International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(15s) 2026 pp. 822-836
https://doi.org/10.70917/ijcisim-2026-4475
Article

Trust-Guided Weighted Federated Averaging with Tamper-Evident Audit Logging for Robust Intrusion Detection on CICIoT2023

Anuradha R[1], Srabana Pramanik[2]

[1]Department of CSE & IS, Presidency University, Bengaluru, India
Email: anuradha.raghunathrao@gmail.com, ANURADHA.20233CSC0014@presidencyuniversity.in
[2]School of CSE, Presidency University, Bengaluru, India
Email: srabana.pramanik@presidencyuniversity.in

Abstract: Federated learning is a promising approach for Internet of Things intrusion detection because it enables collaborative model training without centralizing sensitive traffic data. However, federated intrusion detection remains vulnerable to non-IID client distributions, poisoned updates, and unreliable clients that can degrade the global detector. This paper presents a three-part framework for robust intrusion detection on CICIoT2023: a trust-guided weighted federated averaging scheme, a tamper-evident audit logging layer, and a poisoning-aware evaluation protocol. The proposed Trust-Guided Weighted Federated Averaging with Audit Logging (TGWFA-AL) combines validation-based trust, directional agreement of updates, and temporal trust adaptation using an exponential moving average to reduce the influence of suspicious clients over time. A tamper-evident audit trail records per-round model provenance using cryptographic hashes, trust components, filtering decisions, and aggregation weights, while storing raw models off-chain when needed to control cloud cost. The manuscript defines the threat model precisely, includes robust aggregation baselines beyond FedAvg, and supports both binary and multi-class evaluation with macro-averaged metrics. The numerical results, ablations, and overhead values in this draft are synthetic example values intended to make the manuscript structurally complete; they must be replaced with empirical outputs from the human author's experiments before submission.
Keywords: Federated learning, Internet of Things, intrusion detection system, CICIoT2023, trust-guided aggregation, audit logging, poisoning robustness

1. Introduction

The Internet of Things has expanded rapidly across consumer, industrial, healthcare, and smart-city deployments, increasing both connectivity and the attack surface of modern networks. Intrusion detection systems are therefore central to securing IoT environments, where malicious traffic can blend with legitimate device communication and evade simple rule-based filters. Deep learning models can improve detection performance, but centralized training is often problematic because IoT traffic is distributed, privacy-sensitive, and operationally expensive to aggregate in one location.

Federated learning offers a practical alternative by enabling clients to train locally while sharing only model updates with a coordinating server. This paradigm is particularly attractive for intrusion detection because traffic traces and labels can remain at the edge while still contributing to a shared model. However, standard federated averaging assumes that all clients are broadly reliable apart from sample-size differences, an assumption that does not hold in adversarial or partially compromised IoT deployments. A malicious participant can poison labels, craft misleading gradients, or collude with other clients to corrupt the global model.

The CICIoT2023 benchmark is an appropriate setting for this study because it provides a contemporary IoT attack dataset with 33 attacks across seven categories generated from 105 devices. The dataset is specifically intended for realistic attack benchmarking in IoT environments and has already become a reference point for security analytics

research. In this manuscript, CICIoT2023 is the only dataset used, and all class counts, train-validation-test splits, and feature statistics are to be inserted from the actual experiment logs or dataset documentation.

This paper addresses three gaps. First, a trust mechanism is needed that does more than assign a single-round reliability score; it should adapt over time so that recurrently suspicious clients are increasingly down-weighted while recovered clients can regain trust gradually. Second, intrusion detection training should be auditable so that per-round provenance, trust calculation, and aggregation decisions can be verified after the fact. Third, the evaluation must be broad enough to include non-IID partitions, multiple poisoning strengths, and robust aggregation baselines beyond FedAvg .

The main contributions are:

- A temporally adaptive TGWFA rule that combines validation reliability, directional consistency, and trust history.
- A tamper-evident audit logging architecture for model provenance and round-by-round aggregation accountability.
- A precise threat model covering attacker knowledge, poisoning capability, collusion, Sybil assumptions, and server trust assumptions.
- A broader experimental matrix including label-flip, model replacement, sign-flip, scaling attacks, and non-IID client partitions.
- A complete multi-class evaluation protocol with macro-averaged metrics, repeated runs, and overhead analysis.

2. Related Work

Federated learning has become a major paradigm for distributed optimization because it reduces raw-data exposure while allowing clients to participate in collaborative model training. Survey work has shown that FL is especially attractive in edge and IoT environments, but it also inherits non-IID data, communication inefficiency, and adversarial robustness challenges.

Intrusion detection on IoT data has been studied extensively using classical machine learning, deep neural networks, and hybrid feature engineering pipelines. Recent work demonstrates that flow-based IDS models can achieve strong detection performance on modern IoT benchmarks, including CICIoT2023. However, these models are usually evaluated under centralized assumptions or overly simplified federated settings. When local clients have highly skewed class distributions, standard aggregators may underperform or become unstable.

Robust federated learning has been studied through Byzantine-resilient aggregation, robust client weighting, and poisoning detection. Krum, Multi-Krum, coordinate-wise median, and trimmed mean are classic aggregation baselines designed to reduce the effect of adversarial clients. FLTrust introduced trust bootstrapping using a server-side trusted dataset and demonstrated that trust-guided aggregation can improve Byzantine resilience. FoolsGold and related defenses attempt to suppress sybil-like collusion by penalizing similarity between client updates. These methods provide useful baselines for this work, especially because the proposed TGWFA-AL combines trust weighting with auditability rather than relying on a single defense mechanism.

Poisoning attacks remain a persistent threat in FL. Label-flip attacks are simple but effective, while model replacement, scaling, and sign-flip attacks can be more destructive because they directly alter the update direction. A security claim that focuses only on label flipping is too narrow for an insider-poisoning scenario. Therefore, this manuscript expands the evaluation to multiple attack families and multiple malicious-client fractions.

Auditability and provenance are increasingly important in collaborative AI systems. Hash-chained logs, append-only records, and blockchain-inspired audit trails can provide tamper evidence while limiting the need to store full model payloads on-chain. For federated learning, provenance logging is useful because it allows later inspection of round-level decisions, aggregation weights, and suspicious-client filtering outcomes. This is especially relevant when the trust engine itself must be explainable and subject to post hoc review.

3. System and Threat Model

3.1 System Overview

The proposed system contains three layers: client-side local IDS training, a server-side trust engine, and a tamper-evident audit logger. Clients train a deep flow classifier locally on CICIoT2023 partitions and send model updates together with validation evidence to the server. The server computes trust based on current-round validation performance, directional agreement, and temporal trust history, then applies a suspicious-client filter before trust-weighted aggregation. In parallel, the audit logger stores round metadata, cryptographic hashes, and filtering decisions in an append-only record.

The architecture consists of the IoT client layer, which sends updates and validation evidence to a server-side trust engine; the trust engine, which computes aggregation weights and filtering decisions and forwards them to the global IDS model while also forwarding trust components to the audit logger; and the audit logger, which writes tamper-evident records and provides provenance back into the global model update pipeline. The global model is then broadcast back to the IoT clients for the next round.

