# SwarmShield: A Fully Decentralized Trust-Based Defense Against Adversarial Poisoning in Federated Learning

Cynara Justine[1], Sathyanarayanan Manamohan[2,3], Linu Shine[1], Jiji C. V.[3]
[1]College of Engineering Trivandrum, APJ Abdul Kalam Technological University, Kerala, India
[2]Hewlett Packard Enterprise, Bangalore, India
[3]Shiv Nadar University, Chennai, India
Email: cynara@cet.ac.in, sathya@hpe.com, linushine@cet.ac.in, jijicv@snuchennai.edu.in

**Keywords:** Cybersecurity, adversarial attack, data poisoning attack, federated learning, swarm learning, decentralized defense

**Received:** June 1, 2025

*Adversarial poisoning attacks in federated learning systems can severely compromise model integrity, especially when malicious nodes inject corrupted updates. Existing defenses often rely on a trusted central aggregator, introducing a Single Point of Failure and limiting scalability. To overcome these challenges, we propose SwarmShield, a decentralized, trust-aware defense framework based on Swarm Learning. SwarmShield eliminates the need for a central coordinator by redistributing trust evaluation and model merging across peer nodes. It selectively transmits intermediate model layers, applies dimensionality reduction, and clusters parameter vectors to assess similarity. Trust scores are dynamically computed for each node based on its proximity to the cluster centroid, and nodes with low trust are excluded from aggregation. A secure, trust-weighted averaging mechanism is used for model updates, with integrity ensured through cryptographic hashing and blockchain logging. Extensive experimentation with different types of adversarial data poisoning attacks on CIFAR10 dataset with Resnet50 model demonstrate an average improvement in accuracies by 24.8%. Additionally, its generalizability is robustly demonstrated through successful application to the real-world DermaMNIST medical imaging dataset, where SwarmShield consistently maintained or improved model accuracy across diverse attack scenarios. We also evaluate SwarmShield on the TwoLeadECG time series dataset, highlighting its behavior under temporal adversarial settings. These results validate SwarmShield's effectiveness, scalability, and resilience in adversarial federated learning settings. Further analysis through ablation studies validates the framework's design by quantifying the contribution of each component, while robustness tests demonstrate its resilience across varying ratios of malicious nodes. Our experimental results demonstrate that the proposed approach significantly outperforms existing state-of-the-art methods.*

*Povzetek: Članek predstavi SwarmShield, decentralizirano in zaupanja zavedno obrambo proti zastrupljanju v federiranem učenju, ki brez centralnega agregatorja z gručenjem ocenjuje podobnost posodobitev, dinamično dodeli točke zaupanja in izloča zlonamerne vozle, pri čemer integriteto podpira še kriptografsko zgoščevanje in beleženje v verigi blokov.*

## 1 Introduction

Federated Learning (FL) enables decentralized model training without centralized data collection, preserving data privacy by keeping data on local devices [1]. However, its distributed nature introduces critical security risks, notably adversarial poisoning attacks from compromised nodes. These attacks degrade global model performance through malicious local updates, and are difficult to detect. Centralized defenses, such as server–based parameter aggregation, further introduce Single Points of Failure (SPOF) and elevate the attack surface [2]. Several centralized defense strategies, such as *Krum* [3] and *Foolsgold* [4], etc. have been proposed to resist poisoning and sybil attacks; however, they rely on a trusted aggregator, which remains a critical vulnerability in adversarial settings. Centralized aggregation solutions in FL are vulnerable to manipulation, as a compromised aggregator can distort model updates, leak private data, or collapse system reliability. These solutions also provide attackers a global view of updates, heightening privacy risks.

In contrast, decentralized approaches distribute the trust boundary, improving system resilience. Notable prior works like *Ditto* [5], and Byzantine-tolerant gradient descent techniques [3] have proposed decentralization through trust initialization, personalization, or robust aggregation. However, many existing approaches remain vulnerable to attacks due to their dependence on partial centralization or static trust assumptions.

To address these challenges, we propose *SwarmShield*,

a decentralized trust-aware defense framework that detects and mitigates malicious nodes using collaborative clustering and trust scoring. It enhances adversarial robustness without compromising privacy or requiring centralized coordination, contributing towards secure and resilient FL deployments in real-world distributed environments. *SwarmShield* differs in that it offers a fully decentralized and adaptive peer trust mechanism, integrating layer-wise parameter extraction, clustering-based behavioral analysis, and blockchain-backed tamper-proofing, without requiring global trust anchors or centralized bootstrapping.

The key contributions of this work are as follows:

- **Fully decentralized defense architecture:** We design a central-aggregator-free framework where trust computation, parameter filtering, and model merging are executed collaboratively among peer nodes.
- **Dynamic trust evaluation through clustering:** Each node's behavior is assessed in real time through unsupervised clustering of selected-layer parameter vectors, with trust scores reflecting deviation from cluster centroids.
- **Blockchain-backed integrity and auditability:** SwarmShield employs cryptographic hashing and blockchain logging to secure parameter transmissions and preserve an immutable trust history, enabling tamper-proof, auditable FL.

These contributions collectively strengthen the security of federated learning while maintaining scalability, privacy, and interpretability in adversarial environments.

## 2 Literature survey

Distributed learning architectures underpin modern federated systems, enabling collaboration without direct data sharing. For example, Wu [6] introduced a distributed intelligent optimization algorithm leveraging the Spark framework for large-scale e-commerce data mining. Their approach adapts fitness functions and topological structures for scalable, robust optimization. They proposed a distributed intelligent optimization algorithm for e-commerce user purchase data mining, where the learning objective is designed to balance cluster membership and data representation in a principled manner. This approach relies on modeling spatial membership among data points and systematically updates parameters in parallel across distributed nodes. Such a design enables efficient, real-time adaptation and robust convergence within large-scale distributed systems. Similarly, Ahmed et al. [7] demonstrate the application of vanilla split learning to cyber-physical systems, enabling collaborative model training on local sensor data. Their paradigm ensures privacy and reduces communication, directly addressing resource and privacy constraints prevalent in modern federated settings.

The challenge of ensuring adversarial robustness in Federated Learning, particularly against poisoning attacks, remains a pressing concern. Recent research underscores the importance of secure aggregation in federated learning to ensure privacy and robustness. Notably, a 2025 study introduces a federated learning framework that employs both differential privacy and homomorphic encryption, achieving strong data confidentiality and maintaining high model accuracy even when adversarial clients are present [18]. Ongoing studies continue to address threats such as label-flipping and Byzantine attacks, with decentralization identified as key to enhancing system resilience and mitigating single points of failure. These contributions collectively advance the field's understanding and implementation of secure, robust federated learning architectures.

Early defense efforts primarily focused on centralized aggregation techniques. For example, in [3], Blanchard et al. introduced a Byzantine-resilient rule namely *Krum* that selects gradient updates closest in Euclidean space. While effective in simple settings, it suffers from high computational overhead and limited scalability in high-dimensional, non-IID settings. To address this, in [8] the authors layered an additional filtering step over *Krum*, and proposed a new approach namely *Bulyan* improving robustness but not scalability. Yin et al. [19] sought to optimize statistical guarantees in Byzantine environments, yet their reliance on centralized aggregation makes them ill-suited for dynamic or trustless networks. FoolsGold [9], using cosine similarity to suppress updates from colluding adversaries, innovated in contribution-based weighting but remained sensitive to hyperparameter choices and incurred communication overhead.

Shifting away from centralized designs, Warnat-Herresthal et al. [20] introduced Swarm Learning—embedding consensus mechanisms into model training. In parallel, BAFFLE [10] and FLTrust [11] tried to anchor trust via blockchain and trusted initialization, respectively. However, BAFFLE incurred latency from blockchain overhead, while FLTrust reintroduced a central trust bottleneck.

*Ditto* [5] took a personalization route to isolate client poisoning effects but raised concerns about divergence and model consistency. Zhou et al. [12] proposed *Fed_BVA*, integrating bias-variance analysis for adversarial training. Though robust across adversary types and data distributions, it incurred complexity from second-order statistics computation.

To overcome shortcomings of prior defenses, recent works integrate cryptographic and differential privacy tools. Jin et al. [21] incorporated homomorphic encryption to secure model aggregation, providing strong theoretical guarantees. However, its deployment is limited by excessive computational load, which poses a barrier to real-time FL at the edge level. Zhang et al. [22] proposed *DBFAT*, which reweights local losses and globally regularizes model updates. Although effective in accuracy-robustness trade-offs, the approach requires domain-specific hyperparame-

Table 1: Comparison of existing adversarial defense methods in federated learning

| Method | Approach | Advantages | Limitations |
|---|---|---|---|
| *Krum* [3] | Centralized, Byzantine-resilient rule that selects gradients closest in Euclidean space. | Effective in simple settings. | High computational overhead; limited scalability in high-dimensional, non-IID settings. |
| *Bulyan* [8] | Adds a filtering step over Krum. | Improves robustness. | Does not improve scalability. |
| *FoolsGold* [9] | Centralized, uses cosine similarity to weight contributions. | Innovative contribution-based weighting. | Sensitive to hyperparameters; incurs communication overhead. |
| *BAFFLE* [10] | Anchors trust using a blockchain. | Attempts to anchor trust without a central aggregator. | Incurs latency from blockchain overhead. |
| *FLTrust* [11] | Anchors trust via a trusted initialization process. | Attempts to anchor trust. | Reintroduces a central trust bottleneck. |
| *Ditto* [5] | Personalization to isolate clients. | Isolates effects of client poisoning. | Raises concerns about model divergence and consistency. |
| *Fed_BVA* [12] | Integrates bias-variance analysis for adversarial training. | Robust across various adversary types and data distributions. | Incurs complexity from second-order statistics computation. |
| *FL-Defender* [13] | Uses PCA and angular gradient similarity for detection. | Computationally efficient. | Performance degrades with high-dimensional models. |
| *FedBayes* [14] | Reframes aggregation through a Bayesian lens. | Enhances trust minimization. | Introduces probabilistic modeling complexity and latency. |
| *Zeno* [15] | Uses suspicion scores from validation loss to evaluate updates. | Provides a certifiable robustness measure under certain assumptions. | Practical deployment requires managing validation overhead. |
| GAN-based Defenses [16] | Use generative models for input sanitization and anomaly repair. | Can detect and neutralize poisoned samples by learning benign data distributions. | Significant training complexity; potential trade-offs with model utility. |
| Moving Target Defense [17] | Decentralized, periodically perturbs model parameters. | A novel approach to confuse potential adversaries. | May slow down model convergence. |
| **SwarmShield (Ours)** | **Fully decentralized, trust-based defense using collaborative clustering and blockchain logging.** | **Eliminates central point of failure; adaptive trust; provides auditable integrity.** | **Performance depends on the quality of clustering and inter-node communication.** |

ter tuning. FL-Defender [13] innovates with PCA and angular gradient similarity for targeted attack detection. Although computationally efficient, its performance degrades with high-dimensional models.

