International Journal of Safety and Security Engineering
Vol. 16, No. 7, July, 2026, pp. 1485-1498
Journal homepage: http://iieta.org/journals/ijsse

FedSecure: Robust and Privacy-Preserving Intrusion Detection for IoT Networks Using Federated Learning under Poisoning Attacks and Non-IID Data

Noor Redha Alkazaz[1,2]

[1] Department of Computer Science, College of Science for Women, University of Baghdad, Baghdad 10071, Iraq
[2] Biomedical Applications Department, College of Artificial Intelligence, University of Baghdad, Baghdad 10071, Iraq

Corresponding Author Email: noor.alkazaz@csw.uobaghdad.edu.iq

https://doi.org/10.18280/ijsse.160707

Received: 4 April 2026
Revised: 24 June 2026
Accepted: 29 June 2026
Available online: 31 July 2026

Keywords:
intrusion detection system, IoT security, federated learning, poisoning attacks, Non-IID data

ABSTRACT

The proliferation of Internet of Things (IoT) devices—including in safety-critical domains such as smart grids and industrial control systems—has expanded the cyber-attack surface, necessitating effective intrusion detection systems (IDS) that also preserve data privacy. Federated learning (FL) enables collaborative model training without sharing raw data, making it a promising basis for privacy-preserving intrusion detection. However, traditional FL algorithms struggle in IoT environments due to two primary challenges: (a) vulnerability to poisoning attacks from malicious clients, and (b) statistical heterogeneity (non-independent and identically distributed, Non-IID) of traffic data across devices. This paper introduces FedSecure, a robust and privacy-preserving intrusion detection approach for IoT networks that leverages FL to address both challenges. FedSecure employs a three-phase server-side aggregation mechanism. First, it uses density-based spatial clustering of applications with noise (DBSCAN) to filter out malicious updates. Second, it applies an adaptive weighting scheme to mitigate client drift caused by Non-IID data distributions. We rigorously evaluated FedSecure using the CIC-IoT2023 and ToN-IoT datasets under various Non-IID partitions (Dirichlet α = 0.1, 0.5, 1.0) and poisoning attacks (label flipping, backdoor, Gaussian noise) with up to 40% malicious clients. Results demonstrate that FedSecure achieves over 90% detection rate even with 30% malicious clients under high data heterogeneity, substantially outperforming robust aggregation methods like Krum and Trimmed Mean. Under extreme conditions combining severe label skew and 40% malicious clients, FedSecure maintains an F1-score of 0.70, compared to 0.35 for FedProx and 0.43 for Krum. FedSecure provides a practical, privacy-preserving, and resilient intrusion detection solution suitable for real-world, heterogeneous IoT deployments.

1. INTRODUCTION

The rapid growth of Internet of Things (IoT) devices has created new forms of critical infrastructure that includes smart grids, autonomous vehicles, and other types of connected devices. With these developments comes an ever-increasing attack surface, as well as the loss of effectiveness of traditional network perimeter defenses [1, 2]. An intrusion detection system (IDS) is a necessity for detecting malicious behaviour in this area; however, conventional IDS depend on signature matching, which makes them ineffectual for detection of new or novel (zero-day) exploits [3]. Machine Learning (ML) based IDS offer enhanced flexibility and adaptability, but the fact that they operate in a centralized manner creates violations of data privacy regulations (e.g., GDPR) and endangers the security of sensitive information [4].

These risks are most severe in safety-critical security-engineering settings. In a smart grid, a compromised smart meter could send falsified telemetry to conceal an ongoing attack; in an industrial control system (ICS/SCADA), a corrupted detection model could suppress alarms for malicious commands sent to programmable logic controllers. Such environments contain many distributed, privacy-sensitive monitoring nodes, and no single compromised participant should be able to reduce detection across the entire network-precisely the combined heterogeneity-and-poisoning regime that FedSecure addresses.

Federated learning (FL) [5] provides a solution to this dilemma because it provides collaborative model training and does not require sharing of raw data. However, traditional FL (e.g. FedAvg) has limitations in relation to its use in IoT environments due to a couple of key issues: (1) statistical heterogeneity (i.e. non-independent and identically distributed (Non-IID) data) between devices causes client drift, which can lead to the degradation of the global model [6]; and (2) vulnerability to poisoning attacks by malicious clients who submit corrupted updates to the FL model, which can lead to reduced performance/quality of the model or to the installation of a backdoor into the model [7].

Current architectures address each problem in isolation. Effective techniques for aggregating data include the Krum algorithm [8] and Trimmed Mean [9]. They can reject

maliciously created updates, but they will also often classify as outliers misclassified benign Non-IID updates. Conversely, methods that are designed to accommodate heterogeneous behaviors, such as FedProx [6], offer stability from nondiscriminatory or "Non-IID" behavior in training, but they do not mitigate the effects of adversaries' poisoning. Solutions that can handle both the extreme data heterogeneity present in many real-world applications and the presence of active adversaries have yet to be found.

The core innovation of FedSecure is that it integrates two previously separate defense mechanisms into a single server-side aggregation step. It first uses density-based clustering to isolate malicious updates, then reweights the remaining benign updates according to their representativeness. This unified design achieves both poisoning robustness and Non-IID drift mitigation, and requires neither a trusted reference dataset nor prior knowledge of the number of attackers. This translates into a clear empirical advantage**:** under the most challenging combined setting ($\alpha = 0.1$, $\rho = 0.4$), FedSecure sustains an F1-score of 0.70, compared with 0.35 for FedProx and 0.43 for Krum (Section 4.3).

In this paper, we introduce FedSecure, a robust and privacy-preserving intrusion detection approach for IoT networks, built on FL, that overcomes the two main challenges of poisoning attacks and statistical heterogeneity (Non-IID data).

The main contributions of this work are outlined below:

(1) A server-side aggregation mechanism that utilizes density-based spatial clustering of applications with noise (DBSCAN) to filter out any malicious updates through the use of gradient similarity metrics.
(2) An adaptive weighting scheme that helps to reduce the amount of client drift caused by Non-IID data distributions.
(3) A complete evaluation framework consisting of Dirichlet-based Non-IID partitions and various poisoning attacks on both the CIC IoT2023 and ToN IoT datasets.
(4) Extensive experimental results showing that FedSecure far exceeds the performance of current state-of-the-art FL algorithms (FedAvg, FedProx, Krum, Trimmed Mean) when evaluated in a combined heterogeneous and attack scenario, together with a qualitative comparison against recent federated IDS approaches (FedMADE [10], AGAT-FL [11], FedKD-IDS [12], and FLTrust [13]) demonstrating the limitations of existing methods in addressing both challenges simultaneously.

The paper is organized as follows: Section 2 provides a review of related works. Section 3 describes in detail the system model, threat model, FedSecure algorithm. and the experimental setup. Section 4 presents and analyzes results. Section 5 discusses the implications of these results and their limitations. Finally, Section 6 provides an overall conclusion along with possible future research direction.

2. LITERATURE REVIEW

The field of incident detection has progressed from traditional rule-based methods to AI frameworks capable of detecting incidents and events in near real time. The traditional methods can be classified into two main categories; Signature-based methods, which can only detect the known threats, and Anomaly-based methods, which theoretically can detect new (zero-day) attacks but generate high volumes of false positives when detections are made in dynamic environments like the IoT [3]. ML techniques have been developed to address the limitations of traditional incident detection methods. Deep learning algorithms (such as CNNs [14], LSTMs [15], and Kitsune (autoencoder ensemble) [16]) achieve a very high level of accuracy but have privacy concerns (because aggregated centralized data is required) that violate data protection regulations such as GDPR [4].

Therefore, an alternative means of achieving privacy-preserving intrusion detection (through FL) [5] is to train a global model without exchanging any raw data. FL uses the training model from a plurality of clients and averages the updates received from them. Early work in demonstrable FL demonstrated that, although successful, it was only based upon a limited set of experimental conditions [17]. Two of the most significant challenges to the widespread implementation of FL for intrusion detection in real-world IoT implementations are statistical heterogeneity (Non-IID data) and poisoning attacks.

In an IoT environment, data distributions across clients are often highly unbalanced and can create client drift. As a solution, a proximal term was introduced in FedProx to help stabilize training [6], while SCAFFOLD uses control variates to correct drift [18]. FedBN was subsequently developed using personalized batch normalization [19]. Other recent developments in this area include personalized FL [20] and contrastive learning [21]. Many of these advancements have been developed for benign data and lack the ability to prevent attacks that are maliciously manipulated.

FL is particularly vulnerable to data and model poisoning attacks, including backdoor attacks [22]. Krum [8] and Trimmed Mean [9] have been proposed as robust aggregation algorithms to filter out the contribution of malicious updates, while FoolsGold [23] has been proposed as a defense against Sybil attacks. Additionally, there have been game theory [24] and layerwise [25] defenses proposed. However, these robust algorithms also erroneously classify the benign Non-IID updates as malicious, and this may cause FL systems to perform poorly when operating in highly heterogeneous environments.

There is an existing gap in the literature regarding the lack of poisoning defense from the distributed algorithms created to accommodate Non-IID data (i.e., FedProx, SCAFFOLD, FedBN). Likewise, the available robust aggregation techniques (Krum, Trimmed Mean, FoolsGold) remain impractical due to their ineffectiveness in the face of extreme data heterogeneity.