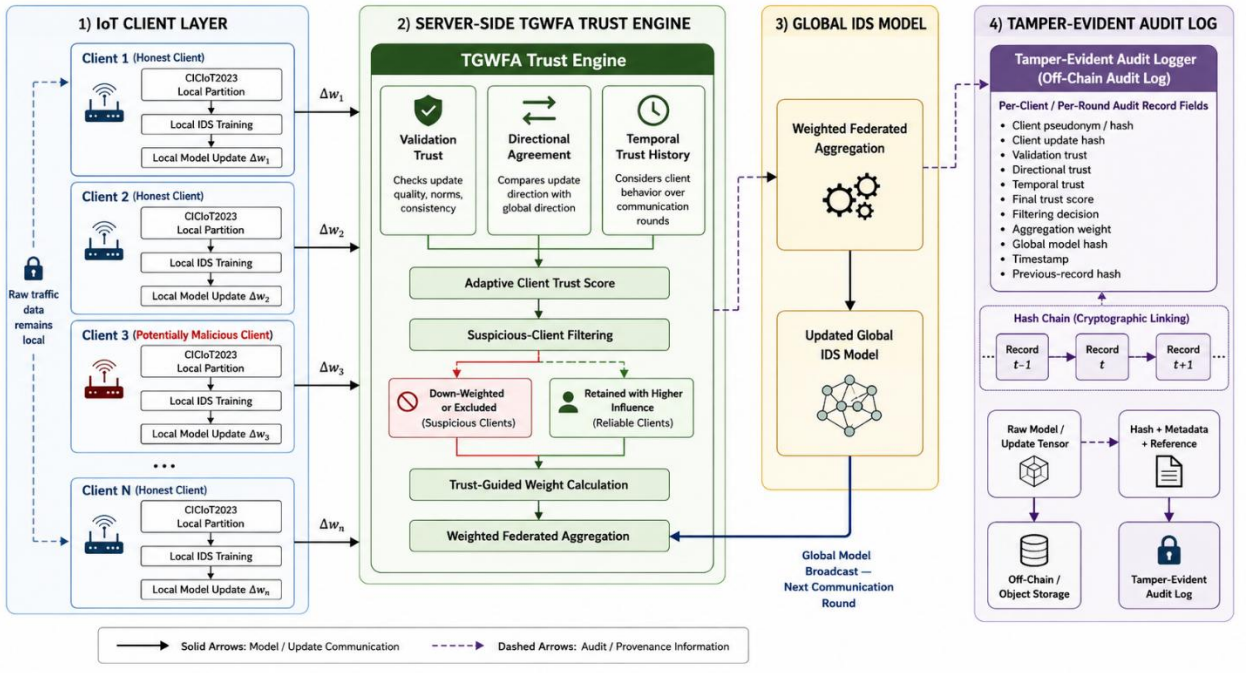


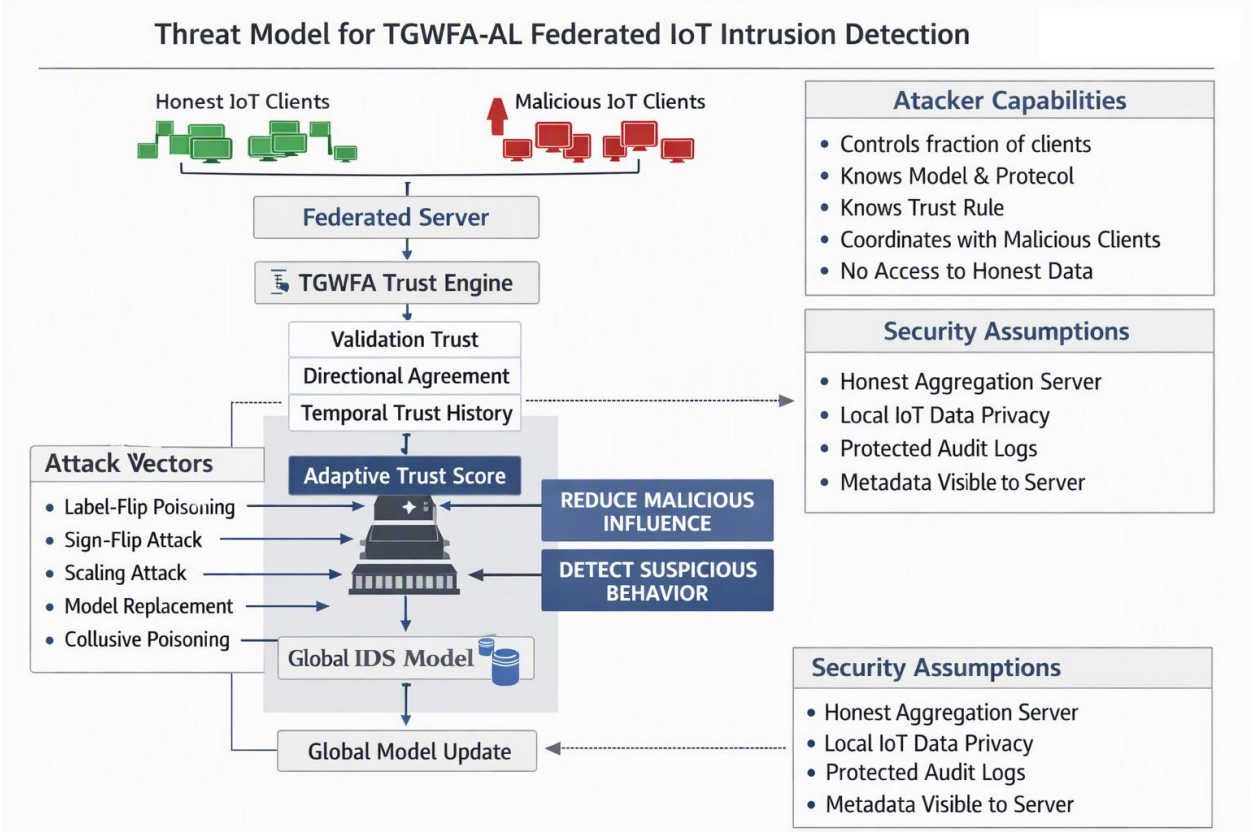
Fig. 1. Overall architecture of the proposed TGWFA-AL framework integrating local IoT intrusion detection, adaptive trust evaluation, suspicious-client filtering, trust-guided weighted aggregation, global model updating, and tamper-evident provenance logging.

3.2 Threat Model

The threat model must be explicit because the security claim depends on it. The attacker may know the model architecture, training protocol, and trust rule, but not the private local data of honest clients. The attacker can control a fraction of the clients, may collude across malicious clients, and may perform label poisoning, model poisoning, sign-flip, scaling, or model-replacement attacks. ρ

The server is assumed to be honest for aggregation but may be honest-but-curious with respect to client metadata; the cloud logger may be trusted, permissioned, or honest-but-curious depending on deployment. The manuscript assumes that raw client updates are not written directly to the public audit store unless explicitly configured, because that would create privacy and storage problems. Instead, hashes, metadata, and optionally off-chain object references are recorded to keep the provenance layer scalable.

