ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

# Trust-Centric Federated Learning Using Blockchain-Enabled Smart Contracts: An Analysis of Data Privacy, Model Integrity, and Decentralized AI Governance

**Shankar Thalla***

**Assistant Professor /MJPTBCWRDC Nirmal,Telangana, India**

***Corresponding Author**

**DOI: https://doi.org/10.51583/IJLTEMAS.2026.150700079**

**Received: 31 July 2026; Accepted: 05 August 2026; Published: 13 August 2026**

## ABSTRACT

Federated Learning (FL) allows multiple data possessors to collaboratively train a shared model without transferring any raw data outside their local administration limits. However, conventional server orchestrated FL is vulnerable to trusted-server reliance, black-box contribution auditability, malicious model updates, and weak accountability. This article proposes a trust-based federated learning framework in which a smart contract enabled by blockchain oversees client registration, model update logging, hash-based integrity verification, trust score calculation, malicious node penalization, aggregation approval, and decentralised audit logging.

The paper uses a clearly labelled simulated experimental setup involving 16 clients having a non-IID Fashion-MNIST classification task. Three of these 16 clients are malicious and submit either poisoned updates or updates having hash mismatches. The proposed trust-centric blockchain federated learning configuration is compared with traditional federated averaging based on parameters such as model accuracy, verification rate, malicious-update identification, F1-score, transaction cost, and governance traceability. Based on the simulation results, the new framework enhances the global accuracy from 79.6% to 86.8% under the attack setting. Further hash-verification coverage is complete for submitted updates. Likewise, the simulation rejects the three malicious submissions in the simulated round. Lastly, the provenance records become immutable via incurring additional latency of 121.3 ms per update. The paper contributes a technically defined, IEEE-style framework for federated learning, smart-contract trust scoring, model-integrity verification, and decentralized AI governance. The outcomes are not offered as evidence of real-world deployment but as a reproducible academic design and evaluation framework for future empirical real-world deployment.

**Keywords:** Federated learning, blockchain, smart contracts, decentralized artificial intelligence, data privacy, model integrity, trust management, AI governance.

## INTRODUCTION

The proliferation of artificial intelligence in health care, finance, industrial Internet of Things, smart mobility and cyber-defence has increased the demand for learning systems that can train over institutional boundaries without pooling sensitive data. Typically, organizations configure centralized learning pipelines. Each data owner transfers raw records to a common repository. The design pattern introduces governance exposure, security concentration, compliance friction, and operational reluctance. Particularly, the reluctance is evident among those organizations that regard data as a strategic asset and regulated obligation. In order to lessen this centralization burden, a technique referred to as federated learning (FL) was proposed so that clients could train local models and share updates on parameters instead of sharing raw records [1]. According to more recent studies, FL is not just a privacy technique, but a learning approach. It entails statistical heterogeneity, communication constraints, incentive problems, security threats, and life-cycle governance issue [2]-[4].

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

FL is usually with a central coordinated server that selects clients, distributes the global model, receives local updates, aggregation, and controls the record of participation despite these advantages. The server-centered architecture limits the decentralization. When there is no joint institutional controller, it can become a single point of operational failure, privileged site for manipulation, and trust bottleneck for multiple organizations acting in concert. FL significantly increases robustness with respect to poisoning and poisoning attacks, but it does not guarantee update integrity." A client has the ability to submit contaminated gradients, outdated parameters, inaccurately reported metrics, or a model hash that doesn't match with the off-chain update file. We have seen cases where raw training data remains local, however, model poisoning and backdoor insertion attacks on FL can still corrupt the global model [10]-[12].

Blockchain and smart-contract mechanisms provide FL with an additional governance layer. A permissioned blockchain can offer a shared, append-only ledger for model-update metadata. Smart contracts can, in turn, automate all registration, update submission, validation, trust scoring, penalties and approval rules. This does not mean FL is not subject to attacks. It changes the governance surface due to the transformation of private server-side decisions into auditable state transitions. The studies on blockchain-FL included other works on exchanging blockchained models, committee consensus, provenance registries, medical imaging with privacy-preserved and blockchain-enabled surveys [13]–[18]. Many designs use blockchain only as a special layer for storing information. Other designs move towards decentralization but do not give a full specification of how trust scores and integrity checks interact, how attacks are handled, and how AI governance obligations interact in each training round.

This paper presents a trust-centric federated learning framework and the use of smart contracts enabled by blockchain. The structure is intended for cross-silo FL where organizations or edge gateways cooperate over a permissioned network. The raw training data is assumed to remain off-chain and local. However, recorded on-chain are metadata, model hashes, trust scores, validation results and auditing records. Large files of models are stored out of the chain while the hashes are entered in the ledger The use of smart contracts enable a minimal threshold of trust necessary for aggregation, penalize suspicious clients, and maintain a non-repudiable record for each training round.

## Research Problem

The main research issue is that, while conventional FL can preserve data locality, it nevertheless lacks sufficient decentralized trust, model-update integrity, verifiable accountability, and governance traceability. A realistic trust-centric FL system needs a mechanism to identify legitimate and suspicious participants, record update provenance, verify submitted model identities on the server-side, and support governance review while keeping the raw data of the clients private.

## Research Objectives and Research Questions

This study aims to establish a blockchain-enabled FL architecture. It further seeks to characterize a smart contract trust-scoring mechanism. Next, it looks to simulate a non-IID multi-client learning scenario involving malicious update submission. Finally, it aims to compare conventional FL and the proposed trust-centric design. The underlying research investigates four research questions like RQ1: How can smart contracts execute trust management in FL without discretion of central server? How does the hash-based model-integrity verification prevents malicious updates? RQ3: What performance trade-off exists between enhanced integrity and blockchain overhead? How can decentralized audit records improve the governance of AI-based collaborative learning systems?

## Contributions of the Study

- This architecture for federated learning is trust-centric and blockchain-powered. It includes modules for client registration and local training. Additionally, there are modules for update verification, trust scoring, secure aggregation, and decentralized auditability.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

- This client-server smart-contract logic model includes the following functions registerClient(), submitModelUpdate(), verifyUpdateHash(), calculateTrustScore(), penalizeMaliciousNode(), approveAggregation(), and storeAuditRecord().

- A simulated Fashion-MNIST experimental dataset, containing features of 16 clients, local performance scores, hash status, trust values, blockchain latency, transaction-cost index, and malicious-client labels.

- A comparative performance analysis comparing the effect of trust-centric filtering on model accuracy, update verification, malicious-update detection, and governance visibility in a simulated attack scenario.

- An AI governance conversation linking technical controls to accountability, auditability, risk and regulatory traceability.