*FedBayes* [14] reframes aggregation through a Bayesian lens, enhancing trust minimization. However, this introduces probabilistic modeling complexity and latency.

Xu et al.[23] proposed *Dual Defense*, which simultaneously enhances privacy and defends against poisoning by combining differential privacy with robust aggregation, but demands tight tuning to balance noise and utility.

Robustness against poisoning attacks in federated learning has evolved through both theoretical and empirical defense approaches. Yin et al. [19] introduced a Byzantine-robust distributed learning method with optimal statistical guarantees, enabling formal treatment of adversarial tolerance in aggregation. Although promising, scalability remains challenging in highly heterogeneous environments. Zeno, proposed by Xie et al. [15] is a defense mechanism that evaluates update trustworthiness using suspicion scores derived from validation loss. This provides a certifiable robustness measure under certain assumptions, though practical deployment requires managing validation overhead. In parallel, generative approaches have emerged as a complementary defense strategy. Generative approaches, particularly those inspired by Generative Adversarial Networks (GANs), are employed for backdoor defense through mechanisms like input sanitization and anomaly repair. Zhang et al. [24] provide a comprehensive survey on how GANs can detect and inpaint anomalous regions to neutralize poisoned samples, leveraging learned benign data distributions. Comprehensive surveys, such as Akhtar et al. (2021), synthesize recent advances in both generative and statistical defenses, noting their promise despite inherent challenges like significant training complexity and trade-offs with model utility [16, 25]. Ye et al. [17] incorporated *Moving Target Defense* in decentralized FL, where the model parameters are periodically perturbed to confuse potential adversaries. Although novel, this may slow the convergence.

Despite their innovative approaches, most existing frameworks face critical limitations, including reliance on a non-malicious central aggregator, lack of mechanisms for malicious node removal, and inability to maintain learning continuity after an attack.

Many recent works have focused on improving privacy and computational efficiency in distributed machine learning frameworks, aligning with the motivations behind our work. Azeri et al. [7] introduced an optimized Vanilla Split Learning method for resource-constrained Cyber-Physical Systems, demonstrating how a split neural architecture can reduce client-side computation while preserving data privacy in collaborative environments.

In a complementary study, Chen et al. [18] proposed a federated learning architecture incorporating differential privacy and homomorphic encryption to ensure privacy preservation in large-scale systems. Their experimental setup, which includes a distributed physical server environment, demonstrates robust model performance even under adversarial scenarios, with only a minor trade-off in accuracy. These works validate the growing emphasis on balancing scalability, security, and computational feasibility in distributed learning, which our work advances further by integrating trust-aware node filtering and robust election mechanisms in the context of adversarial federated environments.

A comparison of existing adversarial defense methods in federated learning is shown in Table 1.

# 3 Methodology

The proposed methodology addresses three key issues in FL security: (i) vulnerability of centralized trust mechanisms to single-point failures, (ii) lack of adaptive malicious node detection during aggregation, and (iii) insufficient privacy-preserving verification. We introduce *SwarmShield*, a decentralized defense framework that distributes trust computation across swarm nodes using cryptographic verification, dynamically adjusts node weightage through clustering-based analysis, and enforces zero-trust principles via blockchain-based logging and secured computation.

In this section, we first outline the common adversarial attacks in FL environments, followed by a detailed presentation of our proposed approach.

## 3.1 Adversarial attacks

The four types of adversarial attacks we have waged for evaluating the improvement in robustness by our solution SwarmShield are label flipping attack, extreme noise attack, Projected Gradient Descent (PGD) attack and Fast Gradient Sign Method (FGSM) attack. Swarmshield framework includes custom implementations of the label flipping and extreme noise attacks. For label flipping, the designated malicious peer contaminates its data stream before training, eg: specifically flipping labels “cars” and “birds”. Additionally, the Cleverhans framework [26] is used for generating FGSM (Fast Gradient Sign Method) [27] and PGD (Projected Gradient Descent) [28] adversarial attacks. A brief description of these attacks and its relevance are given below.

### 3.1.1 Label flipping attack

Label flipping is an adversarial technique in which the labels of a subset of training data are intentionally altered to mislead the model’s learning process [29]. This type of attack is relevant in scenarios involving training data poisoning, where the objective is to compromise model integrity by injecting false labels. The impact on model robustness is often severe, particularly in tasks that rely heavily on accurate label information.

### 3.1.2 Extreme noise attack

In extreme noise attacks, random noise with high variance is added to the input data, disrupting the model’s ability to detect meaningful patterns [30]. This attack targets the model’s sensitivity to input perturbations, making it a crucial test for evaluating a model’s noise tolerance.

### 3.1.3 Projected gradient descent (PGD) attack

PGD is a powerful iterative adversarial attack that generates perturbations by taking multiple small steps along the gradient of the model’s loss function. It is considered one of the most effective attacks for evaluating model robustness against adversarial examples, due to its optimization-based approach that converges to the worst-case perturbation within a given constraint [28].

### 3.1.4 Fast gradient sign method (FGSM) attack

FGSM is a single-step adversarial attack where perturbations are crafted using the sign of the gradient of the loss function with respect to the input data [27]. It is widely used for its simplicity and speed, making it a baseline for evaluating model vulnerability to adversarial examples. The attack underscores the importance of gradient-based defenses in securing deep learning models.

## 3.2 Proposed SwarmShield framework

SwarmShield is a decentralized framework built atop the Swarm Learning (SL) paradigm [31]. It provides defense against adversarial attacks by executing collaboratively across peer nodes using cryptographic guarantees and clustering-based trust mechanisms. In this section, we discuss the various features of Swarmshield framework such as decentralized peer collaboration without central aggregator, trust-based peer monitoring, dynamic peer election, blockchain-backed tamper-proofing and auditability and efficiency optimizations for edge deployment. We also present the algorithm, workflow and flowchart.

#### 3.2.1 Decentralized peer collaboration without central aggregator

In Swarm Learning (SL), the system operates over $R$ communication rounds. In each round $r \in \{1, \ldots, R\}$, every peer node $P_i$ trains its local model on its private dataset $D_i$. The training is performed for $E$ local epochs.

The local model weights for peer $i$, denoted by $\theta_i$, are updated iteratively. For a single local update step $k$ within a communication round $r$, the update rule is:

$$\theta_{i,k+1}^{(r)} = \theta_{i,k}^{(r)} - \eta \cdot \nabla_\theta \mathcal{L}(D_i; \theta_{i,k}^{(r)}) \tag{1}$$

where:

- $\theta_{i,k}^{(r)}$ are the weights of peer $i$'s model at local step $k$ of communication round $r$.
- $\eta$ is the learning rate.
- $\mathcal{L}(D_i; \theta_{i,k}^{(r)})$ is the loss function evaluated on the local dataset $D_i$ using the current weights.

After $E$ local epochs, the final local model weights $\theta_i^{(r)}$ are shared for aggregation.

Preservation of data privacy is highlighted by the fact that no raw data leaves any of the participating nodes. To minimize communication overhead while preserving critical model updates, each peer transmits only the parameters from a selected subset of layers. These parameters are flattened into a single vector for efficient analysis.

The parameter vector $v_i$ for peer $i$ is constructed as follows:

$$v_i = \text{Flatten}(\{\theta_{i,j} \mid j \in \mathcal{J}\}) \tag{2}$$

where:

- $v_i$ is the flattened parameter vector for peer $i$.
- $\theta_{i,j}$ represents the weight parameters of the $j$-th layer of peer $i$'s model.
- $\mathcal{J}$ is the pre-configured index set of layers selected for transmission and trust evaluation.

This formulation draws upon adaptive communication strategies explored in [32], enabling tunable trade-offs between transmission cost and model fidelity. This also ensures communication efficiency and reduces the risk of sensitive information leakage. For our experiments, we follow the common heuristic of selecting the final fully-connected layers of the ResNet-50 model, as these layers typically capture the high-level semantic representations most vulnerable to adversarial poisoning.

While the optimal layers for transmission are application-specific, the selection process follows a clear heuristic to maximize security and efficiency. For our experiments with the ResNet-50 architecture, we selected only the final fully-connected layers for transmission. This choice is based on the rationale that these deeper layers capture high-level semantic features and are therefore most impactful on the model's final decision. Consequently, they are often the primary target of, and most sensitive to, adversarial poisoning attacks. Transmitting only these layers provides the most relevant information for trust evaluation while minimizing communication overhead.

#### 3.2.2 Trust-based peer monitoring

At each round, the elected leader gathers parameter vectors from participating peers to evaluate their trustworthiness. To ensure operational clarity and address which model layers are transmitted, our protocol focuses on a preconfigured subset of layers known to be most indicative of model behavior, typically the final fully-connected or convolutional layers. This targeted approach balances communication overhead with effective anomaly detection.

Before analysis, Principal Component Analysis (PCA) is applied to the flattened parameter vectors ($v_i$) of these selected layers, which we now denote as $x_i$ after PCA reduction. This dimensionality reduction step enhances the subsequent clustering by capturing the principal axes of variance among peer updates. Peers are then grouped using k-means clustering based on these PCA-reduced vectors. The objective is to partition the set of all peer vectors into $K$ disjoint clusters $S = \{S_1, S_2, \ldots, S_K\}$ by minimizing the variance within the clusters:

$$\arg\min_S \sum_{k=1}^{K} \sum_{x \in S_k} \|x - \mu_k\|^2 \tag{3}$$

where $S_k$ is the set of peer vectors belonging to cluster $k$ and $\mu_k$ is the centroid of that cluster [33].

Pairwise similarity between peer vectors $x_i$ and $x_j$ is computed using cosine similarity:

$$\text{sim}(x_i, x_j) = \frac{x_i \cdot x_j}{\|x_i\| \|x_j\|} \tag{4}$$

To detect anomalies, a hybrid outlier score (OS) is calculated combining distance, similarity, and cluster density:

$$\text{OS}(x_i) = \|x_i - c\| \cdot (1 - \text{sim}(x_i, c)) \cdot \frac{1}{|S_k|} \tag{5}$$

where $|S_k|$ is the number of peer vectors in the cluster containing $x_i$, and $c$ is the centroid of that cluster, inspired by local outlier factor methods [34].

