A Federated Learning Based Privacy-Preserving Intrusion Detection Framework for Heterogeneous IoT Environments: Architecture, Non-IID Client Modelling and Evaluation Protocol

Sujata Pattnaik[*1], Dr. Ridhika Sidhi[*2]
[*1] Research Scholar, Department of Computer Science, William Carey University, Meghalaya, India
[*2] Phd. Guide, Department of Computer Science, William Carey University, Meghalaya, India

Abstract—
Intrusion detection in Internet of Things (IoT) networks increasingly relies on machine learning, but centralised training requires sensitive traffic from many sites to be pooled on one server. This paper proposes a federated learning based privacy-preserving intrusion detection framework for distributed IoT environments in which devices, gateways and edge nodes act as federated clients that preprocess traffic and train a lightweight intrusion detection model locally. Model updates are protected through a layered privacy design that combines norm clipping and Gaussian noise for differential privacy, secure aggregation of masked updates, encrypted communication and client authentication; a federated server aggregates the protected updates into a global model that is redistributed for local detection. Beyond the architecture, the paper contributes a reproducible experimental protocol that addresses weaknesses in current evaluations: candidate IoT datasets and selection criteria, a documented preprocessing pipeline, Dirichlet-based non-IID client partitioning, five distributed scenarios covering balanced data, attack-mix heterogeneity, partial participation, client dropout and poisoned updates, and three comparable configurations—a centralised baseline, standard federated learning and privacy-preserving federated learning. Evaluation is multidimensional, pairing detection metrics with privacy parameters, an analytic communication-cost model, training time and robustness. The framework and protocol are designed to make the trade-offs between detection quality, privacy protection and resource cost explicit and measurable, rather than assumed.

Index Terms—Internet of Things, intrusion detection, federated learning, differential privacy, secure aggregation, non-IID data, communication cost, experimental design

1. Introduction

IoT networks link large numbers of heterogeneous devices whose limited security makes them frequent targets of scanning, denial-of-service, botnet and injection attacks [1]. Machine-learning-based intrusion detection systems (IDS) are an established response [2], but they are typically trained on traffic collected centrally. For distributed IoT deployments—homes, factories, hospitals and smart-city infrastructure—centralisation exposes sensitive operational data, consumes constrained bandwidth and creates a single high-value repository.

Federated learning (FL) allows clients to train a shared model while retaining raw data locally [3]. Existing federated IDS studies demonstrate feasibility [4]–[6], yet four shortcomings recur in the literature [7]–[9]. First, FL is often treated as inherently private, although model updates can leak information about training data [10]. Second, client data in IoT is naturally non-IID, and federated averaging can degrade under skewed label distributions [11]. Third, communication and computation costs are seldom reported alongside detection metrics. Fourth, the learning process itself can be attacked through poisoned updates [12], but threat models are frequently implicit.

This paper addresses these shortcomings with an integrated framework and an explicit evaluation protocol. Its contributions are:

1) A layered architecture for privacy-preserving federated intrusion detection that integrates local preprocessing, a lightweight local IDS model, differential privacy, secure aggregation, encrypted communication and authenticated participation.
2) A formal description of the federated training procedure, including clipped and noised client updates and masked aggregation, summarised as an algorithm suitable for implementation.
3) A reproducible experimental design with candidate IoT datasets, Dirichlet-based non-IID client partitioning and five distributed scenarios, including dropout and poisoned updates.
4) A multidimensional evaluation protocol that pairs detection metrics with privacy parameters, an analytic communication-cost

model, training time and statistical comparison.

Section 2 reviews related work. Section 3 states the research objectives and hypotheses. Section 4 presents the framework, Section 5 the experimental design and Section 6 the evaluation protocol. Section 7 discusses expected trade-offs and limitations, and Section 8 concludes.

2. Related Work

DÏoT pioneered federated anomaly detection for IoT by training device-type-specific models at security gateways [4]. Rahman *et al.* compared centralised, on-device and federated IDS and reported that federated training could approach centralised accuracy [5]. Federated deep models have since been applied to IoT attack detection [13] and zero-day botnet detection on edge devices [6]. Campos *et al.* showed that non-IID client partitions and the choice of aggregation function significantly affect federated IDS performance [7]. Surveys by Agrawal *et al.* [14], Vyas *et al.* [8] and Belenguer *et al.* [9] consistently call for privacy mechanisms beyond data locality and for more realistic, reproducible evaluation.

On the privacy side, DP-SGD clips per-example gradients and adds calibrated Gaussian noise [15], while client-level DP for FL applies the same idea to client updates [16]; Wei *et al.* analysed the resulting convergence–privacy trade-off [17]. Secure aggregation hides individual updates from the server using pairwise masking [18]. For heterogeneous data, FedProx adds a proximal term to stabilise local training [19], and Dirichlet label partitioning has become a standard way to generate controlled non-IID splits [20]. The framework in this paper combines these components and, crucially, specifies how their joint effect is to be measured.