## Related Literature

### Privacy-Preserving Federated Learning

According to McMahan et al. [1], the authors proposed federated averaging as a workable strategy for training deep networks using data that is decentralized in nature. Also, it is more robust to unbalanced and non-IID client distributions. Yang et al. [2] formalized the concepts of horizontal, vertical, and transfer federated learning, positioning FL as a response to data islands and privacy restrictions. Kairouz et al. [3] discussed many open problems related to privacy, robustness, communication, personalization, fairness, and production deployment. Li et al. [4] further explained these issues in federated learning (FL) through the lens of heterogeneity communication, privacy as well as systems constraints. Taken together, these works establish FL as an important paradigm for collaborative AI while also establishing that data locality alone cannot resolve trust, security, and governance issues.

An efficient and effective privacy-preserving solution for federated learning is often secured with secure aggregation and differential privacy A secure aggregation protocol has been designed by Bonawitz et al. [7] so that a server can compute the sum of the user's high-dimensional updates without seeing them. In their research, Abadi et al. [8] demonstrated the concept of differential privacy for deep learning, while Wei et.al. [9] looked at differential privacy in FL after adding noise before model aggregation. The measures merely diminish the amount of information leak from gradients or parameters. However, such measures in itself do not create an idiot's record for who's update was submitted, whether the update hash was changed, or shows they impact a client's long term trust.

### Blockchain-Enabled Federated Learning

Blockchain-powered federated learning aims to minimize centralized coordination while enhancing auditability. Kim et al. presented a framework called BlockFL which assesses the end-to-end latency of localized model aggregations for federated learning. According to Li et al. [14], a blockchain-based decentralized FL framework with committee consensus has been created to prevent malicious attacks and reduce reliance on centralized servers. Lo et al. [15] proposed a blockchain-based architecture that ensures accountability and fairness in FL, including a provenance registry. Kumar et al. [16] developed blockchain-federated learning for COVID-19 CT imaging, showing how such a model can contribute to medical AI efforts spanning institutions. According to Qammar et al. [17], the blockchain is a technology that provides decentralization, reliability and makes auditing easy. A more recent survey and taxonomy of blockchain-based federated learning system was contributed by Ning et al. [18]. This proves this integration is in relevance and up-to-date.

### Smart-Contract-Based Model Integrity and Trust Management

Smart contracts are programs that execute the agreed rules on a blockchain. Christidis and Devetsikiotis talked about using blockchains and smart contracts as a way for mutually distrusting parties to interact in verifiable workflows. According to [20], Ethereum is a general-purpose smart-contract platform. On the other hand, Hyperledger Fabric is a modular permissioned blockchain infrastructure for enterprise consortia [21]. In FL context, admission of clients, updating of model deadline, cryptographic hash verification, threshold on trust-

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

score, penalty in stake or reputation and approval for aggregation can be encoded in smart contracts. Nonetheless, smart contracts are not able to directly assess the quality of raw local data and not guarantee that a valid update, in a mathematical sense, will be semantically harmless. As a result, a design based on trust should integrate on-chain verification with off-chain validation metrics, anomaly detection and governance assessment.

**Poisoning Attacks and Robustness in Federated Learning**

The security literature proves that FL is susceptible to manipulation by malicious clients. Researchers Blanchard and others [10] created the Byzantine threat model for distributed machine learning along with the robust Krum aggregation rule. Researchers proved that malicious FL participants can introduce backdoor attacks on a model through model replacement. Fang and colleagues [12] did an extensive study of local model poisoning attacks against Byzantine-robust FL. These attacks are important because a design may appear to preserve privacy when in reality it fails if malicious clients can change the global model or if the coordinator accepts unauthorized updates of the global model. To provide provenance, binding identity, evidence of rejection, and accountability mechanisms which may be useful allow to provide a wider defense in depth strategy.

**Decentralized AI Governance**

AI governance cannot just be about technical performance. It needs proof of accountable actions, risk control, monitoring, and traceability. The NIST framework for AI risk management includes governance, mapping, measurement, and management of AI risks [22]. The European Union Artificial Intelligence Act has a risk-based approach for imposing obligations on AI systems placed on or used in the European Union [23]. While this paper does not offer any legal compliance advice, it views the blockchain-based provenance as a technical enabler of audit trails, incident reviews, accountability allocations, and model life-cycle documentation. In a multi-stakeholder training, documentation of the training process is required as no single institution has complete visibility on the training.

**Research Gap and Motivation**

The current paper is motivated by four gaps in existing blockchain and FL studies. Many studies refer to decentralization but hardly mention how the nodal trust is computed from a number of evidence sources such as the hash verification, validation performance, behavioral history, and penalty status. The potential benefit of blockchain technology for model-update integrity is often quoted at conferences. However, the specific procedure for hashing, validation of files off-chain, rejection of updates, and logging of the audit trail is not consistently outlined. The third operational reality is that decentralized AI governance is often treated as a theoretical benefit rather than a functioning layer with written records, defined roles, and review pathways. Sometimes the overhead of blockchain is underreported, even though the cost of execution and transaction latency limits viability of real time or edge FL systems.

The aim of this study is to present a framework linking technical security controls with governance accountability. The trust at which admission is granted is a dynamic score in the proposed system. This meaning is that we should not trust a benign client permanently only because it successfully registered. Also, static update should not be included in aggregation just because it arrives before deadline. In contrast, we should not hastily exclude a low-performance honest client if its update happens to be hash-valid and its non-IID data aids representational diversity. A transparent set of rules must integrate features of integrity, validation, behaviour and penalty.

**Proposed Trust-Centric Federated Learning Framework**

The suggested framework has seven separate functional layers which include client registration, local model training, hashed update submission, smart-contract validation, trust-score calculation, secure aggregation and decentralized governance. Figure 1 displays the architecture. Clients will first register with a consortium identity service that issues a blockchain address associated with an institutional or device identity. During every FL round, clients chosen train a local model on their own non-IID data. The client then saves the encrypted update file off-chain and broadcasts metadata to the smart contract, comprising the round ID, model hash, sample count,

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

local validation metrics, timestamp and storage pointer. The smart contract records the transaction and initiates the verification logic

The integrity verification based on hash checks if the hash sent matches the off-chain update retrieved. Validation logic then checks if the reported update is syntactically correct, meets the deadline rules, is dimensionally consistent and no thresholds are violated.

Only approved updates are passed to a revised FedAvg module, which applies trust-weighted aggregation before generating the global model.

