Journal of Al-Turath University College (JATUC)
Open Access Journal
ISSN: 2074-5621
DOI: https://doi.org/10.63964/JATUC.43.1.2026.32
https://jatuc.uoturath.edu.iq/

Lightweight Federated Intrusion Detection System for Resource Constrained IoT Networks Using Edge-Assisted Learning

Hanan A. Zainel
First Grades Teacher Department, College of Basic Education, Kirkuk University, Kirkuk, Iraq

*Correspondence email: Hananzainel@uokirkuk.edu.iq

KEYWORDS
Internet of Things, Intrusion detection system, Federated learning, edge computing, lightweight deep learning, model compression, non-IID data, IoT security.

ABSTRACT
The rapid growth of the Internet of Things (IoT) has expanded the attack surface, while many endpoints operate on microcontroller-class hardware with tight limits on CPU, memory and energy, making conventional intrusion detection systems (IDS) difficult to deploy. This work proposes and evaluates a lightweight federated intrusion detection system (LF-IDS) that enables accurate, privacy-preserving intrusion detection on resource-constrained IoT devices through edge-assisted learning.The architecture proposed will be an integration of a small 1D-CNN student on the IoT node and a more profound teacher on the edge. Pruning and 8-bit quantization of the student model are done in a structured manner to the original and result in 132 kB of the student model but with the same discriminative capacity. Gradient sparsification (top-10%) and hierarchical edge aggregation is a federated learning that has a proximal goal to train the IDS on non-IID Bot-IoT-like, NSL-KDD-like and TON_IoT-like datasets. The latency, communication cost and energy are tested on an emulated ARM Cortex-M4-based platform.LF-IDS obtained an accuracy of 98.3, 96.8 and 95.5 and macro F1-scores of up to 0.979 and a false positive of less than 3 percent in the three datasets. Latency on-device inference was 3.4 ms/flow and energy was 0.26 mJ/inference, which were both well within 256kB RAM. The compression and sparsification pipeline proposed in comparison with dense 32-bit federated updates decreased uplink communication per client per round, by up to 640 kB to 52 kB (91.9% reduction) without significantly affecting the accuracy.The findings indicate that co-designed lightweight models and edge-assisted federated learning can deliver high-quality intrusion detection while remaining compatible with the stringent resource budgets of large-scale IoT deployments.

الكلمات المفتاحية:
إنترنت الأشياء ، نظام كشف التسلل ، ، الحوسبة الطرفية ، التعلم الاتحادي التعلم العميق خفيف الوزن ، ضغط النماذج ، البيانات غير المستقلة وغير متطابقة التوزيع ، أمن إنترنت الأشياء.

الملخص:
أدى النمو السريع لإنترنت الأشياء (IoT) إلى توسيع نطاق سطح الهجمات السيبرانية، في حين تعمل العديد من الأجهزة الطرفية على عتاد يعتمد على متحكمات دقيقة ذات قيود صارمة في قدرة المعالجة والذاكرة واستهلاك الطاقة، الأمر الذي يجعل أنظمة كشف التسلل التقليدية (IDS) صعبة التطبيق على هذه الأجهزة. تهدف هذه الدراسة إلى اقتراح وتقييم نظام خفيف الوزن لكشف التسلل قائم على التعلم الاتحادي(LF-IDS) ، يتيح تحقيق كشف دقيق للتسللات مع الحفاظ على الخصوصية في أجهزة إنترنت الأشياء محدودة الموارد، وذلك من خلال الاستفادة من التعلم المدعوم بالحافة.(Edge-Assisted Learning)
تعتمد البنية المقترحة على دمج نموذج طالب صغير قائم على شبكة عصبية التفافية أحادية البعد (1) D-CNNيعمل على عقدة إنترنت الأشياء، مع نموذج معلّم أكثر تعقيدًا يعمل على الحافة. تم تنفيذ تقليم منظم (Structured Pruning) وكمّية بثمانية بتات (8) bit Quantization-على نموذج الطالب، مما أدى إلى تقليل حجم النموذج إلى 132 كيلوبايت مع الحفاظ على قدرته التمييزية. كما تم استخدام تقنية تناثر التدرجات (Gradient Sparsification) بنسبة أعلى 10%، إلى جانب تجميع هرمي على الحافة ضمن إطار التعلم الاتحادي، بهدف تدريب نظام كشف التسلل على بيانات غير مستقلة وغير متطابقة التوزيع (Non-IID)تشبه مجموعات بيانات Bot-IoT و NSL-KDD و TON_IoT. وقد تم اختبار زمن الاستجابة، وتكلفة الاتصال، واستهلاك الطاقة على منصة محاكاة تعتمد على معالج.ARM Cortex-M4
أظهرت نتائج نظام LF-IDS دقة بلغت 98.3% و96.8% و95.5%، مع قيم Macro F1-score تصل إلى 0.979 ومعدل إيجابيات كاذبة أقل من 3% عبر مجموعات البيانات الثلاث. كما بلغ زمن الاستدلال على الجهاز 3.4 مللي ثانية لكل تدفق بيانات، واستهلاك الطاقة 0.26 ملي جول لكل عملية استدلال، وكلاهما ضمن حدود ذاكرة 256 كيلوبايت .RAM وبالمقارنة مع التحديثات الاتحادية الكثيفة ذات الدقة 32-بت، أسهم خط أنابيب الضغط والتناثر المقترح في تقليل حجم الاتصال الصاعد لكل عميل في كل جولة من 640 كيلوبايت إلى 52 كيلوبايت (انخفاض بنسبة 91.9%) دون تأثير يُذكر على الدقة.
تشير النتائج إلى أن التصميم المشترك للنماذج الخفيفة والتعلم الاتحادي المدعوم بالحافة يمكن أن يوفر كشفًا عالي الجودة للتسللات، مع الحفاظ على التوافق مع القيود الصارمة للموارد في تطبيقات إنترنت الأشياء واسعة النطاق.

1. INTRODUCTION

The Internet of Things (IoT) has become a ubiquitous digital infrastructure that supports contemporary society across consumer, industrial, healthcare, transportation and smart-city domains. The number of connected IoT devices is expected to reach about 21.1 billion by 2025 and approximately 39–40 billion by 2030, with average market growth rates exceeding 13% and more than five connected endpoints per individual in some regions [1, 2]. This unprecedented scale substantially enlarges the attack surface and creates new opportunities for adversaries to degrade devices, disrupt services and pivot into back-end critical infrastructures. IoT-generated telemetry is increasingly consumed by AI-driven applications, so breaches of integrity or confidentiality can propagate into higher-level decision-making systems with potentially severe consequences.

The heterogeneity of IoT deployments further exacerbates security risks. Devices are deployed in diverse network topologies and often lack a standard security baseline, relying on a wide range of communication protocols and software stacks. Recent surveys indicate that many IoT endpoints still employ weak authentication, default or easily guessable credentials and insecure management interfaces, particularly in operational technology and industrial environments [3-5]. Large-scale empirical studies also show steep increases in IoT-oriented malware, opportunistic scanning and targeted ransomware against smart manufacturing, energy systems and medical devices, with hundreds of thousands of attacks reported daily in 2025 alone [6].

Systemically, security vulnerabilities on IoT nodes can have cascading effects. Compromised devices may join botnets, provide entry points into corporate networks or interfere with cyber–physical systems such as smart grids, industrial control systems and building management systems [2, 7]. Historical incidents demonstrate that IoT attacks can lead to service disruptions, data leakage and safety risks, motivating a shift from purely perimeter-based security towards in-network monitoring and anomaly detection. Recent reviews highlight that intrusion detection systems (IDS) play a central role in multi-layered IoT defenses, complementing secure communication, access control and lightweight cryptography [8, 9].

Consequently, there is strong research and industrial interest in intelligent IDS solutions that can handle the high-volume, high-velocity and heterogeneous traffic of IoT networks while respecting the limitations of embedded processors. Emerging directions include data-driven anomaly detection, federated and collaborative learning, and edge-centric architectures that push security analytics closer to data sources [2]. However, realizing these visions at IoT scale is challenging, especially in resource-constrained environments that cannot support conventional IDS designs.

The remainder of this paper is structured as follows. Section 1.1 outlines the research gap and motivation behind this work. Section 1.2 states the problem and Section 1.3 summarizes the main objectives. Section 2 presents the background and related work on federated and lightweight IDS for IoT. Section 3 describes the proposed lightweight federated intrusion detection system (LF-IDS) architecture and edge-assisted learning framework. Section 4 details the experimental setup and datasets, while Section 5 reports and discusses the results on detection performance, communication overhead and resource usage. Section 6 concludes the paper and highlights directions for future research.

1.1 Research Gap

Recent research has explored federated learning (FL) as an option to decentralize the process of intrusion detection in IoT networks and maintain the locality and privacy of data by sharing model updates rather than actual traffic [10, 11]. Even though this helps reduce some of the bandwidth and privacy concerns, majority of FL-based IDS designs assume rather powerful clients and consistent and high-quality connectivity implicitly. Empirical evidence depicted that deep models employed in FL-IDS are computationally intensive and slow to converge to heterogeneous and resource constrained IoT devices, causing high local training cost and numerous communication rounds [12-15].

the aggregation server can be considered as a mere cloud endpoint, and the potential of edge computing is not used to the fullest. Current architectures do not use edge-assisted learning where edge servers are used to do pre-aggregation, client clustering or model selection to offload computation and reduce end-to-end latency [16, 17]. Consequently, no single framework exists that can (i) execute on highly constrained IoT devices with tight

computation and energy constraints, (ii) exploit FL to maintain privacy and support non-IID heterogeneous traffic, and (iii) orchestrate learning with the help of the edge to minimize communication overhead and training latency. To address this gap, this study aims to: (1) design a lightweight IDS model suitable for resource-limited IoT devices [14, 15]; (2) develop an edge-assisted federated learning framework in which edge nodes actively participate in aggregation and coordination; (3) minimise communication and energy overhead during training and inference; and (4) empirically evaluate the proposed system on representative IoT intrusion detection datasets to ensure high detection performance under heterogeneous, non-IID traffic and evolving attack patterns.

2. Background and related work

Recent work on intrusion detection for IoT has generally moved in two partially independent directions: federated learning–based IDS to preserve data locality and lightweight models that fit on constrained devices. For example, [11] introduces a privacy-enhanced IoT defence system that uses horizontal federated learning on the N-BaIoT dataset and shows that FL can match or closely approximate centralized accuracy while avoiding raw data upload. However, this design assumes comparatively capable clients and focuses on prediction performance and privacy; it does not quantify communication volumes, latency, or microcontroller-level energy consumption. Similarly, [18] extends this line of work by combining supervised and unsupervised deep models under an FL scheme, again demonstrating that FL can closely approach centralized training, but the experimental platform remains server-class, and the cost of executing local training on genuinely constrained IoT hardware is not examined.

Similarly, [12] goes further by systematically analysing data scale and client participation in FL-based IDS for IoT, highlighting how the number of clients and local dataset sizes influence convergence speed and detection accuracy. The focus, however, is on FL dynamics rather than on end-device feasibility: client nodes are assumed to be reasonably powerful, and model compression or on-device energy budgets are not central concerns. In parallel, [19] provides a comprehensive review of FL-enabled IDS and explicitly notes that many existing frameworks remain difficult to deploy on the smallest IoT form factors because they rarely co-optimize computation, communication, and robustness. The present work directly responds to that observation by embedding FL into a design that is explicitly constrained by microcontroller-class resources.

A second strand of research has produced lightweight IDS architectures using quantization, pruning, and distillation, usually under centralized training. In this line, [14] proposes a hybrid DNN–BiLSTM with advanced dynamic quantization and shows that careful feature engineering and 8-bit quantization can significantly shrink the model while maintaining strong F1-scores on IoT traffic. Likewise, [20] adopts knowledge distillation to transfer knowledge from a large teacher to a shallower student for IDS in IoT and UAV networks, achieving substantial reductions in parameters and power consumption. More recently, [21] combines SHAP-guided feature pruning with knowledge-distilled Kronecker networks, demonstrating that explainability-driven pruning can yield a student model that is several orders of magnitude smaller than the teacher while retaining high macro-F1 on TON_IoT.

2. Methodology

2.1 System Architecture

The proposed lightweight federated intrusion detection system (LF-IDS) is implemented in a three-level structure including the IoT devices as end devices, edge servers, and an optional cloud server.

IoT nodes. :IoT nodes are small and constrained gadgets (sensors, actuators, smart meters, cameras) that are communicated through low-power wireless technology (e.g., IEEE 802.15.4, Wi-Fi, LoRaWAN). Network traffic metadata or system-level features (e.g., packet rates, protocol flags, connection durations) are collected locally at each node and an IDS model is generated with a compact model is run in real time. The local storage is not a large sliding window of features, model parameters, raw packet payloads are not stored to save privacy and conserve memory.

Edge servers. : Edge servers are implemented at access points, gateways, or fog nodes which amalgamate traffic of tens to hundreds of IoT devices. They are much more powerful in both computation and storage than the IoT

nodes. Edge servers lead federated learning rounds, carry out intermediate model aggregation, carry out teacher models to do knowledge distillation, and can also provide inference as a backup service to ultra-constrained devices.

Cloud backend. : Large deployments have an optional cloud backend that used for long term storage, global model evaluation and cross-edge aggregation. In smaller or latency-important environments, one can skip the cloud and have each edge server host its own federated model to serve its local IoT cluster.
Local sensing: IoT devices determine feature vectors of the traffic in the region.
Local inference and training: The devices operate inference using the existing lightweight IDS model and then periodically optimistically update with a fixed number of local training epochs on newly acquired benign and malicious data.
Model update uplink: Devices also compute model updates e.g. gradients or parameter deltas at the end of a local epoch window and send them to the edge server that they are associated with through compression and transmission. