

Mitigating GAN-based poisoning attacks in federated learning using a decentralized multi-layer security framework with blockchain

Muthana R. Alsultan ^{*,1,a}, Shivan A. Mohammed ^{2,b}, Fidha Ezzaldin Aslan ^{3,c}

¹Department of Reconstruction and Projects, University of Baghdad, Baghdad, Iraq

²Department of AI Engineering, University of Zakho, Duhok, Iraq

³Department of Administrative and Financial Affairs, University of Telafer, Nineveh, Iraq

Article Info

Article History:

Received 21 Jul 2026

Accepted 21 Sep 2026

Keywords:

Federated learning;
Blockchain security;
Adversarial attacks;
GAN-based poisoning;
Anomaly detection;
Decentralized AI;
Privacy-preserving FL;
Robust machine learning

Abstract

Federated learning (FL) has gained prominence in decentralized artificial intelligence applications; however, its susceptibility to adversarial attacks, particularly GAN-based poisoning, remains a critical challenge. This study proposes a decentralized, multi-layered security framework integrating blockchain-backed validation, anomaly detection, and consensus-driven aggregation to enhance FL security. The framework effectively filters adversarial model updates, leveraging Mahalanobis distance-based anomaly detection and reputation-weighted selection to improve resilience. Experimental results demonstrate superior performance over existing methods, achieving a 98.7% accuracy, a 97.8% detection rate, and a reduced false positive rate of 2.1%. The system maintains efficiency with minimal computational overhead, ensuring practical applicability. The integration of adaptive defenses enhances robustness towards evolving threats, making this framework a comprehensive solution for securing FL in dynamic environments. Future research will focus on optimizing computational efficiency and incorporating privacy-preserving techniques to further strengthen adversarial resilience.

© 2026 MIM Research Group. All rights reserved.

1. Introduction

The proliferation of FL-based applications has been significantly enabled by the evolution towards 6G networks supporting Internet of Everything (IoE) devices with intelligent, distributed and real-time applications [1]. FL is a privacy-preserving ML-based computing paradigm which enables multiple devices/nodes to collaboratively learn a shared model while keeping the training data decentralized [2]. FL is suitable for applications where data security and privacy are important like healthcare applications, finance, and autonomous systems, etc. Due to FL's decentralized nature, there are security risks which can be leveraged by adversaries.

GAN-based data poisoning attacks have been considered one of the major threats among others [3]. These attacks create synthetic malicious data samples that look practically real [4]. Adversaries have an opportunity to upload these poisonous updates to the global model, resulting in degraded performance or biased outputs over time [5]. Given the fact that FL will be implemented on many essential systems, poisoning attacks using GANs need to be addressed for FL to be trustworthy and robust enough [6].

Blockchain integrated federated learning was recently introduced to countermeasure attacks by keeping immutable records of model updates and participant behavior [7]. Current solutions have operated using singular security solutions and are vulnerable to multipronged attack approaches

*Corresponding author: muth85cnet@uobaghdad.edu.iq

^aorcid.org/0009-0009-8801-1209; ^borcid.org/0000-0001-5713-8519; ^corcid.org/0009-0002-3796-7760

DOI: <http://dx.doi.org/10.17515/resm2026-1996ic0721rs>

Res. Eng. Struct. Mat. Vol. x Iss. x (xxxx) xx-xx

that can poison FL models [8]. In response to rising cyberattack patterns by adversaries, we aim to design a decentralized and robust security solution.

In fact, GAN-based poisoning attacks have been demonstrated as one of the most severe attack vectors for federated learning (FL) [2]. Poisoning attacks leverage FL's distributed learning process by impersonating an honest participant within the federated learning system, subsequently uploading data that is polluted and harmful [9]. The polluted data may lead to decreased accuracy, biased outcomes, or even backdoor attacks. Moreover, current Intrusion Detection Systems (IDS) or other anomaly detection methods may not be able to detect poisoning attacks happening in real-time [10]. Existing solutions attempted to solve poisoning attacks through centralized trust validation, which goes against FL's core values and brings many new problems to FL, such as single points of failure. Hence, without decentralized multi-layer defense, FL models are vulnerable to poisoning attacks.

To deal with poisoning attacks, different solutions such as robust aggregation or blockchain-assisted validation protocols have been proposed for FL [11]. For example, robust aggregation protocols that rely on Byzantine-resilient aggregation algorithms such as Krum, Trimmed Mean and FoolsGold aggregation, are capable of removing poisoned updates [12]. However, they cannot differentiate GAN-generated adversarial samples from natural samples and suffer from high false positives and/or accuracy trade-offs.

Conversely, FL models with blockchain implemented have enabled tamper-proof logging mechanism to mitigate poisoning attacks [7]. Although distributed ledger improves transparency and auditability, they are inherently limited by scalability, computing overhead and flexibility to accommodate FL's ever-changing environment [13]. Most of these methods also lack a dynamic consensus mechanism to adjust according to the attack patterns [14]. Hence, poisoning attacks can still wreak havoc in FL models with improved evasion techniques.

To address these limitations, this paper proposes a decentralized multi-layered security framework that integrates blockchain technology, consensus-driven validation, and adversarial detection techniques. Unlike conventional approaches, our solution introduces two distinct security layers:

- A preliminary validation layer that authenticates local model updates before aggregation.
- A consensus-driven aggregation mechanism that selects only the most trustworthy updates while systematically identifying and excluding malicious participants.

Additionally, by leveraging distributed ledger technology (DLT), our approach ensures tamper-proof logging, enhancing system transparency and resilience. Our experimental evaluation demonstrates that this framework achieves higher detection rates, reduces false positives, and significantly improves model reliability compared to existing FL security mechanisms.

This study aims to develop and evaluate a decentralized, multi-layered security framework that mitigates GAN-based poisoning attacks in federated learning, ensuring model integrity, reliability, and robustness in 6G-enabled distributed systems. The research objectives can be described as:

- To analyze GAN-based poisoning attacks and their impact on federated learning models.
- To design a decentralized federated learning security framework incorporating blockchain, validation layers, and adversarial detection techniques.
- To evaluate the proposed framework's effectiveness, accuracy, and efficiency against state-of-the-art FL security models.

To achieve these research objectives, we have established the following research questions:

- How do GAN-based poisoning attacks affect the accuracy and reliability of federated learning models?
- How can blockchain and consensus-driven validation enhance the security and transparency of FL models?
- To what extent does the proposed multi-layered security framework outperform existing FL security techniques in detecting and mitigating GAN-based poisoning attacks?

The increasing reliance on federated learning in critical applications such as healthcare, finance, and autonomous systems makes securing these models a high priority. This research bridges the gap between FL security and blockchain-driven decentralization, providing a novel, scalable, and effective solution for countering adversarial attacks. Unlike conventional approaches, our multi-layered security framework enables real-time attack detection, improved model integrity, and a moderate computational overhead, making it a practical and robust solution for securing FL environments. Moreover, this study contributes to federated learning research and cybersecurity by introducing an adaptive, consensus-based mechanism that dynamically filters out adversarial updates, enhancing the reliability of FL models. The insights from this research will be valuable for organizations deploying distributed AI models in high-risk environments, ensuring trustworthy and attack-resilient AI-driven systems.

The main contributions of this work, and its novelty with respect to robust aggregation rules (e.g., Krum, Trimmed Mean and FoolsGold), secure aggregation frameworks (SAFElearn and FLGUARD) and existing blockchain-based FL systems, are as follows:

- A two-layer defense that couple's cryptographic validation (RSA signatures, Pedersen commitments and zk-SNARK proofs of bounded, well-formed updates) with content-based anomaly detection. Secure aggregation schemes such as SAFElearn protect the confidentiality of updates but do not assess whether an authenticated update is poisoned, whereas robust aggregation rules assess update statistics but cannot authenticate their origin; the proposed framework performs both before aggregation.
- An adaptive detection mechanism that fuses three complementary measures (KL divergence, autoencoder reconstruction error and Mahalanobis distance) into one anomaly score whose threshold is re-estimated in every round, in contrast to the similarity heuristic of FoolsGold and the clustering-based filtering of FLGUARD, neither of which maintains a per-client history of behavior.
- A reputation-coupled consensus in which the reputation score simultaneously determines a client's aggregation weight, its Proof-of-Stake stake and its exclusion from the federation, while every validation outcome is recorded on a permissioned blockchain. Existing blockchain-based FL systems mainly use the ledger for auditability and incentives and do not tie consensus to the anomaly evidence of each update.
- A formal specification of the complete protocol, including the GAN-based poisoning threat model, the Byzantine fault-tolerance bound, the computational complexity and the on-chain storage cost, together with an experimental evaluation on image and IoT healthcare security datasets.

The principal scientific contribution is therefore not a new cryptographic primitive or an isolated detector, but the formal and engineering integration of authentication, anomaly evidence, reputation and consensus into a single decentralized decision rule in which each component compensates for a known weakness of the others.

Organization of the Rest of Paper. Section 2 presents background works and critical discussion on FL security techniques, blockchain and the detection of adversarial attacks. Section 3 introduces the design of our proposed framework in detail, such as threat model, cryptographic validation layer, anomaly detection layer, the aggregation driven by consensus along with its complexity analysis. Section 4 illustrates the experiment settings. Section 5 reports experimental results, discussion and limitations. Section 6 compares our methods with others. Finally, Section 7 concludes the paper with discussions on practical considerations and future work.