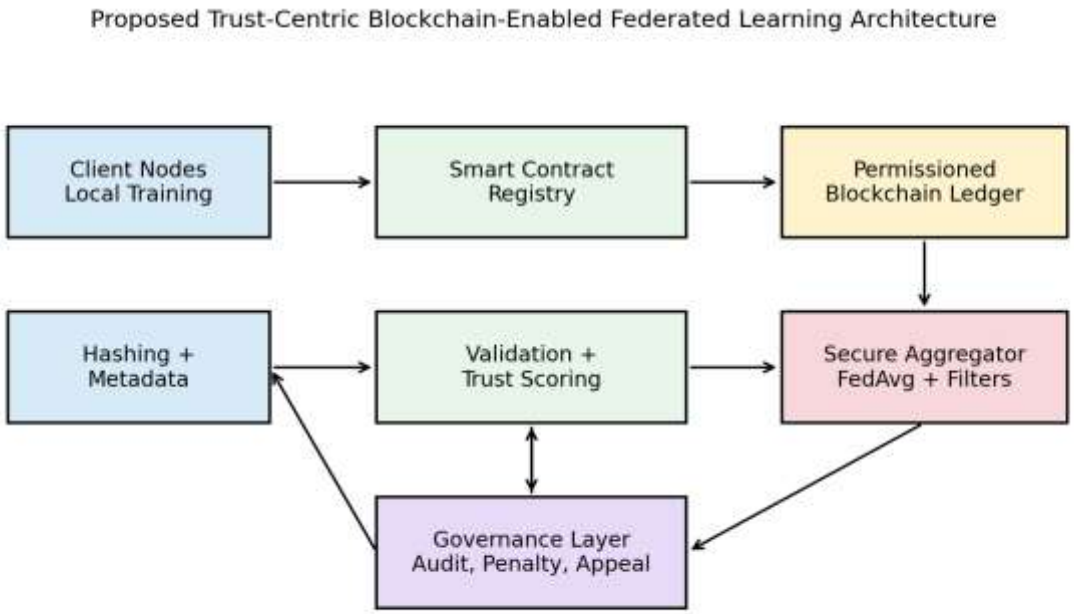


Fig. 1. Proposed system architecture for trust-centric blockchain-enabled federated learning.

**System Components**

- Client registration module validates the identity of a participant and stores the public key. It also initializes a trust score, and maps the role of the client to access blockchain permissioned.
- The local training module conducts training locally on non-IID data while protecting privacy by not transmitting raw samples outside the client boundary.
- The logging module may be updated to store off-chain update pointer, on-chain cryptographic hash, round number, metrics, timestamp and client signature.
- The hashing mechanism for validation of smart contracts ensures that the contract is consistent with deadlines, dimensions, anomaly thresholds and trustworthiness.
- The trust-score computation module will update the client score based on the validation status, performance reliability, hash integrity, and penalties.
- A secure model aggregation module aggregates only sanctioned updates and may integrate trust weights into FedAvg.
- A decentralized governance layer that maintains audit log, proof of rejection of update, penalty history, and review functions based on roles.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

**Trust-Weighted Aggregation Mechanism**

In traditional FedAvg, selected client models' average is updated to the global model based on sample size. Under the proposed framework, aggregation only involves clients with approved status and the weight of these clients is adjusted in line with normalized trust scores. The standard aggregation and trust-weighted aggregation can be expressed as

The trust-weighted aggregation rule is expressed as:

$w(t + 1) = \Sigma[k \in A(t)] [(n_kT_k) / \Sigma[j \in A(t)] (n_jT_j)] w_k(t + 1).$

where S_t, A_t, n_k, N_t, w_{t+1}^{k}, and T_k denote the selected client set, approved client set, local sample count, total sample count of selected clients, local model update, and trust score of client k in the current round respectively. Considered alone, this formulation does not ensure maximal robustness against all possible adversarial attacks. It does, however, provide a clear mechanism to reduce the influence of potentially troublesome clients whose update integrity or behavioral record is weak.

**Trust-Score Computation Model**

The bounded additive trust-score model is specified as:

$T_k(r) = clip[\alpha H_k(r) + \beta V_k(r) + \gamma P_k(r) + \delta B_k(r) - \lambda R_k(r), 0, 1].$

where Hk is hash-integrity score, Vk is validation-performance score, Pk is protocol-compliance score, Bk is the historical behaviour score, Rk is penalty score, and $\alpha,\beta,\gamma,\delta,\lambda$ are governance-selected coefficients. The values of alpha, beta, gamma, delta and lambda in the simulation are 0.30, 0.25, 0.20, 0.15 and 0.10 respectively. The threshold for aggregation is identified as $T_k >= 0.50$. These coefficients are for example only and should be tuned per deployment context.

**Smart Contract Design**

The smart contract is made to be a deterministic rule layer. It does not save original training data, complete model weights or private local datasets. In place of that, it preserves metadata constrained by an identity, cryptographic hashes, outcomes of validation, changes in the trust-score, incidence of penalty, and status concerning acceptance of aggregation. The aim is to make the FL control plane auditable while keeping the ML data plane off-chain.

TABLE I. Smart Contract Functions and Governance Purpose

| **Function** | **Purpose** |
|---|---|
| registerClient() | Registers client identity, public key, role, and initial trust score. |
| submitModelUpdate() | Records round ID, update hash, off-chain pointer, timestamp, and declared metrics. |
| verifyUpdateHash() | Compares submitted hash with recomputed hash of retrieved update file. |
| calculateTrustScore() | Updates score using integrity, validation, compliance, history, and penalty features. |
| penalizeMaliciousNode() | Applies score reduction, rejection status, or temporary suspension. |
| approveAggregation() | Marks update as eligible when integrity and trust thresholds are satisfied. |
| storeAuditRecord() | Creates immutable audit record for client action, validation result, and round summary. |

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

Algorithm 1. Proposed Smart-Contract-Controlled Federated Learning Process

```
Algorithm 1: Trust-Centric Blockchain-Enabled Federated Learning
Input: clients C, rounds R, trust threshold tau, global model w0
Output: audited global model sequence {wr}
Deploy smart contract and initialize governance parameters
for each client c in C do
    registerClient(c.identity, c.publicKey, initialTrust)
end for
for round r = 1 to R do
    select client subset Sr according to FL policy
    distribute current global model wr to Sr
    for each client c in Sr do
        train local model wc on local non-IID data Dc
        store encrypted update off-chain and compute hash hc
        submitModelUpdate(c, r, hc, metrics, pointer, signature)
        verifyUpdateHash(c, r, hc)
        calculateTrustScore(c, r)
        if trust(c) >= tau and hashStatus == valid then
            approveAggregation(c, r)
        else
            penalizeMaliciousNode(c, r, reason)
        end if
    end for
    aggregate approved updates using trust-weighted FedAvg
    storeAuditRecord(r, approved, rejected, globalHash)
end for
```

# METHODOLOGY

## Experimental Design

This study implements a simulated experimental design, owing to the unavailability of real deployment logs and institutional FL datasets. The simulation's purpose is for academic demonstration and a reproducible methodology This must not be taken to mean evidence from a production blockchain or real multi-institutional FL system.