Each peer's trust score $T_i^t$ is updated based on the similarity of its transmitted vector $x_i$ to the main peer cluster centroid $c^t$, and penalized in case of hash mismatches:

$$T_i^t = (1-\alpha) T_i^{t-1} + \alpha \left(1 - \frac{\|x_i - c^t\|}{\max_j \|x_j - c^t\|}\right) - \lambda \cdot \kappa_i^t \tag{6}$$

Here, $\alpha$ is a forgetting factor, and $\kappa_i^t$ is an indicator function such that $\kappa_i^t = 1$ if a hash mismatch is detected for node $i$ at time $t$, and 0 otherwise. The parameter $\lambda$ controls the severity of the penalty applied due to tampering or inconsistency. This memory-aware and tamper-aware formulation ensures that trust scores adapt over time based on both

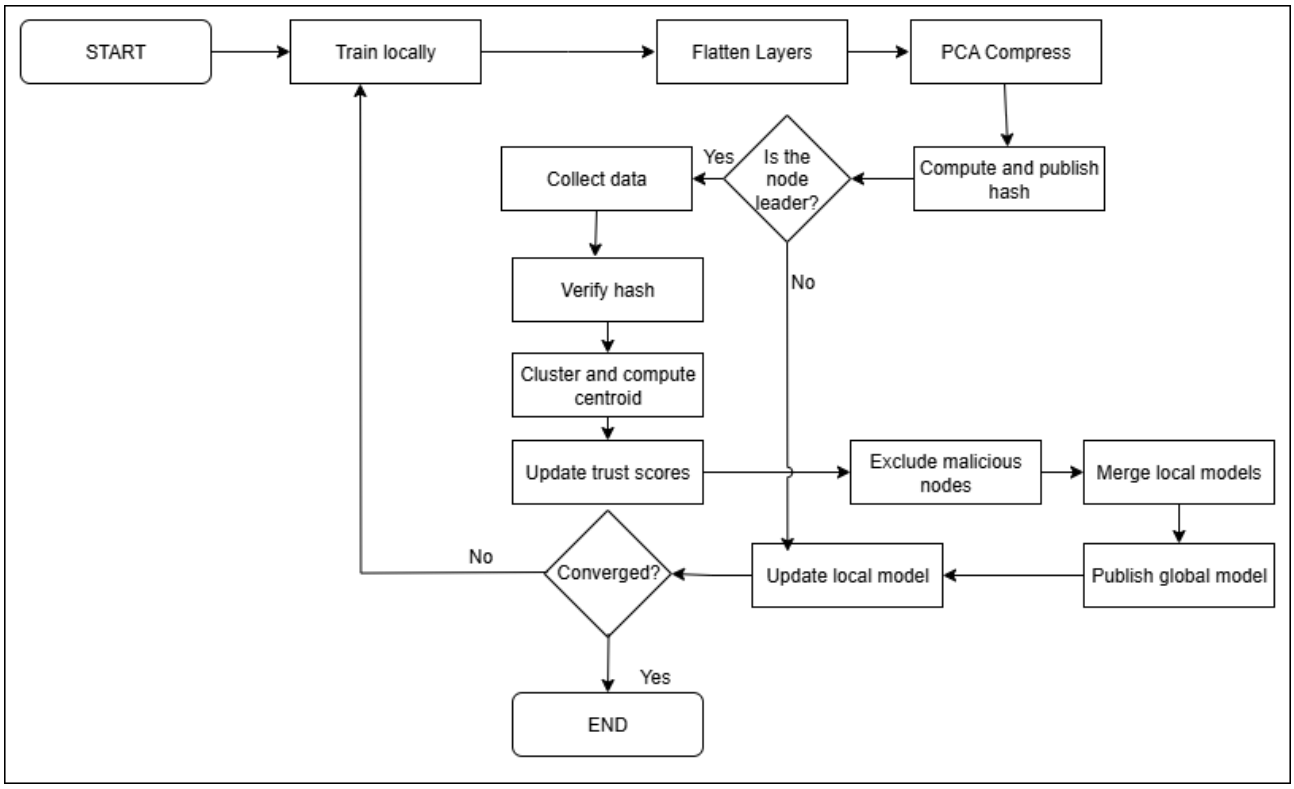



Figure 1: Flowchart illustrating the Compact SwarmShield Protocol. Each component corresponds directly to the operations detailed in Algorithm 1, providing a high-level visual abstraction of the training, compression, trust update, and model aggregation stages.

behavioral consistency and cryptographic integrity. Peers with low trust scores are either downweighted or excluded entirely from the aggregation process.

#### 3.2.3 Dynamic peer election

The leader election mechanism is fully decentralized and implemented using a programmable smart contract on a tamperproof distributed ledger. This eliminates reliance on a central coordinator, addressing a key limitation in baseline systems such as FL-Defender [13].

The protocol for electing a leader in each communication round proceeds as follows:

1. Each node accesses the historical trust scores of its peers from the distributed ledger and casts a vote for another node. Self-voting is explicitly disallowed to prevent self-promotion.

2. The votes are tallied on the ledger, and all nodes arrive at a consensus. The node with the most votes is designated the leader for the round.

3. In the event of a tie, the node with the higher average historical trust score is selected as the tie-breaker.

To further reduce the risk of collusion and deterministic bias, randomness from verifiable external sources is incorporated into the election process. The system also implements a trust decay mechanism, requiring nodes to consistently prove their trustworthiness to maintain a high score.

#### 3.2.4 Blockchain-backed tamper-proofing and auditability

Before synchronization, a cryptographic hash of the selected parameters is computed using Secure Hashing Algorithm SHA-256 [35] to ensure integrity, as shown by:

$$h_i = \text{SHA-256}(x_i) \tag{7}$$

where $h_i$ is the hash digest of parameter $x_i$.

This hash is published to a blockchain ledger, ensuring tamper-proofing and auditability. Trust scores are pushed and retained in the blockchain which is the immutable ledger. During parameter merging, hashes are recomputed and verified. Any mismatch penalizes the responsible peer by reducing its trust score [36].

Based on these scores, node weightages are updated dynamically. To further enforce integrity and fairness, SwarmShield implements a grace threshold mechanism. Peers whose trust scores fall below $\tau^t$ (the dynamic merge threshold) are not immediately excluded but allowed a cooldown period to improve behavior. Such peers may rejoin the network after demonstrating improved behavior, balancing robustness with ethical collaboration.

In addition, during parameter merging, each peer's model update is hashed and published to a blockchain ledger to ensure tamper-proofing and auditability. If the recomputed hash does not match the published value, the responsible peer is penalized by reducing its trust score using a penalty term $\lambda \cdot \kappa_i^t$, where $\kappa_i^t = 1$ in case of mismatch, and 0 otherwise.

The global model aggregation is performed using trust-weighted averaging:

$$w_{\text{global}} = \frac{\sum_{i=1}^{n} T_i \cdot w_i}{\sum_{i=1}^{n} T_i} \tag{8}$$

Only nodes with $T_i > \tau^t$ participate in the merge, where the dynamic trust threshold $\tau^t$ is defined as:

$$\tau^t = \mu^t - \beta\sigma^t \tag{9}$$

Here, $\mu^t$ and $\sigma^t$ denote the mean and standard deviation of trust scores at time $t$, and $\beta$ is a sensitivity parameter.

The sensitivity parameter $\beta$ is crucial for setting the dynamic trust threshold, $\tau^t$. It controls how strictly nodes are filtered for participation in global model aggregation. A higher $\beta$ creates a lower, more lenient threshold, promoting broader participation. Conversely, a lower $\beta$ results in a higher, more stringent threshold, favoring only nodes with trust scores closer to the average. This parameter scales with the standard deviation of trust scores, $\sigma^t$, allowing the system to dynamically adjust its strictness based on the network's trustworthiness. The value of $\beta$ is empirically determined to balance network robustness against the need for sufficient participation. This dynamic system, combined with continuous trust score updates, supports the zero-trust principle by requiring nodes to constantly prove their trustworthiness.

### 3.2.5 Efficiency optimizations for edge deployment

SwarmShield incorporates two key optimizations to reduce computation and support real-time operation:

- **Dimensionality reduction:** PCA is applied on the flattened vectors before clustering to minimize computational overhead. Specifically, PCA is applied to the collection of flattened parameter vectors from all peers during the leader's aggregation step, occurring immediately before the k-means clustering. This reduces the feature space dimensionality, which not only lowers computational cost but also improves the signal-to-noise ratio, thereby enhancing the quality of the subsequent clustering for outlier detection.
- **Incremental clustering:** Clustering state from previous rounds is reused, avoiding redundant computation in stable scenarios.

These enhancements ensure that SwarmShield remains scalable even on low-power edge clusters. Algorithm 1 formally outlines the essence of the SwarmShield framework. The flowchart giving a visual depiction of the workflow for clearer conceptual understanding is shown in 1.

**Algorithm 1** SwarmShield Protocol

**Require:** Model $M_i$ (representing parameters $\theta_i$), data $D_i$, layer index set $\mathcal{J}$
**Ensure:** Robust global model $w_{\text{global}}$

**while** not converged **do**
  Train locally for $E$ epochs: $\theta_i^{(t+1)} = \theta_i^{(t)} - \eta\nabla\mathcal{L}(D_i; \theta_i^{(t)})$
  Flatten selected layers' parameters: $v_i = \text{Flatten}(\{\theta_{i,j} \mid j \in \mathcal{J}\})$
  PCA compress: $x_i = \text{PCA}(v_i)$
  Hash $x_i$: $h_i = \text{SHA-256}(x_i)$; publish $h_i$ to blockchain
  **if** $P_i$ is Leader **then**
    Collect $\{x_j\}, \{h_j\}$ from peers
    **for all** peers $j$ **do**
      Verify SHA-256$(x_j) = h_j$
      **if** hash mismatch **then**
        Mark $P_j$ for exclusion
      **end if**
    **end for**
    Cluster $\{x_j\}$; compute centroid $c^t$
    **for all** verified $j$ **do**
      Compute OS$(x_j)$, update $T_j^t$
      **if** $T_j^t < \tau^t$ **then**
        Mark $P_j$ for exclusion
      **end if**
    **end for**
    $w_{\text{global}}^{(t+1)} = \frac{\sum_j T_j^t \theta_j^{(t+1)}}{\sum_j T_j^t}$
    Publish $w_{\text{global}}^{(t+1)}$, $\{T_j^t\}$ to blockchain
  **else**
    Wait for $w_{\text{global}}^{(t+1)}$ from Leader
  **end if**
  Update $\theta_i^{(t+1)} \leftarrow w_{\text{global}}^{(t+1)}$
**end while**

**Notation:** $i, j, n$: peer indices; $(t)$: round index; $P_i$, $D_i$: peer $i$ and its dataset; $\mathcal{J}$: selected layer indices; $\theta_i^{(t)}$: local model; $v_i$, $x_i$: flattened/PCA vectors; $w_{\text{global}}^{(t)}$: global model; $h_i$: hash for integrity; $c^{(t)}$: cluster centroid; $T_i^{(t)}$, $\tau^{(t)}$: trust score, threshold; $\kappa_i^{(t)}$: hash-fail flag; $\eta$, $\mathcal{L}$: learning rate, loss; $\alpha$, $\lambda$: forgetting factor, trust penalty.