3. Research Objectives, Questions and Hypotheses

The overall aim is to design and evaluate a federated IDS that balances detection performance, privacy protection and resource requirements in distributed IoT environments. Five objectives operationalise this aim: (O1) analyse IoT security characteristics and heterogeneity relevant to distributed intrusion detection; (O2) develop a federated IDS architecture in which raw traffic stays local; (O3) integrate and evaluate privacy mechanisms for model updates; (O4) design a lightweight model suitable for edge resources; and (O5) evaluate the framework under realistic distributed conditions. Table I links the research questions and hypotheses to these objectives and to the measurements that test them.

TABLE I RESEARCH QUESTIONS, HYPOTHESES AND MEASUREMENTS

Obj.	Research question	Hypothesis	Primary measurement
O2	RQ1: Can federated training approach centralised detection without sharing raw traffic?	H1: Standard FL achieves macro-F1 comparable to the centralised baseline under IID partitions.	Macro-F1, recall, FPR
O3	RQ2: How do privacy settings affect detection utility?	H2: Stronger privacy (smaller ε) reduces minority-class recall monotonically.	Macro-F1 and per-class recall vs ε
O5	RQ3: How does non-IID data affect the global model?	H3: Performance decreases as Dirichlet α decreases; per-client variance increases.	Macro-F1, per-client F1 spread
O4	RQ4: What communication and computation are required?	H4: A lightweight model reduces traffic proportionally to its parameter count.	Bytes exchanged, rounds, time
O5	RQ5: How robust is aggregation to poisoned updates?	H5: Robust aggregation limits degradation under a minority of malicious clients.	Macro-F1 and attack success rate

4. Proposed Framework

4.1 Architectural Overview

Fig. 1 shows the proposed architecture. Heterogeneous IoT clients—for example a smart-home gateway, an industrial IoT edge node and a healthcare IoT hub—each hold local traffic, run a preprocessing pipeline and train a local IDS model. Before transmission, each update passes through a privacy layer. Protected updates are sent over authenticated, encrypted channels to a federated server that performs secure aggregation and produces the global IDS model. The updated model is redistributed to all clients, which use it for local detection and as the starting point for the next round. Raw traffic never leaves the client.

 ISSN: 3139-1141

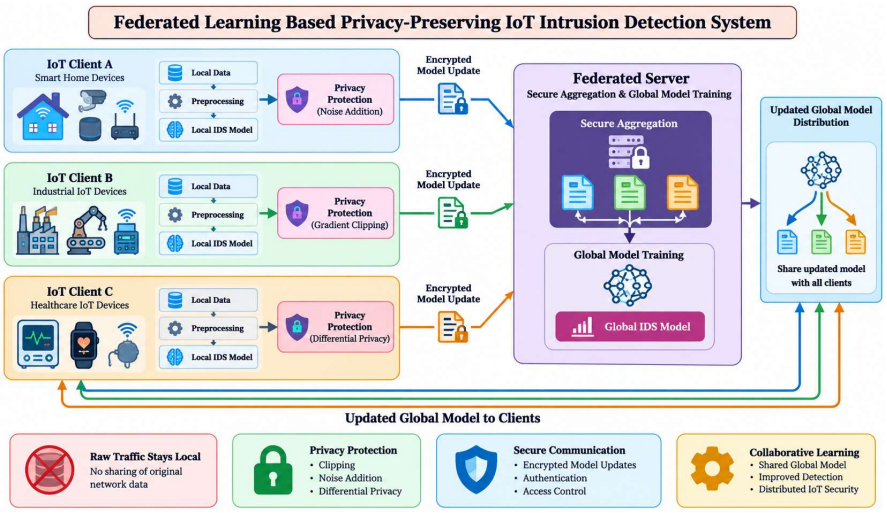

Fig. 1. Proposed federated learning based privacy-preserving IoT intrusion detection framework. Clients retain raw traffic, preprocess it and train a local IDS model; a privacy layer (clipping, noise addition, differential privacy) protects each update; the server securely aggregates encrypted updates and redistributes the global model.

4.2 Client Layer and Local Preprocessing

Clients are devices, gateways or edge nodes with sufficient resources to train a small model; constrained end devices are represented by their gateway. Local preprocessing removes duplicate and unusable records, imputes or drops missing values, encodes categorical protocol and service fields, and scales numerical features. To keep clients consistent without sharing data, the feature schema, encoding dictionaries and scaling method are fixed globally in advance, and scaling statistics are computed locally. Labels are mapped to a common taxonomy of normal traffic and attack categories.