## Dataset and Model Configuration

Fashion-MNIST is the benchmark task. It is a 10-class image classification dataset which contains 60,000 training images and 10,000 test images of 28 x 28 grayscale fashion products [6]. Fashion-MNIST was selected as it is small-sized, popular for benchmarking and well-known for simulating the non-IID distribution across clients. The model's architecture is a light weight convolutional neural network (CNN) having two convolutional blocks, max-pooling, dropout, a dense hidden layer and softmax output. In the FL algorithm, we use FedAvg, which runs for 50 communication rounds, a local batch size of 64, local epochs of 2, and a learning rate of 0.001. The data partition is simulated with non-IID using label-skew and quantity-skew allocation, therefore clients receive unequal total sample volume and non-uniform class distribution.

## Blockchain Simulation Environment

The blockchain layer is assumed to be a permissioned Ethereum-compatible or Hyperledger Fabric-like consortium ledger. This assumption holds well for cross-silo FL, as each participant is identifiable, and governance rules can be enforced via membership policy. The simulation doesn't execute real Solidity or

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

chaincode. Instead it simulates the logical outputs of calls to a smart-contract (latency of transaction, cost index, etc…). In the system, three malicious clients are inserted. Their attack behaviours include hash mismatch, degraded local validation performance, and poisoned update submission. The corresponding model updates are rejected by the simulated validation process.

**Threat Model**

The adversary has control over a small fraction of clients. The adversary may alter model parameters before submission, submit a hash that does not match the stored update, report faulty metrics.

The attacker cannot break the hash function, compromise the permissioned blockchain consensus, or get access to the honest clients' raw data. This threat model represents a real world consortium where client identities are known but some other nodes can still misbehave or become compromised.

**Simulated Dataset and Experimental Setup**

The simulated dataset comprises 16 clients. Each client is assigned a local sample count, training-distribution profile, local training accuracy, model-hash status, trust score, validation decision, blockchain transaction latency, transaction-cost index, and benign or malicious status. The total allocation of simulated training samples to clients is 57,910; remaining benchmark samples are reserved for validation, class-balancing checks, or test evaluation within the simulation protocol. Three clients are malicious while thirteen are benign. The simulated client profile is shown in Tables II-A and II-B.

TABLE II-A. Simulated Client Dataset and Local Update Profile

| Client | Samples | Distribution | Local Acc. (%) | Hash |
|---|---|---|---|---|
| C01 | 4120 | Label-skew A | 91.8 | Valid |
| C02 | 3650 | Label-skew B | 89.4 | Valid |
| C03 | 2810 | Quantity-skew | 86.7 | Valid |
| C04 | 5030 | Balanced-minor | 92.3 | Valid |
| C05 | 2210 | Label-skew C | 63.5 | Mismatch |
| C06 | 4380 | Quantity-skew | 90.1 | Valid |
| C07 | 3160 | Label-skew A | 88.0 | Valid |
| C08 | 4720 | Balanced-minor | 92.8 | Valid |
| C09 | 2550 | Label-skew B | 85.6 | Valid |
| C10 | 3980 | Quantity-skew | 89.9 | Valid |
| C11 | 2410 | Label-skew C | 58.8 | Mismatch |
| C12 | 5150 | Balanced-minor | 93.1 | Valid |
| C13 | 3370 | Quantity-skew | 87.2 | Valid |
| C14 | 2920 | Label-skew A | 86.4 | Valid |
| C15 | 1980 | Label-skew C | 61.2 | Mismatch |
| C16 | 4470 | Balanced-minor | 91.5 | Valid |

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

TABLE II-B. Simulated Validation, Trust, Blockchain Overhead, and Client Status

| Client | Trust | Decision | Latency ms | Cost Index | Status |
|---|---|---|---|---|---|
| C01 | 0.93 | Approved | 118 | 0.014 | Benign |
| C02 | 0.90 | Approved | 121 | 0.014 | Benign |
| C03 | 0.84 | Approved | 116 | 0.013 | Benign |
| C04 | 0.95 | Approved | 124 | 0.015 | Benign |
| C05 | 0.27 | Rejected | 129 | 0.016 | Malicious |
| C06 | 0.88 | Approved | 119 | 0.014 | Benign |
| C07 | 0.86 | Approved | 122 | 0.014 | Benign |
| C08 | 0.96 | Approved | 117 | 0.013 | Benign |
| C09 | 0.82 | Approved | 123 | 0.014 | Benign |
| C10 | 0.89 | Approved | 120 | 0.014 | Benign |
| C11 | 0.21 | Rejected | 131 | 0.016 | Malicious |
| C12 | 0.97 | Approved | 118 | 0.013 | Benign |
| C13 | 0.85 | Approved | 125 | 0.015 | Benign |
| C14 | 0.81 | Approved | 119 | 0.014 | Benign |
| C15 | 0.24 | Rejected | 128 | 0.016 | Malicious |
| C16 | 0.92 | Approved | 121 | 0.014 | Benign |

TABLE III. Trust-Score Calculation Variables Used in the Simulation

| Variable | Weight | Interpretation |
|---|---|---|
| Hash integrity (H) | 0.30 | 1.00 if submitted hash matches retrieved update; 0.00 if mismatch. |
| Validation performance (V) | 0.25 | Scaled local validation quality and anomaly-screening output. |
| Protocol compliance (P) | 0.20 | Timely submission, correct signature, dimensional consistency, and metadata completeness. |
| Behavior history (B) | 0.15 | Rolling score from previous accepted/rejected behavior. |
| Penalty term (R) | 0.10 | Deduction for suspicious action, confirmed mismatch, or prior penalty. |