### 3.2.6 Summary of secure, collaborative workflow

The SwarmShield protocol integrates all the above stages into a cohesive cycle executed per communication round:

1. Local training and layer selection on private data.
2. Secure hashing and transmission of compact updates.
3. Leader election and similarity-based peer clustering.
4. Trust score updates and parameter filtering.

5. Trust-weighted model merging and blockchain logging.

The system enforces collaborative defense without centralization, enhances robustness to poisoning, and ensures verifiable model integrity.

### 3.3 Convergence analysis

We provide a simplified convergence analysis of the SwarmShield framework under standard assumptions for federated optimization. While a complete convergence analysis for decentralized, trust-aware federated learning with dynamic leader election is challenging, we outline the proof under the following assumptions:

- The global objective is to minimize a convex function $\mathcal{L}(w)$, which is decomposable as $\mathcal{L}(w) = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_i(w)$, where $\mathcal{L}_i(w)$ local objective (loss) function for peer $i$. Here, $w$ represents the global model weights and $\mathcal{L}(w)$ is the global objective (loss) function;
- Each $\mathcal{L}_i$ is $L$-smooth and convex, which are common assumptions for proving convergence in federated settings [37].
- Cosine similarity–based trust scores $T_i^t \in [0,1]$ are bounded and updated using stable statistics of parameter deviations.
- The trust-weighted model aggregation, a form of robust aggregation [38], at round $t$ is given by:

$$w_{\text{global}}^{(t+1)} = \sum_{i=1}^{N} \alpha_i^t \theta_i^{(t)}, \quad \text{where } \alpha_i^t = \frac{T_i^t}{\sum_{j=1}^{N} T_j^t}, \tag{10}$$

where $\alpha_i^t$ – normalized trust weight for peer $i$ at round $t$, $N$ – total number of peers and $T_i^t$ – trust score of peer $i$ at round $t$;

**Lemma 1 (Trust-weighted averaging preserves descent direction):** If each local model $\theta_i^{(t)}$ performs a gradient step on $\mathcal{L}_i$, then the trust-weighted average update $w_{\text{global}}^{(t+1)}$ remains a descent direction for the global loss $\mathcal{L}$ under bounded trust variation.

*Proof Sketch:* From convexity and $L$-smoothness, the gradient descent step at each peer $i$ satisfies:

$$\mathcal{L}_i(\theta_i^{(t)}) \le \mathcal{L}_i(w_{\text{global}}^{(t)}) + \langle \nabla\mathcal{L}_i(w_{\text{global}}^{(t)}), \theta_i^{(t)} - w_{\text{global}}^{(t)} \rangle + \frac{L}{2}\|\theta_i^{(t)} - w_{\text{global}}^{(t)}\|^2 \tag{11}$$

Using Jensen's inequality and trust-weighted aggregation:

$$\mathcal{L}(w_{\text{global}}^{(t+1)}) \le \sum_{i=1}^{N} \alpha_i^t \mathcal{L}_i(\theta_i^{(t)}) \tag{12}$$

This implies that if each $\mathcal{L}_i(\theta_i^{(t)}) \le \mathcal{L}_i(w_{\text{global}}^{(t)})$ (i.e., local progress), then global progress follows via the trust-weighted combination.

**Remark 1.** *The inequality in Equation 12 implies that, under convex loss functions and trust-weighted aggregation, global progress is guaranteed if each peer achieves local improvement. That is, if $\mathcal{L}_i(\theta_i^{(t)}) \le \mathcal{L}_i(w_{global}^{(t)})$ for all $i$, then $\mathcal{L}(w_{global}^{(t+1)})$ will also decrease, ensuring convergence through decentralized coordination.*

**Theorem 1 (Convergence under stable trust weights):** Suppose that each peer $i$ performs SGD with learning rate $\eta_t = \frac{1}{\sqrt{t}}$ and trust scores $T_i^t$ satisfy:

$$T_i^t \ge \epsilon > 0, \quad \forall i, t \tag{13}$$

where $\epsilon$ – a small positive constant, representing the lower bound for trust scores, $\eta_t$ – learning rate at round $t$; Then the averaged global model $w_{\text{global}}^{(t)}$ converges in expectation to the global minimizer $w^*$ at a rate $\mathcal{O}(1/\sqrt{t})$:

$$\mathbb{E}[\mathcal{L}(w_{\text{global}}^{(t)}) - \mathcal{L}(w^*)] \le \frac{C}{\sqrt{t}} \tag{14}$$

where $C$ – a generic positive constant bounding the convergence rate, $w_{\text{global}}^{(t)}$ – global model weights at round $t$ and $w^*$ – global minimizer of the objective function.

*Proof Sketch:* This follows from standard convergence results for convex SGD (e.g., [1, 37]) when using weighted model averaging, with the additional assumption that trust weights remain bounded away from zero to prevent degenerate aggregation.

**Discussion:** While SwarmShield operates in a non-convex setting (ResNet-50 on CIFAR-10), such theoretical guarantees under convexity assumptions provide a baseline justification. Further extensions would require analyzing the interplay of non-convexity, dynamic leader election, and peer drop-out behavior.

## 4 Experiments, results and discussion

Experiments were conducted on a three-node testbed, with each node configured with 4 CPU cores, 32 GB RAM, and 300 GB of persistent storage. All nodes run Docker containers with PyTorch and the Swarm Learning Framework preinstalled. A 10-Gigabit Ethernet network links the nodes, and secure certificates are used for peer authentication. This environment is designed to emulate real-world federated learning settings with decentralized and privacy-preserving characteristics. Each node, or *peer*, operates independently on its own private dataset and does not share raw data with other nodes. One of the three peers is deliberately configured to inject adversarial perturbations into its data stream, simulating a malicious participant. The identity of the malicious peer is unknown to the others, thus reflecting realistic adversarial conditions.

## 4.1 Experimental setup

The SwarmShield framework has been developed to facilitate seamless evaluation of adversarial defenses in a decentralized federated learning environment. The implementation is structured to support modularity, real-time orchestration, and ease of deployment across multiple nodes. This section outlines the architectural design and deployment configuration used in our experiments.

### 4.1.1 Framework architecture

The SwarmShield system is implemented in a modular and extensible manner, enabling flexibility in integrating different datasets, models, and adversarial attack types. The core architecture comprises four major components:

- **Machine Learning Engine:** Encapsulates the training logic, model definitions, dataset loading, and adversarial attack routines. A modular design pattern allows seamless swapping or integration of different model architectures, datasets (e.g., CIFAR-10), and attack strategies (e.g., FGSM, PGD).
- **Swarm Learning Control Plane (SLCP):** Serves as the decentralized coordination layer responsible for peer enrollment, dynamic leader election, parameter synchronization, and trust score communication. By eliminating the need for a central server, the SLCP ensures privacy-preserving collaboration and fault-tolerant orchestration in federated setups.
- **Swarm Learning Integration Layer:** Acts as the interface between the ML engine and SLCP. It handles secure container deployment, communication callbacks, and the exchange of selected model parameters during each training round. It also ensures the isolation of peer logic from coordination logic, improving system integrity.
- **Automation Layer:** A suite of orchestration scripts is used to initiate experiments, deploy peer agents, verify system states, and gracefully shut down the network. These scripts automate operations such as launching the SLCP, setting up containers, and executing training rounds with and without defenses enabled.

SwarmShield enables truly distributed execution across multiple independent nodes using Swarm Learning's native synchronization, ensuring authentic federated learning conditions. The framework automates deployment through containerized templates and provides a lightweight callback interface for custom model integration, supporting diverse application domains while maintaining centralized result collection for robustness evaluation.

Configuration occurs through intuitive interfaces for model, dataset, and attack scenario selection, enabling reproducible experiments without modifying core orchestration logic. This containerized approach simplifies deployment while preserving the system's distributed nature and adaptability.

## 4.2 Datasets

To evaluate the robustness and versatility of our proposed SwarmShield framework, we conducted experiments across three distinct and publicly available datasets: a standard natural image dataset (CIFAR-10), a biomedical image dataset (DermaMNIST), and a time series dataset (TwoLeadECG).

### 4.2.1 CIFAR-10

The CIFAR-10 dataset [39] is a widely used benchmark for image classification. It consists of 60,000 color images of size $32 \times 32$ pixels, categorized into 10 mutually exclusive classes (e.g., plane, car, bird). The dataset is pre-divided into 50,000 training images and 10,000 testing images. For our experiments, CIFAR-10 serves as a standard baseline to evaluate the performance of our defense mechanism against various attacks in a well-understood, non-biomedical context.

### 4.2.2 DermaMNIST

The DermaMNIST dataset is part of the MedMNIST v2 [40] collection, a set of standardized biomedical image datasets. It is based on the HAM10000 dataset [41] and contains 10,015 dermatoscopic images of common pigmented skin lesions. The images are resized to $28 \times 28$ pixels and are categorized into 7 distinct classes, including melanoma, basal cell carcinoma, and benign keratosis. This dataset allows us to test SwarmShield's effectiveness on a multi-class, biomedical imaging task with inherent class imbalances.

### 4.2.3 TwoLeadECG

The TwoLeadECG dataset, sourced from the `aeon` library, provides a time series classification challenge. It contains 1,162 instances of electrocardiogram (ECG) readings, each with 2 leads (channels) and a length of 82 time steps. This dataset is derived from the MIT-BIH Arrhythmia Database [42] and is also available in the UCR Time Series Classification Archive [43]. For our experiments, we formulate a binary classification task to distinguish a specific arrhythmia (class 2) from normal heartbeats (class 1). This dataset serves to demonstrate SwarmShield's applicability beyond image-based data, highlighting its effectiveness in federated learning scenarios involving time series data—an important modality in healthcare monitoring.

## 4.3 Results on CIFAR10 dataset

This section presents an empirical evaluation of SwarmShield's effectiveness in enhancing adversarial robustness within a decentralized federated learning


Figure 2: Adversarial attacks on CIFAR10 samples: original images (top row), Noise attack adding static (middle row), and subtle FGSM/PGD perturbations (bottom rows) designed to induce misclassification.