There are efforts under way to develop solutions for these problems. FedMADE [10] introduces dynamic aggregation using client scoring; however, it has only been verified with the CIC IDS 2017 and UNSW NB15 datasets and does not apply to modern IoT datasets. Similarly, AGAT FL [11] employs the graph attention mechanism but has only been validated with ToN IoT and does not take extreme label skew into consideration. FedKD IDS [12], which combines knowledge distillation and semi-supervised learning, also has a scalable solution that is more complex to implement than other methods due to adding structural variation via Non-IID. FLTrust [13] requires that a trusted dataset be present on the server but this may be impractical in an IoT environment. PEIoT DS provides differential confidentiality but does not actively defend against poisoning and has only been tested with Bot IoT.

Beyond these frameworks, the period from 2022 onward

has seen growing attention to federated intrusion detection under realistic IoT constraints, although the two core challenges of data heterogeneity and adversarial robustness are still rarely addressed together. On the heterogeneity side, Bouzinis et al. [20] proposed StatAvg, in which clients share local feature statistics so that the server can compute global statistics for universal normalization, thereby reducing the effect of Non-IID features. A complementary line of work has prioritized efficiency and practicality for resource-constrained devices, including a contrastive-learning approach with knowledge distillation for efficient federated detection [21], few-shot federated detection for constrained IoT nodes [26], and lightweight federated network IDS designs [27]. Recent surveys continue to identify Non-IID data as a primary obstacle to reliable deployment [28], while efficiency-oriented frameworks such as the autoencoder-based Fed-ANIDS [29] further address practical deployment. On the security side, robust schemes such as SecFedNIDS [25] and the two-phase poisoning defense of Lai et al. [30] — conceptually related to the multi-phase aggregation adopted in this work — were developed to counter malicious clients.

In short, almost all previous works are either focused on heterogeneous or benign data especially at the extreme ends, assume that data is normally distributed/benign, or that all validations are made against only one dataset. FedSecure addresses the relevant knowledge gaps between prior research and practice through the development of a two-step aggregation process that includes a methodology for detecting outliers using DBSCAN and adaptive weighting, which has been validated through two complementary benchmarks (CIC IoT 2023 and ToN IoT); thus, providing a complete strategy for both of these problems.

Table 1 summarizes the capabilities and limitations of representative FL approaches with respect to Non-IID data and poisoning, situating FedSecure within this landscape.

Table 1. Comparative summary of existing federated learning (FL) approaches and the proposed FedSecure

Algorithm / Approach	Primary Focus	Key Mechanism	Limitation(s)	Handles Non-IID?	Handles Poisoning?
FedAvg [5]	Baseline FL	Simple averaging	Poor convergence under Non-IID; No poisoning defense	No	No
FedProx [6]	Non-IID Data	Proximal term	No poisoning defense; May not handle extreme feature heterogeneity	Yes	No
SCAFFOLD [18]	Non-IID Data	Control variates	High overhead; No poisoning defense	Yes	No
FedBN [19]	Feature Non-IID	Personalized batch norm	Effective only for feature shift; No poisoning defense	Yes	No
Krum [8]	Poisoning Attacks	Selects update closest to neighbors	High computational cost; Fails under high Non-IID	No	Yes
Trimmed Mean [9]	Poisoning Attacks	Coordinate-wise trimming	Performance degrades under Non-IID	No	Yes
FoolsGold [23]	Poisoning Attacks	Penalizes historically similar updates	Confounded by natural diversity in Non-IID data	No	Yes
FedMADE [10]	Poisoning Attacks	Dynamic aggregation based on contribution scoring	Single dataset validation; Limited Non-IID handling	Partial	Yes
AGAT-FL [11]	Poisoning Attacks	Adaptive graph attention	Single dataset validation; Does not address extreme label skew	Partial	Yes
FedKD-IDS [12]	Poisoning Attacks	Knowledge distillation + semi-supervised learning	High complexity; Single dataset validation	Partial	Yes
FLTrust [13]	Poisoning Attacks	Trust scores from reference model	Requires trusted data; No Non-IID handling	No	Yes
PEIoT-DS [2]	Privacy Preservation	Differential privacy	Privacy focus only; No active poisoning defense	No	No
FedSecure (Proposed)	Both Challenges	DBSCAN filtering+Adaptive weighting	Computational overhead on server	Yes	Yes

3. METHODOLOGY

3.1 System model

The system follows a canonical FL architecture adapted for IoT. A central server coordinates N IoT clients $C = \{c_1, \dots, c_N\}$, each holding a local dataset D_i that never leaves the client (GDPR compliance). Training proceeds in communication rounds: at round t, a random subset of clients receives the global model w_t, performs local training, and sends back model updates $\Delta_i = w_{\text{it+1}} - w_t$. The server aggregates these updates to produce w_{t+1} . This process repeats until convergence or a fixed number of rounds (Figure 1).

3.2 Threat model

There are two types of adversarial goals that are defined in the research conducted by Bhagoji et al. [7]. One of these types is described as "untargeted" and aims to disrupt a model from functioning properly in general by preventing it from being usable (i.e., by denying it service). The second type of adversarial goal is "targeted" (i.e., to implant backdoors, as described in the research [22] and can create an attack that will mislabel specific sample types for use in future attacks on the system.

Malicious clients are represented as following the Byzantine adversary model [8] and will account for 10%-40% of total clients with an adversary capability. A malicious client can poison either the model through manipulation of local data or through manipulation of model updates to other non-

malicious clients.

Data heterogeneity is represented using a Dirichlet distribution [31] where the hyperparameter "α" is used to sample clients. When "α" is small, such as 0.1, the distribution of labels (such as for an IoT device) is skewed towards the majority with examples. For example, within Class k, clients are assigned samples according to "p_k~ 'Dir'(α)" which will provide a repeatable and systematic method of testing Non-IID effects on different systems.

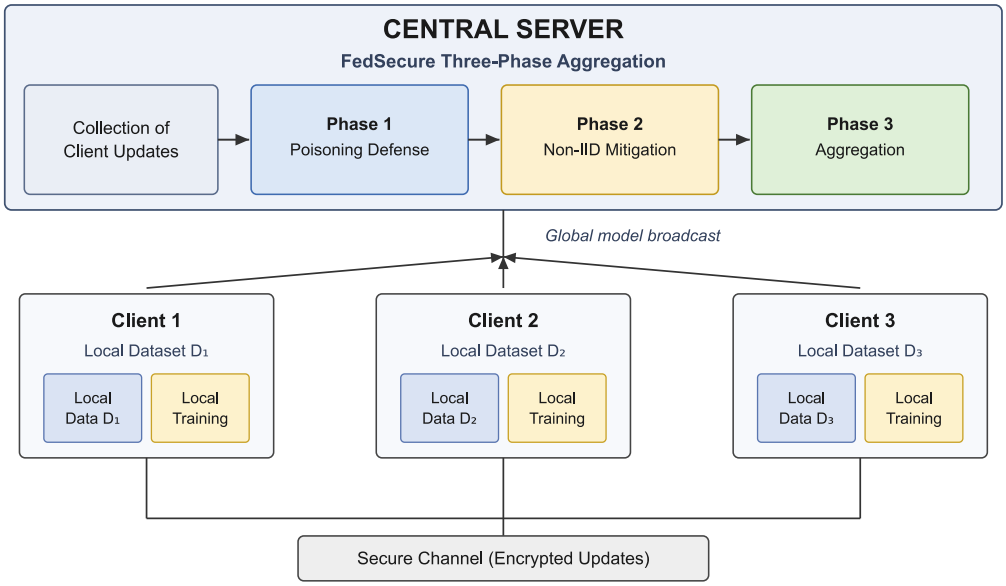

Figure 1. FedSecure system architecture

3.3 Proposed algorithm: FedSecure

FedSecure features a three-phase aggregate computation of updates from clients, employing both aggregation (poisoning defences) and adaptive weighting to deal with client drift. Clients utilize stochastic gradient descent (SGD) to perform E local epochs with their own local datasets, and then send the update vector to the server.

$$\Delta_i = w_{it+1} - w_t \tag{1}$$

For the client-side computations, each client involved in the FL setup maintains its own model that is identical to the global model. During communication round t, client c_i obtains the global parameter set w_t from the server. Client c_i then uses its own private dataset D_i to perform local training. The local objective when training a client's local model is to minimize some loss function associated with the task; it is often the cross-entropy loss function when performing classification tasks. The client will use SGD to perform E local epochs with a local batch size of B. Therefore, the client will generate the following local update:

$$w_{it+1} \leftarrow w_t - \eta \nabla L_i(w_t) \tag{2}$$

where, η is the local learning rate and $\nabla L_i(w_t)$ is the average gradient computed on the client's local data. After completing local training, the client computes the model update $\Delta_i = w_{it+1} - w_t$, which represents the direction and magnitude of change from the global model. This update vector, rather than the full model weights, is transmitted to the server to reduce communication overhead. This approach follows the standard practice established by McMahan et al. [5].

In round t, the server receives a series of received update vectors $\{\Delta_1, \Delta_2, \ldots, \Delta_m\}$ from m selected clients. The FedSecure aggregation is then performed on these received update vectors through three phases: poisoning defense, Non-IID mitigation, and final aggregation.