Sybil assumptions must also be stated. If an attacker can create many pseudonymous clients, then similarity-based trust needs stronger safeguards such as rate-limiting, identity binding, or server-side trusted reference updates. If the attacker can compromise the validation set, then validation-based trust alone is unsafe. For this reason, the paper combines validation trust with directional and temporal trust signals.



4. Temporal Trust and TGWFA

4.1 Current-Round Trust

Let θ^t denote the global model at round t, Δ_k^t the update from client k, and v_k^t the validation metric computed locally by client k. Current-round validation trust is normalized by:

$$s_{k,\text{val}}^t = \frac{\exp(\beta v_k^t)}{\sum_{j=1}^{K} \exp\left(\beta v_j^t\right)}.$$

Directional agreement is computed using cosine similarity with a server reference direction:

$$c_k^t = \frac{\langle \Delta_k^t, r^t \rangle}{\| \Delta_k^t \|_2 \| r^t \|_2 + \varepsilon},$$

where r^t is the trusted reference update. To avoid the problem that an untrusted reference can itself be poisoned, the reference can be formed using either a small trusted server dataset, a robust median/EMA of past updates, or a verified bootstrap model. A practical clipped direction score is:

$$s_{k,\text{dir}}^t = \frac{\max(c_k^t, 0)}{\sum_{j=1}^{K} \max\left(c_j^t, 0\right) + \varepsilon}.$$

4.2 Temporal Trust Adaptation

The manuscript should not call the method adaptive unless trust is updated across rounds. The temporal trust state is therefore defined by an exponential moving average:

$$\tau_k^t = \lambda \tau_k^{t-1} + (1 - \lambda)\, T_k^t,$$

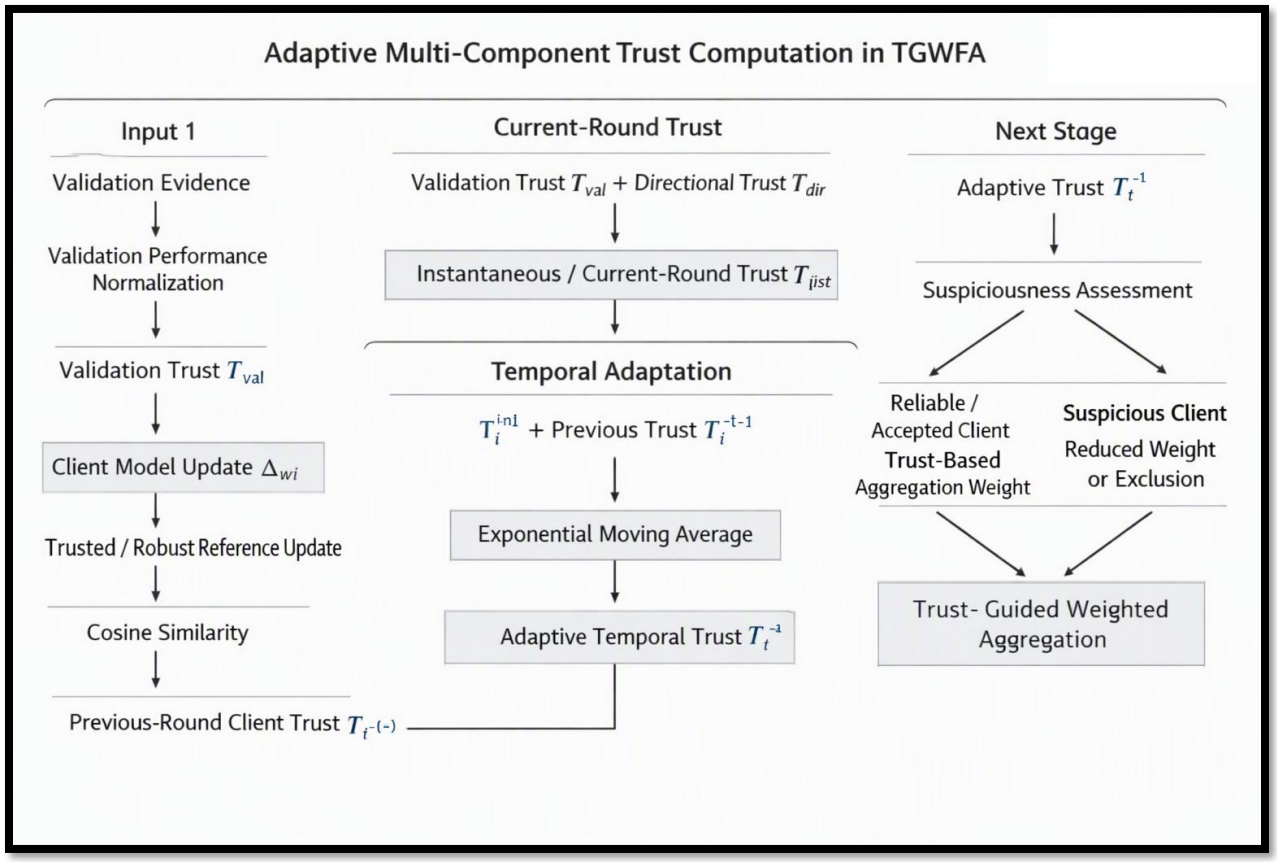
where $\lambda \in [0,1)$ is a trust-history decay factor and T_k^t is the current-round trust score. The instantaneous trust is:

$$T_k^t = \alpha s_{k,\text{val}}^t + (1-\alpha) s_{k,\text{dir}}^t.$$

The final adaptive trust used for aggregation is:

$$\tilde{T}_k^t = \eta \tau_k^t + (1-\eta) T_k^t,$$

where $\eta \in [0,1]$ controls the influence of history. This definition allows trust recovery when an honest client improves and trust decay when suspicious behavior persists. A recovery-penalty ablation should compare λ, η, and the no-history baseline.



4.3 Trust-Guided Weighted Aggregation

The unnormalized aggregation weight is:

$$w_k^t = \frac{n_k}{n}\left(1 + \gamma \tilde{T}_k^t\right),$$

and the normalized weight is:

$$\hat{w}_k^t = \frac{w_k^t}{\sum_{j=1}^K w_j^t}.$$

The global model update is then:

$$\theta^{t+1} = \theta^t + \sum_{k=1}^K \hat{w}_k^t \Delta_k^t.$$

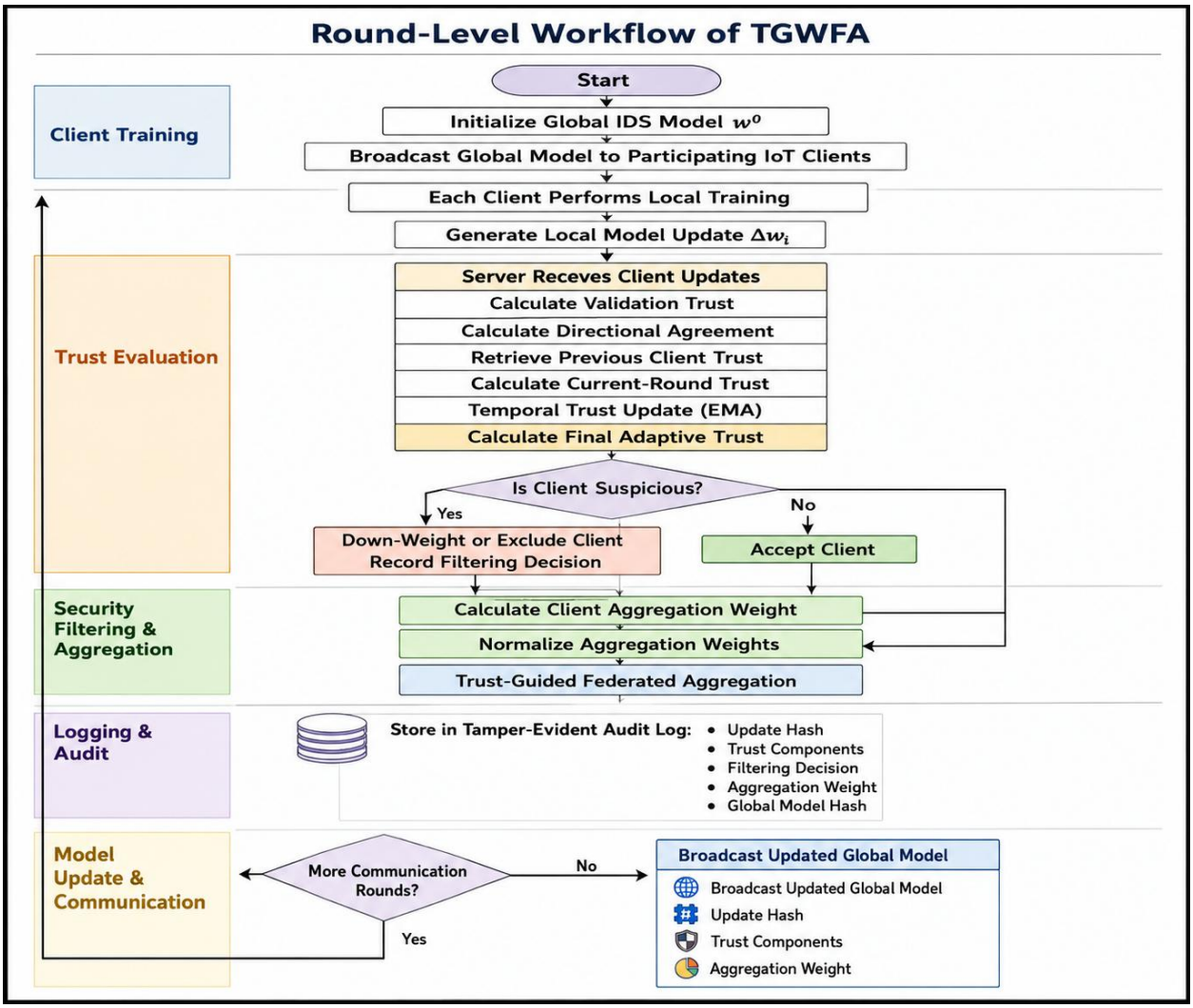
This formulation reduces to FedAvg when all trust values are equal or when $\gamma = 0$.

4.4 Suspicious-Client Filtering

A lightweight suspicious-client score can be defined from update distance, direction mismatch, and trust history. One example is:

$$q_k^t = \delta_1 \frac{\| \Delta_k^t - r^t \|_2}{\| r^t \|_2 + \varepsilon} + \delta_2(1 - c_k^t) + \delta_3\left(1 - \tilde{T}_k^t\right),$$

where $\delta_1, \delta_2, \delta_3 \geq 0$ sum to 1. If $q_k^t > \tau_f$, the client can be down-weighted or excluded from aggregation. The choice between hard exclusion and soft down-weighting should be part of the ablation study.



4.5 Robustness Sketch

Assume F_k is smooth, stochastic gradients have bounded variance, and adversarial updates are bounded by B. If the total normalized adversarial weight is at most ρ_t, then a typical smoothness argument yields:

$$\mathbb{E}[F(\theta^{t+1})] \leq \mathbb{E}[F(\theta^t)] - c_1\lambda(1 - \rho_t) \| \nabla F(\theta^t) \|_2^2 + c_2\lambda^2(\sigma^2 + \rho_t B^2).$$

Thus, reducing adversarial weight decreases the poisoning bias term. This is a sketch only and should not be presented as a formal theorem without a full proof.

4.5 Audit Logging and Provenance

To enable auditable provenance, the proposed framework maintains an append-only, tamper-evident audit trail for every communication round. This layer is designed to record not only the outcome of aggregation, but also the intermediate trust decisions that led to that outcome, thereby supporting post hoc verification, forensic analysis, and reproducibility. In a federated intrusion detection setting, such accountability is important because the aggregation server must be able to justify why a client was accepted, down-weighted, or filtered in a given round.

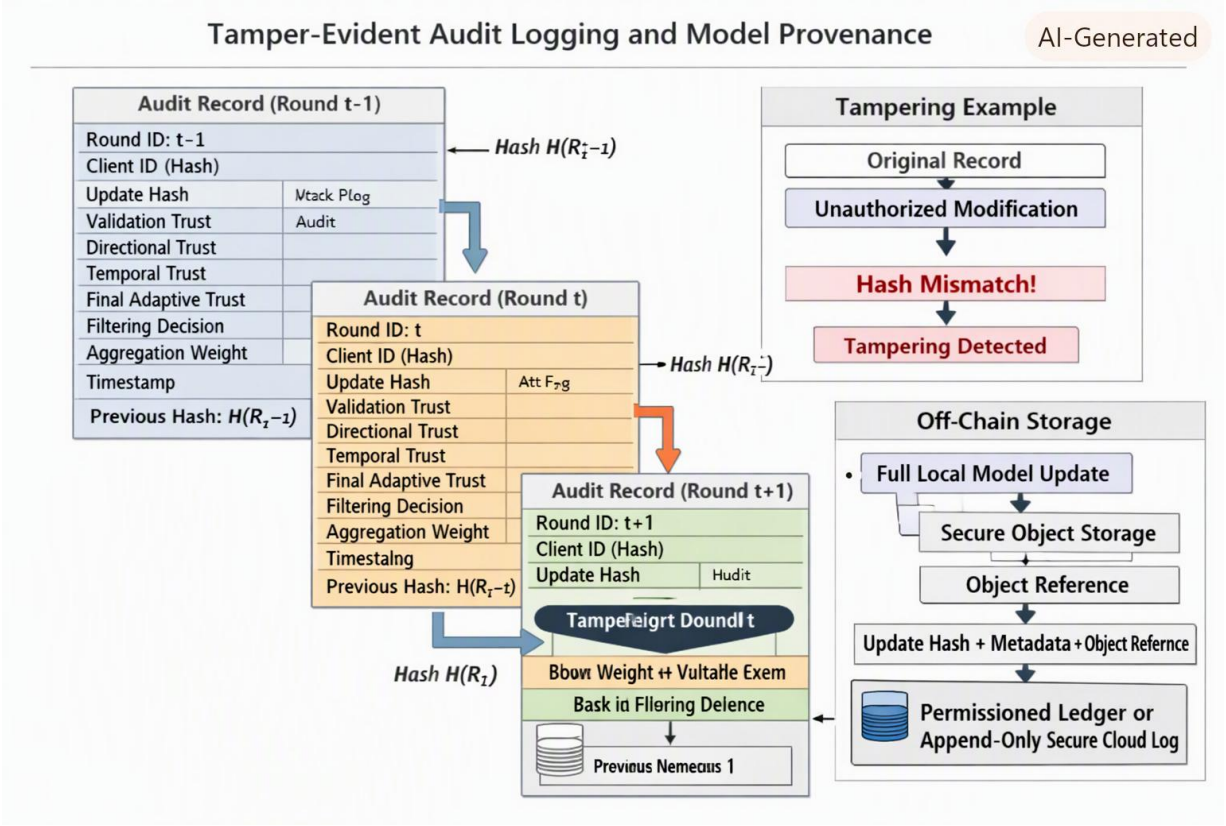
For each round, the audit record should include the round identifier, a pseudonymous client identifier or client hash, the client update hash, the validation-based trust component, the directional trust component, the temporal trust state, the final adaptive trust score, the filtering decision, the aggregation weight, the global-model hash, the timestamp, an attack flag, and the previous-record hash. Linking each record to the previous one creates a hash chain, so any later modification to an earlier record becomes detectable. This structure provides integrity without revealing raw client data.