Table 2: Detailed class-wise accuracy (%) on the CIFAR-10 dataset using the ResNet50 model. The table compares performance: (a) without SwarmShield and (b) with SwarmShield, across five settings: a baseline with no attack and four different adversarial attack types. The Average Percentage Improvement ('API') row quantifies the significant performance recovery achieved by SwarmShield under each attack.

| Attack | NOATTACK | | Label Flipping | | PGD | | FGSM | | ExtremeNoise | |
|---|---|---|---|---|---|---|---|---|---|---|
| Labels | a | b | a | b | a | b | a | b | a | b |
| plane | 63.27 | 61.953 | 60.867 | **77.62** | 66.063 | **78.52** | 61.4 | **78.29** | 65.347 | **77.61** |
| car | 69.37 | 69.013 | 69.627 | **80.64** | 68.243 | **83.91** | 67.077 | **82.04** | 69.337 | **81.12** |
| bird | 42.82 | **45.887** | 46.653 | **60.47** | 42.64 | **59.51** | 42.823 | **60.39** | 45.653 | **59.11** |
| cat | 39.333 | 39.313 | 38.91 | **51.16** | 40.607 | **52.1** | 39.003 | **51.94** | 38.647 | **51.44** |
| deer | 48.887 | 47.33 | 47.74 | **65.98** | 50.16 | **67.91** | 49.74 | **65.73** | 48.783 | **66.74** |
| dog | 48.703 | 48.157 | 47.463 | **61.23** | 47.56 | **60.27** | 49.303 | **56.99** | 49.91 | **59.93** |
| frog | 67.577 | **67.927** | 66.943 | **80.16** | 66.663 | **80.04** | 66.737 | **78.45** | 64.843 | **79.19** |
| horse | 66.007 | 65.257 | 65.643 | **78.18** | 64.257 | **77.38** | 64.927 | **78.27** | 63.66 | **76.79** |
| ship | 72.43 | 71.3 | 72.303 | **83.53** | 69.673 | **82.99** | 73.557 | **84.22** | 69.513 | **84.17** |
| truck | 62.383 | 62.36 | 61.717 | **77.19** | 63.047 | **75.45** | 64.247 | **77.32** | 62.373 | **78.55** |
| **Average** | 58.612 | 57.950 | 59.277 | **71.456** | 57.521 | **71.258** | 57.331 | **70.754** | 57.577 | **71.165** |
| **API** | -0.20 | | **25.11** | | **25.11** | | **24.47** | | **24.50** | |

environment. Figure 2 illustrates the effect of the different kinds of attack on the CIFAR10 dataset. Performance is reported in terms of classification accuracy of the global model aggregated at the end of each round.

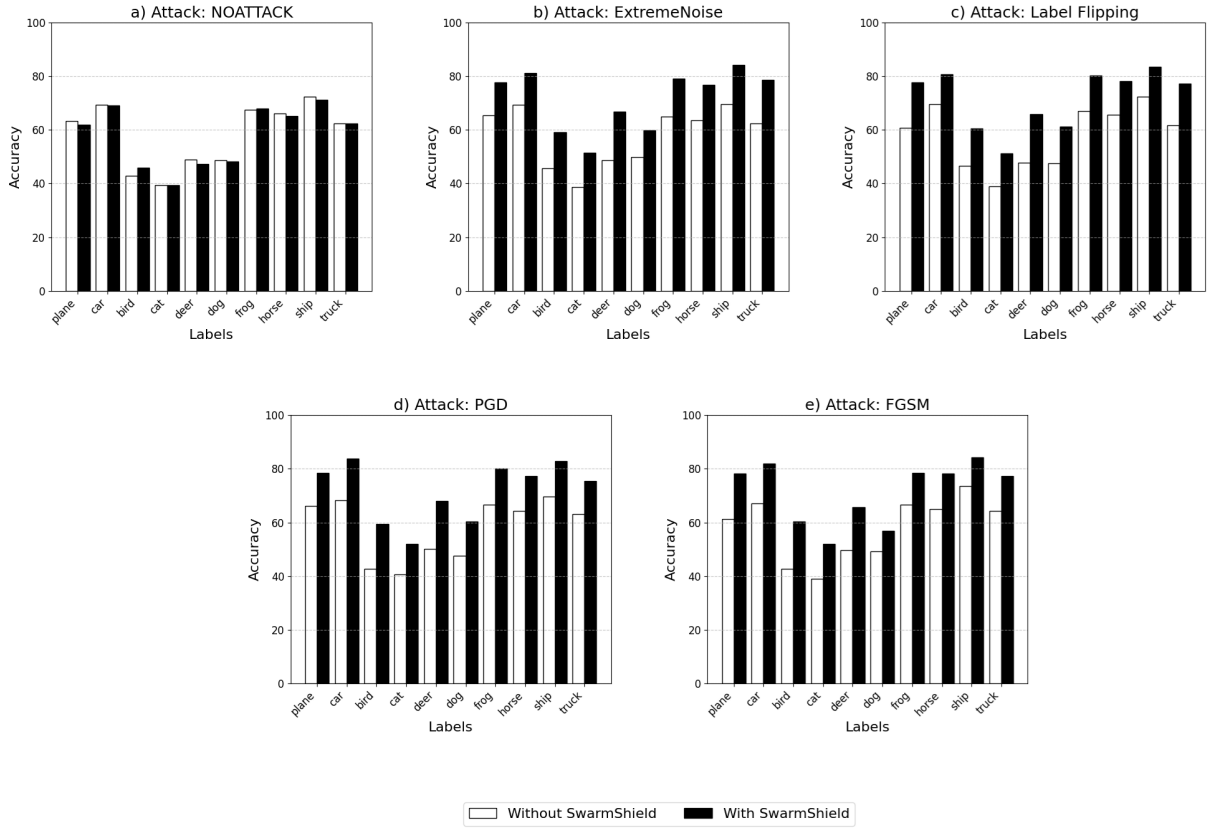
In the *no attack* scenario, all participating peers in the swarm operate on clean, unaltered data. The training process follows the standard Swarm Learning protocol where an initial global model is distributed to all peers, each peer trains the model on its local dataset, and shares selected layer parameters with the elected leader. The leader aggregates these updates using SwarmShield's trust-weighted averaging and returns the updated global model to all peers.

Under this setting, the ResNet50 model maintains stable classification accuracy across CIFAR-10 classes, as summarized in Table 2. Since no malicious behavior is present, SwarmShield's mechanisms—such as trust scoring and exclusion—do not trigger, validating that the framework introduces minimal overhead (58.61% baseline vs 57.95% with SwarmShield) and preserves accuracy in benign environments.

When adversarial attacks are introduced, model performance deteriorates predictably. Gradient-based attacks like PGD and FGSM cause accuracy degradation through subtle input perturbations, while label flipping and extreme noise inject semantic or structural disruption. For example, under PGD, accuracy for the *deer* class drops to 50.16%, while FGSM reduces *bird* accuracy to 42.82%.

The introduction of SwarmShield yields consistent improvements across all attack types. As shown in Table 2 and visualized in Figure 3, SwarmShield enables the model to recover from adversarial perturbations. The Average Percentage Improvement (API) presented in Table 2 shows up to 25.11% average improvement, with notable recoveries in: - Label Flipping: *car* class from 69.63% to 80.64% - PGD: *deer* class from 50.16% to 67.91% - FGSM: *bird* class from 42.82% to 60.39% - Extreme Noise: maintains

Figure 3: CIFAR-10 class accuracy with/without SwarmShield across five scenarios: No Attack, Extreme Noise, Label Flipping, PGD, and FGSM. Black bars show SwarmShield's improved performance versus undefended model (white bars).




24.5% average improvement

For CIFAR-10, SwarmShield demonstrates particular advantage against gradient-based attacks like PGD and FGSM, achieving average performance recoveries of 25.11% and 24.47% respectively. The PCA-based analysis effectively captures deviation patterns in malicious clients' model weights, allowing the k-means algorithm to isolate these nodes as outliers. Through trust-weighted aggregation, their harmful contributions are nullified while preserving the consensus of honest peers.

### 4.3.1 Comparison with State of the Art

Table 3 further positions SwarmShield against recent adversarial defense methods for FGSM, PGD, Label Flipping and Extreme Noise attacks. As no prior work was found to evaluate all four attack types—FGSM, PGD, Label Flipping, and Extreme Noise—within a unified framework, we present a two-part comparison: one focusing on gradient-based attacks (FGSM, PGD), and the other on label noise attacks (Label Flipping, Extreme Noise). This allows for a more representative and fair evaluation of SwarmShield across diverse threat scenarios.

In the gradient-based category, SwarmShield performs on par with the best-performing centralized methods such as GReAT (ADAM), achieving 71.81% under PGD-10 and 71.44% under FGSM. Notably, SwarmShield achieves this robustness in a fully decentralized setup, without relying on a central aggregator or customized optimizers.

For label noise attacks, SwarmShield shows even stronger gains. As seen in the right half of Table 3, it outperforms other state-of-the-art methods by a margin of 5–10%, reaching 71.85% under label flipping and 70.41% under extreme noise. Competing methods in this category—such as Soften to Defend, GLC, and FixMatch + NAD—typically depend on centralized supervision, access to a trusted clean subset, or semi-supervised learning frameworks. SwarmShield, in contrast, maintains high performance using only peer-consensus-based trust scoring, without assuming access to clean labels or external supervision.

Additionally, while most of the compared methods are evaluated using shallower backbones like ResNet-18, ResNet-34, or PreAct ResNet-18, SwarmShield demonstrates strong resilience even with the deeper ResNet-50 architecture. This highlights its scalability and suitability for deployment in practical federated learning scenarios where larger models and decentralized coordination are both essential.

Table 3: Comparison of SwarmShield's adversarial accuracy against state-of-the-art methods on CIFAR-10. The table is split into two categories for a fair comparison: gradient-based attacks (FGSM, PGD) on the left, and data poisoning attacks (Label Flipping, Extreme Noise) on the right. SwarmShield demonstrates competitive or superior performance in a fully decentralized setting.

| **Method** | **FGSM (%)** | **PGD-10 (%)** | **Method** | **Label Flipping (%)** | **Extreme Noise (%)** |
|---|---|---|---|---|---|
| TLA-RN [44] | 55.41 | 52.5 | Soften to Defend [45] | 60.5 | 57.0 |
| TLA-SA [44] | 58.8 | 53.53 | GLC (Gold Loss Corr.) [46] | 61.0 | 59.5 |
| TLA [44] | 58.88 | 53.87 | Symmetric CE [47] | 67.4 | 62.1 |
| GReAT (SGD) [48] | 62.78 | 60.58 | Co-teaching [49] | 66.2 | 59.2 |
| GReAT (ADAM) [48] | **72.47** | 71.31 | KNN Defense [50] | 51.0 | - |
| **SwarmShield (Proposed)** | 71.44 | **71.81** | **SwarmShield (Proposed)** | **71.85** | **70.41** |

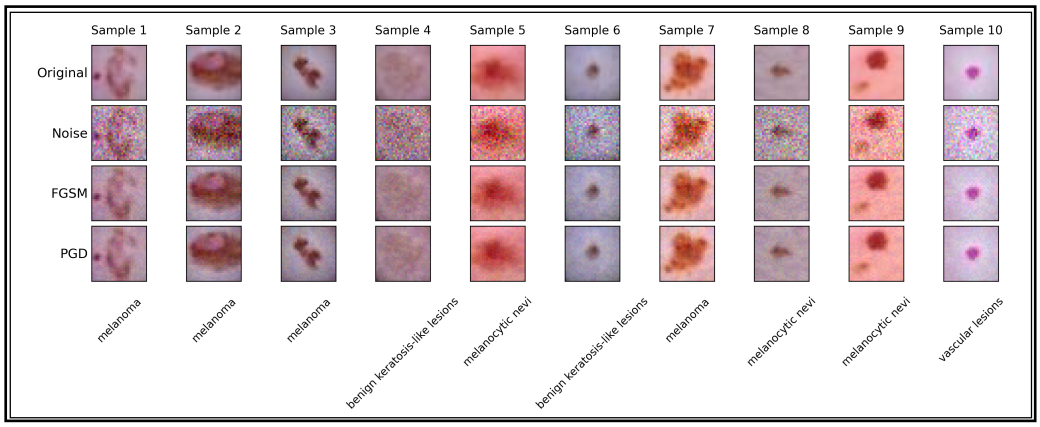



Figure 4: Adversarial attacks on DermaMNIST samples: original images (top row), Noise attack adding static (middle row), and subtle FGSM/PGD perturbations (bottom rows) designed to induce misclassification.