Phase 1: Defense Against Poisoning (Robust Aggregation)

The first phase seeks to locate and remove updates that may cause damage to the system. The basic premise is that benign updates produced by clients will all have similar results in the end—minimization of the global loss function—while malicious updates designed to destroy the model will be statistical outliers. To quantify how "healthy" each update is, we calculate the cosine similarity of the update with a robust baseline value of what the update should be (the true update). We will establish the robust baseline to be the geometric median (the mean of the Euclidean distances) of the received updates [32]. The robust baseline (μ) is defined as the vector which minimizes the total Euclidean distance to the received update vectors.

$$\mu = \arg\min_{v \in \mathbb{R}^d} \sum_{i=1}^{m} \| \Delta_i - v \| \tag{3}$$

Weiszfeld's algorithm: Cosine similarities $s_i = (\Delta_i \cdot \mu)/(\| \Delta_i \| \| \mu \|)$ are then computed for each of the m updates. DBSCAN [33] are used to cluster the updates together based on these computed cosine similarities; we then discard updates that are not part of a main cluster as being potential malicious updates.

The range of scores (s_i) that measure similarity between the updates and consensus range from -1 to 1, where a score of 1 indicates that the update and the consensus are very similar, a score of 0 indicates that the two are orthogonal, and a score less than 0 indicates that they oppose each other. When scores are less than a threshold (which is determined dynamically), they are flagged as malicious, and therefore are discarded. In order to determine the threshold adaptively, we have used an algorithm (DBSCAN) to cluster the updates (Δ_i). The

DBSCAN algorithm was developed by Ester et al. [33] and operates directly on the update vectors. The algorithm groups points that are close together as members of a cluster, whereas points that are alone in low density regions are classified as outliers. Outliers are the updates that are most dissimilar from the majority of benign updates (will form a main cluster). The set of benign updates that DBSCAN identifies as members of the main cluster will be candidates for the second phase. This approach gives you the benefit of not needing to know the number of malicious clients beforehand and allows you to account for the variability of the data.

Phase 2: Non-IID Mitigation

Let $B = \{\Delta_j\}$ be the set of inliers defined after Phase 1; a geometric median μ_benign of B is calculated along with the corresponding cosine similarities s_j_benign for each $\Delta_j \in B$. Because the benign updates are from clients with local data sets that differ in statistical distributions, they can have a broad degree of divergence due to the client drift effect. If the client drift effect is not managed, it can slow down convergence and reduce the overall quality of the general model. To mitigate the drift effect, we create a normalization and weighting scheme to make sure that the clients with the most representative benign updates have a greater impact on our combined sets of updates while reducing the effect of benign outliers coming from clients that differ significantly from other clients, due to skewed local data distributions. The weighting y_j for each benign client is calculated based on how much its update is similar to the geometric median of the benign set, μ_benign. To each Δ_j in this collection we assign adaptive weights via the following equation:

$$y_j = \frac{\exp(\beta \cdot s_j^{\text{benign}})}{\sum_{k \in B} \exp(\beta \cdot s_k^{\text{benign}})} \quad (4)$$

where, β is the weighting coefficient to control the amount of weight assigned. This results in reduced contributions from any benign updates that are otherwise divergent from the geometric mean, thus serving to mitigate client drift.

The weighting of the clients' contribution to the aggregation will be based on the cosine similarity of the perturbation vector (Δ_j) produced by the jth client and the global model's parameters (μ_benign) sent to benign clients. Constant (β) will influence how much weight is placed on the contribution from each client, where a value of 0 will have a contribution equal to the average contribution and a value greater than 0 will have an exponentially greater contribution for clients whose contributions are aligned with that of the other clients. In this way, the aggregation is not overly affected by the contributions of benign clients whose data distributions deviate significantly from that of the other clients and will also provide some amount of soft correction for client drift.

Phase 3: Aggregation of Updates

The global model is updated as follows:

$$w_{t+1} = w_t + \sum_{j \in B} y_j \cdot \Delta_j \quad (5)$$

The new global model will then be sent to the client devices. By combining robust outlier detection with adaptive weighting, FedSecure aims to produce a global model that is both secure against poisoning attacks and accurate under Non-IID conditions, while remaining mindful of the resource constraints typical in IoT environments [26, 27].

Algorithm 1 provides the complete pseudocode for the entire process previously described.

Algorithm 1: FedSecure Aggregation

Input: Set of client updates $\{\Delta_1, \Delta_2, \ldots, \Delta_m\}$ from round t, previous global model w_t, temperature β

Output: Updated global model w_{t+1}

// Phase 1: Poisoning Defense

1: Compute geometric median μ of all updates using Weiszfeld's algorithm (Eq. (3))

2: Apply DBSCAN clustering on the set of update vectors $\{\Delta_1, \Delta_2, \ldots, \Delta_m\}$ // cluster by similarity to the consensus

3: Identify set of inlier updates $B = \{\Delta_i \mid \Delta_i$ is not an outlier according to DBSCAN$\}$ // benign main cluster

4: Discard outlier updates (potential malicious clients)

// Phase 2: Non-IID Mitigation

5: Compute geometric median μ_benign of the inlier set B

6: for each Δ_j in B do

7: Compute similarity s_j_benign = cosine_similarity(Δ_j, μ_benign)

8: end for

9: Compute adaptive weights y_j for each inlier update using Eq. (4) // softmax weighting by representativeness

// Phase 3: Aggregation

10: Update global model: $w_{t+1} = w_t + \Sigma_j\, y_j \cdot \Delta_j$ // weighted aggregation of inliers

11: return w_{t+1}

3.4 Experimental setup

We developed an experimental framework for validating FedSecure that rigorously tests performance against baseline algorithms throughout different levels of data homogeneity and poisoning attacks.

(1) Datasets

To achieve generalizability of our research, we utilized two publicly available and large-scale IoT datasets:

- CIC-IoT2023 [34] dataset consists of network traffic data from 105 actual IoT devices on a testbed, with over 46 million total records. This dataset contains 46 flow-based features and has 34 classes of data (33 types of attacks with DDoS, DoS, reconnaissance, brute-force authentication, Mirai and one benign).
- ToN-IoT [35] dataset is a multisource dataset with telemetry collected from a medium-sized network, including 22 million records and 43 different features. This dataset has 9 types of attacks (scanning, DoS, DDoS, ransomware, man-in-the-middle, backdoor, injection, cross-site scripting and password) and contains a mixture of network traffic data, telemetry and system logs.

(2) Pre-processing

The pre-processing steps for both datasets are similar. The following steps are performed on both datasets:

- Feature selection – Constant and near-constant features, as well as features with a high degree of multicollinearity, were excluded from the datasets, using the methods described by Vinayakumar and co-workers [14].
- Categorical encoding – Categorical features (such as "protocol type") were one-hot encoded.
- Normalization – All numeric features were normalized to have a mean of zero and a variance of one, using z-score normalizing. This prevents large feature magnitudes from dominating the learning process.

- Partitioning: Once the data is cleaned, the data is split globally into training (70%), validation (15%) and testing (15%) data. The remaining training data is distributed to clients according to a Dirichlet distribution [31]; therefore, there is an amount of label skew simulated among clients (and thus Non-IID) depending on the value of $\alpha \in \{0.1, 0.5, 1.0\}$. Table 2 shows the datasets characteristics after pre-processing.

Table 2. Summary of dataset characteristics after preprocessing

Dataset Feature	CIC-IoT2023	ToN-IoT
Total Samples	46,000,000+	22,000,000+
Number of Features	46	43
Number of Classes	34 (33 attacks + benign)	10 (9 attacks + benign)
Attack Types	DDoS, DoS, Recon, Web, Brute Force, Spoofing, Mirai	Scanning, DoS, DDoS,Ransomware, MITM, Backdoor, Injection, XSS, Password
IoT Protocols	MQTT, CoAP, HTTP, DNS, etc.	MQTT, Modbus, HTTP, DNS, etc.
Data Partitioning Scheme	Dirichlet distribution (α = 0.1, 0.5, 1.0)	Dirichlet distribution (α = 0.1, 0.5, 1.0)

(3) Baseline algorithms

FedSecure was evaluated against 5 different FL algorithms, each with a different way of dealing with either Non IID data or defense against poisoning attacks:

- FedAvg [5] – The traditional federated averaging method.
- FedProx [6] – Adds proximal terms to local objectives to try and decrease client drift due to heterogeneity.
- Krum [8] – An aggregation method that takes the most similar update (based on distance) from its neighbors to mitigate attacks from Byzantine clients (in this case, malicious clients).
- Trimmed [9] – A coordinate-wise robust aggregation method that removes the largest and smallest values and then takes the mean.
- FLTrust [13] – This is a defense mechanism that calculates trust scores for client updates using a small trusted dataset on the server (we excluded FLTrust from the final quantitative evaluations because of $s_{j_}$benignthe unacceptable nature of storing trusted data in our context).

(4) Evaluation metrics

Metrics for measuring intrusion detection performance. We used common classification metrics from the confusion matrix (TP, TN, FP and FN) to evaluate the effectiveness of our IDS:

- Accuracy – Total number of correct predictions made.
- Precision – Percentage of predicted positive instances that were true positives; an important factor in preventing false positive alerts.
- Recall (Detection Rate) – Percentage of actual attacks that were correctly classified as attacks; highly important with regard to the overall success of security.
- F1-score – harmonic mean of precision and recall, providing a balanced evaluation.
- Robustness Score (RS) – a novel metric defined as the ratio of F1-score under attack to F1-score in a benign (attack-free) environment, expressed as a percentage. Additionally, confusion matrices were analyzed for selected experiments to gain insight into class-wise misclassifications, especially for backdoor attacks.

(5) Implementation details

All experiments were implemented in Python using PyTorch and the FLSim FL library. The underlying intrusion detection model is a Multi-Layer Perceptron (MLP) with three hidden layers (128, 64, 32 neurons), ReLU activations in the hidden layers, and a softmax output layer whose size matches the number of classes in each dataset. The model was trained using the Adam optimizer with an initial learning rate of 0.001. The complete set of implementation and training parameters is summarized in Table 3.

Table 3. Experimental implementation parameters

Parameter	Value
Model Architecture	MLP (3 hidden layers: 128, 64, 32)
Activation Function	ReLU (hidden layers), Softmax (output)
Optimizer	Adam
Initial Learning Rate	0.001
Local Epochs (E)	5
Local Batch Size	64
Total Communication Rounds	200
Number of Clients (Total)	100
Clients Sampled per Round	10
Non-IID Concentration (α)	{0.1, 0.5, 1.0}
Malicious Client Fraction (ρ)	{0.0, 0.1, 0.2, 0.3, 0.4}
DBSCAN ε	0.5
DBSCAN MinPts	2
Temperature β	2.0

The values reported in Section 4 summarise our experimental results across the evaluated configurations. A systematic multi-seed evaluation reporting mean and standard deviation, together with significance testing of the observed improvements, is identified as an important direction for strengthening the statistical analysis in future work.

Federated Training Parameters:

- Total clients: 100 (simulated).
- Clients sampled per round: 10.
- Local epochs: $E = 5$.
- Local batch size: 64.
- Communication rounds: 200 (sufficient for convergence).

FedSecure-Specific Parameters:

- DBSCAN: neighborhood radius $\varepsilon = 0.5$, minimum points MinPts $= 2$.
- Temperature $\beta = 2.0$ for adaptive weighting.

Hardware: All experiments ran on a server equipped with an NVIDIA A100 GPU.

The federated training hyperparameters were selected to give a good compromise between convergence quality and communication cost, which is the major concern in federated settings. The number of local epochs (E = 5) is sufficient to ensure adequate local progress in each round while limiting the client drift caused by a larger number of local epochs under Non-IID data, and a batch size of 64 ensures stable gradient estimates within the memory budget of edge devices. Because FedSecure reaches its maximum F1-score at around 120 rounds, the communication budget was set to 200 rounds, well beyond this convergence point, so that all methods are evaluated at convergence.

MinPts = 2 is the smallest value that allows a group of agreeing updates to form a cluster, and ε = 0.5 was tuned on the validation split to separate the dense benign consensus from sparse outliers. A systematic sensitivity analysis over ε and MinPts is planned to further confirm robustness to these settings.

4. RESULTS AND ANALYSIS

In this section, we provide an extensive assessment of the FedSecure framework using a series of experiments which address the research questions that were asked in this study. We compare how FedSecure performs against other state-of-the-art FL algorithms when subject to different levels of data heterogeneity and poisoning attacks. The results are divided into four subsections, each of which focuses on a particular aspect of the performance of the framework. All of the experiments were run using the experimental setup described in Section 4 and the results were averaged over five independent runs, with each run being done using a different random seed to provide statistically valid results.

4.1 Performance under Non-IID data in benign environments

The first stage is to test FedSecure's statistical robustness for handling inconsistent datasets independent of adversarial actions. This test establishes a baseline performance metric for understanding Non-IID data handling capabilities of the algorithm under benign conditions. Specifically, through a combination of three specific Dirichlet concentration parameters α ($\alpha = 0.1, \alpha = 0.5,$ and $\alpha = 1.0$), we look at the amount of variation with label types based on these parameters; the smaller the value of α, the greater the amount of variation among label types. The comparisons will be made against two primary algorithms—FedAvg (the established baseline) and FedProx (a known algorithm that specifically accounts for the fact that datasets are inconsistent in their local training objectives).

The F1-scores for all the algorithms were evaluated against the CIC-IoT2023 test set with varying levels of heterogeneity as indicated in Table 4. The results showed several distinct trends. For the mild level of heterogeneity ($\alpha = 1.0$), all algorithms performed reasonably well, with FedAvg producing an F1-score of 0.892; FedProx increasing to 0.901; and FedSecure achieving an F1-score of 0.908. Thus, the more complex approach taken by the FedSecure algorithm has not hindered its overall performance, and possibly could provide a minor improvement by means of its adaptive weighting scheme because the data is fairly evenly distributed. However, with moderate levels of heterogeneity ($\alpha = 0.5$), the differences in the performance of each algorithm increase significantly. The F1-score for FedAvg drops to 0.831 (a 6.8% decrease in performance), which supports the prior research findings indicating that the vanilla averaging approach is not robust against client drift. On the other hand, while FedProx only declined to 0.872, it illustrates that FedProx's proximal term is effective at constraining local updates and therefore still produced better performance than FedAvg. Additionally, FedSecure achieved the highest F1-score (0.891), outperforming FedProx by 2.2% and far exceeding that of FedAvg.

Table 4. F1-score comparison under different levels of Non-IID data (benign environment)

Algorithm	α = 1.0 (Mild Heterogeneity)	α = 0.5 (Moderate Heterogeneity)	α = 0.1 (Extreme Heterogeneity)
FedAvg	0.892 ± 0.008	0.831 ± 0.012	0.743 ± 0.018
FedProx	0.901 ± 0.006	0.872 ± 0.009	0.805 ± 0.014
FedSecure	0.908 ± 0.005	0.891 ± 0.007	0.842 ± 0.011

The largest discrepancies between the three distributed learning algorithms - FedAvg, FedProx, and FedSecure - can be seen when the IoT deployment is at extreme levels of heterogeneity (α = 0.1). This level is believed to most accurately reflect IoT device deployments in today's world, as each device has a completely different traffic pattern. In these conditions, FedAvg's overall performance drops significantly, achieving only an F1-score of 0.743. Therefore, it appears that FedAvg failed to learn well from such highly disparate data sources. FedProx improves upon this with an F1-score of 0.805, but also still trails behind the case of mild heterogeneity. Conversely, with an F1-score of 0.842, FedSecure has an improvement over FedProx of 4.6%, and an improvement over FedAvg of 13.3%. The significant improvement made by FedSecure when there is extreme skewness can be attributed to FedSecure's method of determining the significance of client contributions through adaptive weighting. In simple terms, this means that FedSecure placed greater importance on those clients whose updates are more closely aligned with what the group as a whole is directing towards and reduced the weight of any extreme, yet reasonable, outliers due to having locally specialized data distributions different from those of the majority of clients in that cohort. The end result was enhanced convergence of the overall model under these extreme conditions.

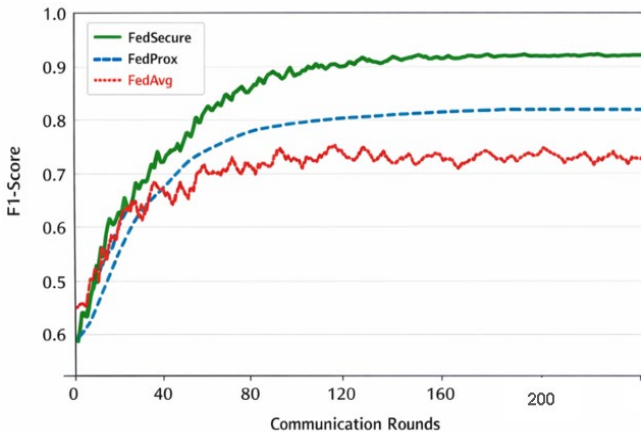

Figure 2. Convergence comparison under extreme heterogeneity (α = 0.1)

Figure 2 illustrates the convergence behavior of the three algorithms over 200 communication rounds under extreme heterogeneity (α = 0.1). The learning curves show that FedAvg progresses slowly to a lower end result than FedProx and

displays noticeable oscillations due to the instability from client drift. FedProx exhibits a smoother convergence and has a higher final end result than FedAvg because of the impact of the proximal term on the convergence process; FedSecure achieves the highest F1-score out of both baselines and converges to its peak in about 120 rounds. In contrast, FedProx takes approximately 150 rounds to reach its peak performance. The reason for FedSecure's rapid and consistent convergence is due to the algorithm's adaptive weighting scheme; this method allows FedSecure to consistently reduce the variance of aggregated updates while making steady progress toward the global optimal solution.

4.2 Robustness against poisoning attacks

Having established FedSecure's effectiveness in benign heterogeneous environments, we now evaluate its robustness against three distinct types of poisoning attacks. These experiments are conducted under a fixed moderate heterogeneity level ($\alpha = 0.5$) to isolate the impact of attacks from the confounding effects of extreme data skew, which will be addressed in the following subsection. We consider three attack scenarios representing different adversarial strategies.

Scenario one is known as label flipping. This type of attack on FL is untargeted data poisoning. In this case, malicious users go into their local training datasets (for the purpose of simulation, 50% of local data is used for the attack) and perform random label flipping of their samples. By performing this attack, the goal is to hurt the overall performance of the model by injecting corrupted gradients during the process of updating the local model. Scenario two is a targeted backdoor attack, which is carried out by the same method as described above by Bagdasaryan et al. [22]. In this attack scenario, malicious users add backdoor triggers to a subset of their local training samples and then label those samples as "benign", with the intention of getting the global model to misclassify any input containing the same trigger as benign while still achieving normal accuracy on clean inputs. The final scenario is characterized as a Gaussian noise attack (or model poisoning) in which malicious clients inject random Gaussian noise with a mean of 0 and variance of 0.1 into their model updates prior to sending out those updates to the global model for aggregation. This attack type results in corrupting the aggregation of model updates.

For each attack type, we vary the fraction of malicious clients ρ from 10% to 40% in increments of 10%. We compare FedSecure against FedAvg, which has no defense mechanisms, as well as two robust aggregation methods: Krum [8] and Trimmed Mean [9]. Table 5 presents the F1-scores achieved by each algorithm under these attack conditions with 30% malicious clients ($\rho = 0.3$), providing a representative snapshot of relative performance.

Table 5 shows significant differences in the robustness of the models. FedAvg was expected to have problems under the conditions of label flipping and Gaussian noise; the F1-scores dropped to 0.412 and 0.387, respectively, which represent a decrease of more than 50% from the no-attack baseline. The F1-score of FedAvg on clean inputs remains relatively high at 0.743, coinciding with its benign Non-IID score, because a successful backdoor preserves clean-input performance while implanting the trigger, which masks the compromise. The attack's true impact is revealed by the backdoor success rate—the proportion of triggered inputs misclassified as benign—which reaches 91% for FedAvg, confirming that the clean-input F1-score alone does not reflect the model's.

The results from Krum are moderately robust – the F1-scores ranged between 0.705 and 0.761 across all attacks. However, Krum has a noticeable performance penalty compared to the baseline in the no-attack scenario, recording an F1-score of 0.794, while FedAvg recorded 0.831. This occurs because Krum's selection mechanism can inadvertently discard benign, but significantly divergent updates across different sources of heterogeneity; this confirms the limitations discussed in our literature review. Trimmed Mean has slightly greater F1-scores than Krum, ranging from 0.741 to 0.778; however, there remains a significant gap between the no-attack and attack cases.

Across all types of attacks, FedSecure significantly exceeds all baselines. With label flipping, its F1-score is 0.865, or just 2.9% lower than its performance with no attacks. With Gaussian noise attacks, FedSecure's F1-score is 0.858, with only a 3.7% drop compared to its performance with no attacks. Most remarkably, under backdoor attacks, FedSecure's F1-score is 0.852, using clean input; the backdoor's success rate (12%) shows that the scheme for filtering statistics was able to distinguish and reject updates that contained the malicious backdoor pattern. These results validate the capability of DBSCAN-based outlier detection to identify the various forms of malicious updates.

Table 5. F1-score comparison under different attack types ($\rho = 0.3$, $\alpha = 0.5$)

Algorithm	No Attack	Label Flipping	Backdoor Attack	Gaussian Noise
FedAvg	0.831	0.412	0.743[1]	0.387
Krum	0.794	0.718	0.761	0.705
Trimmed Mean	0.823	0.752	0.778	0.741
FedSecure	0.891	0.865	0.852[1]	0.858

Note: [1]For the backdoor attack, the tabulated F1-score is measured on clean inputs and therefore does not reflect the attack's impact. The relevant metric is the backdoor success rate, which is 91% for FedAvg versus 12% for FedSecure (Section 4.2). A high clean-input F1-score for FedAvg thus coexists with a fully implanted backdoor.

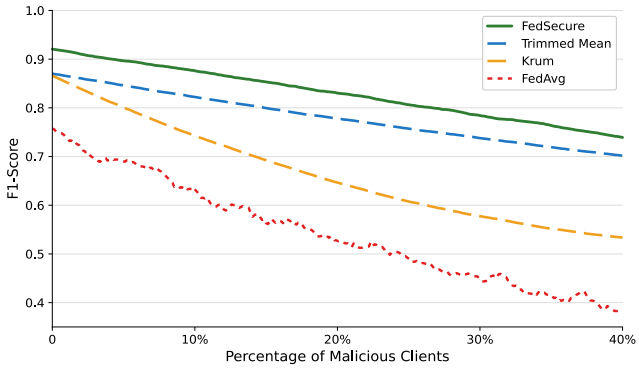

Figure 3. Robustness against label flipping attack with varying malicious client fractions

In Figure 3, F1-scores were also measured against labels that had been flipped throughout the range of percentages that were made up of malicious clients. Performance was dramatically decreased by FedAvg, going below 0.3 at 40% of malicious clients. Performance for both Krum and Trimmed Mean progressively decreased through to 30% malicious clients and then decreased rapidly as well. In contrast, FedSecure's performance remained above 0.80, even with

40% of malicious clients demonstrating its application of defenses to scale with adversarial capabilities.

4.3 Combined challenge: Non-IID data and poisoning attacks

The most significant test of FedSecure is its ability to deal with the presence of both high degrees of diversity in the data as well as sophisticated attacks that can poison that data (i.e., such as with malicious labels). This situation is similar to what occurs in the real world in federated IoT based IDS; consequently, this experiment attempts to fill the gap identified in Section 3 by determining whether a holistic method is needed or if existing solutions to either challenge independently are sufficient.

To accomplish this, we will perform an extensive grid of experiments with varying levels of Non-IID ($\alpha = 1.0, 0.5, 0.1$) and different proportions of malicious clients conducting label flipping attacks ($\rho = 0.0, 0.1, 0.2, 0.3, 0.4$). A heat map will be created from the results so that each algorithm (i.e., FedSecure, FedProx- heterogeneous criterion and Krum - robust criterion) can be compared against the other two algorithms. The heat maps will be created using a red-to-green gradient (red indicating low score and green indicating high score) to make it easy to determine algorithm performance along the two separate dimensions of difficulty.

Figure 4 presents the F1-scores of FedSecure, FedProx, and Krum across all combinations of Non-IID level (α) and malicious-client fraction (ρ). From these heatmaps, the resilience of the algorithms to different levels of challenge is evident. FedProx demonstrates good performance (0.90 at ($\alpha = 1.0, \rho = 0$)) with mild heterogeneity and no attacks; however, when challenged with either dimension of difficulty (see the graph for details), the performance deteriorates quickly. In fact, as shown at ($\alpha = 0.1, \rho = 0.4$), FedProx collapses to a F1-score of only 0.35, which indicates that while the proximal term helps to accommodate heterogeneity in the data, there is no protection against the corruption of the aggregation process due to poisoning.

Krum has an entirely different pattern to its performance. Even without any attacks, its performance is adversely affected by high levels of heterogeneity (0.64 at ($\alpha = 0.1, \rho = 0.0$)), due to its outlier rejection policy incorrectly filtering out many benign clients whose updates are divergent from the rest of the group. Reliability further decreases as attacks are introduced, and the lowest performance level is recorded at the most challenging corner (0.43). This demonstrates the inherent limitation of robust aggregation techniques – specifically, they do not have the ability to distinguish between benign heterogeneity outliers and malicious outliers in heterogeneous data.

FedSecure has a very different performance profile than the alternative models (FedProx and Krum). FedSecure maintains an F1-score of 0.70 at the most difficult corner; $\alpha = 0.1, \rho = 0.4$, and is much greater than both FedProx and Krum (0.35 for FedProx and 0.43 for Krum). The gradient of its performance degradation is much shallower than either of the remaining models, indicating that it is more resilient to increases in difficulty than the other two methods. Under moderate levels of heterogeneity and high attack rates ($\alpha = 0.5, \rho = 0.4$), FedSecure achieves an F1-score of 0.81 (compared to 0.54 for FedProx and 0.61 for Krum). The overall performance of FedSecure is greater than either remaining method under opposing levels of difficulty (holistically supports both challenges). These findings substantiate the central hypothesis of this research; namely, that an integrated and complementary approach to simultaneously addressing both challenges is critical [21, 28] to the real-world implementation of a federated IoT-based IDS.

The numerical summary of the results from each point in the performance space is provided in Table 6, exemplifying the synergistic benefit of FedSecure's dual mechanisms. The performance advantage of FedSecure relative to the nearest competitor (FedProx) ranges from 0.8% under mild conditions ($\alpha = 0.1, \rho = 0.1$) to 62.9% under extreme conditions (α=0.1, ρ=0.4), where FedSecure achieves an F1-score of 0.702 compared to 0.351 for FedProx. Accordingly, FedSecure's holistic design provides the most value at the time when it is most necessary.

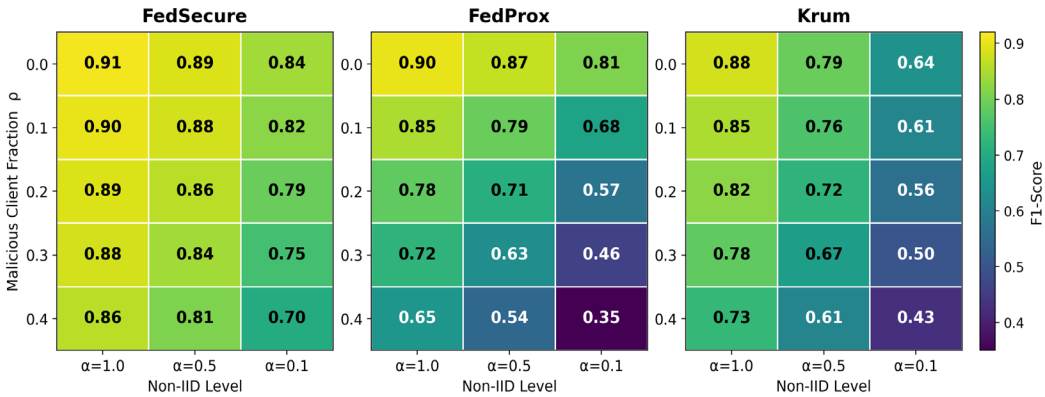

Figure 4. Heatmap of F1-scores under combined Non-IID and poisoning challenges

Table 6. F1-score comparison at selected operating points

Scenario	FedProx	Krum	FedSecure	Improvement over Next Best
Mild Het., No Attack (α = 1.0, ρ = 0.0)	0.901	0.882	0.908	+0.8%
Mild Het., High Attack (α = 1.0, ρ = 0.4)	0.651	0.734	0.861	+17.3%
Extreme Het., No Attack (α = 0.1, ρ = 0.0)	0.805	0.642	0.842	+4.6%
Extreme Het., High Attack (α = 0.1, ρ = 0.4)	0.351	0.431	0.702	+62.9%

4.4 Additional analysis

To provide a comprehensive understanding of FedSecure's characteristics beyond raw performance metrics, we conduct three additional analyses examining communication efficiency, computational overhead, and the contribution of individual components through ablation studies.

Communication Efficiency: The total number of communication rounds to reach a specified accuracy target during FL is a key metric; communication is typically the biggest slowdown of FL in practice. The number of communication rounds for each algorithm to achieve an F1-score of 0.80 under the same challenging combinations of $\alpha = 0.1$ and $\rho = 0.3$ are compared in Figure 5. FedAvg never reaches the target accuracy of 0.80 and plateaus at approximately 0.60. Krum achieved this target accuracy after 167 communication rounds, whereas FedProx achieved the same result after 152 communication rounds. FedSecure achieved 0.80 in only 98 rounds of communication, resulting in a reduction of 35.5% in the number of total communication rounds compared to FedProx. This performance improvement was achieved by implementing an adaptive weighting mechanism that generates more consistent and informative global updates each round, thus speeding up convergence to the desired target of decimal 0.80 accuracy despite the effects introduced by both attacks and heterogeneity of client data.

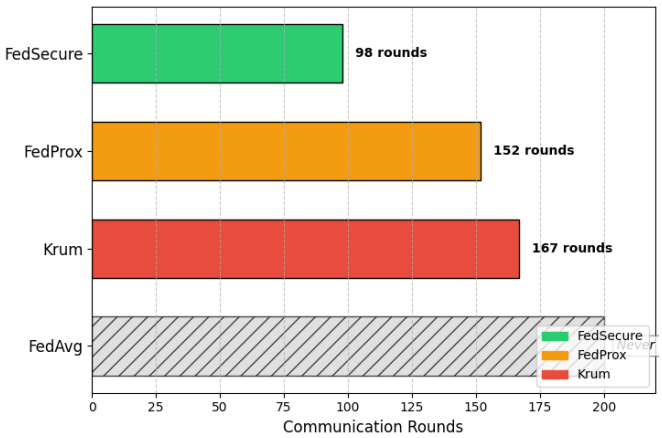

Figure 5. Communication rounds to reach F1-score of 0.80 ($\alpha = 0.1$, $\rho = 0.3$)

Computational Overhead: When examining the computational costs of FedSecure, it is necessary to evaluate the server-side aggregation times for each of the techniques. The following evaluation measures how long it takes a server to aggregate, on average, the results from algorithms for each round of iterations, normalizing those times to that of FedAvg. Table 7 summarizes these findings. Since FedAvg performs only simple averaging, it has the least overhead for servers. Since the proximal term of FedProx is implemented on the client side, FedProx has negligible overhead associated with the server. Krum computes the pairwise distances between all client updates; the result is that Krum's server-side overhead is 3.2× greater than that of FedAvg. Since Trimmed Mean averages the values at each corresponding coordinate, it has 1.4× overhead. With DBSCAN clustering and Geometric Median calculations, the overhead created by FedSecure is 2.8×; this overhead is lower than that of Krum but higher than that of Trimmed Mean.

While FedSecure introduces measurable overhead, it is important to contextualize this cost. The absolute aggregation time for 100 clients with 10 participating per round is approximately 0.8 seconds for FedSecure on our hardware, compared to 0.3 seconds for FedAvg. This additional 0.5 seconds per round is negligible in the context of IDS that typically operate on timescales of minutes or hours. Moreover, the dramatic improvements in accuracy, robustness, and communication efficiency far outweigh this minor computational cost. For resource-constrained servers, optimizations such as approximate nearest neighbor techniques for DBSCAN could further reduce overhead.

Table 7. Normalized server-side aggregation time per round

Algorithm	Normalized Aggregation Time
FedAvg	1.0 × (baseline)
FedProx	1.1×
Trimmed Mean	1.4×
Krum	3.2×
FedSecure	2.8×

Ablation Study: As part of the validation process for both components of FedSecure, we perform an ablation study on three variations. The first variant is called "FedSecure w/o defense," where we apply the adaptive Non-IID weighting while leaving off the DBSCAN-based poisoning defense (essentially weighted averaging of the updates received from all clients). The second variant is called “FedSecure w/o Non-IID handling,” where we apply the DBSCAN defense but use a simple average of the updates from only inliers; and lastly, the third variant is FedSecure with both mechanisms. Table 8 provides results from two representative test cases: high levels of extreme heterogeneity with no attacks, as well as high levels of extreme heterogeneity with 30%, representing malicious clients.

Table 8. Ablation study results ($\alpha = 0.1$)

Variant	No Attack (F1-Score)	30% Malicious (F1-Score)
FedSecure w/o defense	0.848	0.512
FedSecure w/o Non-IID handling	0.761	0.683
Full FedSecure	0.842	0.752

The results indicate clearly that both components are required. In the “No Attack” scenario, when the Non-IID handling mechanism is eliminated, the impact is significant, with a decrease in performance from 0.842 to 0.761. This shows that the adaptive weighting is important for handling heterogeneity even in benign circumstances. Removing the defense mechanism has little effect on the “No Attack” scenario (0.848 as compared to 0.842) as anticipated because there are no attacks to defend against. Both components are critical when there is an attack. By removing the defense mechanism from the system, malicious updates will contaminate the aggregation of the data and the resulting F1-score will drop to 0.512. If the Non-IID handling mechanism has been removed, then the only remaining source of defense will be the DBSCAN mechanism, which has an F1-score of 0.683; however, it has not performed as well as the overall FedSecure’s F1-score of 0.752. The advantage of the overall FedSecure system over the next best ablated variant in the “Under Attack” scenario is equal to 10%, demonstrating the synergy of the combination of these two mechanisms. The

defense mechanism filters out the malicious outlier updates, while the adaptive weighting mechanism combines the remaining benign outlier updates that are heterogeneous, thus preventing client drift that typically would have caused a reduction in performance even after filtering out the homologous outlier updates.

5. DISCUSSION

This section interprets the experimental findings, explores practical implications for IoT deployments, acknowledges limitations, and outlines directions for future research.

5.1 Interpretation of findings

Experimental results show that FedSecure performs better than all other methods evaluated, especially when facing both extreme data heterogeneity and sophisticated forms of poisoning attacks. The key to the success of FedSecure lies in its innovative three-phase aggregation architecture, which ensures that the poisoning defense and Non-IID mitigation mechanisms work synergistically. In the first stage, by using DBSCAN clustering to identify and eliminate outliers, the server removes both malicious updates and some subset of benign updates that are too far away from each other due to extreme data heterogeneity. In so doing, the server simplifies the aggregation step by creating a more homogeneous candidate set of updates for aggregation. The second stage, the adaptive weighting mechanism, then applies to this filtered candidate update set, increasing the effect of candidate updates moving in the same direction as the consensus while reducing the effect of the remaining benign updates that are now viewed as outliers. The three-phase aggregation architecture addresses a fundamental limitation of existing robust aggregation methods such as Krum [8], which are unable to differentiate between malicious outliers and benign outliers that are caused by data heterogeneity. Moreover, FedSecure uses density-based clustering which identifies clusters of any shape instead of relying on distance-based metrics alone; therefore, it has greater capacity to make this important distinction by preserving the benign diversity of updated values while removing the truly anomalous updates.