2. Literature Review

FL is being rapidly implemented in the upcoming wireless networks, i.e., 6G networks. Although FL brings new features, it also faces security threats from attackers like GAN-based poisoning attacks [1]. The authors conducted various research to secure FL systems using methods like Byzantine-robust aggregation techniques, Blockchain-based frameworks, and implementing efficient anomaly detection systems. This literature review discusses those works where the authors proposed methods to secure FL systems from adversarial attacks. Although initial research analyzed secure

FL systems using WFL that combined both communication and computing for future 6G networks [15], they did not account for malicious entities. Works that performed FL specifically for 6G networks mentioned security concerns but did not provide solutions to prevent poisoning attacks on data [16]. Other works have implemented resource allocation for FL over wireless networks without accounting for adversarial users [17].

Blockchain technology has been proposed as a method for strengthening FL security, which was explored through the design of BC-integrated drone networks for decentralized learning [18]. However, these architectures mainly tackle trust and transparency issues without specifically targeting advanced poisoning threats. Privacy-preserving FL solutions for 6G have also been discussed, though many overlook defenses against instantaneous attacks [19]. FedPARL presented an optimization framework focusing on resource-limited efficiency improvements but did not incorporate resilient security features [20].

Approaches towards ameliorating data poisoning attacks on FL have also been considered with robust aggregation frameworks like FoolsGold and adversarial detection techniques [21]. These typically utilize some form of similarity metric to detect Sybil attacks but can produce false positives due to the use of statistical estimates. There have also recently been attempts at anti-GAN frameworks that thwart adversarial learning, however compromised participants can still poison overtrained networks [22].

Other techniques applied to Byzantine-robust FL have implemented defensive techniques with cryptography-backed validation mechanisms alongside participant origin-tracing. Examples of these frameworks include SAFElearn [25] and FLGUARD [26]. These methods work well in defending against attacks which they have been explicitly tailored to, but are not flexible in reacting to adaptive malicious behavior. Solutions such as validation-inspired poisoning detection and VAMP validation offer improved security but come at the cost of computational resources [27].

Approaches employing decentralized consensus methods inside FL, like Proof-of-Stake (PoS) validation and Merkle-tree-based provenance logging can effectively guard against malicious model updates [28]. However, many FL defense solutions do not account for adaptive attacks from GAN-based adversaries, where the need for sustained monitoring with online anomaly detection arises. While security models based on smart contracts have shown success in helping to secure FL transactions, their applications for security in adversarial FL have not [29].

In addition to these techniques, each family of defenses has been shown to have complementary strengths and weaknesses when comparatively evaluated. FedAvg [30], which averages client updates weighted by their data size, has no protective mechanism at all. Therefore, any collusion of a single malicious client can move the global model to an arbitrary location. Krum [31] accepts the update closest to its neighbors and provably tolerates f Byzantine clients out of n when $n > 2f + 2$. However, Krum disregards the majority of benign updates, and under non-IID data distributions it will often mistakenly reject honest clients whose updates are simply far from each other. Algorithms such as coordinate-wise Trimmed Mean and Median [32] offer order-optimal statistical error rates, but require *a priori* knowledge* of the number of attackers and are vulnerable to attacks tailored to fall into the benign range of every coordinate [24]. Bulyan [33] relies on Krum followed by trimmed averaging to fend off such attacks, but requires $n \geq 4f + 3$ clients to function and has very high runtime costs. FLTrust [34] constructs trust scores bootstrapped from a small clean “root” dataset stored on the server. It works well but requires a fully trusted central party with data that is representative of all clients, preventing its use in fully decentralized settings. Robust federated aggregation (RFA) [35] aggregates client updates using the geometric median. This is robust to corruption of less than half of the updates but allows malicious updates that land close to the median to go unnoticed. FedProx [36] adds a proximal term to the objective function that helps keep training stable under data heterogeneity but is not itself a security mechanism.

On the attack side, GAN-based attacks were first shown to reconstruct private samples of other participants in collaborative learning [37], and PoisonGAN [38] demonstrated that a generator trained without access to other clients’ data can craft poisoned samples that bypass similarity-based defenses. Model replacement attacks [39] further show that a single scaled update can implant a persistent backdoor. Blockchain-based FL frameworks, such as committee-consensus

designs [29] and blockchained on-device FL [28], provide decentralization and auditability, but a ledger only guarantees that updates are recorded faithfully; it does not assess whether a recorded update is poisoned, and learning latency grows with the block generation time [28]. Consequently, no single family simultaneously provides authentication, content-based detection, adaptivity to non-IID data and decentralization, which motivates the integrated design proposed in this paper.

Overall, although there have been several approaches to hardening FL to adversarial attacks, many solutions lack scalability, flexibility, or detection capabilities during model training. This paper hopes to address these problems by presenting a decentralized, layered security architecture for FL that utilizes blockchain-inspired verification, consensus-based aggregation, and adversarial detection. Our approach will adapt many of the techniques from previous works while improving robustness, detection accuracy, and computational overhead in defending against GAN-poisoning attacks targeting FL networks. Table 1 summarizes all papers discussed.

Table 1. Summary of key contributions in FL security

TE	Contribution	Key Features	Ref
6G FL	WFL to integrate communication and computation	Enhances wireless FL efficiency, lacks security focus	[15]
6G FL	Overview of federated learning in 6G communications	Identifies FL integration challenges, no concrete security solutions	[16]
6G FL	Resource distribution for FL in mobile networks	Optimizes FL for WN, lacks adversarial threat considerations	[17]
6G FL	Blockchain-based federated learning for UAV networks	Uses blockchain to enhance FL security, no targeted defense	[18]
6G FL	Comprehensive study of 6G-FL applications and challenges	Explores 6G-FL applications, does not address poisoning attacks	[19]
FL	FedPARL: Lightweight FL model for resource-limited IoT	Focuses on FL efficiency, lacks robust adversarial resilience	[20]
FL	Analysis of data poisoning and a detection defense	Detects and blocks attacks but struggles with dynamic threats	[21]
FL	Anti-GAN architecture for adversarial defense in FL	Obfuscates GAN learning but vulnerable to overtraining manipulation	[22]
FL	FoolsGold: Sybil attack mitigation using contribution similarity	Uses contribution similarity for attack detection, risks false positives	[23]
FL	Parameter estimation for model update improvement	Enhances FL updates but does not mitigate poisoning attacks	[24]
FL	SAFElearn: Secure aggregation-based FL model	Provides secure aggregation, does not adapt to evolving threats	[25]
FL	FLGUARD: Federated learning protection model	Defends against targeted attacks but lacks dynamic adaptability	[26]
FL	Blockchain-based on-device FL with latency analysis	Ensures FL transaction security, scalability issues remain	[28]
ML	VAMP: authentication and provenance against ML poisoning	Verifies data origin but not the content of updates	[27]

3. Proposed Methodology

The problem of adversarial resilience in FL arises due to the decentralized nature of model training, where malicious participants can introduce poisoned updates to compromise the global model. Specifically, Generative Adversarial Network (GAN)-based attacks generate synthetic perturbations to mimic benign contributions while degrading model performance. The problem can be formally defined as follows:

3.1 Problem Formulation and Threat Model

Let $C = C1, C2, \dots, Cn$ represent a set of n participating clients in an FL system, each with a local dataset D_i . Each client trains a local model M_i with parameters θ_i^t at iteration t using stochastic gradient descent (SGD):

$$\theta_i^{t+1} = \theta_i^t - \eta \nabla L(D_i, \theta_i^t) \quad (1)$$

where η is the learning rate, ∇ is the gradient operator, and $L(D_i, \theta_i^t)$ is the empirical cross-entropy loss of client C_i , defined as:

$$L(D_i, \theta_i^t) = \frac{1}{|D_i|} \sum_{(x,y) \in D_i} \ell_{CE}(f_{\theta}(x), y) \quad (2)$$

where $f_{\theta}(x)$ is the SoftMax output of the model for sample x , y is its true label, and ℓ_{CE} is the cross-entropy function.

GAN-based poisoning attack model. Following the generative poisoning strategies reported in [37], [38], a malicious client C_a initializes a discriminator D from the global model θ^t received at round t and trains a local generator G using only its own local dataset D_a ; the adversary has no access to the data of other clients. The generator and the discriminator solve the minimax problem:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_a} [\log D(x)] + \mathbb{E}_{z \sim p_z} [\log(1 - D(G(z)))] \quad (3)$$

where p_a is the distribution of the attacker's local data and p_z is a 100-dimensional standard normal noise distribution. The generator is a DCGAN-style network (a fully connected layer followed by two transposed convolution layers with batch normalization and ReLU activations, and a Tanh output layer), and the discriminator has the architecture of the classification model shared by the federation. The trained generator synthesizes samples that resemble a source class, and the attacker assigns them an attacker-chosen target label y_{tgt} (targeted poisoning), producing the poisoned local dataset:

$$\tilde{D}_a = D_a \cup \{(G(z_k), y_{tgt})\}_{k=1}^{\lceil \rho_p |D_a| \rceil} \quad (4)$$

where ρ_p is the poisoning ratio, i.e., the number of injected synthetic samples relative to the size of the local dataset. The attack is persistent: once started, the malicious clients submit poisoned updates in every subsequent round.

Each adversarial participant $C_a \in C$ then trains on $\tilde{D}_a$ and submits the poisoned update:

$$\theta_a^{t+1} = \theta_a^t + \delta_a, \quad \delta_a = -\eta \nabla L(\tilde{D}_a, \theta_a^t) \quad (5)$$

where δ_a is an adversarial perturbation designed to manipulate global model aggregation. The central objective is to identify and mitigate these adversarial manipulations while ensuring computational efficiency and scalability. Because the poisoned update is produced by a legitimate, registered client, it passes authentication and must be detected from its content.

3.2 Constraints and Challenges

The FL security problem is constrained by several factors:

- Security vs. Efficiency Trade-off: Ensuring robust adversarial detection while minimizing computational overhead.
- Scalability Constraints: Handling many participants without performance degradation.
- Dynamic Adversarial Adaptation: Attackers modify their strategies over time, requiring an adaptive defense mechanism.
- Decentralized Verification: A trustless approach is required, making central validation infeasible.