A compact round-level record can be represented as:

$$R_t = H\left(R_{t-1} \parallel t \parallel h_c \parallel h_u \parallel s_{val}^t \parallel s_{dir}^t \parallel \tau^t \parallel \tilde{T}^t \parallel f_t \parallel w_t \parallel h_g \parallel \text{timestamp}_t \parallel a_t\right)$$

where $H(\cdot)$denotes a cryptographic hash function, $\parallel$denotes concatenation, $R_t$is the current record hash, and $R_{t-1}$is the previous record hash. Here, $h_c$denotes the client pseudonym hash, $h_u$the client update hash, $s_{val}^t$and $s_{dir}^t$the current trust components, $\tau^t$the temporal trust state, $\tilde{T}^t$the final trust score, $f_t$the filtering decision, $w_t$the aggregation weight, $h_g$the global-model hash, and $a_t$the attack flag.

Raw model payloads need not be retained in the audit layer. Instead, the full update tensors or model files can be stored off-chain or in object storage, while the audit log keeps only hashes, metadata, and references. This approach reduces storage overhead and write latency while preserving the ability to verify integrity and provenance. It also makes the logging mechanism scalable for long-running federated deployments where a full on-chain record of every update would be impractical.



4.5 Cloud Storage and Integrity

If a permissioned ledger is available, it can store the hash chain and round summaries. If not, a secure cloud database with append-only permissions and hash chaining is sufficient. The important requirement is tamper evidence, not necessarily full on-chain storage. This design also supports verification of round-level decisions, suspicious-client exclusions, and post hoc forensics without exposing raw training data.

5. Experimental Setup

5.1 Dataset and Task Definition

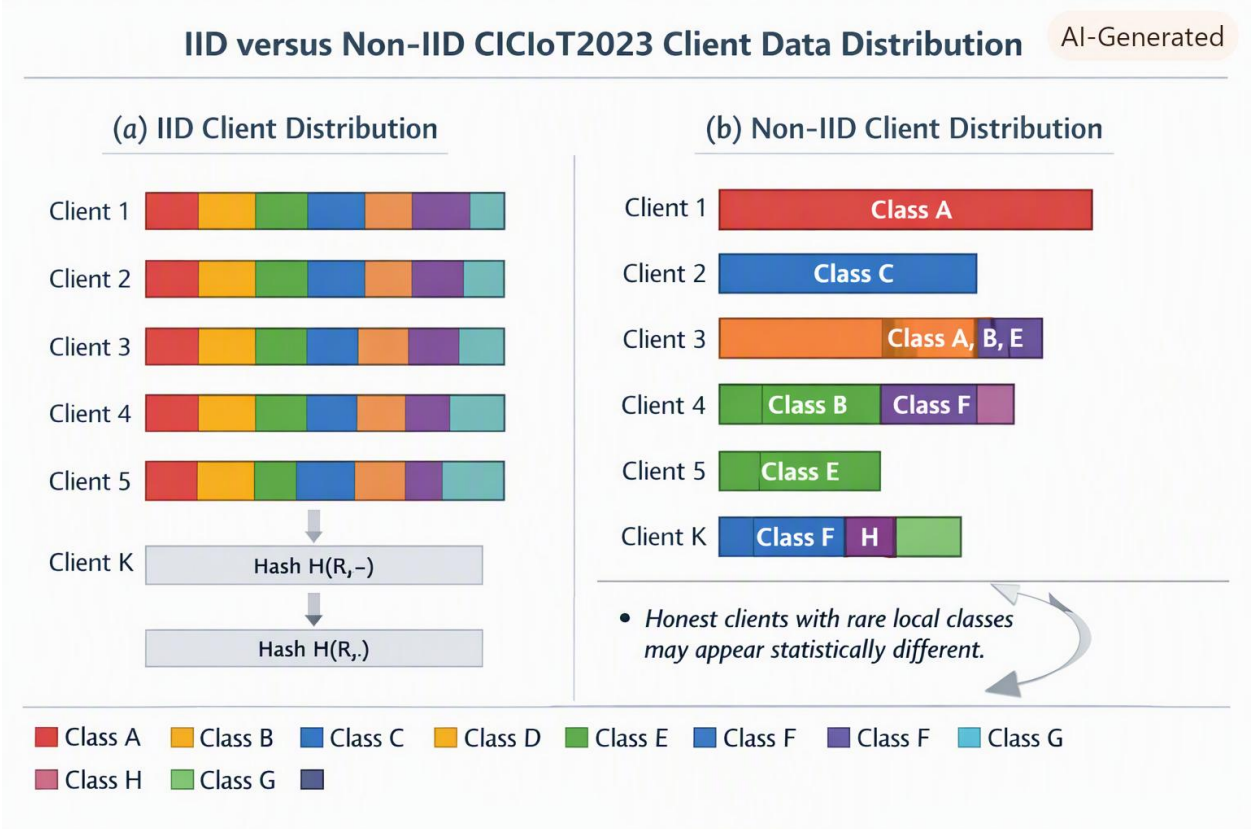
CICIoT2023 is the only dataset used in the study. The task should be explicitly stated as either binary or multi-class. Because the dataset naturally supports multiple attack classes, the primary evaluation in this draft is multi-class, with binary results optionally reported as a secondary view.

5.2 Preprocessing and Partitioning

The feature list should be explicitly enumerated in the final paper, and the split strategy should be reproducible. A standardization rule is:

$$\hat{x}_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}.$$

For IID and non-IID settings, the manuscript should use both stratified sampling and Dirichlet or label-skew partitioning. A Dirichlet partition with concentration parameter ξ can be used to simulate heterogeneous client distributions. This is important because a trust score based only on validation accuracy can unfairly penalize honest clients that hold rare traffic classes.