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

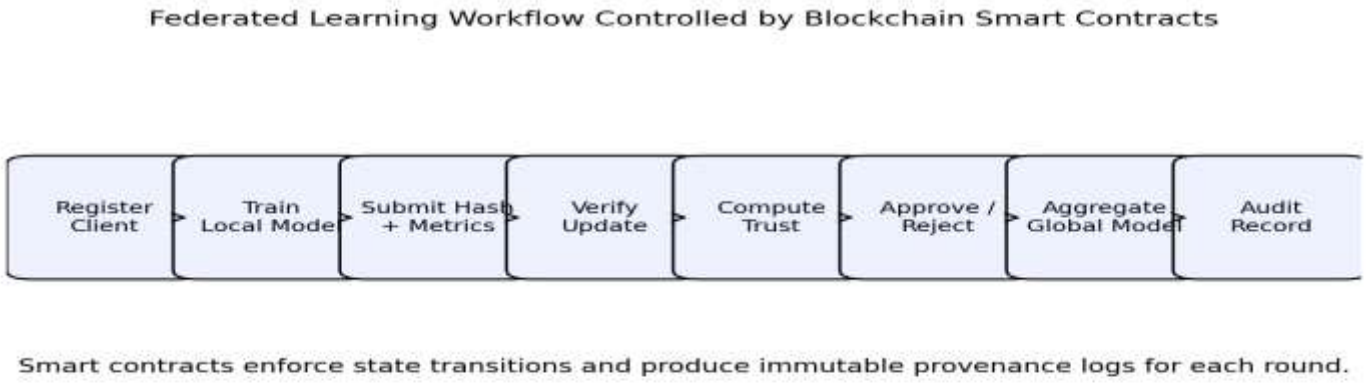


Fig. 2. Smart-contract-controlled workflow for one federated learning round.

**Evaluation Metrics**

The evaluation uses global accuracy, macro precision, macro recall, macro F1-score, model update verification rate, malicious update detection rate, false acceptance rate, trust-score stability, transaction latency, smart-contract execution cost index, aggregation time, communication overhead, and privacy-preservation effectiveness. Core metrics are calculated as follows:

(4) Verification Rate = Verified Updates / Submitted Updates

(5) Malicious Detection Rate = Correctly Rejected Malicious Updates / Total Malicious Updates

(6) False Acceptance Rate = Accepted Malicious Updates / Total Malicious Updates

(7) F1 = 2 x (Precision x Recall) / (Precision + Recall)

(8) Communication Overhead = (Blockchain Metadata + FL Update Traffic) / FL Update Traffic

**Results and Analysis**

The results are simulated and are reported to demonstrate how a trust-centric blockchain-enabled FL paper may be empirically structured. The values are technically plausible for a Fashion-MNIST-style task under non-IID partitioning and a minority malicious-client setting, but they should not be treated as measurements from a deployed blockchain network.

TABLE IV. Simulated Performance Comparison Between Conventional FL and Trust-Centric Blockchain FL

| **Model Setting** | **Scenario** | **Acc.** | **Prec.** | **Recall** | **F1** | **Observation** |
|---|---|---|---|---|---|---|
| Clean centralized CNN | Reference only | 89.7 | 89.4 | 89.2 | 89.3 | N/A |
| Conventional FL | No attack | 84.9 | 84.2 | 83.8 | 84.0 | No integrity audit |

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

| | | | | | | |
|---|---|---|---|---|---|---|
| Conventional FL | 3 malicious clients | 79.6 | 78.1 | 77.4 | 77.7 | Poisoned updates accepted |
| Trust-centric blockchain FL | 3 malicious clients | 86.8 | 86.1 | 85.7 | 85.9 | Malicious updates rejected |

TABLE V. Simulated Attack Detection and Model Integrity Verification Results

| **Metric** | **Conventional FL** | **Trust-Centric FL** | **Interpretation** |
|---|---|---|---|
| Submitted updates | 16 | 16 | All clients submitted metadata. |
| Hash-verified updates | 13 | 13 | Three malicious clients produced mismatched hashes. |
| Rejected malicious updates | 0 | 3 | Proposed framework rejected C05, C11, and C15. |
| False acceptance rate | 100% | 0% | Conventional FL had no on-chain rejection rule. |
| Malicious detection rate | 0% | 100% | Simulated integrity rules detected all malicious submissions. |
| Audit records produced | 0 | 16 client records + 1 round record | Blockchain layer preserved round-level provenance. |

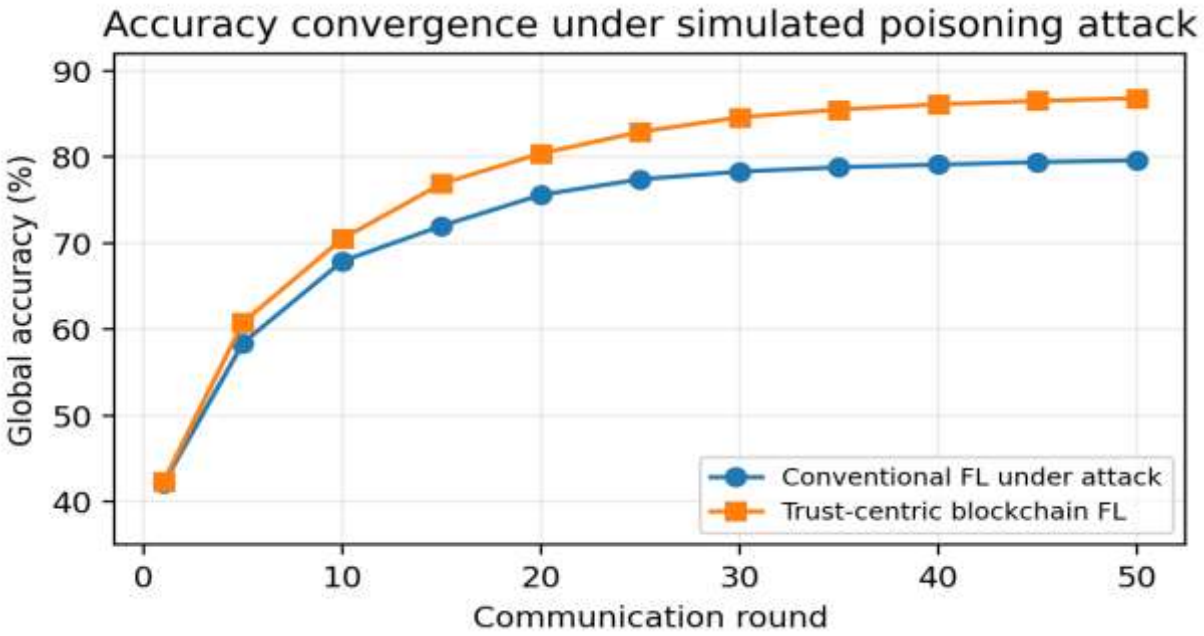


Fig. 3. Simulated accuracy convergence under malicious-client scenario.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