4.3 Lightweight Local IDS Model

Because update size and on-device computation scale with model parameters, the default local model is a compact multilayer perceptron (MLP) over flow features, with a 1-D convolutional network as a higher-capacity alternative. A representative MLP with 46 inputs, hidden layers of 64 and 32 units and an 8-class softmax output contains 5,352 parameters (about 21 KB in 32-bit floating point). Model size, in parameters and serialised bytes, is reported as a first-class result.

4.4 Privacy Layer

Each selected client k computes its update $\Delta_k = w_k^{t+1} - w^t$, clips it to L2 norm C and, when client-level differential privacy is enabled, adds Gaussian noise [15], [16]:

$$\bar{\Delta}_k = \Delta_k \cdot \min(1, C / \|\Delta_k\|_2) \quad (1)$$

$$\tilde{\Delta}_k = \bar{\Delta}_k + N(0, \sigma^2 C^2 I) \quad (2)$$

Clipping bounds each client's influence, which is also useful against poisoned updates; the noise multiplier σ, sampling rate q and number of rounds R determine the cumulative (ε, δ) guarantee, which is computed with a privacy accountant and reported for every configuration [15], [21]. Noise may be added locally or, when combined with secure aggregation, distributed across clients so that the aggregate carries the required noise.

4.5 Secure Aggregation and Federated Server

To prevent the server from inspecting individual updates, clients mask them with pairwise random values s_{ij} agreed through key exchange [18]:

$$y_k = \tilde{\Delta}_k + \Sigma_{j>k} s_{kj} - \Sigma_{j<k} s_{jk} \quad (3)$$

The masks cancel in the sum, so the server recovers only the aggregate and updates the global model as

$$w^{t+1} = w^t + (\eta_g / |S_t|) \Sigma_{k \in S_t} y_k \quad (4)$$

where η_g is the server learning rate and S_t the set of participating clients. Uniform weighting is used with client-level DP so that sensitivity remains bounded; data-size weighting as in FedAvg [3] is used in the non-private baseline. Secret sharing of mask seeds allows aggregation to complete when some clients drop out. When robust aggregation is evaluated, the masked sum is replaced by a robust statistic computed at a trusted aggregation point, and this trust assumption is stated explicitly.

4.6 Federated Training Procedure

Fig. 2 shows the training sequence, and Algorithm 1 gives the complete procedure. Each round consists

 ISSN: 3139-1141

of global model distribution, local training, update protection, secure aggregation and global update; the process repeats until convergence or a fixed number of rounds.

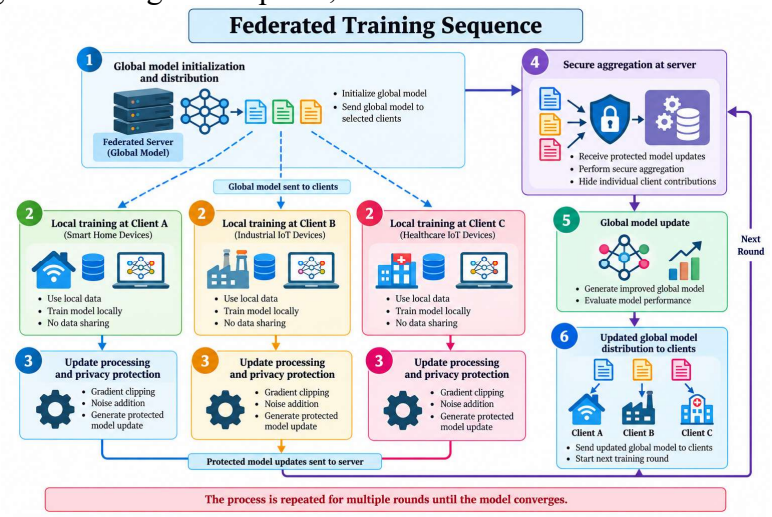

Fig. 2. Federated training sequence: (1) global model initialisation and distribution, (2) local training at clients, (3) update processing and privacy protection, (4) secure aggregation at the server, (5) global model update and (6) redistribution for the next round.

Algorithm 1: Privacy-Preserving Federated IDS Training

Input: clients $k = 1 \ldots K$ with local data D_k; rounds R; sampling rate q; local epochs E; clip norm C; noise multiplier σ; server rate η_g
Output: global IDS model w^R
Server initialises w^0; all clients agree on feature schema and label taxonomy
for $t = 0, 1, \ldots, R - 1$ **do**
  Server samples client set S_t with probability q; sends w^t over an authenticated, encrypted channel
  for each client $k \in St$ **in parallel do**
    $w_k \leftarrow w^t$; train E local epochs on preprocessed D_k (class-weighted loss)
    $\Delta_k \leftarrow w_k - w^t$; clip to norm C (Eq. 1); add noise $N(0, \sigma^2 C^2 I)$ (Eq. 2)
    Mask update with pairwise seeds (Eq. 3); send y_k to server
  end for
  Server sums masked updates, recovers dropped clients' masks, updates w^{t+1} (Eq. 4)
  Log round time, bytes exchanged, participating clients and privacy accountant state
end for
Distribute w^R to all clients for local intrusion detection

5. Methodology and Experimental Design

5.1 Research Workflow

The research follows the workflow in Fig. 3: local data collection, preprocessing and feature selection, local IDS training, privacy protection, secure aggregation, global model distribution and local detection and evaluation, repeated over multiple training rounds. Every step records its settings so that experiments can be reproduced.