To formally define the constraints, we introduce a security function $S(\theta)$ that enforces verification of updates, such that:

$$P(S(\theta_i^{t+1}) = 1 | C_i \text{ is benign}) \approx 1 \quad (6)$$

$$P(S(\theta_a^{t+1}) = 1 | C_a \text{ is adversarial}) \approx 0 \quad (7)$$

where $S(\theta)$ should accept benign updates and reject adversarial contributions with high probability.

3.3 Proposed Multi-Layer Security Model

To address these constraints, the proposed framework introduces a decentralized, multi-layered security model for FL, integrating:

- Cryptographic validation via digital signatures and homomorphic commitments.
- Adversarial detection through statistical divergence (DKL) and Mahalanobis distance measures.

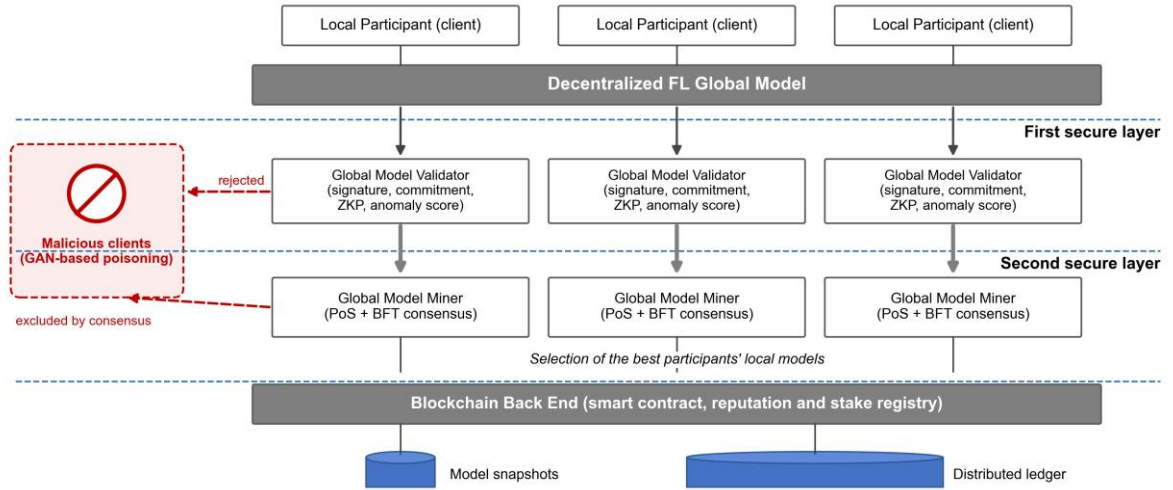


Fig. 1. Architecture of the decentralized FL framework with two security layers. The first layer (validators) authenticates clients and verifies each update using signatures, commitments, zero-knowledge proofs (ZKP) and anomaly scores; the second layer (miners) applies Proof-of-Stake (PoS) committee selection and Byzantine Fault Tolerant (BFT) consensus before the accepted models are recorded on the blockchain

- Decentralized consensus aggregation using Byzantine Fault Tolerant (BFT) mechanisms and Proof-of-Stake (PoS) blockchain validation.
- Client authentication and auditability through a smart-contract registry of client public keys and immutable logging of all validation outcomes.

Blockchain architecture and communication protocol. The ledger layer is a permissioned (consortium) Ethereum network in which only enrolled validator and miner nodes can create blocks. A Solidity smart contract maintains four registries: (i) the public keys of registered clients, used for authentication; (ii) the per-round update commitments; (iii) the reputation scores and stakes; and (iv) the blacklist of excluded devices. To avoid storing large models on-chain, model parameters are exchanged off-chain between clients and aggregation nodes over authenticated TLS channels, and only fixed-size metadata (the commitment digest, the signature, the verification outcome and the reputation score) are written to the ledger. Consensus on the global model is reached in two stages: a Proof-of-Stake (PoS) selection of a validator committee according to the stakes in Eq. (24), followed by a Byzantine Fault Tolerant (BFT) voting round among the selected validators (Section 3.6).

3.4 Client Authentication and Secure Local Model Validation

Each client node C_i trains a local model M_i on its private dataset D_i using the SGD rule in Eq. (1). Before its update is accepted, the client must be authenticated and the update must pass a multi-stage cryptographic verification. During enrolment, each client generates an RSA key pair and registers the public key PK_i in the smart contract:

$$(PK_i, SK_i) \leftarrow KeyGen(1^\lambda) \quad (8)$$

where λ is the security parameter and SK_i is the private key of client C_i . RSA-2048 keys with PSS padding and SHA-256 hashing [40] are used. A client whose public key is not registered, or whose device is on the blacklist, is rejected before any further processing. Each update is then validated in three steps:

- Signature-based authenticity verification using a digital signature scheme:

$$\sigma_i^t = Sign(H(\theta_i^{t+1}), SK_i) \quad (9)$$

where $H(\cdot)$ is the SHA-256 hash function. $Sign(\cdot)$ computes the RSA-PSS signature of the digest of the update, and any node can check it with the public function $Verify(\sigma_i^t, H(\theta_i^{t+1}), PK_i) \in \{0, 1\}$, which returns 1 only if the update was produced by the registered client and was not modified in transit.

- Commitment verification for consistency:

$$C_i^t = Commit(\theta_i^{t+1}, v_i^t) = g^{H(\theta_i^{t+1})} h^{v_i^t} \bmod p \quad (10)$$

where v_i^t is a new random nonce used to ensure each commitment is distinct, p is a large prime, and g and h are generators of a prime order group for which the discrete logarithm of one with respect to the other is unknown. This Pedersen commitment [41] is both binding, meaning a client cannot later try to claim a different update was made, and hiding, meaning it leaks no information about the parameters. Because the scheme is additively homomorphic, multiple clients' commitments can be verified jointly with a single product, lightening the load on validator nodes.

- Zero-Knowledge Proof (ZKP) of update well-formedness:

$$ZKP - Verify(\pi_i^t, C_i^t, vk) = \begin{cases} True, & \text{if } \pi_i^t \text{ is valid for } C_i^t \\ False, & \text{otherwise} \end{cases} \quad (11)$$

where π_i^t is the proof generated by client C_i and vk is the public verification key. The ZKP protocol is a zk-SNARK based on the Groth16 construction [42]. The proven statement is that the client knows an opening (θ_i^{t+1}, v_i^t) of the commitment C_i^t whose update norm $\|\theta_i^{t+1} - \theta^t\|_2$ does not exceed a public bound τ ; validators thus confirm that the update is well formed and bounded without observing the parameters. A Groth16 proof consists of three group elements, and its verification requires a constant number of pairing operations regardless of model size, so the verification cost is $O(1)$ per client. Proof generation is linear in the circuit size and is the dominant cryptographic cost at the client; it therefore adds latency to each local round, and this latency is included in the secure training times of Table 4. The three checks are combined into a single validation outcome that is recorded on the blockchain:

$$V_i^t = Verify(\sigma_i^t, H(\theta_i^{t+1}), PK_i) \wedge ZKP - Verify(\pi_i^t, C_i^t, vk) \quad (12)$$

An update with $V_i^t = 0$ is discarded, the client is marked as adversarial for that round, and the event is logged on the ledger.

Figure 1 illustrates the multi-layered security architecture of the proposed decentralized federated learning framework.

3.5 Adversarial Detection Via GAN Resistance Mechanism

Updates that pass the cryptographic checks may still be poisoned, because a GAN-based attacker signs and commits its poisoned update correctly (Section 3.1). The content of each update is therefore analyzed. Since model updates are high dimensional, each update is first mapped to a k -dimensional representation ($k = 512$) with a fixed Gaussian random projection matrix $R \in \mathbb{R}^{k \times d}$ shared through the smart contract:

$$z_i^t = R(\theta_i^{t+1} - \theta^t) \quad (13)$$

Three complementary anomaly measures are computed on z_i^t . First, a statistical divergence compares the distribution of the received update with that of benign updates:

$$D_{KL}(P \parallel Q_i) = \sum_{b=1}^B P(b) \log \frac{P(b)}{Q_i(b)} \quad (14)$$

where P is the normalized histogram ($B = 50$ bins) of the coordinates of the aggregated benign update of the previous round, Q_i is the normalized histogram of z_i^t over the same bins, and a constant $\varepsilon = 10^{-8}$ is added to every bin to avoid division by zero. Second, an undercomplete autoencoder computes the reconstruction error of the update:

$$A(z_i^t) = \|z_i^t - \hat{z}_i^t\|_2^2, \quad \hat{z}_i^t = g_\varphi(f_\psi(z_i^t)) \quad (15)$$

where f_ψ is the encoder and g_φ is the decoder, so that $\hat{z}_i^t$ is the reconstruction of z_i^t . The encoder is a fully connected network with 512, 256 and 64 neurons and ReLU activations, and the decoder mirrors it (64, 256 and 512 neurons with a linear output). The autoencoder is retrained with the Adam optimizer on the accepted updates of a sliding window of the last $W = 5$ rounds by minimizing:

$$L_{AE}(\psi, \varphi) = \frac{1}{|Z_W|} \sum_{z \in Z_W} \|z - g_\varphi(f_\psi(z))\|_2^2 \quad (16)$$

where Z_W is the set of accepted projected updates in the window. Third, the Mahalanobis distance measures how far an update lies from the benign population while accounting for correlations between coordinates:

$$D_M(z_i^t) = \sqrt{(z_i^t - \mu)^T \Sigma^{-1} (z_i^t - \mu)} \quad (17)$$

where μ and Σ are the mean vector and the shrinkage covariance matrix (with εI added for numerical stability) of the updates in Z_W . The three measures are min-max normalized over the clients of the current round (denoted by a tilde) and fused into a single anomaly score $\gamma_i^t \in [0, 1]$:

$$\gamma_i^t = \frac{1}{3} (\tilde{D}_{KL,i} + \tilde{A}_i + \tilde{D}_{M,i}) \quad (18)$$

An update is flagged as adversarial when $\gamma_i^t > \delta^t$. The threshold δ^t is not fixed; it is re-estimated in every round from the anomaly scores of the updates accepted in the window W :

$$\delta^t = \mu_\gamma + \kappa \sigma_\gamma \quad (19)$$

where μ_γ and σ_γ are the mean and standard deviation of these scores and $\kappa = 2$. During the first W rounds, when no history is available, the median and the median absolute deviation of the scores of the current round are used instead of μ_γ and σ_γ . The validation and consensus mechanism for detecting and filtering adversarial contributions is depicted in Figure 2.