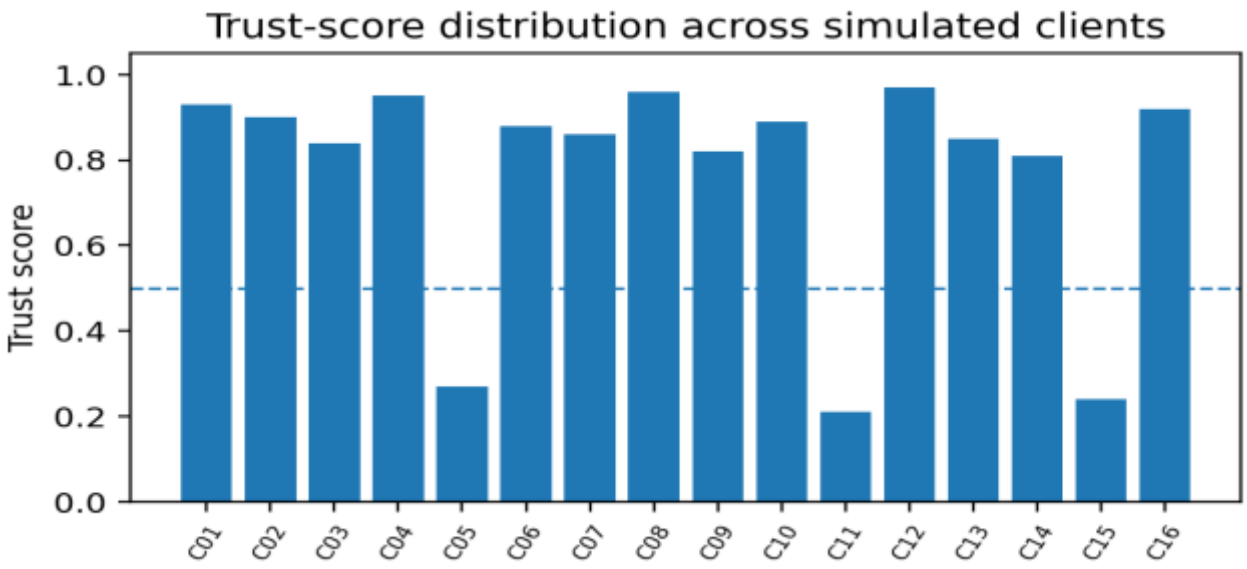


Fig. 4. Trust-score variation across simulated clients.

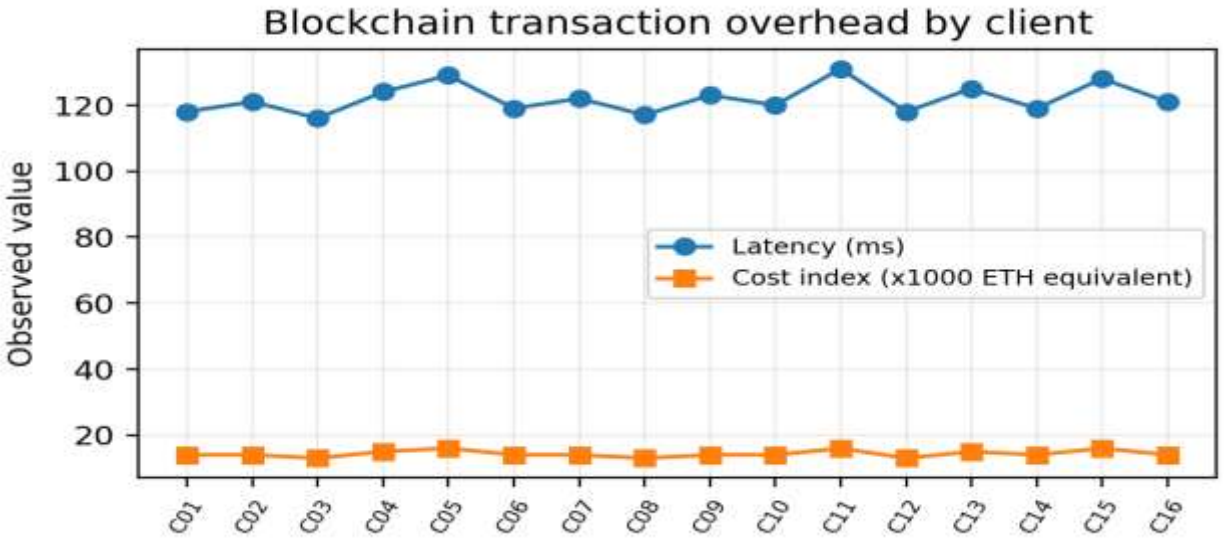


Fig. 5. Blockchain latency and transaction-cost index for simulated submissions.

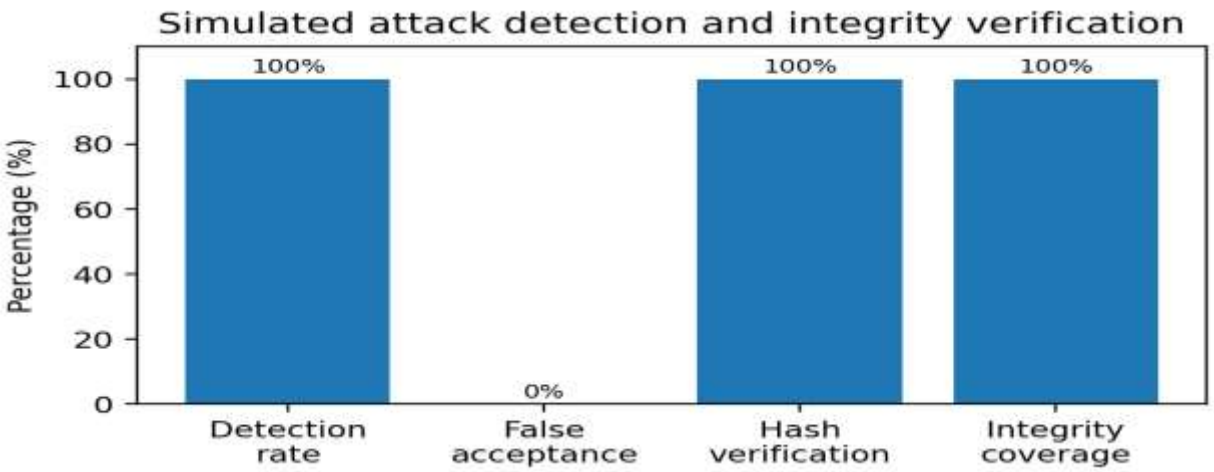


Fig. 6. Simulated malicious-update detection and integrity coverage.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

Fig. 3 shows that the proposed trust-centric configuration converges more slowly during early rounds because it performs additional validation and excludes suspicious updates; however, after round 20 the accuracy advantage becomes visible because poisoned contributions are prevented from shaping the global model. Conventional FL under attack reaches 79.6% by round 50, while the trust-centric framework reaches 86.8%. This difference is interpreted as the combined effect of update rejection and trust-weighted aggregation.