 ISSN: 3139-1141

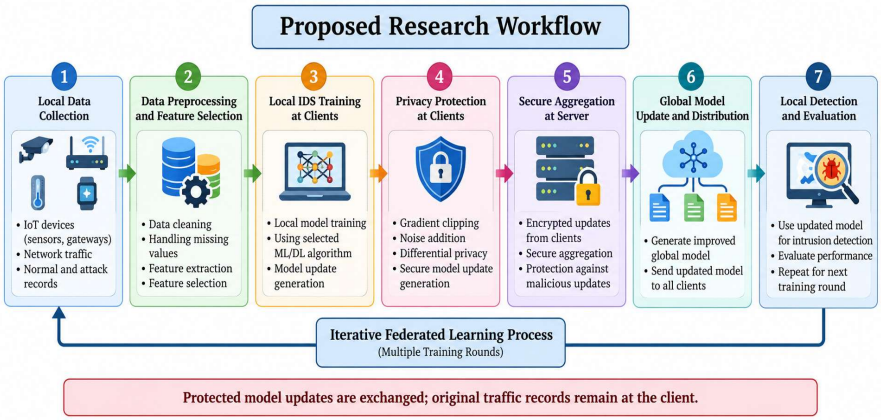

Fig. 3. Research workflow of the proposed framework. Protected model updates are exchanged while original traffic records remain at the client; the process iterates over multiple federated rounds.

5.2 Dataset Selection

Datasets are selected using five criteria: IoT relevance of devices and protocols; presence of both normal traffic and multiple attack categories; sufficient size for partitioning across clients; documented labels and features; and public availability for reproducibility. Table II lists the candidate datasets. The final choice, version and download date are recorded, and at least one experiment is repeated on a second dataset to test generalisation.

TABLE II CANDIDATE IOT INTRUSION DETECTION DATASETS

Dataset	Characteristics	Role in this study
CICIoT2023 [22]	Large-scale IoT testbed of 105 devices; 33 attacks in seven categories (e.g., DDoS, DoS, recon, web, brute force, spoofing, Mirai)	Primary candidate; multi-class detection and non-IID partitioning
TON_IoT [23]	Telemetry, operating-system and network traffic from an IoT/IIoT testbed with attacks such as scanning, DoS, DDoS, ransomware, backdoor, injection, XSS, password and MITM	Heterogeneous sources; natural client grouping
IoT-23 [24]	Labelled captures of malware and benign traffic from real IoT devices	Malware and botnet traffic; generalisation test
Edge-IIoTset [25]	IoT/IIoT testbed with 14 attacks in five threat categories; designed for centralised and federated learning	Alternative candidate for IIoT clients

5.3 Preprocessing and Feature Groups

Identifiers that would allow trivial memorisation—source and destination IP addresses, timestamps and flow identifiers—are removed. Remaining features fall into four groups (Table III). Feature selection, if applied, uses only training data, and removed features with their reasons are documented. The training, validation and test split is made before partitioning so that the test set is identical for all configurations.

TABLE III FEATURE CATEGORIES USED FOR INTRUSION DETECTION

Feature group	Examples	Information captured
Flow statistics	Duration, packet counts, byte counts, header lengths	Volume and shape of traffic
Protocol features	Protocol type, service, TCP flag counts	Connection context
Direction features	Forward/backward packet and byte ratios	Flow relationship between endpoints
Timing features	Packet rate, inter-arrival time statistics	Rate anomalies such as floods and scans

 ISSN: 3139-1141

5.4 Client Formation and Non-IID Modelling

Clients are formed as logical partitions that represent different IoT environments (Fig. 4). In the IID setting, records are shuffled and divided equally. In the non-IID setting, label skew follows the Dirichlet scheme [20]: for each client k, class proportions are drawn as

$$pk \sim Dir(\alpha \bullet 1M), \quad k = 1, \ldots, K \tag{5}$$

where M is the number of classes and α controls heterogeneity. Large α approaches IID data, while small α gives each client only a few dominant classes. Quantity skew is added by drawing client sizes from a log-normal distribution. Fig. 5 illustrates partitions produced by this procedure for ten clients and five traffic classes at α = 100, 1 and 0.1. Partition seeds and the resulting class counts per client are published with the results.