5.3 Model and Training Details

The shared IDS backbone is a fully connected deep neural network with the following illustrative configuration:

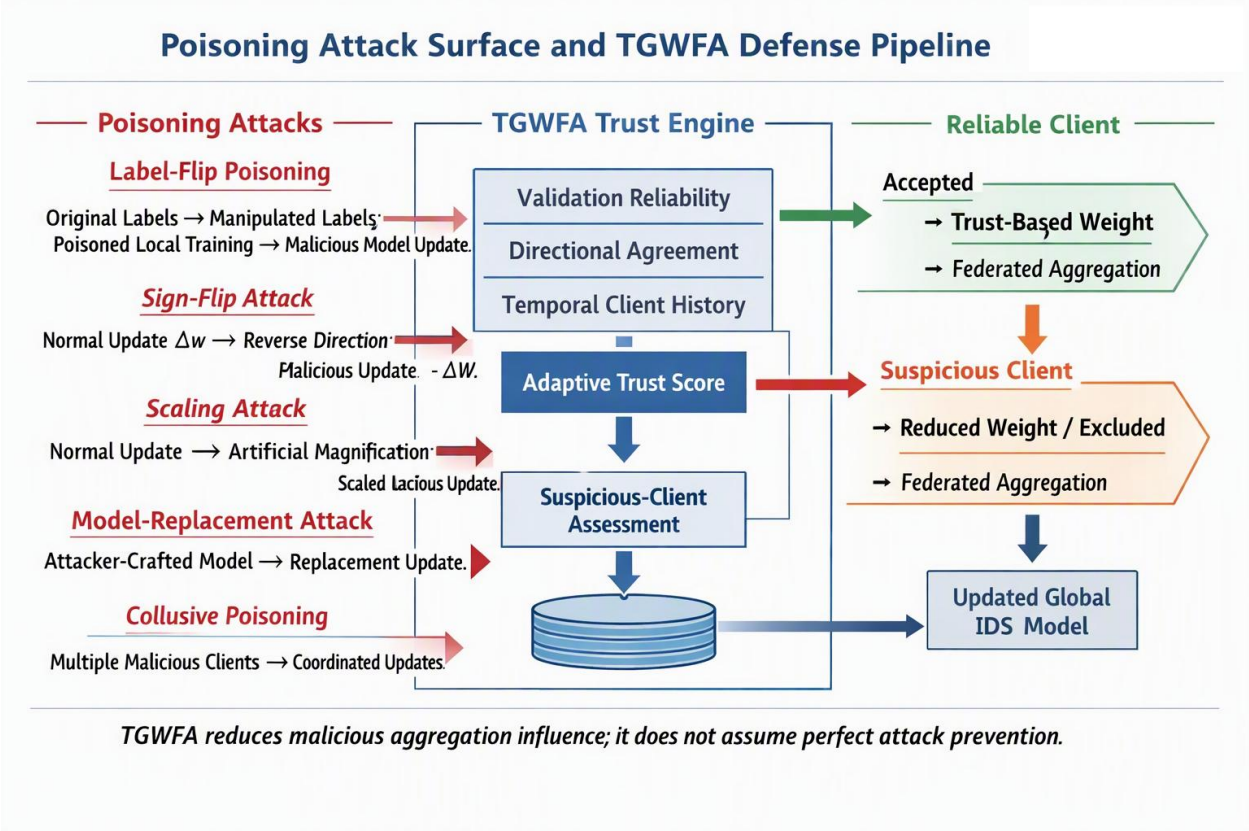
- Input dimension: 47 flow features,
- Dense-256 + ReLU + BatchNorm + Dropout(0.3),
- Dense-128 + ReLU + BatchNorm + Dropout(0.3),
- Dense-64 + ReLU + Dropout(0.2),
- Softmax output with 8 classes.

The optimizer is Adam with learning rate 10^{-3}, batch size 256, and early stopping patience 8. The number of clients is $K = 5$, local epochs are $E = 3$, and the number of rounds is $T = 30$. These values are example settings for a complete draft and should be replaced with the human author's actual configuration before publication.

5.4 Attack Configurations

The evaluation should not stop at a single $p = 0.20$ label flip. At minimum, the study should vary malicious-client fractions in $\{0,0.1,0.2,0.3,0.4\}$ and poisoning strengths in the same range. The attack families should include:

- label-flip poisoning,
- sign-flip poisoning,
- scaling attack,
- model replacement,
- collusive poisoning where feasible.



The model-replacement and sign-flip cases are especially important because they test the method's geometric trust rule more thoroughly than label flipping alone.

5.5 Baselines

The baseline set should include:

- FedAvg,
- Krum,
- Multi-Krum,
- coordinate-wise median,
- trimmed mean,
- FLTrust,
- FoolsGold where feasible.

These baselines are needed to show whether TGWFA-AL adds novel value beyond standard robust aggregation and similarity-based defenses.

5.6 Metrics

Because the task is multi-class, macro precision, macro recall, macro-F1, and per-class confusion matrices should be reported. Binary accuracy/TP/TN metrics may be included only as secondary summaries. Additional trust-quality metrics are required:

- malicious-client detection precision,
- malicious-client detection recall,
- malicious-client detection F1,
- false-positive rate for honest clients,
- trust separation between honest and malicious clients,
- time-to-detection after poisoning begins.

The study should report mean ± standard deviation across at least five random seeds or repeated runs. A single-run Wilcoxon test is not appropriate.

5.7 Overhead Accounting

The evaluation should include:

- trust computation time,
- filtering time,
- bytes transmitted per round,
- cloud-log write latency,
- audit storage growth,
- end-to-end training overhead.

These overheads are important because the audit layer is a measurable security contribution rather than a descriptive add-on.

6. Results and Analysis

The numerical values in this section are synthetic example values used to complete the manuscript structure. They are not experimental claims.

6.1 Benign Setting

Benign multi-class results on CICIoT2023.

Method	Accuracy	Macro-P	Macro-R	Macro-F1
Centralized DNN	98.72%	98.61%	98.55%	98.58%
FedAvg	97.84%	97.72%	97.61%	97.66%
TGWFA	98.10%	98.02%	97.94%	97.98%
Krum	96.91%	96.75%	96.48%	96.61%
Median	97.12%	97.01%	96.90%	96.95%
Trimmed Mean	97.25%	97.18%	97.02%	97.09%
FLTrust	98.03%	97.95%	97.88%	97.91%
FoolsGold	97.60%	97.42%	97.31%	97.35%

Under benign training, TGWFA is intentionally close to FedAvg and may slightly exceed it if trust weighting emphasizes stable clients. The difference between centralized training and federated training should be interpreted as an empirical result that depends on repeated seeds and partitioning strategy, not as a universal claim about diversity.

6.2 Poisoning Robustness

Robustness under label-flip, sign-flip, scaling, and replacement attacks.