3.6 Decentralized Consensus Aggregation and Reputation Update

To prevent malicious model updates from influencing the global model, the second layer applies a Byzantine Fault Tolerant (BFT) consensus based on Practical Byzantine Fault Tolerance (PBFT) [43]. In each round, a committee of N_v validator nodes is selected by PoS according to the stakes in Eq. (24). Each validator independently recomputes V_i^t and γ_i^t for every client and votes to accept or reject the update. An update is accepted, and the block with the round results is committed, only if it receives at least $2f + 1$ matching votes, where f is the maximum number of faulty or malicious validators:

$$N_v \geq 3f + 1, \quad \text{votes}_i \geq 2f + 1 \quad (20)$$

The consensus layer therefore tolerates up to $f = \lfloor (N_v - 1)/3 \rfloor$ Byzantine validators, i.e., fewer than one third of the committee. At the client level, each update is filtered individually; however, because the adaptive threshold in Eq. (19) is estimated from accepted updates, the framework assumes that

malicious clients are a minority (less than 50%) of the participants in each round. The accepted updates form the validated set $\mathcal{V}^t$, which is aggregated by reputation-weighted averaging:

$$\theta^{t+1} = \sum_{i \in \mathcal{V}^t} w_i^t \theta_i^{t+1} \quad (21)$$

where $\mathcal{V}^t$ is the set of validated clients in round t and w_i^t is the reputation-based weight:

$$w_i^t = \frac{r_i^t}{\sum_{j \in \mathcal{V}^t} r_j^t} \quad (22)$$

where $r_i^t \in [0, 1]$ is the reputation score of client C_i . All reputation scores are initialized to $r_i^0 = 0.5$ and updated after each round as:

$$r_i^{t+1} = \rho r_i^t + (1 - \rho) q_i^t, \quad q_i^t = \begin{cases} 1 - \gamma_i^t, & \text{if } i \in \mathcal{V}^t \\ 0, & \text{otherwise} \end{cases} \quad (23)$$

where $\rho = 0.9$ is a forgetting factor. Accepted clients with low anomaly scores gain reputation, whereas the reputation of rejected clients decays geometrically. A client whose reputation falls below $r_{min} = 0.2$ is added to the blacklist in the smart contract and excluded from subsequent rounds.

The aggregation is reinforced using a Proof-of-Stake (PoS) blockchain to ensure transparency. Each node receives a stake that combines its long-term reputation with its verification outcome in the current round:

$$S_i^t = \alpha r_i^t + \beta V_i^t \quad (24)$$

where $\alpha \in [0, 1]$ and $\beta \in [0, 1]$ are weighting coefficients with $\alpha + \beta = 1$ that balance long-term behavior against the current verification result, and $V_i^t \in \{0, 1\}$ is given by Eq. (12). In this work, $\alpha = 0.7$ and $\beta = 0.3$, so the stake is dominated by reputation; this prevents a new or intermittently honest node from acquiring a large stake in a single round. Because S_i^t is bounded in $[0, 1]$, no single node can dominate the committee selection.

The process of model selection and integration with blockchain consensus mechanisms is shown in Figure 3.

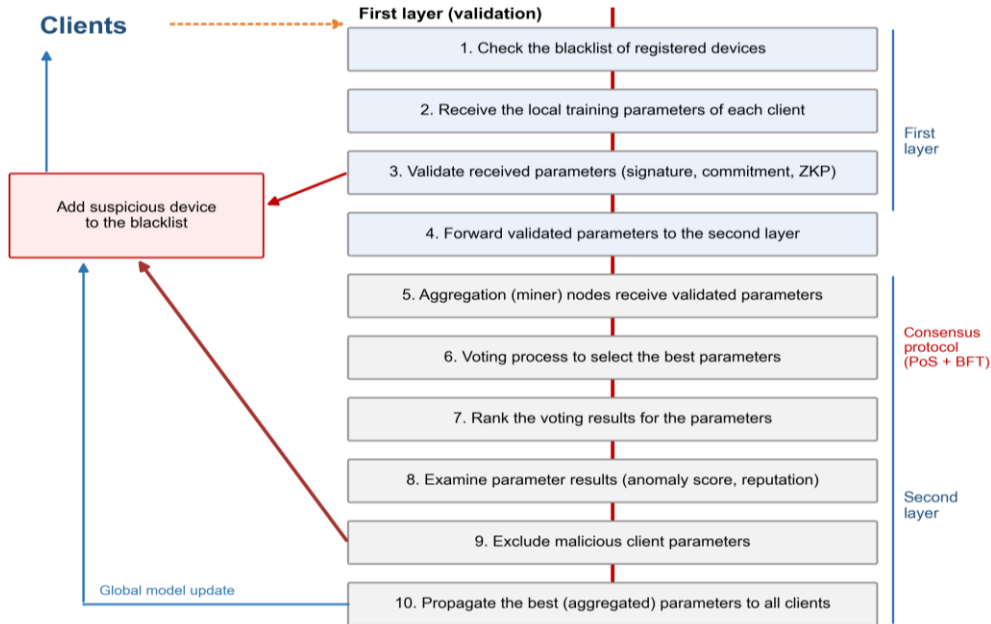


Fig. 2. Validation and consensus workflow. Steps 1–4 form the first (validation) layer and steps 5–10 the second (consensus) layer; devices detected as suspicious in steps 3 or 9 are blacklisted, and the aggregated global model is returned to the clients

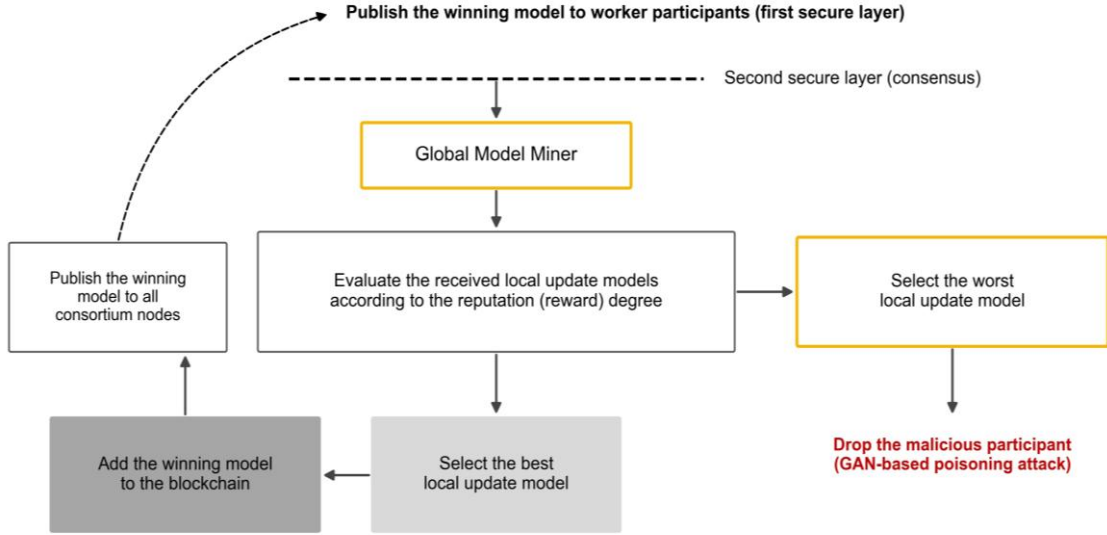


Fig. 3. Model selection and blockchain integration performed by a global model miner: local updates are ranked by reputation, the worst update is dropped as a suspected GAN-based poisoning contribution, and the winning model is recorded on the blockchain and published to all nodes

3.7 Algorithm for Secure Federated Learning

Algorithm 1 Secure Federated Learning with GAN-Resistant Aggregation

- Input: clients $C = \{C_1, \dots, C_n\}$ with local datasets D_i ; learning rate η ; local epochs E ; maximum rounds T ; window W ; κ ; ρ ; α ; β ; r_{min} ; tolerance ϵ
- Output: global model θ ; reputation scores r_i ; ledger record of accepted updates
- Initialize θ_0 and $r_{i0} = 0.5$; register public keys PK_i (Eq. 8)
- for $t = 0, 1, \dots, T - 1$ do
 - for each non-blacklisted client C_i in parallel do
 - Train θ_{it+1} for E epochs with SGD (Eqs. 1–2)
 - Compute σ_{it} , C_{it} and proof π_{it} (Eqs. 9–11)
 - end for
 - for each received update do
 - Compute V_{it} (Eq. 12); if $V_{it} = 0$ then reject C_i
 - Compute z_{it} and y_{it} (Eqs. 13–18)
 - if $y_{it} > \delta t$ (Eq. 19) then flag C_i as adversarial
 - end for
 - Select the committee by PoS (Eq. 24) and run BFT voting (Eq. 20) to obtain $\mathcal{V}_t$
 - Aggregate θ_{t+1} over $\mathcal{V}_t$ (Eqs. 21–22)
 - Update r_{it+1} (Eq. 23); blacklist clients with $r_{it+1} < r_{min}$
 - Record commitments, signatures, V_{it} and r_{it+1} on the blockchain
 - if $\|\theta_{t+1} - \theta_t\|_2 / \|\theta_t\|_2 < \epsilon$ for 5 consecutive rounds then break
- end for
- return global model θ_{t+1}

Algorithm 1 takes the client datasets and the hyperparameters of the defense as inputs and returns the final global model together with the reputation scores and the ledger record. Training stops when the maximum number of rounds T is reached or when the relative change of the global model remains below $\epsilon_c = 10^{-4}$ for five consecutive rounds.