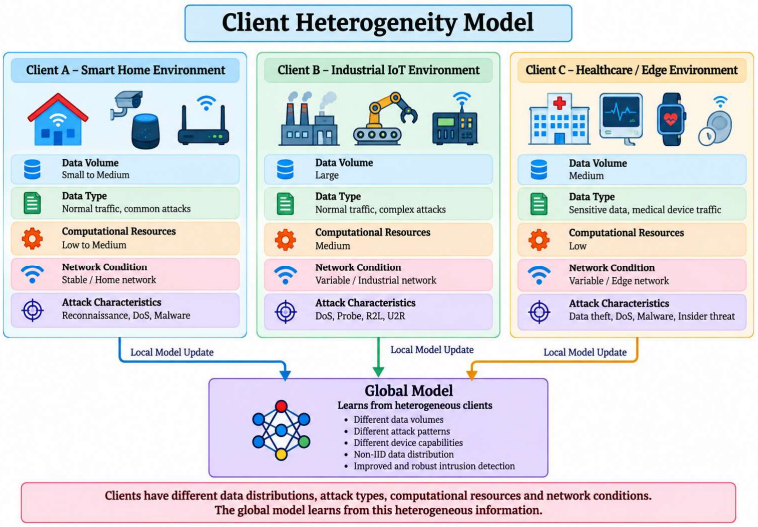

Fig. 4. Client heterogeneity model: smart-home, industrial IoT and healthcare/edge clients differ in data volume, data type, computational resources, network condition and attack characteristics; the global model learns from these heterogeneous clients.

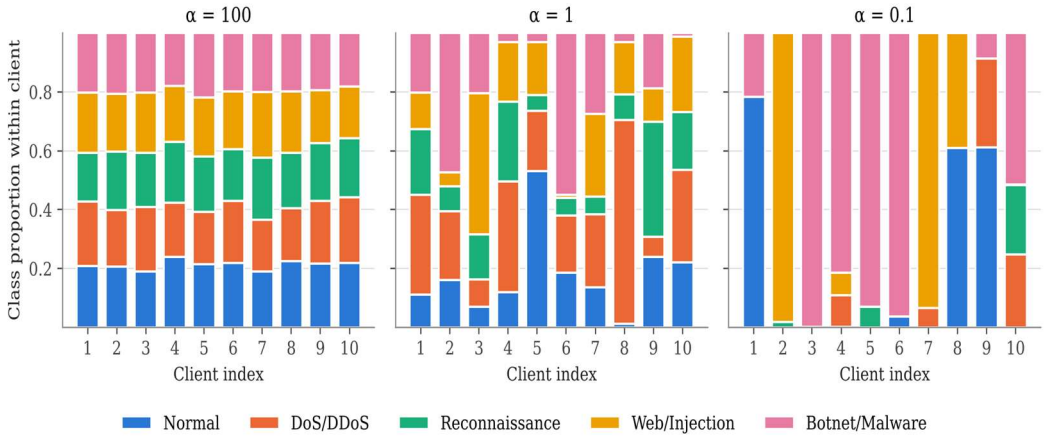

Fig. 5. Illustration of Dirichlet label-skew partitioning for ten clients and five traffic classes (random seed 42). As α decreases from 100 to 0.1, clients move from near-identical class mixes to being dominated by one or two classes. The plot shows the partitioning procedure only, not experimental results.

5.5 Experimental Configurations and Scenarios

Three configurations are compared under identical data, splits, model architecture and test set: (i) a centralised baseline trained on pooled training data, which serves as a reference rather than a deployment target; (ii) standard FL with FedAvg and no update protection; and (iii) privacy-preserving FL with clipping, differential privacy and secure aggregation. Each configuration is run under the scenarios in Table IV. Planned parameter ranges are as follows: $K \in \{10, 20, 50\}$clients; participation rate $q \in \{0.2, 0.5, 1.0\}$; $E \in \{1, 5\}$local epochs; up to $R = 100$ rounds with early stopping on validation macro-F1; α ∈ {100, 1, 0.5, 0.1}; target ε ∈ {1, 4, 8, ∞} at $\delta = 10^{-5}$; and clip

norm C tuned on the validation set. FedProx [19] is included as a heterogeneity-aware variant of configuration (ii).

TABLE IV DISTRIBUTED EXPERIMENTAL SCENARIOS

ID	Client condition	Purpose	Varied parameter
A	Relatively balanced (IID) partitions	Baseline distributed condition	K
B	Non-IID attack mix and quantity skew	Heterogeneity analysis	α
C	Partial client participation per round	Availability analysis	q
D	Random client dropout during rounds	Robustness to missing clients	Dropout rate
E	A minority of clients submit poisoned updates	Learning-process robustness	Fraction of malicious clients

5.6 Threat Model for Robustness Experiments

The server is assumed honest-but-curious: it follows the protocol but may try to infer client data from what it observes. Communication channels are authenticated and encrypted. In Scenario E, up to 20% of clients are compromised and send manipulated updates—label-flipping (attack records relabelled as normal) and scaled model-replacement updates [12], [26]. Defences evaluated are norm clipping and coordinate-wise median or trimmed-mean aggregation [27]. Attacks outside this model, such as server compromise or side-channel leakage, are explicitly out of scope.