## 4.4 Results on DermaMNIST dataset

Table 4: Experiment Summary: Global Model Accuracy (%) on DermaMNIST

| **Scenario** | **Baseline (No Defense)** | **With SwarmShield** |
|---|---|---|
| No Attack (Clean) | 74.06 | **74.71** |
| Label Flipping | 72.87 | **73.77** |
| Extreme Noise | 74.61 | 74.51 |
| FGSM | 73.67 | **74.26** |
| PGD | 73.62 | **74.36** |

Building on the dataset characteristics from Section 4.2, we evaluate SwarmShield's performance on medical imaging through the DermaMNIST benchmark. Figure 4 visually demonstrates how different attack types affect sample dermatoscopic images, showing both obvious noise perturbations and subtle gradient-based distortions. The quantitative results in Table 4 reveal SwarmShield's consistent improvements across all attack scenarios. In clean conditions, the framework achieves a 74.71% accuracy compared to the 74.06% baseline, demonstrating minimal overhead. Against label flipping attacks, SwarmShield improves performance by 0.90% absolute (73.77% vs 72.87%), while maintaining comparable results for extreme noise attacks (74.51% vs 74.61%). Gradient-based attacks show particularly strong resilience, with FGSM accuracy improving to 74.26% from 73.67% and PGD reaching 74.36% versus 73.62% baseline.

The framework's effectiveness in medical imaging stems from its ability to leverage clinically significant feature

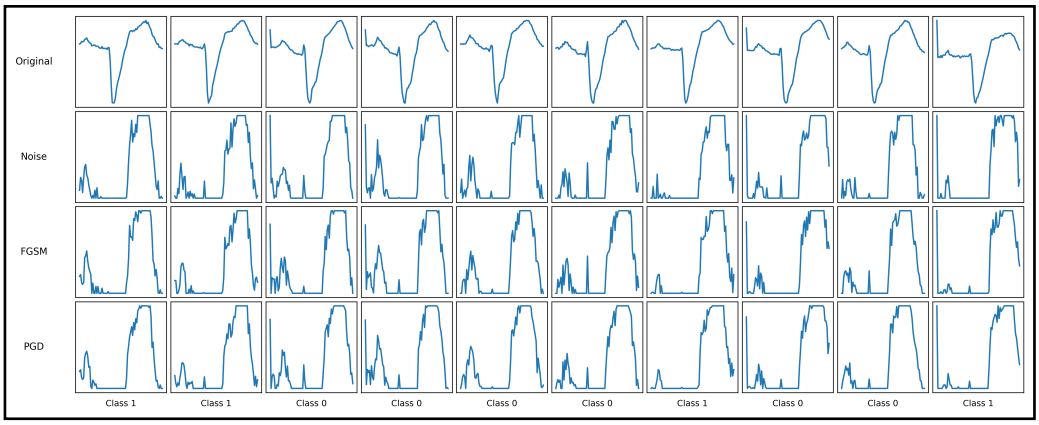



Figure 5: Adversarial attacks on TwoLeadECG samples: original images (top row), Noise attack adding static (middle row), and subtle FGSM/PGD perturbations (bottom rows) designed to induce misclassification.

differences between disease categories like melanoma and benign keratosis. The PCA and k-means clustering reliably identifies malicious updates by detecting deviations in these diagnostic patterns, while the trust-weighted aggregation preserves the integrity of medically relevant model parameters. This combination of techniques proves especially valuable for dermatological applications where both accuracy and robustness are critical for clinical decision-making.

## 4.5 Results on time series dataset (TwoLeadECG)

Table 5: Global model accuracy (%) for each attack scenario on the TwoLeadECG dataset.

| Scenario | No Defense | SwarmShield |
|---|---|---|
| Clean | 99.30 | 98.77 |
| Label Flipping | 40.83 | 2.02 |
| Extreme Noise | 91.13 | **99.65** |
| FGSM | 86.83 | **98.24** |
| PGD | 93.15 | 49.96 |

Our evaluation of SwarmShield on time-series ECG classification builds upon the dataset characteristics outlined in Section 4.2, using the TwoLeadECG dataset from the UCR Time Series Classification Archive [43]. Figure 5 visually demonstrates the varying effects of different attack types on ECG waveform samples.

The results in Table 5 show clear variability in SwarmShield's effectiveness across attack types. The framework achieves strong gains under Extreme Noise (+8.52% over baseline) and FGSM (+11.41%), but suffers substantial degradation under Label Flipping (2.02% vs. 40.83%) and PGD (49.96% vs. 93.15%).

The poor performance under label flipping stems from the fact that this attack modifies only the target labels during training, leaving the ECG waveforms themselves intact. Since SwarmShield's detection relies on identifying anomalous parameter updates or unusually large gradients, malicious updates generated from clean-looking but mislabelled data appear statistically similar to benign updates. This allows corrupted labels to propagate into the global model, leading to severely distorted decision boundaries [51].

In the case of PGD, the attack introduces small but structured perturbations in the time domain that closely resemble natural ECG variability, such as minor waveform shifts or amplitude modulations. These perturbations remain within physiologically plausible bounds, evading magnitude-based detection, while PGD's iterative nature ensures alignment with the classifier's most discriminative temporal features [52]. Figure 6 schematically illustrates these evasion mechanisms: label flipping bypasses detection by preserving data morphology, and PGD exploits natural variability to blend perturbations with genuine patterns.

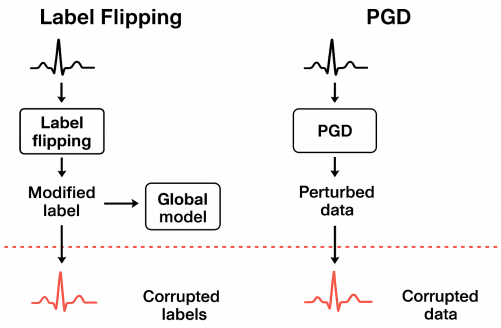



Figure 6: Visual summary of how Label Flipping and PGD attacks evade detection in ECG classification.

Rather than being irrelevant, these findings expose critical blind spots in defending time-series medical models. They underscore the need for future defenses that (1) incorporate semantic consistency checks between labels and temporal morphology, and (2) leverage temporal dependency analysis to detect adversarially consistent patterns.

## 4.6 Ablation study

To dissect the contribution of each component within the SwarmShield framework, we conducted a comprehensive ablation study. We systematically disabled key mechanisms—integrity verification, dimensionality reduction (PCA), trust-based aggregation, and density-based clustering—to evaluate their individual and synergistic impact on defense efficacy. The experiments were performed on the CIFAR-10 dataset using a ResNet-18 model, with 3 out of 10 clients configured as malicious agents for 50 communication rounds. The results, summarized in Table 6, present the final global model accuracy against three distinct adversarial strategies: FGSM, PGD (with a reduced $\epsilon$ of 0.05 to represent a more subtle attack), and stochastic Noise.

Table 6: Final Global Model Accuracy (%) at Round 50 for SwarmShield Ablation Configurations Under Various Attacks. The new results dramatically highlight the critical role of density-based clustering in preventing catastrophic failure against PGD attacks.

| Ablation Configuration | FGSM | NOISE | PGD |
|---|---|---|---|
| fedavg (No Defense) | 74.06 | 78.78 | 75.72 |
| only_hashing | 68.49 | 78.97 | 76.34 |
| only_pca | 67.69 | 78.27 | 75.57 |
| only_trust | 71.29 | 80.09 | **78.69** |
| no_integrity | 75.44 | 80.23 | 77.86 |
| no_pca | 76.39 | 80.59 | 78.03 |
| no_trust | 72.63 | 80.74 | 77.99 |
| no_clustering | 73.23 | 78.27 | 67.05 |
| **full_swarmshield** | **76.48** | **80.90** | 78.36 |

### 4.6.1 Analysis of component contributions

The results in Table 6 reveal a complex interplay between SwarmShield's components. The updated data powerfully underscores the synergistic necessity of the full defense pipeline, particularly the critical role of clustering as a prerequisite for a reliable trust mechanism.

**The critical synergy of PCA and clustering** The most significant finding is the pivotal role of density-based clustering. The `no_clustering` configuration, which builds its Kernel Density Estimation (KDE) trust model on all client updates, suffers a catastrophic performance collapse against the PGD attack, dropping to 67.05% accuracy. This demonstrates that without a filtering mechanism, the KDE model becomes "poisoned" by the inclusion of malicious updates in its training set. The subtle PGD updates are sufficient to corrupt the learned probability distribution of "normal" behavior, causing the server to assign high trust to adversaries and effectively rendering the defense useless.

In stark contrast, `full_swarmshield` first employs PCA to accentuate the geometric properties of model updates. Subsequently, its density-based clustering algorithm identifies the core group of honest clients. By constructing the KDE trust model *exclusively* from this validated cluster, SwarmShield ensures its definition of normalcy is robust and uncontaminated. The dramatic performance gap between `full_swarmshield` and `no_clustering` against PGD (78.36% vs. a mere 67.05%) powerfully exemplifies that clustering is not just a performance enhancement but a mandatory step for a secure trust-based aggregation.

**Defense against gradient-based attacks (FGSM & PGD)** Against the FGSM attack, `full_swarmshield` achieves the highest accuracy (76.48%), proving its multi-stage pipeline is essential for mitigating these aggressive updates. Simpler defenses like `only_pca` and `only_hashing` perform worse than vanilla `fedavg`, indicating that isolated mechanisms are insufficient.

An important and revealing result occurs with the PGD attack, where the `only_trust` configuration (78.69%) marginally outperforms `full_swarmshield` (78.36%). This phenomenon highlights a key trade-off. The weakened PGD attack ($\epsilon = 0.05$) creates updates that are statistically separable in the high-dimensional parameter space. In this specific context, the lossy compression of PCA may discard subtle information, and the overhead of clustering is not strictly necessary. However, the catastrophic failure of the `no_clustering` run proves that while a direct trust mechanism can work in ideal conditions, it is extremely brittle. For general-purpose robustness against a wider spectrum of threats, the full pipeline is essential.