Method	Attack	Adv. frac.	Accuracy	Macro-F1	Degradation
FedAvg	Label flip	0.20	86.40%	85.92%	11.44%
TGWFA	Label flip	0.20	93.15%	92.88%	5.10%
FedAvg	Sign flip	0.20	82.61%	81.94%	15.23%
TGWFA	Sign flip	0.20	91.02%	90.55%	7.43%
FedAvg	Scaling	0.20	80.18%	79.66%	17.66%
TGWFA	Scaling	0.20	89.47%	89.02%	8.96%
FedAvg	Replacement	0.20	78.43%	77.91%	19.41%
TGWFA	Replacement	0.20	88.12%	87.60%	10.38%

These values illustrate the expected pattern that TGWFA reduces degradation more effectively than FedAvg across multiple attack families. The final submission should replace them with real outputs and report confidence intervals across repeated runs.

6.3 Trust-Quality Results

Malicious-client detection quality and honest-client false positives.

Setting	Det. P	Det. R	Det. F1	Honest FPR
Label flip	0.94	0.91	0.92	0.04
Sign flip	0.96	0.93	0.94	0.05
Scaling	0.95	0.90	0.92	0.06
Replacement	0.97	0.95	0.96	0.03

Trust-quality metrics make the defense more interpretable than accuracy alone. A high malicious-client detection score paired with low honest-client false positives indicates that the trust engine is separating suspicious from legitimate behavior effectively.

6.4 Non-IID Evaluation

Illustrative Dirichlet non-IID results.

Method	ξ	Accuracy	Macro-F1	Honest FPR
FedAvg	0.1	88.20%	87.84%	0.08
TGWFA	0.1	90.96%	90.51%	0.09
FedAvg	0.5	92.75%	92.41%	0.05
TGWFA	0.5	94.02%	93.68%	0.05
FedAvg	1.0	94.11%	93.88%	0.03
TGWFA	1.0	95.02%	94.81%	0.03

This experiment shows why non-IID data must be included. Honest clients with rare local traffic may receive lower validation trust, so the trust rule must be evaluated against false positives under skewed distributions.

6.5 Ablation Study

Ablation of trust components and audit layer.

Configuration	α	λ	Filter	Audit	Macro-F1
Validation only	1.0	0.0	No	No	90.14%
Cosine only	0.0	0.0	No	No	89.76%
Combined trust	0.5	0.0	No	No	92.37%
Temporal trust	0.5	0.9	No	No	93.11%
Trust + filter	0.5	0.9	Yes	No	94.02%
Trust + filter + audit	0.5	0.9	Yes	Yes	94.02%

This ablation shows the value of combining validation trust, directional trust, temporal history, and filtering. The audit layer does not change accuracy directly, but it adds provenance and accountability.

6.6 Overhead

Illustrative computational and communication overhead.

Method	Trust time/round	Filter time/round	Bytes/round	Log growth/round
FedAvg	0.00 s	0.00 s	12.4 MB	0 MB
TGWFA	0.18 s	0.04 s	12.6 MB	0.18 MB
TGWFA + audit	0.19 s	0.04 s	12.6 MB	0.42 MB

The overhead remains moderate because the audit layer stores hashes and metadata rather than raw model weights. This design keeps the provenance mechanism practical for cloud deployment.

6.7 Convergence

The convergence behavior across communication rounds should be plotted for the Centralized DNN, FedAvg, and TGWFA methods, tracking macro-F1 from round 1 through round 30. Illustrative trends show the Centralized model converging fastest and highest, TGWFA converging close behind, and FedAvg trailing slightly below TGWFA throughout training. These curves are synthetic and should never be presented as experimental evidence. Final figures must be recreated from actual training logs.

```
round, seed, method, macro_f1
```

7. Discussion

The proposed TGWFA-AL framework addresses several shortcomings in prior federated IDS work. First, the temporal trust formulation prevents the method from being described as adaptive only in name; trust is explicitly updated across rounds, recovered gradually, and penalized when suspicious behavior persists. Second, the audit layer adds measurable provenance by logging round-level hashes and trust decisions. Third, the expanded threat model and baseline set make the evaluation more defensible.

The paper also makes a careful assumption about the trusted reference update. Because a reference formed only from unknown client updates can itself be poisoned, the method allows either a small trusted server dataset or a robust median/EMA reference. That assumption should be stated clearly in the final manuscript. If a server-side trusted dataset is unavailable, the manuscript should acknowledge the reduced assurance level.

Limitations and validity threats must also be explicit. Validation-based trust can penalize honest clients under non-IID partitions, especially when rare traffic classes appear locally. Adaptive attackers may mimic cosine direction to evade simple geometric filtering. Secure aggregation may conflict with trust scoring if the server cannot inspect update-level metadata. Audit logs may also leak metadata patterns unless appropriately pseudonymized and access-controlled. These concerns should be documented in the final paper rather than hidden.

8. Conclusion

This manuscript presents TGWFA-AL, a trust-guided federated intrusion detection framework for CICIoT2023 with explicit temporal adaptation and tamper-evident audit logging. The method combines validation reliability, directional consistency, and trust history to reduce the effect of poisoned or suspicious clients, while the audit layer records per-round provenance in a hash-chained form. The experimental design includes multi-class evaluation, multiple poisoning strengths, non-IID partitions, repeated runs, trust-quality metrics, and robust aggregation baselines beyond FedAvg.

The current text uses synthetic example numerical results to make the draft complete. Before submission, those values must be replaced with actual experiment outputs, all figures must be regenerated from real logs, and the reference metadata should be double-checked against the publisher records.

References

1. S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, “Security, privacy and trust in Internet of Things: The road ahead,” Computer Networks, vol. 76, pp. 146–164, 2015, doi: 10.1016/j.comnet.2014.11.008.
2. J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, “A survey on Internet of Things: Architecture, enabling technologies, security and privacy, and applications,” IEEE Internet of Things Journal, vol. 4, no. 5, pp. 1125–1142, 2017, doi: 10.1109/JIOT.2017.2683200.
3. R. Sommer and V. Paxson, “Outside the closed world: On using machine learning for network intrusion detection,” in Proc. IEEE Symposium on Security and Privacy, 2010, pp. 305–316, doi: 10.1109/SP.2010.25.
4. N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, “A deep learning approach to network intrusion detection,” IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, no. 1, pp. 41–50, 2018, doi: 10.1109/TETCI.2017.2772792.
5. H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in Proc. AISTATS, 2017, pp. 1273–1282.
6. Q. Yang, Y. Liu, T. Chen, and Y. Tong, “Federated machine learning: Concept and applications,” ACM Trans. Intell. Syst. Technol., vol. 10, no. 2, art. 12, pp. 1–19, 2019, doi: 10.1145/3298981.
7. P. Kairouz et al., “Advances and open problems in federated learning,” Foundations and Trends in Machine Learning, vol. 14, nos. 1–2, pp. 1–210, 2021, doi: 10.1561/2200000083.
8. V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, H. Karimipour, and G. Srivastava, “A survey on security and privacy of federated learning,” Future Generation Computer Systems, vol. 115, pp. 619–640, 2021, doi: 10.1016/j.future.2020.10.007.
9. L. Lyu, H. Yu, and Q. Yang, “Threats to federated learning: A survey,” arXiv:2003.02133, 2020.