6. Evaluation Protocol

6.1 Detection Metrics

With true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN), where an attack is the positive class, the following metrics are computed:

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN)$$

$$\text{Precision} = TP / (TP + FP), \quad \text{Recall} = TP / (TP + FN) \tag{7}$$

$$F1 = 2 \cdot \text{Precision} \cdot \text{Recall} / (\text{Precision} + \text{Recall}),$$

$$FPR = FP / (FP + TN) \tag{8}$$

Because IoT intrusion data is strongly imbalanced, macro-averaged F1 and per-class recall are the primary measures, and accuracy is reported only for completeness. The false-positive rate is reported because false alarms directly limit the operational value of an IDS. ROC-AUC and precision–recall curves are both reported, since precision–recall analysis is more informative under class imbalance [28]. Confusion matrices are given for the global model and for selected clients, and per-client macro-F1 is summarised by its mean, minimum and spread.

6.2 Communication and Resource Metrics

For a model with P parameters encoded with b bytes each, K participating clients per round and R rounds, the total traffic of plain federated averaging (download plus upload) is

$$C_{total} = 2 \cdot R \cdot K \cdot P \cdot b \tag{9}$$

Secure aggregation adds key-agreement messages that scale with the number of clients, and these are measured directly. Fig. 6 applies the model to three model sizes with $K = 10$ and $b = 4$: after 100 rounds the lightweight MLP needs about 43 MB in total, compared with 400 MB for a 50 k-parameter CNN and 4 GB for a 500 k-parameter model. This makes model size a primary design variable for IoT links. Local training time, aggregation time, total wall-clock time and serialised model size are also recorded.

(6)

 ISSN: 3139-1141

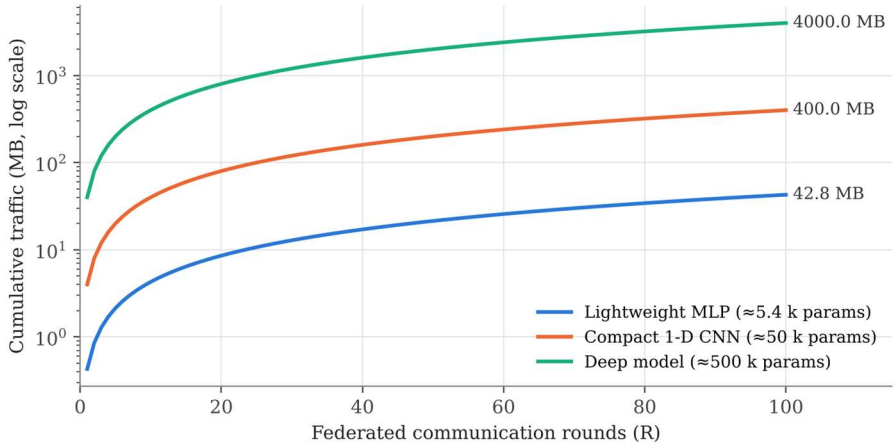

Fig. 6. Analytic estimate of cumulative federated communication (upload plus download) for ten clients and 32-bit parameters as a function of rounds, for three model sizes (log scale). Values follow Eq. (9) and exclude protocol overhead; they are design estimates, not measurements.

6.3 Privacy Reporting

For every privacy-preserving run, the clip norm C, noise multiplier σ, sampling rate q, number of rounds and resulting (ε, δ) are reported, together with the level at which privacy is applied (client or record). Privacy–utility curves plot macro-F1 and minority-class recall against ε, so that a configuration can be chosen for a required level of protection rather than assuming that the strongest setting is best.

6.4 Ablation and Statistical Comparison

An ablation study adds components one at a time—FL only, FL with clipping, FL with clipping and DP, and the full configuration with secure aggregation—to attribute changes in utility and cost to individual mechanisms. Every configuration is run with at least five random seeds, and results are reported as mean ± standard deviation. Differences between paired configurations are tested with the Wilcoxon signed-rank test, and comparisons across several configurations with the Friedman test and a post-hoc procedure, at a significance level of 0.05. The eight evaluation dimensions are summarised in Fig. 7. Implementation uses an open FL framework such as Flower [29], and code, seeds, library versions and hardware details are released with the results.