3.8 Computational Complexity, Convergence and Blockchain Overhead

Let n be the number of clients, d the number of model parameters, k the projection dimension and N_v the committee size. Local training costs $O(E \cdot |D_i| \cdot d)$ per client, as in FedAvg [30]. Signing and committing require one hash of the update, $O(d)$, and a constant number of modular

exponentiations. At the validators, signature and ZKP verification cost $O(1)$ per client, the random projection $O(kd)$, the KL divergence $O(k + B)$, the autoencoder and the Mahalanobis distance $O(k^2)$ per client after one $O(k^3)$ covariance inversion per round, the aggregation $O(nd)$, and the BFT voting $O(N_v^2)$ messages. The per-round cost of the security layers is therefore $O(n(kd + k^2) + k^3 + N_v^2)$, which is linear in the number of clients and in the model size for fixed k and N_v .

Regarding convergence, Eq. (21) is a weighted average of validated updates. When the validated set contains only benign clients, the procedure reduces to weighted FedAvg over the benign subset, and the convergence analysis of FedAvg on non-IID data [44], which assumes smooth, strongly convex local losses with bounded stochastic gradient variance, applies with w_i^t in place of the data-size weights. If a malicious update passes the filter, its influence is limited by its reputation weight, which decays geometrically according to Eq. (23).

Regarding blockchain overhead, the on-chain data per client per round are of fixed size: a 32-byte commitment digest, a 256-byte RSA-2048 signature, a 1-byte verification outcome and a 32-byte reputation value, i.e., approximately 321 bytes, independent of the model size. For $n = 20$ clients and $T = 50$ rounds this amounts to about 321 kB of ledger data. Groth16 proofs are verified by the validators and only the outcome is stored. Under the Ethereum gas schedule, writing a new 32-byte storage word costs 20,000 gas, so the metadata of one client require roughly 2×10^5 gas per round. In the permissioned PoS network used here, gas has no monetary value and acts only as a resource limit, and the block interval is set by the consortium. However, transaction latency and throughput requirements grow with the number of clients, and a deployment on the public Ethereum network with many clients would incur substantial gas costs and confirmation delays; this limitation is discussed in Sections 5.1 and 7.

4. Experimental Setting

The experimental evaluation of the proposed federated learning security framework is conducted on multiple datasets to assess its robustness against adversarial attacks, particularly GAN-based poisoning. The datasets used encompass a variety of domains, ensuring a comprehensive validation of the model's effectiveness in detecting and mitigating adversarial threats while maintaining high predictive accuracy.

4.1 Datasets and Data Partitioning

MNIST contains 60,000 training and 10,000 test greyscale images of handwritten digits. To simulate a non-IID federated scenario, the training set is sorted by label and divided into 40 shards of 1,500 images, and each of the $n = 20$ clients receives two shards, so that every client holds samples of only two-digit classes, following the pathological non-IID partition of [30]. CIFAR-10 and CIFAR-100 contain 50,000 training and 10,000 test color images in 10 and 100 classes, respectively, and are used to assess generalization to more complex image classification tasks. Their training sets are partitioned among the 20 clients using a Dirichlet distribution over class proportions with concentration parameter $\alpha_D = 0.5$ [45], which yields clients with different class distributions and dataset sizes. For all image datasets, the official test sets are used only to evaluate the global model.

To evaluate the framework in a connected healthcare environment, the WUSTL-EHMS-2020 Internet of Medical Things security dataset [46] is used, referred to in this paper as the IoT Healthcare dataset. It combines network flow metrics with patients' biometric measurements collected from a real-time healthcare monitoring testbed, and each record is labelled as normal or attack (spoofing and data alteration). The task performed on this dataset is binary intrusion detection. Identifier fields such as addresses and ports are removed, the remaining features are standardized with z-score normalization, and the records are segmented into windows of 10 consecutive records for the LSTM model. The dataset is split into 80% training and 20% test data with stratification by class, and the training part is distributed across the 20 clients with the same Dirichlet partition ($\alpha_D = 0.5$).

Evaluation Protocols We split all datasets into disjoint subsets for multiple simulated clients in federated learning deployment setting. During training, privacy constraints are maintained by

decentralizing raw data and limiting information exchange to model updates between clients and the global server. Adversarial attacks generated by GANs are incorporated at different levels of intensity to test the framework's robustness. Accuracy, precision, recall, and F1-score metrics are used to evaluate model performance while additional security metrics assess attack resistance and detection effectiveness. By evaluating these metrics, we can systematically assess the impact of security enhancements on both model performance and its resistance to adversarial threats.

4.2 Federated Training Configuration and Attack Settings

All experiments use $n = 20$ clients, all of which participate in every communication round. The global model is trained for $T = 50$ communication rounds; in each round every client performs $E = 1$ local epoch of mini-batch SGD with learning rate $\eta = 0.01$, momentum 0.9 and batch size 64, and the stopping rule of Section 3.7 is monitored. The classification models are a CNN with two convolutional layers (32 and 64 filters, 5×5 kernels, max pooling) and two fully connected layers for MNIST, ResNet-18 for CIFAR-10 and CIFAR-100, and a two-layer LSTM with 64 hidden units for the IoT Healthcare dataset. The defense parameters are $k = 512$, $B = 50$, $W = 5$, $\kappa = 2$, $\rho = 0.9$, $r_{min} = 0.2$, $\alpha = 0.7$, $\beta = 0.3$ and a validator committee of $N_v = 7$ nodes, which tolerates $f = 2$ Byzantine validators. In the default attack scenario, 20% of the clients (4 of 20) are malicious. Each malicious client trains the GAN of Section 3.1 on its local data for 200 iterations per round with the Adam optimizer (learning rate 2×10^{-4}), injects synthetic samples with a poisoning ratio $\rho_p = 0.5$ relabeled to an attacker-chosen target class (targeted attack), starts attacking at round 6 and continues in every subsequent round. The label-flipping and model inversion attacks in Table 3 are launched by the same set of malicious clients. For reproducibility, the random seeds of Python, NumPy and PyTorch are fixed to 42 for data partitioning, model initialization and attacker selection.

The framework was implemented in three programming languages: Python (PyTorch) for federated training and anomaly detection, TypeScript for the smart-contract interface, and C++ for the cryptographic routines. All experiments were run on a workstation with Ubuntu 20.04, an 11th-generation Intel Core i7 processor (2.6 GHz), 16 GB RAM and an NVIDIA GTX 1650 Ti GPU with 4 GB of memory. The GPU performs local model training, while the same machine hosts the nodes of the private Ethereum network; clients are simulated as separate processes that communicate with the aggregation nodes through local network sockets.

RSA-2048 digital signatures are used to authenticate participants, and all verification outcomes are recorded in each participant's ledger entry. The distributed ledger thereby provides a tamper-evident record of the security-relevant metadata of each participant, while private training data never leave the clients.

4.3 Evaluation Metrics and Code Availability

Model utility is measured by the test accuracy, precision, recall and F1-score of the global model. Detection performance is measured by the detection rate (the proportion of malicious updates that are rejected) and the false positive rate (the proportion of benign updates that are rejected). Computational cost is measured as the average wall-clock time per communication round with and without the security layers. The source code, configuration files and data partitioning scripts are available from the corresponding author upon reasonable request and will be released in a public repository upon publication.

5. Results and Discussion

The proposed federated learning security framework was evaluated using multiple datasets, measuring its performance under various adversarial conditions. The results are analyzed with respect to accuracy, adversarial resilience, and computational efficiency. This section presents detailed evaluations, visualized results, and explanatory insights into observed trends. The first phase of evaluation focused on federated learning performance in benign conditions, establishing a baseline. In this scenario, the model trained on MNIST, CIFAR-10, CIFAR-100, and IoT healthcare datasets demonstrated consistent accuracy across different training rounds. However, when GAN-based poisoning attacks were introduced, there was a notable decline in model accuracy and an increase in misclassified instances, as shown in Figure 4.

The proposed framework successfully mitigated these adversarial effects by applying multi-layered security validation. Table 2 illustrates the model's accuracy before and after adversarial mitigation across different datasets.