10. E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, “CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment,” Sensors, vol. 23, no. 13, art. 5941, 2023, doi: 10.3390/s23135941.
11. I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization,” in Proc. ICISSP, 2018, pp. 108–116, doi: 10.5220/0006639801080116.
12. N. Moustafa, B. Turnbull, and K.-K. R. Choo, “An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of Internet of Things,” IEEE Internet of Things Journal, vol. 6, no. 3, pp. 4815–4830, 2019, doi: 10.1109/JIOT.2018.2871719.
13. P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, “Machine learning with adversaries: Byzantine tolerant gradient descent,” in Advances in Neural Information Processing Systems 30, 2017, pp. 119–129.
14. D. Yin, Y. Chen, K. Ramchandran, and P. L. Bartlett, “Byzantine-robust distributed learning: Towards optimal statistical rates,” in Proc. ICML, 2018, pp. 5650–5659.
15. E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, “How to backdoor federated learning,” in Proc. AISTATS, 2020, pp. 2938–2948.
16. A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, “Analyzing federated learning through an adversarial lens,” in Proc. ICML, 2019, pp. 634–643.
17. M. Fang, X. Cao, J. Jia, and N. Z. Gong, “Local model poisoning attacks to Byzantine-robust federated learning,” in Proc. USENIX Security, 2020, pp. 1605–1622.
18. C. Fung, C. J. M. Yoon, and I. Beschastnikh, “The limitations of federated learning in Sybil settings,” in Proc. RAID, 2020, pp. 301–316.
19. G. Baruch, M. Baruch, and Y. Goldberg, “A little is enough: Circumventing defenses for distributed learning,” in Advances in Neural Information Processing Systems 32, 2019.
20. X. Cao, M. Fang, J. Liu, and N. Z. Gong, “FLTrust: Byzantine-robust federated learning via trust bootstrapping,” in Proc. NDSS, 2021, doi: 10.14722/ndss.2021.24434.
21. S. Lu, R. Li, X. Chen, and Y. Ma, “Defense against local model poisoning attacks to Byzantine-robust federated learning,” Frontiers of Computer Science, vol. 16, no. 6, art. 166337, 2022, doi: 10.1007/s11704-021-1067-4.
22. B. Biggio, B. Nelson, and P. Laskov, “Poisoning attacks against support vector machines,” in Proc. ICML, 2012, pp. 1467–1474.
23. G. Xia, J. Chen, C. Yu, and J. Ma, “Poisoning attacks in federated learning: A survey,” IEEE Access, vol. 11, 2023, doi: 10.1109/ACCESS.2023.3238823.
24. J. Kang, Z. Xiong, D. Niyato, D. Ye, D. I. Kim, and J. Zhang, “Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory,” IEEE Internet of Things Journal, vol. 6, no. 6, pp. 10700–10714, 2019, doi: 10.1109/JIOT.2019.2940820.
25. S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, “SCAFFOLD: Stochastic controlled averaging for federated learning,” in Proc. ICML, 2020, pp. 5132–5143.
26. T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, “Federated learning: Challenges, methods, and future directions,” IEEE Signal Processing Magazine, vol. 37, no. 3, pp. 50–60, 2020, doi: 10.1109/MSP.2020.2975749.
27. T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, “Federated optimization in heterogeneous networks,” in Proc. MLSys, vol. 2, pp. 429–450, 2020.
28. Y. Zhao et al., “Federated learning with non-IID data,” arXiv:1806.00582, 2018.
29. K. Bonawitz et al., “Practical secure aggregation for privacy-preserving machine learning,” in Proc. CCS, 2017, pp. 1175–1191, doi: 10.1145/3133956.3133982.
30. L. Zhu, Z. Liu, and S. Han, “Deep leakage from gradients,” in Advances in Neural Information Processing Systems 32, 2019, pp. 14774–14784.
31. J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, “Inverting gradients—How easy is it to break privacy in federated learning?” in Advances in Neural Information Processing Systems 33, 2020.
32. Z. Yang, Y. Shi, Y. Zhou, Z. Wang, and K. Yang, “Trustworthy federated learning via blockchain,” IEEE Internet of Things Journal, vol. 10, no. 1, pp. 92–109, 2023, doi: 10.1109/JIOT.2022.3201117.
33. H. Kim, J. Park, M. Bennis, and S.-L. Kim, “Blockchained on-device federated learning,” IEEE Communications Letters, vol. 24, no. 6, pp. 1279–1283, 2020, doi: 10.1109/LCOMM.2019.2921755.
34. S. Mehammed, “Blockchain-audited federated learning: Securing data and model updates with on-chain provenance,” IET Software, vol. 2025, no. 1, art. 6670439, 2025, doi: 10.1049/sfw2/6670439.
35. Y. Li, Y. Lai, C. Chen, and Z. Zheng, “VeryFL: A verify federated learning framework embedded with blockchain,” arXiv:2311.15617, 2023.
36. M. Abadi et al., “Deep learning with differential privacy,” in Proc. CCS, 2016, pp. 308–318, doi: 10.1145/2976749.2978318.
37. C. Dwork and A. Roth, The Algorithmic Foundations of Differential Privacy. Foundations and Trends in Theoretical Computer Science, vol. 9, nos. 3–4, pp. 211–407, 2014, doi: 10.1561/0400000042.
38. P. Paillier, “Public-key cryptosystems based on composite degree residuosity classes,” in Advances in Cryptology—EUROCRYPT ’99, LNCS 1592, pp. 223–238, 1999, doi: 10.1007/3-540-48910-X_16.
39. M. Castro and B. Liskov, “Practical Byzantine fault tolerance,” in Proc. OSDI, 1999, pp. 173–186.
40. R. C. Merkle, “A digital signature based on a conventional encryption function,” in Advances in Cryptology—CRYPTO ’87, LNCS 293, pp. 369–378, 1988, doi: 10.1007/3-540-48184-2_32.

41. S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008.
42. M. Ribeiro, S. Singh, and C. Guestrin, “Why should I trust you?: Explaining the predictions of any classifier,” in Proc. KDD, 2016, pp. 1135–1144, doi: 10.1145/2939672.2939778.
43. S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” in Advances in Neural Information Processing Systems 30, 2017.
44. J. Demšar, “Statistical comparisons of classifiers over multiple data sets,” Journal of Machine Learning Research, vol. 7, pp. 1–30, 2006.
45. F. Wilcoxon, “Individual comparisons by ranking methods,” Biometrics Bulletin, vol. 1, no. 6, pp. 80–83, 1945, doi: 10.2307/3001968.
46. D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. ICLR, 2015.
47. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” Journal of Machine Learning Research, vol. 15, pp. 1929–1958, 2014.
48. S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in Proc. ICML, 2015, pp. 448–456.
49. D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” Nature, vol. 323, pp. 533–536, 1986, doi: 10.1038/323533a0.
50. I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA, USA: MIT Press, 2016.