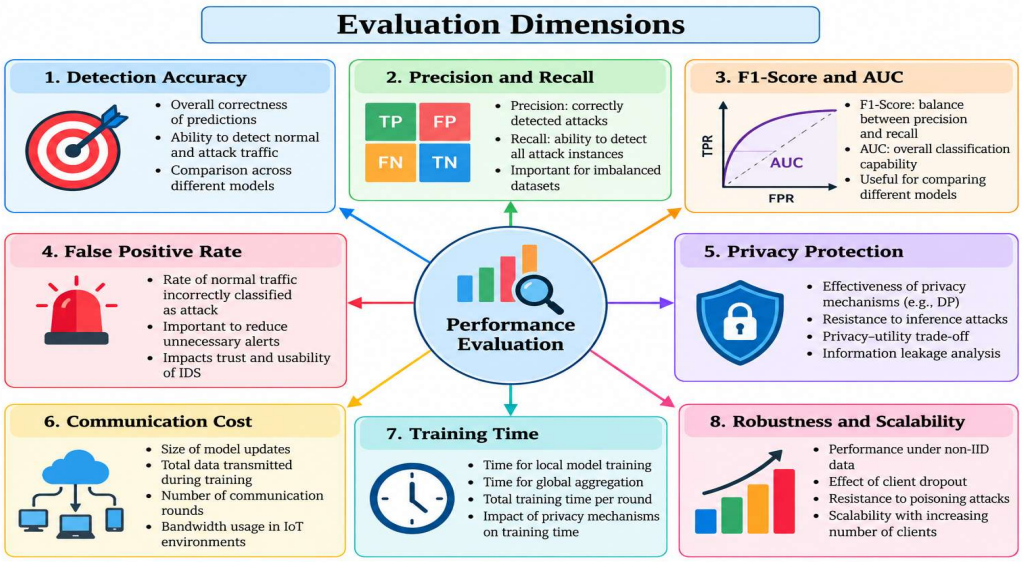

Fig. 7. Evaluation dimensions of the framework: detection accuracy, precision and recall, F1-score and AUC, false-positive rate, privacy protection, communication cost, training time, and robustness and scalability.

7. Discussion

7.1 Expected Trade-offs

Several trade-offs are anticipated and are formulated as testable hypotheses rather than claims. Differential privacy should reduce utility as ε decreases, and the reduction is expected to be largest for rare attack classes whose gradient signal is weak relative to the added noise [17]. Non-IID data is expected to slow convergence and widen the gap between the best and worst clients [7], [11]; FedProx-style regularisation may narrow this gap at little extra cost. Clipping may simultaneously stabilise training and limit poisoning, so its effect is reported in both the privacy and robustness analyses. Communication cost grows linearly with model size, rounds and clients, which favours compact models and fewer, longer local training phases, within the limits set by client drift.

7.2 Deployment Considerations

In practice, training is best placed on gateways and edge nodes, with constrained devices contributing traffic to their gateway. A hierarchical arrangement—aggregation at site gateways followed by global aggregation—reduces wide-area traffic and aligns with organisational boundaries. Secure aggregation must tolerate intermittent clients, and the privacy budget must be managed across the full training lifetime, including any retraining as new attacks appear.

7.3 Limitations and Ethical Considerations

Logical partitions of public datasets approximate but do not reproduce real multi-site deployments, and simulated communication omits some network effects. Differential privacy protects against defined inference risks but not every possible attack, and secure aggregation assumes an honest-but-curious server. Robustness is evaluated only for the threat model stated in Section 5.6. All experiments use publicly available, anonymised datasets in an isolated environment; no live networks are attacked, and attack simulations are limited to manipulated model updates within the testbed.

8. Conclusion and Future Work

This paper presented a federated learning based privacy-preserving intrusion detection framework for heterogeneous IoT environments together with a reproducible evaluation protocol. The framework keeps raw traffic at the client, trains a lightweight local IDS model, protects updates through clipping, differential privacy, secure aggregation, encryption and authentication, and aggregates them into a global detection model. The protocol specifies candidate datasets, preprocessing, Dirichlet-based non-IID partitioning, five distributed scenarios, three directly comparable configurations and a multidimensional set of detection, privacy, communication and robustness measures with statistical testing. Its purpose is to make the balance between detection quality, privacy and resource cost observable and comparable. Future work will report the experimental results obtained with this protocol, extend it to hierarchical and personalised federated learning, and investigate aggregation methods that remain robust to poisoning while preserving secure-aggregation guarantees.

References

[1] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, “DDoS in the IoT: Mirai and other botnets,” *Computer*, vol. 50, no. 7, pp. 80–84, 2017, doi: 10.1109/MC.2017.201.

[2] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, “Network intrusion detection for IoT security based on learning techniques,” *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2671–2701, 2019, doi: 10.1109/COMST.2019.2896380.

[3] 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. 20th Int. Conf. Artif. Intell. Statist. (AISTATS)*, Fort Lauderdale, FL, USA, 2017, pp. 1273–1282, doi: 10.48550/arXiv.1602.05629.