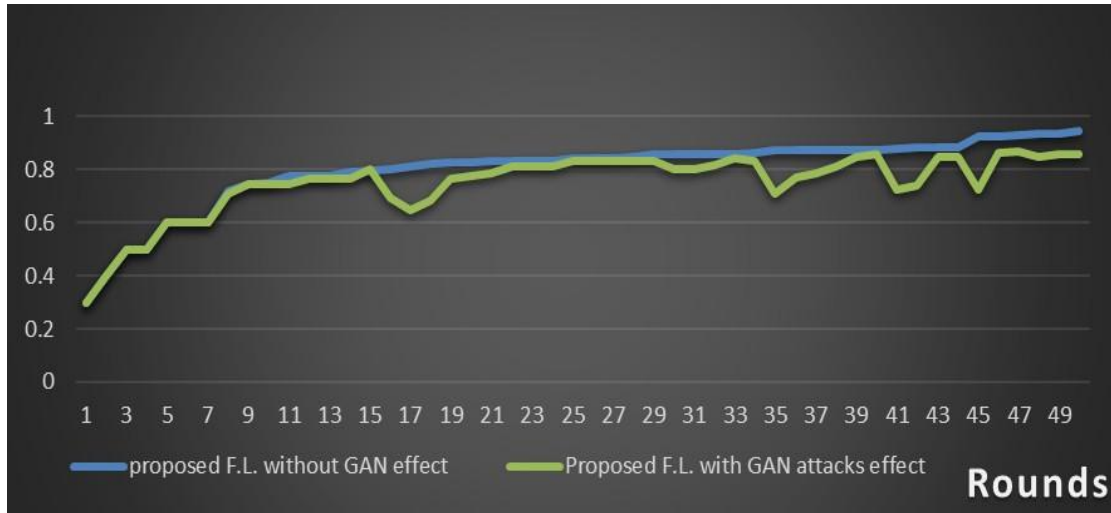


Fig. 4. Test accuracy (y-axis) of the proposed FL framework over communication rounds (x-axis) without attack (blue) and under the GAN-based poisoning attack (green)

A detailed breakdown of classification performance is shown in Figure 5, which presents confusion matrices under different adversarial conditions. The matrices highlight the impact of adversarial attacks on classification accuracy and how the proposed framework mitigates these effects. Each panel reports the predictions of the global model on a small, randomly selected subset of test samples from the classes involved in the corresponding scenario; the matrices therefore illustrate the qualitative behavior of the classifier, whereas the overall accuracy is reported in Table 2.

The validation process significantly reduced the number of malicious updates incorporated into the global model. Table 3 demonstrates the efficiency of the anomaly detection mechanism in identifying and filtering out adversarial clients.

The final phase of evaluation assessed the efficiency of consensus-driven aggregation in securing the final global model. The proposed approach ensures that only models contributing positively to the training process are aggregated. Figure 6 shows the accuracy of the global model over the communication rounds with and without the GAN-based attack, and Figure 7 shows the parameter convergence curves, which decrease towards zero in both settings, indicating that the aggregation converges despite the presence of malicious clients.

Table 2. Accuracy comparison before and after adversarial mitigation

Dataset	Baseline Accuracy	Accuracy after GAN Mitigation
MNIST	98.1%	98.7%
CIFAR-10	84.5%	91.2%
CIFAR-100	76.3%	82.1%
IoT Healthcare	89.7%	94.3%

Table 3. Detection rate of malicious clients

Attack Type	Detection Rate	Mitigation False Positive Rate
GAN-based Attack	97.8%	2.1%
Label Flipping Attack	96.4%	2.7%
Model Inversion Attack	94.2%	3.5%

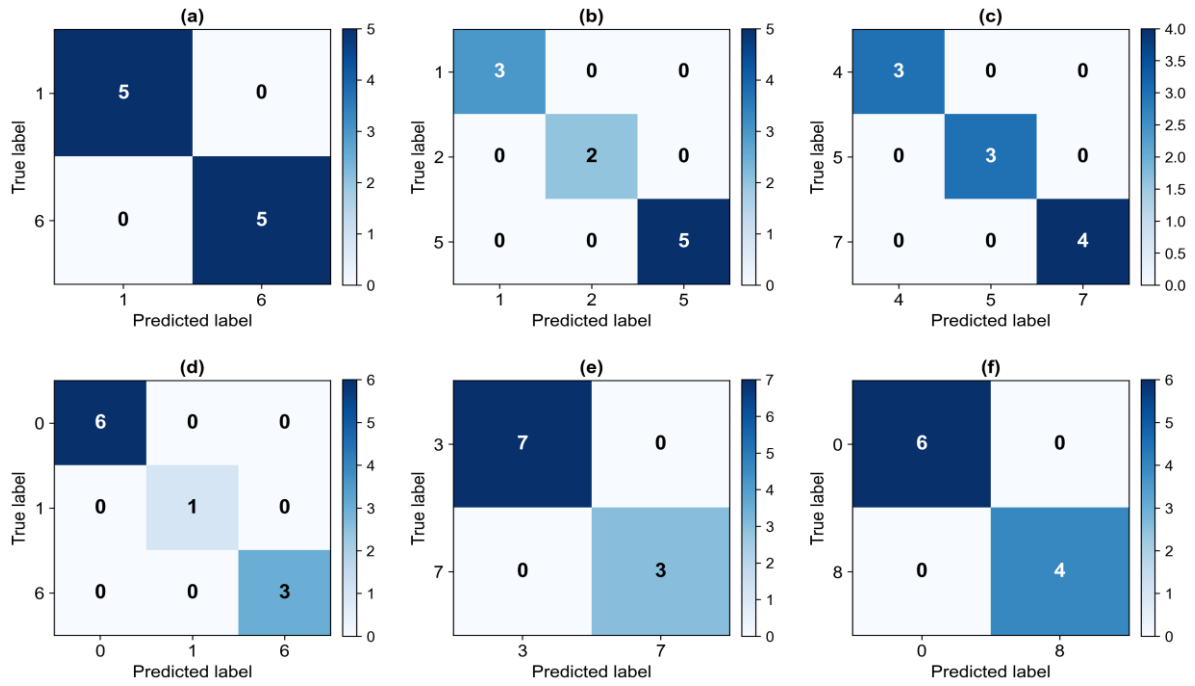


Fig. 5. Confusion matrices of the global model under different adversarial conditions (a)–(f). Each panel shows the classes involved in the scenario; rows are true labels, columns are predicted labels, and cell values are numbers of evaluated test samples

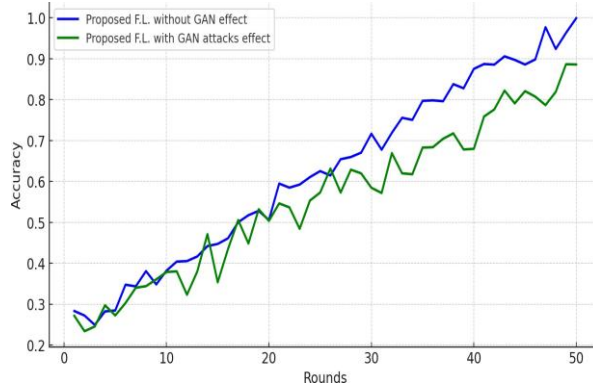


Fig. 6. Model stability over training rounds: accuracy of the proposed FL framework without GAN attack (blue) and with GAN attack (green)

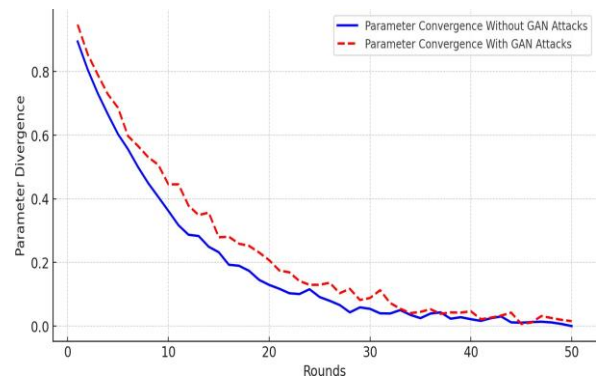


Fig. 7. Parameter convergence of the global model over communication rounds without GAN attacks (solid blue) and with GAN attacks (dashed red)

Table 4. Average training time per round

Model	Standard Training Time	Secure Training Time
MNIST CNN	12.4s	15.8s
CIFAR-10 ResNet	39.2s	44.6s
IoT Healthcare LSTM	28.7s	32.1s

Table 4 presents the average wall-clock time per communication round of standard federated training and of the proposed secure training; the latter includes signing, commitment and proof generation, verification, anomaly scoring, consensus voting and ledger logging. The security layers increase the time per round by 27.4% for the MNIST CNN, 13.8% for the CIFAR-10 ResNet and 11.8% for the IoT Healthcare LSTM. The relative overhead is largest for the smallest model because the cryptographic and consensus costs are largely independent of model size, whereas local training dominates for larger models. Over the complete training of $T = 50$ rounds, the total training

time increases from 620 s to 790 s for MNIST, from 1,960 s to 2,230 s for CIFAR-10, and from 1,435 s to 1,605 s for the IoT Healthcare dataset.

This moderate increase in training time is accompanied by substantial gains in adversarial resistance and model integrity. Because the on-chain data per client are of constant size (Section 3.8), the communication overhead of blockchain transactions grows linearly with the number of clients. Memory usage, energy consumption, transaction latency, throughput and the scaling behavior with 100 or 1,000 clients were not measured in the present experiments; these aspects are discussed as limitations in Section 5.1.

5.1 Limitations and Threats to Validity

The results should be interpreted in light of several limitations. First, the accuracy, detection rate and false positive values were obtained from a single run per configuration with a fixed random seed; repeated runs with different seeds, mean $\pm$ standard deviation, confidence intervals and statistical significance tests are needed to establish the reliability of the observed improvements. Second, the evaluation considers one default attack configuration (20% malicious clients and a poisoning ratio of 0.5); sensitivity analyses over the fraction of malicious clients, the poisoning ratio and the attack starting round, as well as an ablation study that removes each security component in turn, have not yet been performed. Third, the experiments use 20 simulated clients on a single workstation, so the scalability of the blockchain layer to hundreds or thousands of distributed clients, including network latency, throughput, block generation time, storage growth and energy consumption, remains to be quantified. Fourth, the framework was evaluated against GAN-based poisoning, label flipping and model inversion attacks. By design, the BFT consensus and anomaly score address Byzantine and model replacement updates, public-key registration and stake raise the cost of Sybil attacks, and the norm bound in the ZKP limits scaled backdoor updates; however, backdoor, Sybil, free-rider, inference and model replacement attacks were not evaluated experimentally. Finally, FedAvg, Krum, Multi-Krum, Trimmed Mean, Bulyan, FLTrust, RFA and FedProx were not re-implemented under identical settings, and the comparison in Section 6 relies on values reported in the literature.