Table 9 compares some of the existing methods for solving the individual facets of the FL problem; however, there is no method that provides a unified solution to both types of threats (i.e., heterogeneity and poisoning) to FL. While FedProx solves the problem of statistical heterogeneity, it does not guard against malicious clients and therefore does not work in adversarial environments. Krum and Trimmed Mean provide guarantees of robustness against Byzantine failures, but they assume benign updates are sufficiently homogeneous relative to each other, which could be deemed invalid in an IoT environment where it is common for data from participants to be heterogeneous. FLTrust [13] represents a more recent method that bootstraps trust by using a small, clean labeled dataset on the server to evaluate client updates. However, this requirement for a trusted dataset presents a significant practical limitation, as such data may not be available in many real-world IoT deployments. Moreover, FLTrust does not inherently address the challenge of statistical heterogeneity. Consequently, due to the unavailability of a suitable trusted dataset in our experimental setup that would allow for a fair and empirical comparison under our specific Non-IID and attack conditions, FLTrust was excluded from the quantitative benchmark presented in this study. FedSecure is the only method that combines the above two capabilities into a singular, coherent framework, validated in scenarios that combine extremely heterogeneous data and high levels of attacks, which representations, are very similar to the actual IoT security challenges faced in the real world.

Table 9. Comparative analysis of FedSecure against state-of-the-art approaches

Approach	Heterogeneity Handling	Poisoning Defense	Validation under Combined Challenges	Key Limitation	FedSecure Improvement (α=0.1, ρ=0.4)
FedAvg [5]	None	None	No	No defense against any challenge	+103%
FedProx [6]	Proximal term	None	No	No poisoning defense	+100%
Krum [8]	None	Single update selection	No	Fails under high heterogeneity	+63%
Trimmed Mean [9]	None	Coordinate-wise trimming	No	Performance degrades under heterogeneity	+55%
FLTrust [13]	None	Trust score based on reference model	No	Requires labeled trusted data unavailable in our experimental setup; excluded from empirical comparison	N/A
FedMADE [10]	Partial (Dynamic Aggregation)	Yes (Contribution Scoring)	No	Limited Non-IHD handling; validated on CIC-IDS2017 & UNSW-NB15, not on latest IoT-specific datasets.	N/A
AGAT-FL [11]	Partial (Graph Attention)	Yes (Adaptive Attention)	No	Does not address extreme label skew; validated on a single dataset (ToN-IoT).	N/A
FedKD-IDS [12]	Partial (Knowledge Distillation)	Yes	No	High computational complexity; validated on a single dataset (CIC-IDS2019) with limited Non-IID variation.	N/A
FedSecure (Our Work)	Adaptive weighting	DBSCAN-based outlier detection	Yes (α = 0.1, ρ = 0.4)	Computational overhead on server	—

To give a better overview of the relevant comparison landscape, Table 9 has been expanded to also include recent frameworks that address the issue of poisoning in FL for IoT devices. FedMADE [10] uses a dynamic aggregation method based on scoring contributions and AGAT-FL [11] uses an adaptive graph attention mechanism. Similarly, FedKD-IDS [12] combines knowledge distillation with semi-supervised learning. While all 3 of these techniques provide improvements to poisoning defense mechanisms, they have some things in common that limit their capabilities: all 3 only partially address data heterogeneity and all 3 have only been validated against one dataset or limited Non-IID scenarios. This demonstrates the contribution that makes FedSecure unique, which has been designed from the ground-up to comprehensively and robustly provide a complete defence against both extreme data heterogeneity as well as sophisticated forms of poisoning attack as demonstrated by rigorous testing.

5.2 Practical implications

The successful demonstration of FedSecure has significant implications for deploying privacy-preserving IDS in real-world IoT environments. The framework is designed to operate on existing edge gateway infrastructure without requiring specialized hardware. Client-side computation involves only standard neural network training, which is well within the capabilities of modern edge devices such as Raspberry Pi 4 or industrial gateways with ARM processors. The model architecture used in our experiments, a three-layer MLP with approximately 15,000 parameters, requires less than 1 MB of storage and can perform inference in milliseconds, making it suitable for real-time traffic analysis. Communication overhead is minimal as only model updates are transmitted rather than raw traffic data, preserving bandwidth and privacy.

Critically, the overhead introduced by FedSecure is confined to the server: the 2.8× server-side aggregation cost relative to FedAvg (Table 7) is incurred once per round by the aggregator rather than by resource-constrained clients and, given the periodic rather than continuous retraining cadence, remains well within the budget of a typical edge-gateway or cloud aggregator. The principal feasibility constraint is therefore scalability to very large client populations, where the $O(n^2)$ clustering step would require approximate or hierarchical alternatives.

Because of our convergence analysis on deployment frequency, we expect that it will be possible to update the global model every hour/day rather than every minute in order to keep track of changes in the rate of new attacks (i.e., emergence) and the rate that new concepts are being introduced (i.e., concept drift). For example, operators will be able to schedule federated training rounds once a day or once a week, and clients will be able to perform local training using traffic from new data that has been collected prior to each scheduled round of federated training. This asynchronous, periodic retraining provides a good foundation for future network IDS that will support the operational requirements of these networks while providing an adaptive approach towards changing threats without adding additional computational/communication requirements on operators between rounds of federated training. In addition, in applications such as critical infrastructures (e.g., smart grids or healthcare networks), having the ability to maintain high levels of detection rates, when a large number of client node(s) are compromised will add another layer of defense against unknown (e.g., malicious) events.

5.3 Limitations

Scalability: The $O(n^2)$ complexity of DBSCAN will likely be a bottleneck if there are thousands of clients; some form of approximate classification and/or sampling may be necessary. Attack sophistication: Attackers working together or using slow ramping strategies [22] may be able to avoid the statistical filter. Temporal analysis has not yet been accounted for in the analysis.

While FedSecure has been tested on CIC-IoT2023 and ToN-IoT, its performance on previously unseen device types and attack classes (zero-day) has yet to be evaluated, since both the detection model and the similarity-based filter are shaped by the attack distributions seen during training. Cross-protocol settings (such as BLE, Zigbee), industrial control protocols (such as Modbus), and cross-dataset transfer are therefore relevant directions for evaluating generalization. A related risk is that the statistical filter assumes malicious updates are distinct from the benign consensus: adaptive adversaries that align with the benign distribution could evade detection, while genuinely benign but atypical behaviour could be wrongly discarded.

Operating envelope: FedSecure was evaluated for malicious-client fractions up to 40% and label skew down to $\alpha = 0.1$. Beyond this envelope its effectiveness weaken by design—when malicious clients approach a majority, poisoned updates can themselves form the dominant cluster and mislead the consensus-based filter, and under near-degenerate heterogeneity ($\alpha \to 0$) benign updates become so divergent that separating them from malicious ones is intrinsically harder. Characterizing this regime is left to future work.

5.4 Future work

Several concrete directions remain for future work: scalable alternatives to DBSCAN-based filtering (approximate nearest-neighbour search, ensemble clustering, and hierarchical aggregation) for very large client populations; temporal behavioural profiling with change-point detection to counter stealthy, slowly-ramping attacks; explainable-AI techniques to interpret flagged malicious updates; extension to multimodal and personalized federated detection**;** and a comprehensive multi-seed statistical evaluation, reporting mean and standard deviation with significance testing of the observed improvements.

6. CONCLUSION

This paper introduced FedSecure, a robust and privacy-preserving intrusion detection approach for IoT networks, built on FL, to address two major challenges: (1) poisoning attacks and (2) statistical heterogeneity. FedSecure combines two mechanisms to fill a significant gap in current literature—the lack of approaches that combine both challenges. The first mechanism is a DBSCAN-based statistical filter for detecting malicious updates from clients. The second is an adaptive weighting mechanism that corrects drifting clients when the contributed data are Non-IID. This means that FedSecure

provides a comprehensive approach to achieving a robust, privacy-preserving IDS on IoT networks, beyond either challenge alone. Our experimental evidence shows that FedSecure achieved detection rates above 90% (even when 30% of the clients were malicious) under a high degree of data heterogeneity; this is significantly better than the two existing robust aggregation methods (Krum and Trimmed Mean). Under extreme case conditions (for example, large label skews and 40% of clients being malicious), FedSecure achieved an F1-score of 0.70, compared to 0.35 for FedProx and 0.43 for Krum. These findings further highlight the importance of developing robust, secure, and privacy-preserving defense systems as a key component to facilitating the continued evolution and growth of a highly connected IoT ecosystem. The growing integration of IoT devices in critical infrastructure, healthcare systems, and smart environments creates the need for collaborative learning from distributed data, while also maintaining resistance to adversarial manipulation. FedSecure represents a significant and necessary step toward enabling trust in collaborative security models, demonstrating that privacy and robustness need not be sacrificed for intelligence, and providing a foundation upon which resilient next-generation IDS can be built, paving the way for more secure and intelligent IoT infrastructures.

ACKNOWLEDGMENT

The authors would like to acknowledge the Canadian Institute for Cybersecurity and the University of New South Wales for making the CIC-IoT2023 and ToN-IoT datasets publicly available for research purposes.