[4] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, “DÏoT: A federated self-learning anomaly detection system for IoT,” in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Dallas, TX, USA, 2019, pp. 756–767, doi: 10.1109/ICDCS.2019.00080.

[5] S. A. Rahman, H. Tout, C. Talhi, and A. Mourad, “Internet of Things intrusion detection: Centralized, on-device, or federated learning?,” *IEEE Netw.*, vol. 34, no. 6, pp. 310–317, 2020, doi: 10.1109/MNET.011.2000286.

[6] S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh, and O. Jogunola, “Federated

deep learning for zero-day botnet attack detection in IoT-edge devices," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3930–3944, 2022, doi: 10.1109/JIOT.2021.3100755.

[7] E. M. Campos *et al.*, "Evaluating federated learning for intrusion detection in Internet of Things: Review and challenges," *Comput. Netw.*, vol. 203, Art. no. 108661, 2022, doi: 10.1016/j.comnet.2021.108661.

[8] A. Vyas, P.-C. Lin, R.-H. Hwang, and M. Tripathi, "Privacy-preserving federated learning for intrusion detection in IoT environments: A survey," *IEEE Access*, vol. 12, pp. 127018–127050, 2024, doi: 10.1109/ACCESS.2024.3454211.

[9] A. Belenguer, J. A. Pascual, and J. Navaridas, "A review of federated learning applications in intrusion detection systems," *Comput. Netw.*, vol. 258, Art. no. 111023, 2025, doi: 10.1016/j.comnet.2024.111023.

[10] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, San Francisco, CA, USA, 2019, pp. 691–706, doi: 10.1109/SP.2019.00029.

[11] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," 2018, *arXiv:1806.00582*, doi: 10.48550/arXiv.1806.00582.

[12] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. 23rd Int. Conf. Artif. Intell. Statist. (AISTATS)*, 2020, pp. 2938–2948, doi: 10.48550/arXiv.1807.00459.

[13] V. Mothukuri, P. Khare, R. M. Parizi, S. Pouriyeh, A. Dehghantanha, and G. Srivastava, "Federated-learning-based anomaly detection for IoT security attacks," *IEEE Internet Things J.*, vol. 9, no. 4, pp. 2545–2554, 2022, doi: 10.1109/JIOT.2021.3077803.

[14] S. Agrawal *et al.*, "Federated learning for intrusion detection system: Concepts, challenges and future directions," *Comput. Commun.*, vol. 195, pp. 346–361, 2022, doi: 10.1016/j.comcom.2022.09.012.

[15] M. Abadi *et al.*, "Deep learning with differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, Vienna, Austria, 2016, pp. 308–318, doi: 10.1145/2976749.2978318.

[16] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," 2017, *arXiv:1712.07557*, doi: 10.48550/arXiv.1712.07557.

[17] 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.

[18] K. Bonawitz *et al.*, "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, Dallas, TX, USA, 2017, pp. 1175–1191, doi: 10.1145/3133956.3133982.

[19] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proc. Mach. Learn. Syst. (MLSys)*, vol. 2, 2020, pp. 429–450, doi: 10.48550/arXiv.1812.06127.

[20] T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," 2019, *arXiv:1909.06335*, doi: 10.48550/arXiv.1909.06335.

[21] C. Dwork, "Differential privacy," in *Proc. 33rd Int. Colloq. Automata, Lang. Program. (ICALP)*, Venice, Italy, 2006, pp. 1–12, doi: 10.1007/11787006_1.

[22] 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. no. 5941, 2023, doi: 10.3390/s23135941.

[23] N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets," *Sustain. Cities Soc.*, vol. 72, Art. no. 102994, 2021, doi: 10.1016/j.scs.2021.102994.

[24] S. Garcia, A. Parmisano, and M. J. Erquiaga, "IoT-23: A labeled dataset with malicious and benign IoT network traffic," Zenodo, 2020, doi: 10.5281/zenodo.4743746.

[25] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, "Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning," *IEEE*

Access, vol. 10, pp. 40281–40306, 2022, doi: 10.1109/ACCESS.2022.3165809.

[26] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," in *Proc. 29th USENIX Secur. Symp.*, 2020, pp. 1605–1622, doi: 10.48550/arXiv.1911.11815.

[27] D. Yin, Y. Chen, K. Ramchandran, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proc. 35th Int. Conf. Mach. Learn. (ICML)*, Stockholm, Sweden, 2018, pp. 5650–5659, doi: 10.48550/arXiv.1803.01498.

[28] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," in *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*, Pittsburgh, PA, USA, 2006, pp. 233–240, doi: 10.1145/1143844.1143874.

[29] D. J. Beutel *et al.*, "Flower: A friendly federated learning research framework," 2020, *arXiv:2007.14390*, doi: 10.48550/arXiv.2007.14390.