6. Comparative Analysis

To demonstrate the efficacy of the proposed federated learning security framework, we compare its performance with existing state-of-the-art methods in adversarial mitigation within federated learning environments. The evaluation is conducted based on key performance indicators such as accuracy, attack detection rate, false positive rate, computational overhead, and resilience to adversarial attacks.

Several studies have proposed methods to enhance federated learning security. The FoolsGold approach [23] focuses on detecting Sybil attacks through contribution similarity analysis, but it suffers from a high false positive rate when handling dynamic adversaries. SAFElearn framework [25] supports secure aggregation, but has no dynamic defense against adaptive adversaries. FLGUARD [26] proposes a defense mechanism but is not adaptable to attack patterns. Adversarial robust federated learning [22] trains models adversarially, yet remains susceptible to backdoor poisoning leveraging highly trained models. The authentication and provenance-based approach from Stokes et al. [27] (VAMP) protects against poisoning by digitally signing training data to verify where it comes from and whether it has been tampered with. VAMP's prevention strategy is conceptually similar to layer one of our proposed frameworks, but like the majority of related work VAMP does not inspect the content of authenticated data. This means that they will not detect a registered participant uploading GAN-generated poisoned updates. The blockchained FL system proposed by Kim et al. [28] offers decentralized aggregation with latency analysis, but also does not feature poisoning detection. The reasons why these two were left out of Table 5 is that they do not evaluate accuracy and detection metrics under a similar poisoning scenario.

On the other hand, our approach adopts the defense-in-depth strategy by employing layered security verifications, blockchain-supported consensus scheme, and deviation scores for anomaly detection. Our system proves more resilient to adversarial attacks. Also, our approach achieves

higher accuracy with lower false positive rate without compromising time efficiency. Please refer to Table 5. Note: The numbers in table shows the comparison between our approach and other competing methods. The sources of the values for competing methods are listed in the note below.

Table 5 shows that our proposed framework outperforms the reported accuracy and detection rate while yielding lower false positive rates than competing methods. Lower false positive rate minimizes the chance of rejecting genuine client updates, and the adaptive threshold enables the system to track the variations in adversarial behaviors while maintaining moderate overhead. However, since competing methods' results are drawn from different experimental setups, the comparison should be taken as a reference instead of rigorous head-to-head.

Specifically, we design a robust and privacy-preserving defense based on contributions of blockchain-verified tamper-proof auditing, Mahalanobis distance-based anomaly detection for filtering malicious updates, and reputation-aware trust weighted consensus mechanism to give higher weights to trustworthy clients. By contrast to prior work that uses a fixed detection threshold, our method is fully dynamic, automatically adapting to new conditions.

Note: The values of FoolsGold [23], SAFElearn [25], FLGUARD [26] and GAN-resistant FL [22] were compiled from published reports on these methods and were not reproduced by the authors under identical settings; datasets, numbers of clients and attack configurations therefore differ between rows. The Overhead and Adaptive Defense columns are qualitative assessments based on the design of each method.

Table 5. Comparative analysis of adversarial mitigation techniques in federated learning

Method	Accuracy	Detection Rate	False Positive Rate	Overhead	Adaptive Defense
FoolsGold [23]	85.6%	92.1%	5.4%	Low	No
SAFElearn [25]	88.4%	94.5%	4.7%	High	No
FLGUARD [26]	90.1%	95.2%	3.8%	Medium	No
GAN-resistant FL [22]	91.5%	96.1%	3.5%	Medium	Partially
Proposed Model	98.7%	97.8%	2.1%	Medium-Low	Yes

7. Conclusions

This study proposed a decentralized, multi-layered security framework for federated learning that mitigates GAN-based poisoning attacks by combining cryptographic validation, adaptive anomaly detection, reputation-weighted aggregation and blockchain-backed consensus. The first layer authenticates each client and verifies the integrity and well-formedness of its update using RSA signatures, Pedersen commitments and zk-SNARK proofs, and then fuses the KL divergence, autoencoder reconstruction error and Mahalanobis distance into an anomaly score with an adaptive threshold. The second layer selects a validator committee by Proof-of-Stake, reaches Byzantine fault-tolerant agreement on the accepted updates, aggregates them with reputation-based weights and records the outcomes on a permissioned ledger.

The experimental evaluation on MNIST, CIFAR-10, CIFAR-100 and an IoT healthcare security dataset showed that the framework improves the accuracy of the global model relative to the baseline, reaching 98.7% on MNIST, 91.2% on CIFAR-10, 82.1% on CIFAR-100 and 94.3% on the IoT Healthcare dataset, with a detection rate of 97.8% and a false positive rate of 2.1% against GAN-based attacks. These values compare favorably with those reported for FoolsGold, SAFElearn, FLGUARD and GAN-resistant FL, although this comparison is indicative because the competing results were obtained under different settings.

From a practical deployment perspective, the security layers increased the time per communication round by 12–27%, and the on-chain storage per client is constant and independent of the model size. The framework is therefore suited to consortium settings with a moderate number of institutional participants, such as hospitals, IoT gateways or edge servers in 6G networks, where a permissioned blockchain can be operated by the participating organizations and

gas has no monetary cost. Deployment on a public blockchain, or with very large numbers of resource-constrained devices, would require lighter cryptographic operations and a ledger with higher throughput.

The study has several limitations. The reported results come from single runs with 20 simulated clients on one workstation, without confidence intervals or statistical significance tests. Robustness under different poisoning ratios and fractions of malicious clients, ablation of individual components, and blockchain metrics such as transaction latency, throughput, block generation time, storage growth and energy consumption were not measured. Moreover, backdoor, Sybil, free-rider, inference and model replacement attacks were not evaluated experimentally, and the competing defenses were not re-implemented under identical settings.

Future work will address these limitations in four directions. First, the evaluation will be extended with repeated runs reporting mean $\pm$ standard deviation and statistical tests, ablation and sensitivity studies, and direct comparisons with FedAvg, Krum, Multi-Krum, Trimmed Mean, Bulyan, FLTrust, RFA and FedProx under identical settings. Second, scalability experiments with 100 and 1,000 clients will measure ledger latency, throughput, storage and energy cost, and layer-2 rollups, sharding and asynchronous consensus will be investigated to reduce them. Third, lightweight proof systems and hardware acceleration of proof generation will be explored to reduce the client-side cryptographic cost. Finally, the framework will be combined with differential privacy and evaluated against a broader range of attacks, including backdoor, Sybil, free-rider and inference attacks, in real multi-institution healthcare deployments.

Acknowledgement

The authors acknowledge that this study is supported by 1) Department of Reconstruction and Projects, University of Baghdad, Baghdad, Iraq, 2) Department of AI, College of Engineering, University of Zakho, Zakho, Iraq, and 3) Department of Administrative and Financial Affairs, University of Telafer, Nineveh, Iraq.

References

- [1] Sheraz M, Chuah TC, Lee YL, Alam MM, Al-Habashna A, Han Z. A comprehensive survey on revolutionizing connectivity through artificial intelligence-enabled digital twin network in 6G. *IEEE Access*. 2024;12. <https://doi.org/10.1109/ACCESS.2024.3384272>
- [2] Chen C, Liu J, Tan H, Li X, Wang KIK, Li P, et al. Trustworthy federated learning: Privacy, security, and beyond. *Knowledge and Information Systems*. 2025;67(3):2321-2356. <https://doi.org/10.1007/s10115-024-02285-2>
- [3] Zhang C, Yu S, Tian Z, Yu JJ. Generative adversarial networks: A survey on attack and defense perspective. *ACM Computing Surveys*. 2023;56(4):1-35. <https://doi.org/10.1145/3615336>
- [4] Psychogios K, Velivassaki TH, Bourou S, Voulkidis A, Skias D, Zahariadis T. Gan-driven data poisoning attacks and their mitigation in federated learning systems. *Electronics*. 2023;12(8):1805. <https://doi.org/10.3390/electronics12081805>
- [5] Hussain A, Aslam A, Tripura S, Dhanawat V, Shinde V. Weather forecasting using machine learning techniques: Rainfall and temperature analysis. *Preprints*. 2024. <https://doi.org/10.20944/preprints202402.1566.v2>
- [6] Zheng Y, Chang CH, Huang SH, Chen PY, Picek S. An overview of trustworthy AI: Advances in IP protection, privacy-preserving federated learning, security verification, and GAI safety alignment. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*. 2024. <https://doi.org/10.1109/JETCAS.2024.3477348>
- [7] Khan MF, Abaoud M. Blockchain-integrated security for real-time patient monitoring in the internet of medical things using federated learning. *IEEE Access*. 2023;11:117826-117850. <https://doi.org/10.1109/ACCESS.2023.3326155>
- [8] Manzoor HU, Shabbir A, Chen A, Flynn D, Zoha A. A survey of security strategies in federated learning: Defending models, data, and privacy. *Future Internet*. 2024;16(10):374. <https://doi.org/10.3390/fi16100374>
- [9] Zhang Y, Zeng D, Luo J, Fu X, Chen G, Xu Z, et al. A survey of trustworthy federated learning: Issues, solutions, and challenges. *ACM Transactions on Intelligent Systems and Technology*. 2024;15(6):1-47. <https://doi.org/10.1145/3678181>