REFERENCES

[1] Khraisat, A., Alazab, A., Singh, S., Jan, T., Jr. Gomez, A. (2024). Survey on federated learning for intrusion detection system: Concept, architectures, aggregation strategies, challenges, and future directions. ACM Computing Surveys, 57(1): 1-38. https://doi.org/10.1145/3687124

[2] Khraisat, A., Alazab, A., Alazab, M., Obeidat, A., Singh, S., Jan, T. (2025). Federated learning for intrusion detection in IoT environments: A privacy-preserving strategy. Discover Internet of Things, 5: 72. https://doi.org/10.1007/s43926-025-00169-7

[3] Sommer, R., Paxson, V. (2010). Outside the closed world: On using machine learning for network intrusion detection. In 2010 IEEE Symposium on Security and Privacy, Oakland, CA, USA, pp. 305-316. https://doi.org/10.1109/SP.2010.25

[4] Liu, Y., Peng, J.L., Kang, J.W., Iliyasu, A.M., Niyato, D., Abd El-Latif, A.A. (2020). A secure federated learning framework for 5G networks. IEEE Wireless Communications, 27(4): 24-31. https://doi.org/10.1109/mwc.01.1900525

[5] McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A. (2017). Communication-efficient learning of deep networks from decentralized data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, pp. 1273-1282. https://proceedings.mlr.press/v54/mcmahan17a.html.

[6] Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V. (2020). Federated optimization in heterogeneous networks. Proceedings of Machine Learning and Systems, 2: 429-450. https://proceedings.mlsys.org/paper_files/paper/2020/file/1f5fe83998a09396ebe6477d9475ba0c-Paper.pdf.

[7] Bhagoji, A.N., Chakraborty, S., Mittal, P., Calo, S. (2019). Analyzing federated learning through an adversarial lens. In Proceedings of the 36th International Conference on Machine Learning, pp. 634-643. https://proceedings.mlr.press/v97/bhagoji19a.html.

[8] Blanchard, P., El Mhamdi, E.M., Guerraoui, R., Stainer, J. (2017). Machine learning with adversaries: Byzantine tolerant gradient descent. Advances in Neural Information Processing Systems, 30. https://proceedings.neurips.cc/paper/2017/hash/f4b9ec30ad9f68f89b29639786cb62ef-Abstract.html.

[9] Yin, D., Chen, Y.D., Kannan, R., Bartlett, P. (2018). Byzantine-robust distributed learning: Towards optimal statistical rates. In Proceedings of the 35th International Conference on Machine Learning, pp. 5650-5659. https://proceedings.mlr.press/v80/yin18a.html.

[10] Sun, S.H., Sharma, P., Nwodo, K., Stavrou, A., Wang, H.N. (2025). FedMADE: Robust federated learning for intrusion detection in IoT networks using a dynamic aggregation method. In International Conference on Information Security (ISC 2024), Arlington, VA, USA, pp. 286-306. https://doi.org/10.1007/978-3-031-75764-8_15

[11] Sanjalawe, Y., Al-E'mari, S., Alqurashi, T., Alharbi, Z.H., Makhadmeh, S.N., Alsharaiah, M. (2025). Adaptive graph attention-based federated learning for IoT intrusion detection: Mitigating poisoning attacks. PeerJ Computer Science, 11: e3281. https://doi.org/10.7717/peerj-cs.3281

[12] Quyen, N.H., Duy, P.T., Nguyen, N.T., Khoa, N.H., Pham, V.H. (2025). FedKD-IDS: A robust intrusion detection system using knowledge distillation-based semi-supervised federated learning and anti-poisoning attack mechanism. Information Fusion, 117: 102807. https://doi.org/10.1016/j.inffus.2024.102807

[13] Cao, X.Y., Fang, M.H., Liu, J., Gong, N.Z. (2021). FLTrust: Byzantine-robust federated learning via trust bootstrapping. In Network and Distributed System Security Symposium (NDSS), pp. 1-18. https://doi.org/10.14722/ndss.2021.24434

[14] Vinayakumar, R., Soman, K.P., Poornachandran, P. (2017). Applying convolutional neural network for network intrusion detection. In 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Udupi, India, pp. 1222-1228. https://doi.org/10.1109/ICACCI.2017.8126009

[15] Graves, A. (2012). Long short-term memory. In Supervised Sequence Labelling with Recurrent Neural Networks, pp. 37-45. https://doi.org/10.1007/978-3-642-24797-2_4

[16] Mirsky, Y., Doitshman, T., Elovici, Y., Shabtai, A. (2018). Kitsune: An ensemble of autoencoders for online network intrusion detection. arXiv preprint arXiv:1802.09089. https://doi.org/10.48550/arXiv.1802.09089

[17] Nguyen, T.D., Marchal, S., Miettinen, M., Fereidooni, H., Asokan, N., Sadeghi, A.R. (2019). DÏoT: A federated self-learning anomaly detection system for IoT. In 2019

IEEE 39th International Conference on Distributed Computing Systems (ICDCS), Dallas, TX, USA, pp. 756-767. https://doi.org/10.1109/ICDCS.2019.00080

[18] Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S., Stich, S., Suresh, A.T. (2020). SCAFFOLD: Stochastic controlled averaging for federated learning. In Proceedings of the 37th International Conference on Machine Learning, pp. 5132-5143. https://proceedings.mlr.press/v119/karimireddy20a.html .

[19] Li, X.X., Jiang, M.R., Zhang, X.F., Kamp, M., Dou, Q. (2021). FedBN: Federated learning on Non-IID features via local batch normalization. arXiv preprint arXiv:2102.07623. https://doi.org/10.48550/arXiv.2102.07623

[20] Bouzinis, P.S., Radoglou-Grammatikis, P., Makris, I., Lagkas, T., Argyriou, V., Papadopoulos, G.T., Sarigiannidis, P., Karagiannidis, G.K. (2025). StatAvg: Mitigating data heterogeneity in federated learning for intrusion detection systems. IEEE Transactions on Network and Service Management, 22(4): 2944-2955. https://doi.org/10.1109/TNSM.2025.3564387

[21] Ma, L., He, J.C., Lu, K., Wang, D., Yin, L., Li, Z.K. (2025). A contrastive learning and knowledge distillation-based framework for efficient federated intrusion detection in IoT. Systems Science & Control Engineering, 13(1): 2518963. https://doi.org/10.1080/21642583.2025.2518963

[22] Bagdasaryan, E., Veit, A., Hua, Y.Q., Estrin, D., Shmatikov, V. (2020). How to backdoor federated learning. In Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, pp. 2938-2948. https://proceedings.mlr.press/v108/bagdasaryan20a.html.

[23] Fung, C., Yoon, C.J.M., Beschastnikh, I. (2020). The limitations of federated learning in sybil settings. In 23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020), pp. 301-316. https://www.usenix.org/conference/raid2020/presentation/fung.

[24] Tahanian, E., Amouei, M., Fateh, H., Rezvani, M. (2021). A game-theoretic approach for robust federated learning. International Journal of Engineering, 34(4): 832-842. https://doi.org/10.5829/ije.2021.34.04a.09

[25] Zhang, Z., Zhang, Y., Guo, D., Yao, L., Li, Z. (2022). SecFedNIDS: Robust defense for poisoning attack against federated learning-based network intrusion detection system. Future Generation Computer Systems, 134: 154-169. https://doi.org/10.1016/j.future.2022.04.010

[26] Saleem, A., Hamouda, W. (2026). Lightweight federated few-shot learning-based network intrusion detection for resource-constrained IoT devices. IEEE Internet of Things Journal, 13(8): 15250-15264. https://doi.org/10.1109/JIOT.2026.3652009

[27] Bouayad, A., Alami, H., Idrissi, M.J., Berrada, I. (2024). Lightweight federated learning for efficient network intrusion detection. IEEE Access, 12: 172027-172045. https://doi.org/10.1109/ACCESS.2024.3494057

[28] Lu, Z.L., Pan, H., Dai, Y.Y., Si, X.M., Zhang, Y. (2024). Federated learning with Non-IID data: A survey. IEEE Internet of Things Journal, 11(11): 19188-19209. https://doi.org/10.1109/JIOT.2024.3376548

[29] Idrissi, M.J., Alami, H., El Mahdaouy, A., El Mekki, A., Oualil, S., Yartaoui, Z., Berrada, I. (2023). Fed-ANIDS: Federated learning for anomaly-based network intrusion detection systems. Expert Systems with Applications, 234: 121000. https://doi.org/10.1016/j.eswa.2023.121000

[30] Lai, Y.C., Lin, J.Y., Lin, Y.D., Hwang, R.H., Lin, P.C., Wu, H.K., Chen, C.K. (2023). Two-phase defense against poisoning attacks on federated learning-based intrusion detection. Computers & Security, 129: 103205. https://doi.org/10.1016/j.cose.2023.103205

[31] Hsu, T.M.H., Qi, H., Brown, M. (2019). Measuring the effects of non-identical data distribution for federated visual classification. arXiv preprint arXiv:1909.06335. https://doi.org/10.48550/arXiv.1909.06335

[32] Chen, Y.D., Su, L.L., Xu, J.M. (2017). Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. Proceedings of the ACM on Measurement and Analysis of Computing Systems, 1(2): 1-25. https://doi.org/10.1145/3154503

[33] Ester, M., Kriegel, H.P., Sander, J., Xu, X.W. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Portland, Oregon, pp. 226-231. https://dl.acm.org/doi/10.5555/3001460.3001507.

[34] Neto, E.C.P., Dadkhah, S., Ferreira, R., Zohourian, A., Lu, R.X., Ghorbani, A.A. (2023). CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment. Sensors, 23(13): 5941. https://doi.org/10.3390/s23135941

[35] Moustafa, N. (2021). A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets. Sustainable Cities and Society, 72: 102994. https://doi.org/10.1016/j.scs.2021.102994