	1.000	0.585	0.555
Anchor		1.000	0.640
Target			1.000

Target achieves the highest defense-averaged Attack Impact for all three base attacks in Table 8a. Its overall average is 40.46 points, exceeding Global and Anchor by 1.80 and 2.28 points, respectively. Table 8b reports average Jaccard overlaps of 0.585 for Global–Anchor, 0.555 for Global–Target, and 0.640 for Anchor–Target. The Attack Impact comparison provides the effectiveness evidence, whereas the overlaps confirm that changing the scoring location materially changes the selected blocks. Under this fixed-state diagnostic, conditioning on the target update therefore outperforms the two evaluated non-target scoring locations.

Selected-block overlap across base attacks. To describe how selections vary across base attacks, we compute pairwise Jaccard overlap between their structured block masks. For each defense, attacks are compared at ten matched score-refresh rounds; the resulting overlaps are averaged within each defense, and Table 9 reports the average across nine defenses. These comparisons involve different training trajectories, so matching refresh rounds does not hold the model state fixed.

Table 9: Pairwise Jaccard overlap of the structured blocks selected for different base attacks. Lower off-diagonal values indicate more attack-specific selection patterns.

Attack	IPM	Min-Max	Min-Sum	ALIE	S.F
IPM	1.00	0.20	0.21	0.26	0.27
Min-Max		1.00	0.42	0.20	0.21
Min-Sum			1.00	0.21	0.22
ALIE				1.00	0.21
S.F					1.00

With the exception of Min-Max–Min-Sum, the average overlap ranges from 0.20 to 0.27, indicating that most selected blocks differ across base attacks. Min-Max and Min-Sum exhibit a higher overlap of 0.42, consistent with their shared parameterization around the mean of honest updates and their difference primarily in the distance constraint. This cross-attack comparison supports the descriptive claim that masks are attack-specific, but it does not isolate the causal effect of scoring blocks on the target model because the underlying trajectories also differ. It is therefore not directly comparable to the same-state overlaps in Table 8b.

C.6 Effectiveness of Majority-Aware Scaling

We next characterize how majority-aware scaling modifies different base attacks. The controlled component ablation in Figure 3(b) provides the main effectiveness result: with the selected blocks and injection ray fixed, majority-aware scaling outperforms the unscaled and Pill-style alternatives for all five base attacks. The appendix uses the realized scales and a matched Mimic baseline to explain this behavior rather than repeat the component comparison.

Adaptation across base attacks. We use the configuration from Section 5.2 and summarize the logged round-wise values across nine defenses. This pool differs from the five-defense component ablation in the main paper and is used only to diagnose the behavior of the selected scales.

Table 10 shows three distinct operating regimes. IPM and ALIE predominantly extrapolate the selected residual, with median scales of 4.45 and 2.95, respectively. Sign Flipping also favors extrapolation. Min-Max operates closer to the unscaled injection, with a median of 1.02. In contrast, Min-Sum is predominantly contracted or removed: its median is 0.45. The coexistence of extrapolation ($\lambda^* > 1$), contraction ($0 < \lambda^* \leq 1$), and anchor fallback ($\lambda^* = 0$) shows that no single fixed-scale

Table 10: Behavior of the adaptively selected scale λ^* in the main Attack Impact (AI) evaluation (Section 5.2) and the matched anchor-copying control. Each row pools round-wise solver outputs across nine defenses. The last column reports mean Attack Impact across the same nine defenses for Mimic; parentheses report $\Delta\mathrm{AI} = \mathrm{Full} - \mathrm{Mimic}$, where Full is the corresponding nine-defense mean for GRASP in Table 2. All observed zero-scale outputs correspond to solver-recorded infeasibility; no feasible optimum at zero occurred.

Base attack	**Median [Q_1,Q_3]**	**Range**	**Fallback ($\lambda^* = 0$)**	$\lambda^* > 1$	**Mimic AI (ΔAI)**
IPM	4.45 [4.01, 5.04]	0.00–9.48	10.2%	89.8%	20.05 (+14.66)
Min-Max	1.02 [0.00, 1.33]	0.00–5.49	27.8%	52.8%	18.09 (+30.12)
Sign Flipping	2.32 [0.23, 3.09]	0.00–4.28	20.7%	63.5%	28.95 (+21.44)
ALIE	2.95 [2.39, 3.49]	0.00–7.11	6.7%	93.2%	21.49 (+10.65)
Min-Sum	0.45 [0.00, 0.89]	0.00–1.97	33.7%	14.0%	16.94 (+17.80)

regime describes all five base attacks. The observed scales vary with the injection direction and the evolving reference geometry.

Ruling out anchor copying. To distinguish residual injection from the $\lambda = 0$ fallback, we include a matched Mimic baseline that submits the target-nearest honest update a_t on every round. A fallback sets the injected perturbation $\lambda\delta$ to zero but does not produce an attack-neutral submission: duplicating the same honest anchor implements the Mimic attack. Under the same nine-defense configuration used for the scale statistics, the full method exceeds Mimic for all five attacks by 10.65–30.12 Attack Impact points. The improvement therefore cannot be attributed solely to copying the nearest anchor.

The distribution of λ^* is descriptive rather than an independent effectiveness test. Because the shared scalar couples all selected blocks, one block can restrict the scale applied to others; layer- or block-specific scaling remains an important direction for future work.

C.7 Sensitivity Analysis

Sensitivity to malicious client fraction. Table 11 reports MAR and TNR by base attack and malicious client fraction, together with their averages.

Table 11: FLAME filtering outcomes by base attack and malicious client fraction. Values are percentages averaged over rounds.

Attack	**10%**		**20%**		**30%**		**40%**	
	Base	**Ours**	**Base**	**Ours**	**Base**	**Ours**	**Base**	**Ours**
Malicious admission rate (MAR; higher is more evasive)								
IPM	63.00	96.00	100.00	99.00	80.00	100.00	68.00	100.00
Min-Max	36.00	47.00	19.00	97.00	37.00	97.00	43.00	100.00
Sign Flipping	11.20	23.00	15.30	68.00	20.73	58.00	29.45	99.00
ALIE	91.00	100.00	36.00	100.00	20.00	100.00	0.00	100.00
Min-Sum	0.00	86.00	0.00	98.00	0.00	100.00	0.00	100.00
Average	40.24	**70.40**	34.06	**92.40**	31.55	**91.00**	28.09	**99.80**
True-negative rate (TNR; higher retains more honest clients)								
IPM	64.00	50.49	47.20	41.83	77.86	33.37	67.17	23.67
Min-Max	59.11	59.40	71.75	42.63	66.34	48.57	63.10	44.27
Sign Flipping	64.49	65.18	73.25	56.68	70.71	54.91	76.90	41.90
ALIE	68.42	53.96	67.55	48.93	74.29	73.97	87.00	53.17
Min-Sum	63.62	56.87	72.73	44.60	78.49	55.09	87.23	43.97
Average	**63.93**	57.18	**66.50**	46.93	**73.54**	53.18	**76.28**	41.40

Sensitivity to block selection rate. Table 12 shows that the default 10% BSR is competitive across all three base attacks, although no single rate is uniformly optimal. Its Attack Impact is highest for ALIE (24.74 points), within 0.73 points of the Min-Max maximum at 30%, and within 2.22 points of the Min-Sum maximum at 50%. Averaged across the three attacks, the default yields 31.66 points,

Table 12: Sensitivity to the block selection rate (BSR), defined as the percentage of structured blocks selected by count within each layer rather than the percentage of parameters modified. A BSR of 100% denotes all-block injection. AI here is Attack Impact averaged across Multi-Krum, SignGuard, DnC, and FLAME. The remaining columns report the fallback and extrapolation rates of the adaptively selected scale λ^*, pooled across all 100 attacked rounds and the same four defenses. Fallback denotes the fraction of outputs with $\lambda^* = 0$. The dagger marks the default BSR.

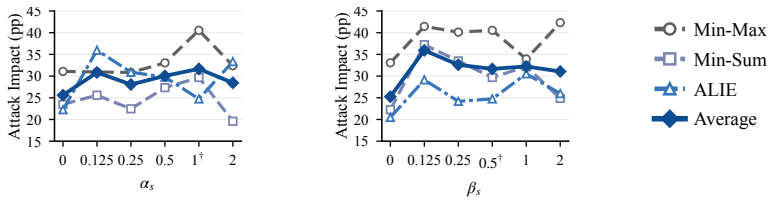
BSR	Min-Max			Min-Sum			ALIE		
	AI	Fallback	$\lambda^* > 1$	AI	Fallback	$\lambda^* > 1$	AI	Fallback	$\lambda^* > 1$
5%	38.40	27.8%	51.5%	30.75	43.0%	18.0%	24.51	12.2%	87.8%
10%†	40.55	27.0%	52.0%	29.68	34.2%	8.2%	24.74	9.5%	90.2%
20%	36.26	27.0%	46.0%	30.04	43.5%	2.0%	23.93	17.8%	81.0%
30%	41.28	15.8%	44.5%	30.08	44.2%	0.0%	23.85	15.5%	81.8%
50%	40.44	14.8%	30.2%	31.90	41.0%	0.0%	22.74	19.8%	80.0%
100%	40.40	17.5%	43.8%	30.84	39.8%	0.0%	14.90	3.5%	96.5%

only 0.08 points below the largest average of 31.74 at 30%. All-block injection (100% BSR) is less effective: the average falls to 28.71 points, with Min-Sum at 30.84 and ALIE decreasing to 14.90. Thus, expanding the mask does not monotonically strengthen the attack, supporting a sparse block selection based on scores evaluated on the target model rather than indiscriminate all-block injection.

The scale statistics further show that BSR changes the geometry of the selected residual differently for each base attack. Min-Max has fallback rates of 14.8–27.8% and extrapolates in 30.2–52.0% of rounds. Min-Sum has higher fallback rates of 34.2–44.2%, and its extrapolation rate falls to zero for BSRs of 30% or larger. In contrast, ALIE extrapolates in at least 80.0% of rounds at every rate. These scale regimes do not translate directly into Attack Impact; for example, ALIE extrapolates in 96.5% of rounds at 100% BSR but attains its lowest Attack Impact. This confirms that the effect of λ^* depends jointly on the selected residual direction and the downstream defense, rather than on scale alone.

Sensitivity to scoring weights. We varied the individual-term weight α_s while fixing $\beta_s = 0.5$. We then varied the cooperative-term weight β_s while fixing $\alpha_s = 1$ on MNIST with LeNet under non-IID data. Figure 4 reports average Attack Impact across Multi-Krum, SignGuard, DnC, and FLAME for each base attack and their average.

For α_s, the average increased from 25.61 points at $\alpha_s = 0$ to 31.66 points at the default $\alpha_s = 1$, before declining to 28.44 points at $\alpha_s = 2$. ALIE instead peaked at 35.96 points for $\alpha_s = 0.125$, whereas Min-Max and Min-Sum attained their maxima at $\alpha_s = 1$, reaching 40.55 and 29.68 points, respectively. For β_s, the average peaked at 35.93 points for $\beta_s = 0.125$, which was 4.27 points above the default value of 31.66. The attack-specific maxima occurred at $\beta_s = 0.125$ for Min-Sum (37.22), $\beta_s = 1$ for ALIE (30.53), and $\beta_s = 2$ for Min-Max (42.32). These distinct optima show that neither weight uniformly controls attack strength across base attacks.



(a) (b)

Figure 4: Sensitivity to the scoring weights: (a) individual-term weight α_s with $\beta_s = 0.5$; (b) cooperative-term weight β_s with $\alpha_s = 1$. The daggers mark the default settings.

Sensitivity to attacker-local scoring data. We study the two quantities that determine how much attacker-local data enters block scoring: the scoring fraction r_{score} and the fixed-panel size cap $B_{\max}$.

Table 13: Sensitivity of Attack Impact to attacker-local scoring data.

Attack	**Defense**	Scoring fraction r_{score} ($B_{\max} = 128$)			Scoring cap $B_{\max}$ ($r_{\text{score}} = 10\%$)		
		5%	**10%**	**20%**	**32**	**64**	**128**
Sign Flipping	FLTrust	16.56	17.99	14.48	14.58	14.27	17.99
	FLAME	72.19	64.44	32.56	45.84	35.73	64.44
	Multi-Krum	42.42	73.03	72.51	65.94	44.71	73.03
IPM	FLTrust	16.79	19.59	15.21	18.54	17.71	19.59
	FLAME	4.81	8.89	6.33	3.35	3.76	8.89
	Multi-Krum	72.33	63.06	72.81	63.66	72.29	63.06
Min-Max	FLTrust	17.59	18.98	14.79	14.91	14.95	18.98
	FLAME	63.21	63.10	56.59	71.54	71.83	63.10
	Multi-Krum	72.33	63.16	62.02	59.63	58.43	63.16
Average		**42.03**	**43.58**	**38.59**	**39.78**	**37.08**	**43.58**

The experiment covers Sign Flipping, IPM, and Min-Max against FLTrust, FLAME, and Multi-Krum on MNIST under the configuration in Section 5.2.

For the fraction sweep, each compromised client assigns r_{score} of its local partition to a label-stratified scoring holdout and uses the disjoint complement for local attack training. The 5%, 10%, and 20% settings therefore use 95%, 90%, and 80% for training; their pooled scoring sets contain 691, 1,383, and 2,764 examples. The panel cap is fixed at $B_{\max} = 128$. In the cap comparison, all settings use 90% for training and a 10% scoring holdout per compromised client.

Table 13 reports average Attack Impact of 42.03, 43.58, and 38.59 points at scoring fractions of 5%, 10%, and 20%; the largest deviation from the 10% default is 4.99 points. For caps of 32, 64, and 128, the comparison gives 39.78, 37.08, and 43.58 points, with a largest deviation of 6.50 points from its 128-example reference.

The aggregate averages conceal substantial setting-specific variation. Across the three scoring fractions, the largest within-setting range is 39.63 points for Sign Flipping with FLAME; across the three caps, it is 28.71 points for the same setting. Overall, average Attack Impact remains between 37.08 and 43.58 points across the tested settings, while the setting-level results show that scoring-data availability interacts with the base attack and defense.

C.8 Runtime and Scalability

Our runtime evaluation aims to quantify the computational overhead of GRASP and assess how it scales with client count and model size. We measure the time for the GRASP wrapper to construct one shared malicious update and materialize the m submissions. Figure 5 separately varies client count and model size on synthetic inputs.

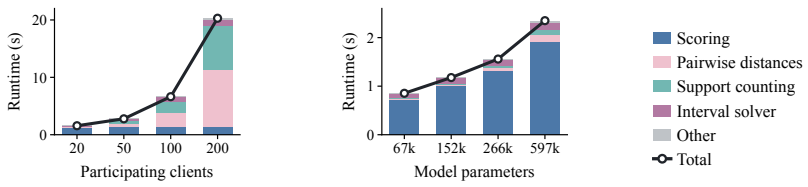


Figure 5: Runtime and scalability of a score-refresh round: (a) client count with the 266,060-parameter CNN fixed; (b) model parameter count with 20 clients fixed. Colors show the stage breakdown; the black line joins total times. Each bar and marker use the median-total repetition of five timed runs.

The runtime evaluation excludes client training and server aggregation. We use a fixed 128-example synthetic scoring panel, keep the malicious fraction at 20%, run one warm-up, and time five repetitions per setting. Each plotted refresh round computes scores once; the stacked phases and total are

taken from the repetition with the median total time, so every stack sums exactly to its total-time marker. "Other" includes anchor selection, block construction and ranking, submission materialization, and remaining overhead.

Figure 5(a) varies the number of clients from 20 to 200 with the 266,060-parameter Fashion-MNIST CNN fixed. Total time rises from 1.57 to 20.30 s. At 200 clients, pairwise distances and support counting together take 17.65 s, about 87% of the measured total; these reference-set operations dominate the larger federation. In Figure 5(b), we fix 20 clients and widen the same CNN from 67,160 to 596,710 parameters. Total time increases from 0.85 to 2.35 s, primarily as scoring grows from 0.73 to 1.92 s.

Overall, GRASP completes a score-refresh round in at most 2.35 s across the tested model sizes with 20 clients, demonstrating second-scale update construction in this setting. However, reference-set processing becomes the main bottleneck as client count increases, limiting scalability even when scoring is reused through cached masks. The results therefore support computational feasibility within the tested small-federation setting, rather than uniformly low overhead as the federation grows.

D GRASP ATTACKER-LOCAL VARIANT

To test whether GRASP depends on observing honest-client updates, we introduce GRASP with attacker-local references, termed the GRASP attacker-local variant. It requires no honest-client information beyond that already assumed by the underlying base attack. Conditional on the target update v, GRASP variant replaces GRASP's honest-update reference set with $R = \{u_j : j \in \mathcal{M}\}$, where each u_j is a compromised client's pre-attack update. Block injection selects the member of this set nearest to v as the anchor, which represents attacker-local training geometry rather than an observed honest update. The same set supplies the distance geometry for the majority-aware scale selection.

Consequently, the complete GRASP variant pipeline is attacker-local for base attacks constructed from compromised-client information alone. In our evaluation, Sign Flipping and ALIE are fully attacker-local, whereas IPM, Min-Max, and Min-Sum retain their base attacks' honest-update assumptions; for the latter attacks, only the GRASP reference construction is attacker-local.

D.1 CALIBRATION

Because GRASP variant cannot measure the honest distance envelope directly, it estimates a local distance quantile and extrapolates the unobserved envelope through Blom's Gaussian order-statistic approximation Blom (1958). The majority proportion is mapped to the smaller local reference set before the unchanged interval-overlap solver is applied. For the frozen pre-attack reference set R, define the off-diagonal distance sample

$$\mathcal{D}_R = \{\|u_i - u_j\|_2 : 1 \le i < j \le |R|\}. \tag{24}$$

Let ρ be the majority-support proportion defined in the main paper. The implementation maps it to the observed sample through

$$q = \min\{|R|, \lceil \rho|R| \rceil\}, \qquad p = 100\rho. \tag{25}$$

We define $\widehat{\tau}$ as the pth percentile of $\mathcal{D}_R$. The radius is corrected only for the difference in reference-set size. For a sample count k, we use Blom's rankit approximation Blom (1958), $b(k) = \Phi^{-1}((k - 3/8)/(k + 1/4))$. The radius is

$$\gamma_{\mathrm{B}} = \max\left\{1, \frac{b(n-m)}{b(|R|)}\right\}, \qquad \tau^{\mathrm{local}} = \gamma_{\mathrm{B}}\widehat{\tau}. \tag{26}$$

The estimator rescales the observed radius and uses the same frozen R for the anchor, pairwise-distance sample, and feasible intervals. The inputs to GRASP variant contain only compromised clients' pre-attack updates.

The estimator requires $|R| > 1$ and $n - m > 1$. The lower bound $\gamma_{\mathrm{B}} \ge 1$ prevents population extrapolation from shrinking the observed radius. The support constraint uses q, rather than the unscaled honest-client count, so it never requests more neighbors than R contains.

D.2 EVALUATION

We compare the base attacks, GRASP, and GRASP variant using the configuration, base attacks, and defenses from Section 5.2, covering 45 combinations of attacks and defenses with five runs per setting.

Table 14: Attack Impact across nine Byzantine-robust defenses. The GRASP attacker-local variant (shown as +GRASP variant) constructs its reference geometry from compromised clients' pre-attack updates. Sign Flipping and ALIE are fully attacker-local, whereas IPM, Min-Max, and Min-Sum retain their base attacks' honest-update assumptions. Base and GRASP results reproduce the main Attack Impact evaluation in Section 5.2.

Attack / Method	FLTrust	FLAME	Multi-Krum	Bulyan	Sign Guard	DnC	C.C	FL Detector	SkyMask
IPM	13.43	7.42	**72.29**	8.22	1.10	8.04	1.93	**46.59**	17.28
+GRASP	19.59	**8.89**	63.06	**81.53**	24.56	**32.28**	16.82	43.13	**22.55**
+GRASP variant	**21.20**	4.03	54.28	54.08	**48.00**	8.12	**40.26**	15.15	12.30
Min-Max	13.77	**72.62**	63.13	68.08	14.34	11.91	80.82	65.26	17.97
+GRASP	18.98	63.10	**63.16**	**81.17**	**15.83**	20.12	**81.43**	66.86	**23.26**
+GRASP variant	**21.19**	60.58	62.96	77.92	14.84	**39.60**	80.82	**75.26**	21.20
Sign Flipping	15.46	13.76	9.66	50.95	14.33	7.68	69.92	71.28	**31.53**
+GRASP	17.99	**64.44**	**73.03**	**67.68**	**32.82**	**20.94**	75.10	**77.68**	23.81
+GRASP variant	**21.15**	64.09	64.76	60.91	23.40	3.37	**81.04**	47.72	20.55
ALIE	24.12	4.13	23.30	71.24	3.16	6.49	5.78	2.06	10.56
+GRASP	**35.70**	**9.60**	**63.93**	75.99	13.22	12.19	59.48	7.49	11.67
+GRASP variant	32.28	0.61	54.77	**81.47**	**72.73**	**34.87**	**81.05**	**12.73**	**25.09**
Min-Sum	13.43	8.65	10.07	**78.84**	15.08	7.36	20.82	71.25	3.19
+GRASP	18.76	**36.68**	**50.55**	60.99	**22.66**	8.81	26.64	68.15	**19.46**
+GRASP variant	**21.12**	20.60	44.38	70.22	20.25	**14.23**	**28.88**	**72.87**	16.14
Average across all 45 settings: Base 27.74; +GRASP 40.04; +GRASP variant 40.07.									

Table 14 reports the complete results. GRASP variant achieved an average Attack Impact of 40.07 points, compared with 27.74 for the base attacks and 40.04 for GRASP. This result is 12.33 points above the base attacks and 0.03 points above GRASP. At the base-attack level, GRASP variant exceeded the base attack for all five attacks and exceeded GRASP for Min-Max and ALIE. Across the 45 combinations of attacks and defenses, GRASP variant led in 18 settings, GRASP in 22, and the base attack in five.

D.3 CONCLUSION

Overall, GRASP variant retains the attack-strengthening effect when its reference geometry is constructed solely from compromised clients' pre-attack updates. Across the 45 evaluated settings, its average Attack Impact exceeds that of the base attacks by 12.33 points and closely matches GRASP. The improvement over each of the five base attacks, averaged across defenses, supports the utility of attacker-local references beyond a single attack construction.

These findings suggest that compromised clients' pre-attack updates can provide useful reference geometry for both residual injection and majority-aware scale selection. The wrapper's empirical benefit therefore does not appear to depend exclusively on constructing its references from honest-client updates. The two reference choices nevertheless produce different outcomes across attack–defense combinations, with neither consistently outperforming the other. Taken together, the results support attacker-local reference construction as a viable alternative that preserves GRASP's aggregate attack-strengthening effect in the evaluated configuration.