- [10] Martins I, Resende JS, Sousa PR, Silva S, Antunes L, Gama J. Host-based ids: A review and open issues of an anomaly detection system in iot. *Future Generation Computer Systems*. 2022;133:95-113. <https://doi.org/10.1016/j.future.2022.03.001>
- [11] Li X, Hu Y, Zeng L, An Y, Yang J, Xiao X. Blockchain-based federated learning approaches in internet of things applications. *Security and Privacy*. 2024;7(6):e435. <https://doi.org/10.1002/spy2.435>
- [12] Ness S, Eswarakrishnan V, Sridharan H, Shinde V, Venkata Prasad Janapareddy N, Dhanawat V. Anomaly detection in network traffic using advanced machine learning techniques. *IEEE Access*. 2025;13:16133-16149. <https://doi.org/10.1109/ACCESS.2025.3526988>
- [13] Liu J, Chen C, Li Y, Sun L, Song Y, Zhou J, et al. Enhancing trust and privacy in distributed networks: a comprehensive survey on blockchain-based federated learning. *Knowledge and Information Systems*. 2024;66(8):4377-4403. <https://doi.org/10.1007/s10115-024-02117-3>
- [14] Lashkari B, Musilek P. A comprehensive review of blockchain consensus mechanisms. *IEEE Access*. 2021;9:43620-43652. <https://doi.org/10.1109/ACCESS.2021.3065880>
- [15] Bouzinis PS, Diamantoulakis PD, Karagiannidis GK. Wireless federated learning (WFL) for 6G networks- Part I: Research challenges and future trends. *IEEE Communications Letters*. 2022;26(1):3-7. <https://doi.org/10.1109/LCOMM.2021.3121071>
- [16] Liu Y, Yuan X, Xiong Z, Kang J, Wang X, Niyato D. Federated learning for 6G communications: Challenges, methods, and future directions. *China Communications*. 2020;17(9):105-118. <https://doi.org/10.23919/ICC.2020.09.009>
- [17] Yang Z, Chen M, Saad W, Shikh-Bahaei M, Poor HV, Cui S. Federated learning in 6G mobile wireless networks. In: *Computer Communications and Networks*. Cham, Switzerland: Springer; 2021. p. 359-378. https://doi.org/10.1007/978-3-030-72777-2_16
- [18] Saraswat D, Verma A, Bhattacharya P, Tanwar S, Sharma G, Bokoro PN, et al. Blockchain-based federated learning in uavs beyond 5g networks: A solution taxonomy and future directions. *IEEE Access*. 2022;10:33154-33182. <https://doi.org/10.1109/ACCESS.2022.3161132>
- [19] Yang Z, Chen M, Wong KK, Poor HV, Cui S. Federated learning for 6G: Applications, challenges, and opportunities. *Engineering*. 2022;8:33-41. <https://doi.org/10.1016/j.eng.2021.12.002>
- [20] Imteaj A, Amini MH. FedPARL: Client activity and resource-oriented lightweight federated learning model for resource-constrained heterogeneous IoT environment. *Frontiers in Communications and Networks*. 2021;2:657653. <https://doi.org/10.3389/frcmn.2021.657653>
- [21] Tolpegin V, Truex S, Gursoy ME, Liu L. Data poisoning attacks against federated learning systems. In: *Computer Security - ESORICS 2020, LNCS, vol. 12308*. Cham, Switzerland: Springer; 2020. p. 480-501. https://doi.org/10.1007/978-3-030-58951-6_24
- [22] Zhang X, Luo X. Exploiting defenses against gan based feature inference attacks in federated learning. *arXiv preprint arXiv:2004.12571*. 2020.
- [23] Fung C, Yoon CJ, Beschastnikh I. Mitigating sybils in federated learning poisoning. *arXiv preprint arXiv:1808.04866*. 2018.
- [24] Fang M, Cao X, Jia J, Gong N. Local model poisoning attacks to Byzantine-Robust federated learning. In: *29th USENIX Security Symposium (USENIX Security 20)*; 2020. p. 1605-1622.
- [25] Fereidooni H, Marchal S, Miettinen M, Mirhoseini A, Möllering H, Nguyen TD, et al. SAFELearn: Secure aggregation for private federated learning. In: *Proc. 2021 IEEE Security and Privacy Workshops (SPW)*; 2021. p. 56-62. <https://doi.org/10.1109/SPW53761.2021.00017>
- [26] Nguyen TD, Rieger P, Yalame H, Möllering H, Fereidooni H, Marchal S, et al. FLGUARD: Secure and private federated learning. *arXiv preprint arXiv:2101.02281*. 2021.
- [27] Stokes JW, England P, Kane K. Preventing machine learning poisoning attacks using authentication and provenance. In: *Proc. MILCOM 2021 - IEEE Military Communications Conference*; 2021. p. 181-188. <https://doi.org/10.1109/MILCOM52596.2021.9653139>
- [28] Kim H, Park J, Bennis M, Kim SL. Blockchain on-device federated learning. *IEEE Communications Letters*. 2020;24(6):1279-1283. <https://doi.org/10.1109/LCOMM.2019.2921755>
- [29] Li Y, Chen C, Liu N, Huang H, Zheng Z, Yan Q. A blockchain-based decentralized federated learning framework with committee consensus. *IEEE Network*. 2021;35(1):234-241. <https://doi.org/10.1109/MNET.011.2000263>
- [30] McMahan HB, Moore E, Ramage D, Hampson S, Agüera y Arcas B. Communication-efficient learning of deep networks from decentralized data. In: *Proc. 20th Int. Conf. Artificial Intelligence and Statistics (AISTATS)*, PMLR, vol. 54; 2017. p. 1273-1282.
- [31] Blanchard P, El Mhamdi EM, Guerraoui R, Stainer J. Machine learning with adversaries: Byzantine tolerant gradient descent. In: *Advances in Neural Information Processing Systems 30 (NIPS 2017)*; 2017. p. 119-129.
- [32] Yin D, Chen Y, Ramchandran K, Bartlett P. Byzantine-robust distributed learning: Towards optimal statistical rates. In: *Proc. 35th Int. Conf. Machine Learning (ICML)*, PMLR, vol. 80; 2018. p. 5650-5659.

- [33] El Mhamdi EM, Guerraoui R, Rouault S. The hidden vulnerability of distributed learning in Byzantium. In: Proc. 35th Int. Conf. Machine Learning (ICML), PMLR, vol. 80; 2018. p. 3521-3530.
- [34] Cao X, Fang M, Liu J, Gong NZ. FLTrust: Byzantine-robust federated learning via trust bootstrapping. In: Proc. Network and Distributed System Security Symposium (NDSS); 2021. <https://doi.org/10.14722/ndss.2021.24434>
- [35] Pillutla K, Kakade SM, Harchaoui Z. Robust aggregation for federated learning. IEEE Transactions on Signal Processing. 2022;70:1142-1154. <https://doi.org/10.1109/TSP.2022.3153135>
- [36] Li T, Sahu AK, Zaheer M, Sanjabi M, Talwalkar A, Smith V. Federated optimization in heterogeneous networks. In: Proc. Machine Learning and Systems (MLSys), vol. 2; 2020. p. 429-450.
- [37] Hitaj B, Ateniese G, Perez-Cruz F. Deep models under the GAN: Information leakage from collaborative deep learning. In: Proc. 2017 ACM SIGSAC Conf. Computer and Communications Security (CCS); 2017. p. 603-618. <https://doi.org/10.1145/3133956.3134012>
- [38] Zhang J, Chen B, Cheng X, Binh HTT, Yu S. PoisonGAN: Generative poisoning attacks against federated learning in edge computing systems. IEEE Internet of Things Journal. 2021;8(5):3310-3322. <https://doi.org/10.1109/IIOT.2020.3023126>
- [39] Bagdasaryan E, Veit A, Hua Y, Estrin D, Shmatikov V. How to backdoor federated learning. In: Proc. 23rd Int. Conf. Artificial Intelligence and Statistics (AISTATS), PMLR, vol. 108; 2020. p. 2938-2948.
- [40] Moriarty K, Kaliski B, Jonsson J, Rusch A. PKCS #1: RSA cryptography specifications version 2.2. RFC 8017, Internet Engineering Task Force. 2016. <https://doi.org/10.17487/RFC8017>
- [41] Pedersen TP. Non-interactive and information-theoretic secure verifiable secret sharing. In: Advances in Cryptology - CRYPTO '91, LNCS, vol. 576. Berlin, Germany: Springer; 1992. p. 129-140. https://doi.org/10.1007/3-540-46766-1_9
- [42] Groth J. On the size of pairing-based non-interactive arguments. In: Advances in Cryptology - EUROCRYPT 2016, LNCS, vol. 9666. Berlin, Germany: Springer; 2016. p. 305-326. https://doi.org/10.1007/978-3-662-49896-5_11
- [43] Castro M, Liskov B. Practical Byzantine fault tolerance. In: Proc. 3rd USENIX Symp. Operating Systems Design and Implementation (OSDI); 1999. p. 173-186.
- [44] Li X, Huang K, Yang W, Wang S, Zhang Z. On the convergence of FedAvg on non-IID data. In: Proc. 8th Int. Conf. Learning Representations (ICLR); 2020.
- [45] Hsu TMH, Qi H, Brown M. Measuring the effects of non-identical data distribution for federated visual classification. arXiv preprint arXiv:1909.06335. 2019.
- [46] Hady AA, Ghubaish A, Salman T, Unal D, Jain R. Intrusion detection system for healthcare systems using medical and network data: A comparison study. IEEE Access. 2020;8:106576-106584. <https://doi.org/10.1109/ACCESS.2020.3000421>