**Defense against stochastic attacks (Noise)** The Noise attack serves as a control, introducing non-strategic, random perturbations. Here, configurations with a trust mechanism generally outperform the `fedavg` baseline, as the KDE model inherently identifies and down-weights statistical outliers. The `full_swarmshield` configuration again achieves the highest accuracy (80.90%). Its ability to form a tight core cluster via density-based clustering allows it to effectively reject the random noise that other configurations might partially incorporate, leading to a more stable and accurate global model. The relatively poor performance of `no_clustering` (78.27%) here further supports the conclusion that a clean input to the trust model is paramount.

### 4.6.2 Conclusion of ablation study

The ablation study validates our core hypothesis: robust defense in federated learning is not achieved by a single mechanism but by a synergistic pipeline. While trust-based aggregation is a cornerstone, its effectiveness is critically dependent on the quality of its input. The catastrophic failure of the `no_clustering` configuration against PGD attacks proves that without a robust filtering mechanism like density-based clustering, a trust model is easily poisoned

and can fail completely. For general robustness against a wide spectrum of threats, the full SwarmShield pipeline is superior. PCA provides a salient feature space, and density-based clustering purifies the input to the trust model, ensuring its integrity and reliability. This integrated design is essential to SwarmShield's overall performance.

## 4.7 Robustness to malicious client ratios: scaling under adversarial pressure

To further characterize the operational boundaries of SwarmShield, we evaluate its performance as the ratio of malicious clients increases from 0% to 90%, across canonical adversary strategies. Table 7 and Fig. 7 summarize the global accuracy for extreme noise, FGSM, and label-flipping attacks.

Table 7: SwarmShield accuracy (%) versus malicious client ratio for three attack types.

| Malicious Ratio | Extreme Noise | FGSM | Label Flipping |
|---|---|---|---|
| 0.0 | 83.70 | 83.53 | 83.75 |
| 0.1 | 82.95 | 83.30 | 83.45 |
| 0.3 | 80.75 | 81.48 | 81.59 |
| 0.5 | 72.87 | 76.45 | 2.44 |
| 0.7 | 68.82 | 67.66 | 1.74 |
| 0.9 | 70.55 | 50.71 | 1.86 |

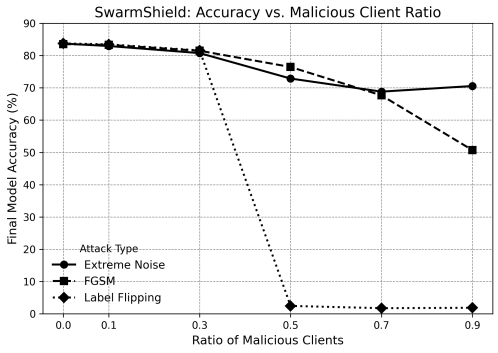



Figure 7: SwarmShield test accuracy under increasing ratios of malicious clients for multiple attack types.

### 4.7.1 Technical analysis and interpretation

– For both **Extreme Noise** and **FGSM** attacks, SwarmShield consistently maintains high accuracy ($> 80\%$) up to a 30% adversarial ratio. This resilience is attributable to the PCA-induced geometric separation of adversarial updates and the adaptive down-weighting of outliers, which together suppress the influence of non-conforming gradients. The effectiveness against gradient-based attacks (FGSM) further validates the robustness of the aggregation pipeline, even when adversaries leverage whitebox knowledge.

– Beyond the $50\%$ malicious threshold, performance degrades more rapidly, with the steepest drop observed for **Label Flipping**, where accuracy collapses to random guessing. This phenomenon is theoretically anticipated: in the presence of a majority of coordinated attackers, even optimal clustering and trust mechanisms may fail, as the adversarial cluster can dominate the geometric structure of the update space (cf. the impossibility results for robust mean estimation when over half the data is adversarial).

– The model's accuracy for **Extreme Noise** and **FGSM** remains non-trivial ($\sim$70%) even at very high adversarial ratios. This is due to the inherently incoherent or inconsistent nature of noise-based attacks, which are more easily isolated as outliers in the projected space. In contrast, label flipping attacks produce more structured, adversarially-aligned updates that can coalesce in the aggregation, circumventing geometric defenses when they form a majority.

– The observed *graceful degradation*—a slow decay in performance up to the theoretical breakdown point—demonstrates that SwarmShield's defenses scale with adversarial pressure and remain effective in the practical regime where the honest majority assumption holds. This is a critical requirement for federated learning deployment in open, untrusted environments.

SwarmShield's design, grounded in geometric and statistical insights, delivers provable robustness under minority adversarial regimes, with each module playing a specific and theoretically motivated role. The system's performance breakdown at extreme adversarial ratios aligns with established lower bounds, confirming both the strength and the limitations of current robust aggregation techniques. These findings highlight the necessity of combining multiple, theoretically justified defenses to achieve resilience in federated learning at scale.

## 4.8 Discussion

The comprehensive evaluation of SwarmShield across multiple datasets and attack scenarios reveals several key insights about the framework's capabilities and limitations. These findings warrant deeper examination across several critical dimensions that impact real-world deployment.

### 4.8.1 Generalizability

The consistent performance across CIFAR-10 and DermaMNIST [40] demonstrates SwarmShield's adaptability to both standard benchmarks and medical imaging domains. The framework maintains effectiveness against diverse attack vectors including label flipping and gradient-based attacks, with particular success in clinical applications where model accuracy is critical. This cross-domain

robustness stems from the trust mechanism's fundamental design, which operates on model parameters rather than data-specific features.

#### 4.8.2 Algorithmic complexity

SwarmShield's computational overhead scales polynomially with network size, dominated by the $O(n^2)$ clustering operation for $n$ peers. The leader election ($O(n \log n)$) and model aggregation ($O(kd)$ for $k$ clusters and $d$-dimensional representations) remain efficient through dimensionality reduction. Our empirical results show stable operation when the clustering successfully isolates malicious updates, approximating robust federated averaging over honest participants [1].

#### 4.8.3 Scalability

While evaluations demonstrate effective operation on moderate-scale networks (3-50 nodes), practical deployment at scale may require optimizations for very large or high-dimensional models. The architecture's modular design permits such extensions while maintaining the core trust mechanisms. Notably, SwarmShield advances beyond existing work [53] by demonstrating robustness on specialized medical datasets (DermaMNIST, TwoLeadECG) where conventional defenses remain unvalidated.

This work establishes new capabilities for secure federated learning in clinical and time-series applications while maintaining compatibility with existing FL frameworks. The results particularly highlight the importance of domain-specific validation, as attack patterns and defense effectiveness vary significantly across data modalities.

## 5 Conclusion and future work

The proposed SwarmShield framework introduces a decentralized trust-aware defense mechanism that enhances adversarial robustness in federated learning through three key innovations: dynamic leader election, unsupervised parameter clustering, and blockchain-based auditing. Our comprehensive evaluation demonstrates consistent improvements across multiple benchmarks, achieving 24.8% average accuracy gains on CIFAR-10 under diverse attacks (FGSM, PGD, label flipping, extreme noise) while maintaining 74.71% accuracy on the medical DermaMNIST dataset. The framework's effectiveness stems from its PCA-based clustering mechanism and trust-weighted aggregation, which ablation studies confirm as essential components, providing graceful degradation even with 50% malicious nodes.

The results establish SwarmShield as a practical solution for real-world deployment, showing particular strength in medical applications where data sensitivity demands robust privacy-preserving techniques. Unlike centralized approaches [53], the framework maintains competitive performance while eliminating single points of failure, achieving 98.24% accuracy against FGSM attacks on TwoLeadECG time-series data while introducing negligible overhead in benign conditions.

Future work will focus on three key directions: (1) extending evaluation to additional modalities including NLP tasks and IoT sensor data, (2) enhancing temporal pattern recognition for improved robustness against PGD attacks in time-series applications, and (3) developing hybrid defenses combining parameter clustering with activation pattern analysis. These extensions will address current limitations in label-flipping scenarios while maintaining the framework's core advantages of decentralization and computational efficiency. The demonstrated success across medical imaging and time-series analysis suggests promising applications in healthcare monitoring and other sensitive domains where both privacy and robustness are paramount.

## Acknowledgment

The authors thank Hewlett Packard Enterprise for supporting the experimental work conducted during the first author's tenure at the organization. College of Engineering Trivandrum is also acknowledged for its academic support during the first author's concurrent part-time Ph.D. program. The contributions of both the institutions were essential to the successful execution of this research.

## References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html

[2] M. Goldblum, D. Tsipras, C. Xie, X. Chen, A. Schwarzschild, D. Song, A. Madry, B. Li, and T. Goldstein, "Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. [Online]. Available: https://doi.org/10.1109/TPAMI.2022.3162397

[3] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Advances in neural information processing systems*, vol. 30, 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/f4b9ec30ad9f68f89b29639786cb62ef-Abstract.html

[4] C. Fung, C. J. Yoon, and I. Beschastnikh, "The limitations of federated learning in sybil settings," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, 2020,

pp. 301–316. [Online]. Available: https://www.usenix.org/conference/raid2020/presentation/fung

[5] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *International conference on machine learning*. PMLR, 2021, pp. 6357–6368. [Online]. Available: https://proceedings.mlr.press/v139/li21h

[6] J. Wu, "Distributed intelligent optimization of e-commerce user purchase data mining using spark framework," *Informatica*, vol. 48, no. 20, pp. 29–40, 2024. [Online]. Available: https://doi.org/10.31449/inf.v48i20.6779

[7] N. Azeri, O. Hioual, and O. Hioual, "Efficient vanilla split learning for privacy-preserving collaboration in resource-constrained cyber-physical systems," *Informatica*, vol. 48, no. 11, 2024. [Online]. Available: https://doi.org/10.31449/inf.v48i11.6186

[8] E. M. E. Mhamdi, R. Guerraoui, and S. Rouault, "The Hidden Vulnerability of Distributed Learning in Byzantium," in *Proceedings of the 35th International Conference on Machine Learning*. PMLR, 2018, pp. 3518–3527. [Online]. Available: https://proceedings.mlr.press/v80/mhamdi18a.html

[9] C. Fung, C. J. Yoon, and I. Beschastnikh, "Mitigating sybils in federated learning poisoning," *arXiv preprint arXiv:1808.04866*, 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1808.04866

[10] P. Ramanan and K. Nakayama, "Baffle: Blockchain based aggregator free federated learning," in *2020 IEEE international conference on blockchain (Blockchain)*. IEEE, 2020, pp. 72–81. [Online]. Available: https://doi.org/10.1109/Blockchain50366.2020.00017

[11] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping," *arXiv preprint arXiv:2012.13995*, 2020. [Online]. Available: https://doi.org/10.14722/NDSS.2021.24434

[12] Y. Zhou, J. Wu, H. Wang, and J. He, "Adversarial robustness through bias variance decomposition: A new perspective for federated learning," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2022, pp. 2753–2762. [Online]. Available: https://doi.org/10.1145/3511808.3557232

[13] N. M. Jebreel and J. Domingo-Ferrer, "Fl-defender: Combating targeted attacks in federated learning," *Knowledge-Based Systems*, vol. 260, p. 110178, 2023. [Online]. Available: https://doi.org/10.1016/j.knosys.2022.110178

[14] M. Vucovich, D. Quinn, K. Choi, C. Redino, A. Rahman, and E. Bowen, "Fedbayes: A zero-trust federated learning aggregation to defend against adversarial attacks," in *2024 IEEE 14th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE, 2024, pp. 0028–0035. [Online]. Available: https://doi.org/10.1109/CCWC60891.2024.10427896

[15] C. Xie, S. Koyejo, and I. Gupta, "Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance," in *International conference on machine learning*. PMLR, 2019, pp. 6893–6901. [Online]. Available: https://proceedings.mlr.press/v97/xie19b/xie19b.pdf

[16] N. Akhtar, A. Mian, N. Kardan, and M. Shah, "Advances in adversarial attacks and defenses in computer vision: A survey," *IEEE access*, vol. 9, pp. 155 161–155 196, 2021. [Online]. Available: https://doi.org/10.1109/ACCESS.2021.3127960

[17] Z. Ye, "Mitigating poisoning attacks in decentralized federated learning through moving target defense," *ZORA University of Zurich*, 2024. [Online]. Available: https://www.zora.uzh.ch/id/eprint/262742

[18] Y. Chen, Y. Yang, Y. Liang, T. Zhu, and D. Huang, "Federated learning with privacy preservation in large-scale distributed systems using differential privacy and homomorphic encryption," *Informatica*, vol. 49, no. 13, 2025. [Online]. Available: https://doi.org/10.31449/inf.v49i13.7358

[19] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5650–5659. [Online]. Available: https://proceedings.mlr.press/v80/yin18a/yin18a.pdf

[20] S. Warnat-Herresthal, H. Schultze, K. L. Shastry, S. Manamohan, S. Mukherjee, V. Garg, R. Sarveswara, K. Händler, P. Pickkers, N. A. Aziz *et al.*, "Swarm learning for decentralized and confidential clinical machine learning," *Nature*, vol. 594, no. 7862, pp. 265–270, 2021. [Online]. Available: https://doi.org/10.1038/s41586-021-03583-3

[21] W. Jin, Y. Yao, S. Han, C. Joe-Wong, S. Ravi, S. Avestimehr, and C. He, "Fedml-he: An efficient homomorphic-encryption-based privacy-preserving federated learning system," *arXiv preprint arXiv:2303.10837*, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2303.10837

[22] J. Zhang, B. Li, C. Chen, L. Lyu, S. Wu, S. Ding, and C. Wu, "Delving into the adversarial robustness of federated learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, no. 9,

2023, pp. 11 245–11 253. [Online]. Available: https://doi.org/10.1609/aaai.v37i9.26331

[23] R. Xu, S. Gao, C. Li, J. Joshi, and J. Li, "Dual defense: Enhancing privacy and mitigating poisoning attacks in federated learning," *NeurIPS 2024*, 2024. [Online]. Available: https://neurips.cc/virtual/2024/poster/96030

[24] L. Zhang, M. Goldstein, and R. Ranganath, "Understanding failures in out-of-distribution detection with deep generative models," in *International Conference on Machine Learning*. PMLR, 2021, pp. 12 427–12 436. [Online]. Available: https://proceedings.mlr.press/v139/zhang21g/zhang21g.pdf

[25] C.-H. Ho and N. Vasconcelos, "Disco: Adversarial defense with local implicit functions," *Advances in neural information processing systems*, vol. 35, pp. 23 818–23 837, 2022. [Online]. Available: https://dl.acm.org/doi/10.5555/3600270.3602000

[26] N. Papernot, F. Faghri, N. Carlini, I. Goodfellow, R. Feinman, A. Kurakin, C. Xie, Y. Sharma, T. Brown, A. Roy, A. Matyasko, V. Behzadan, K. Hambardzumyan, Z. Zhang, Y.-L. Juang, Z. Li, R. Sheatsley, A. Garg, J. Uesato, W. Gierke, Y. Dong, D. Berthelot, P. Hendricks, J. Rauber, and R. Long, "Technical report on the cleverhans v2.1.0 adversarial examples library," *arXiv preprint arXiv:1610.00768*, 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1610.00768

[27] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations (ICLR)*, 2015. [Online]. Available: https://doi.org/10.48550/arXiv.1412.6572

[28] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *International Conference on Learning Representations (ICLR)*, 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1706.06083

[29] B. Biggio, B. Nelson, and P. Laskov, "Support vector machines under adversarial label noise," in *Asian conference on machine learning*. PMLR, 2011, pp. 97–112. [Online]. Available: http://proceedings.mlr.press/v20/biggio11/biggio11.pdf

[30] K. T. Co, L. Muñoz-González, S. de Maupeou, and E. C. Lupu, "Procedural noise adversarial examples for black-box attacks on deep convolutional networks," in *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, 2019, pp. 275–289. [Online]. Available: https://doi.org/10.1145/3319535.3345660

[31] S. Warnat-Herresthal, H. Schultze, K. L. Shastry, S. Manamohan, S. Mukherjee, V. Garg, R. Sarveswara, K. Händler, P. Pickkers, N. A. Aziz *et al.*, "Swarm learning as a privacy-preserving machine learning approach for disease classification," *BioRxiv*, pp. 2020–06, 2020. [Online]. Available: https://doi.org/10.1101/2020.06.25.171009

[32] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[33] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, vol. 5. University of California press, 1967, pp. 281–298. [Online]. Available: https://www.cs.cmu.edu/~bhiksha/courses/mlsp.fall2010/class14/macqueen.pdf

[34] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 93–104. [Online]. Available: https://www.dbs.ifi.lmu.de/Publikationen/Papers/LOF.pdf

[35] National Institute of Standards and Technology, "FIPS PUB 180-4: Secure Hash Standard (SHS)," August 2015. [Online]. Available: https://doi.org/10.6028/NIST.FIPS.180-4

[36] D. Eastlake and T. Hansen, "Us secure hash algorithms (sha and sha-based hmac and hkdf)," Internet Engineering Task Force (IETF), RFC 6234, May 2011, rFC 6234. [Online]. Available: https://doi.org/10.17487/RFC6234

[37] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," in *International Conference on Learning Representations (ICLR)*, 2020. [Online]. Available: https://iclr.cc/virtual_2020/poster_HJxNAnVtDS.html

[38] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and trends® in machine learning*, vol. 14, no. 1–2, pp. 1–210, 2021. [Online]. Available: http://dx.doi.org/10.48550/arXiv.1912.04977

[39] A. Krizhevsky, "Learning multiple layers of features from tiny images," University of Toronto, Technical Report TR-2009, 2009. [Online]. Available: http://www.cs.utoronto.ca/~kriz/learning-features-2009-TR.pdf

[40] J. Yang, R. Shi, D. Wei, Z. Liu, L. Zhao, B. Ke, H. Pfister, and B. Ni, "Medmnist v2-a large-scale lightweight benchmark for 2d and 3d biomedical image classification," *Scientific Data*, vol. 10, no. 1, p. 41, 2023. [Online]. Available: https://www.nature.com/articles/s41597-022-01721-8

[41] P. Tschandl, C. Rosendahl, and H. Kittler, "The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," *Scientific data*, vol. 5, no. 1, pp. 1–9, 2018. [Online]. Available: https://www.nature.com/articles/sdata2018161.pdf

[42] G. B. Moody and R. G. Mark, "The impact of the mit-bih arrhythmia database," *IEEE Engineering in Medicine and Biology Magazine*, vol. 20, no. 3, pp. 45–50, 2001. [Online]. Available: https://physionet.org/content/mitdb/1.0.0/

[43] H. A. Dau, E. Keogh, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, B. H. Chen, Y. Hu, N. Begum, A. Bagnall, A. Mueen, G. Batista, and T. Hexagon, "The ucr time series classification archive," in *Proceedings of the IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 6, 2019, pp. 1293–1305. [Online]. Available: https://www.cs.ucr.edu/~eamonn/time_series_data_2018/

[44] C. Mao, Z. Zhong, J. Yang, C. Vondrick, and B. Ray, "Metric learning for adversarial robustness," *Advances in neural information processing systems*, vol. 32, 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/c24cd76e1ce41366a4bbe8a49b02a028-Paper.pdf

[45] Z. Li, T. Zhou, C. Li, Y. Yu, and J. Zhu, "Soften to defend: Towards adversarial robustness via self-guided label refinement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. [Online]. Available: http://dx.doi.org/10.1109/CVPR52733.2024.02340

[46] D. Hendrycks, M. Mazeika, and T. Dietterich, "Using trusted data to train deep networks on labels corrupted by severe noise," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 31, 2018. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2018/file/ad554d8c3b06d6b97ee76a2448bd7913-Paper.pdf

[47] Y. Wang, X. Ma, Z. Chen, Y. Luo, J. Yi, and J. Bailey, "Symmetric cross entropy for robust learning with noisy labels," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 322–330. [Online]. Available: http://dx.doi.org/10.1109/ICCV.2019.00041

[48] S. Bayram and K. Barner, "Great: A graph regularized adversarial training method," *IEEE Access*, 2024. [Online]. Available: https://doi.org/10.1109/ACCESS.2024.3395976

[49] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," *Advances in neural information processing systems*, vol. 31, 2018. [Online]. Available: https://dl.acm.org/doi/10.5555/3327757.3327944

[50] M. Abrishami, S. Dadkhah, E. C. P. Neto, P. Xiong, S. Iqbal, S. Ray, and A. A. Ghorbani, "Classification and analysis of adversarial machine learning attacks in iot: a label flipping attack case study," in *2022 32nd Conference of Open Innovations Association (FRUCT)*. IEEE, 2022, pp. 3–14. [Online]. Available: https://doi.org/10.23919/FRUCT56874.2022.9953823

[51] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I 25*. Springer, 2020, pp. 480–501. [Online]. Available: https://doi.org/10.1007/978-3-030-58951-6_24

[52] S. G. Finlayson, J. D. Bowers, J. Ito, J. L. Zittrain, A. L. Beam, and I. S. Kohane, "Adversarial attacks on medical machine learning," *Science*, vol. 363, no. 6433, pp. 1287–1289, 2019. [Online]. Available: https://doi.org/10.1126/science.aaw4399

[53] J. Zhang, C. Ge, F. Hu, and B. Chen, "Robustfl: Robust federated learning against poisoning attacks in industrial iot systems," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 9, pp. 6388–6397, 2021. [Online]. Available: http://dx.doi.org/10.1109/TII.2021.3132954