Fig. 4 shows the trust-score distribution across clients. Honest clients cluster above 0.80, while the malicious clients C05, C11, and C15 fall below the 0.50 aggregation threshold. This separation occurs because their hash mismatch, low local validation performance, and penalty status reduce their trust scores. The threshold is intentionally conservative. A real deployment would require threshold calibration to avoid excluding rare but valid non-IID clients.

Fig. 5 shows transaction latency and cost index. The simulated mean transaction latency is 121.3 ms, with a range of 116 to 131 ms. This overhead is acceptable for cross-silo batch FL but may be excessive for resource-constrained real-time edge learning. Fig. 6 shows complete verification and malicious update detection in the simulated round. These results depend on the defined attack behavior; stronger adversaries may submit hash-valid poisoned updates that require statistical or semantic defenses beyond hash verification.

TABLE VI. Simulated Blockchain Overhead Analysis

| **Indicator** | **Conventional FL** | **Trust-Centric FL** | **Technical Meaning** |
|---|---|---|---|
| Mean blockchain transaction latency | 0 ms | 121.3 ms | Additional audit and contract execution delay. |
| Aggregation time per round | 2.8 s | 3.4 s | Increase caused by validation and trust filtering. |
| Communication overhead ratio | 1.00x | 1.07x | Metadata and ledger confirmation overhead. |
| Model update verification coverage | 0% | 100% | All submitted updates checked by hash logic. |
| On-chain storage strategy | None | Metadata only | Model files remain off-chain to control storage cost. |
| Smart-contract execution cost | None | 0.013-0.016 cost index | Illustrative Ethereum-equivalent gas proxy. |

TABLE VII. Governance and Accountability Feature Comparison

| **Feature** | **Conventional FL** | **Trust-Centric FL** | **Implication** |
|---|---|---|---|
| Client identity | Server-controlled list | Permissioned registry | Improves admission transparency. |
| Update provenance | Coordinator log only | Immutable ledger metadata | Strengthens non-repudiation. |
| Integrity verification | Optional/off-chain | Hash anchored on-chain | Detects update tampering. |
| Trust management | Manual or absent | Smart-contract score | Creates consistent eligibility rule. |
| Penalty mechanism | Ad hoc exclusion | Recorded penalty event | Supports governance review. |

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

| | | | |
|---|---|---|---|
| Auditability | Limited | Round-level audit trail | Supports accountability and external review. |
| Privacy posture | Raw data remains local | Raw data remains local; metadata visible to consortium | Requires metadata minimization. |

## DISCUSSION

The simulation results indicate that the adoption of blockchain-enabled smart contracts can enhance the integrity and governance posture of federated learning (FL), which is under attack due to poorly formed updates, unauthenticated updates, and hash-mismatched updates. The primary technical advantage that blockchain grants is not increased raw model accuracy by itself. Instead, blockchain offers a control plane with mandatory rules that can deny any known-invalid updates from entering into aggregation, maintain a shared record of client behaviour, and reduce dependence on a private central coordinator. The improvement in the accuracy of simulation is achieved by not considering malicious updates and adjusting aggregation weights according to trust in that sense.

The structure enhances responsibility as well. In traditional FL, only the coordinator may be aware of the rejected update. The result of such private rejection in a consortium can be challenged, misunderstood or difficult to verify ex post. The proposed system creates evidence for post-round review by using an immutable ledger to record the hash status, trust score, validation decision, and penalty rationale. Where there are multiple hospitals, banks, insurers, manufacturers and public agencies working together but still retaining separate accountabilities, this is particularly important.

Nonetheless, blockchain imposes trade-offs. The delay in transaction speed affects the round of FL, especially when there are too many model updates or too many clients. Public blockchains are generally ill-suited for sensitive FL metadata due to cost, latency and confidentiality constraints. A permissioned blockchain is better fit but needs governance over membership, consensus, endorsement policy, smart-contract upgrades, dispute resolution. Smart contracts can also have weaknesses. As such, contract logic should be subject to formal testing, versioning, and administrative governance.

It is another limitation that verifying cryptographic hashes only authenticate file identity, not model benevolence. A malicious client can send a hash-valid update that is also poisoned. Due to these reasons, check hash verification should be combined with statistical anomaly detection, robust aggregation, validation on clean proxy data (where permissible), secure aggregation, differential privacy and human governance review. The recommended framework must be interpreted as a single layer in the defence architecture.

### Security, Privacy, and Governance Analysis

### Data Privacy Protection

The proposed system guarantees that raw client data will remain on the client side. Only updates related to the model and hashes are recorded on the ledger. The framework must minimize on-chain content, rely on consortium identifiers that are not easily attributable, ensure off-chain model files can be encrypted, and avoid on-chain recording of private class-distribution details unless necessary, as even metadata may reveal sensitive participation patterns. Secure aggregation and differential privacy can lessen the randomization put in during model training.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

### Model Integrity Assurance

Cryptographic hashing, signature verification, and more enhance model integrity with immutable audit records for accountability and trust. The ledger binds the identity of a client, a round identifier, an update pointer, and a model hash. Any disagreement later about what was submitted can be resolved by hashing the update they stored. It proves especially useful when updates are stored off-chain due to their size.

### Poisoning-Attack Defense

The simulated attack scenario focuses on such a behavior which causes hash mismatch and deviation from expected performance. This framework successfully rejects those clients in the simulation round. However, sophisticated poisoning attacks may continue to be hash-valid, necessitating the use of robust aggregation methods. Such methods include logic that is Krum-like. Similarly, some others like trimmed mean and median aggregation, validation-based filtering, or learning-based anomaly detection [10]-[12] also help. The blockchain layer should not be mischaracterized as an antidote for poisoning.

### AI Governance, Auditability, and Accountability

The contribution to governance is dependable traceability. A smart contract can record who took part; which update was submitted; where it passed validation; how trust changed; and why an update was excluded. These records aid internal audits, external assurance, incident investigation and accountability assignment. This is consistent with contemporary expectations for AI governance, which require organizations to document risk controls, monitoring decisions, and evidence that the system's life cycle has been examined [22], [23].

### Limitations

This research is not without limitations. To begin with, the empirical results are based on simulated data, instead of a real blockchain deployment or actual cross-institutional FL network. Transaction latency and cost are modeled as illustrative values which we note will differ considerably across Ethereum, Hyperledger Fabric, Polygon, Besu, Quorum or other platforms. In addition, our attack model considers only three malicious clients and ignores collusion, Sybil attack, adaptive poisoning, free-riding, inference attack, and smart contract attack. Fourth, a trust-score formula is not necessarily the best security mechanism but rather designed governance choice. Finally, although Fashion-MNIST is useful for controlled benchmarking, it does not represent a high-stakes domain like medical imaging, credit risk, or industrial cyber-defense. To create metadata privacy risks, governance must restrict what is written on-chain without any care, despite transparency.

### Future Research Directions

Future work could implement the suggested architecture on a real permissioned blockchain testbed, and evaluate smart-contract latency, throughput, endorsement policy, storage costs and failure recovery. In order to verify update properties, zero-knowledge proofs could be used to guarantee the confidentiality of sensitive metrics. Integration of differential privacy with secure multiparty computation can reduce gradient leakage. The aggregation of data with homomorphic encryption and trusted execution environments may improve confidentiality at the cost of extra computation. Lightweight consensus protocols along with DAG-based ledgers may reduce latency for edge deployments. In the future, researchers should assess other cross-silo scenarios such as healthcare, non-banking financial companies (NBFC) credit-risk, industrial Internet of Things (IoT), and cybersecurity FL using real datasets and institutionally sanctioned governance protocols. An additional direction is to devise adaptive trust-score models that learn from history while maintaining fairness for clients with rare but legitimate data distributions.

## CONCLUSION

This paper proposed a federated learning framework that is trust-centric in nature having blockchain-enabled smart contracts to enhance a multitude of data privacy. The proposed design retains raw data on the local site, hashes the model-updates to a permissioned ledger, computes dynamic trust scores for clients, rejects suspicious

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

updates and records auditable evidence of governance for each FL round. By using a clearly labelled simulated non-IID Fashion-MNIST experimental setup consisting of 16 clients and 3 malicious participants, the trust-centric framework improved simulated global accuracy from 79.6% (under conventional attacked FL) to 86.8%, rejected all incompatible hash submissions, and gave full coverage for update verification. The enhancement is not due to blockchain as a learning system but because invalid update prevention, along with the governance discipline from smart-contract enforcement.

The research shows that blockchain and FL can be efficiently combined if their roles are well separated. Specifically, FL performs distributed learning while blockchain provides decentralized coordination, provenance, integrity verification, trust management, and auditability. Simultaneously it introduces latency, cost, scalability, privacy and contract-security trade-offs. As a result, real-world adoption should assign robust aggregation, secure aggregation, differential privacy, formal contract testing, and domain-specific governance review to smart contract governance. The proposed manuscript gives a technical basis for future implementation-oriented research on trustworthy decentralized AI system.

## REFERENCES


1. H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Aguera y Arcas, "Communication-efficient learning of deep networks from decentralized data," in Proc. 20th Int. Conf. Artificial Intelligence and Statistics (AISTATS), PMLR, vol. 54, pp. 1273-1282, 2017.
2. Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," ACM Trans. Intell. Syst. Technol., vol. 10, no. 2, Article 12, pp. 1-19, 2019, doi: 10.1145/3298981.
3. P. Kairouz et al., "Advances and open problems in federated learning," Found. Trends Mach. Learn., vol. 14, no. 1-2, pp. 1-210, 2021, doi: 10.1561/2200000083.
4. T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," IEEE Signal Process. Mag., vol. 37, no. 3, pp. 50-60, 2020, doi: 10.1109/MSP.2020.2975749.
5. K. Bonawitz et al., "Towards federated learning at scale: System design," in Proc. MLSys, 2019.
6. H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," arXiv:1708.07747, 2017.
7. K. Bonawitz et al., "Practical secure aggregation for privacy-preserving machine learning," in Proc. ACM SIGSAC Conf. Computer and Communications Security (CCS), pp. 1175-1191, 2017, doi: 10.1145/3133956.3133982.
8. M. Abadi et al., "Deep learning with differential privacy," in Proc. ACM SIGSAC Conf. Computer and Communications Security (CCS), pp. 308-318, 2016, doi: 10.1145/2976749.2978318.
9. K. Wei et al., "Federated learning with differential privacy: Algorithms and performance analysis," IEEE Trans. Inf. Forensics Security, vol. 15, pp. 3454-3469, 2020, doi: 10.1109/TIFS.2020.2988575.
10. P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 30, 2017.
11. E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in Proc. AISTATS, PMLR, vol. 108, pp. 2938-2948, 2020.
12. M. Fang, X. Cao, J. Jia, and N. Z. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," in Proc. 29th USENIX Security Symp., pp. 1605-1622, 2020.
13. H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," IEEE Commun. Lett., vol. 24, no. 6, pp. 1279-1283, 2020, doi: 10.1109/LCOMM.2019.2921755.
14. Y. Li, C. Chen, N. Liu, H. Huang, Z. Zheng, and Q. Yan, "A blockchain-based decentralized federated learning framework with committee consensus," IEEE Netw., vol. 35, no. 1, pp. 234-241, 2021, doi: 10.1109/MNET.011.2000263.
15. S. K. Lo, Y. Liu, Q. Lu, C. Wang, X. Xu, H.-Y. Paik, and L. Zhu, "Toward trustworthy AI: Blockchain-based architecture design for accountability and fairness of federated learning systems," IEEE Internet Things J., vol. 10, no. 4, pp. 3276-3284, 2023, doi: 10.1109/JIOT.2022.3144450.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XV, Issue VII, July 2026

16. R. Kumar et al., "Blockchain-federated-learning and deep learning models for COVID-19 detection using CT imaging," IEEE Sens. J., vol. 21, no. 14, pp. 16301-16314, 2021, doi: 10.1109/JSEN.2021.3076767.
17. A. Qammar, A. Karim, H. Ning, and J. Ding, "Securing federated learning with blockchain: A systematic literature review," Artificial Intelligence Review, vol. 56, no. 5, pp. 3951-3985, 2023, doi: 10.1007/s10462-022-10271-9.
18. W. Ning et al., "Blockchain-based federated learning: A survey and new perspectives," Appl. Sci., vol. 14, no. 20, Article 9459, 2024, doi: 10.3390/app14209459.
19. K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the Internet of Things," IEEE Access, vol. 4, pp. 2292-2303, 2016, doi: 10.1109/ACCESS.2016.2566339.
20. V. Buterin, "Ethereum: A next-generation smart contract and decentralized application platform," Ethereum White Paper, 2014.
21. E. Androulaki et al., "Hyperledger Fabric: A distributed operating system for permissioned blockchains," in Proc. 13th EuroSys Conf., pp. 1-15, 2018, doi: 10.1145/3190508.3190538.
22. National Institute of Standards and Technology, Artificial Intelligence Risk Management Framework (AI RMF 1.0), NIST AI 100-1, Jan. 2023.
23. European Parliament and Council of the European Union, Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence, Official Journal of the European Union, 2024.
24. M. Brundage et al., "Toward trustworthy AI development: Mechanisms for supporting verifiable claims," arXiv:2004.07213, 2020.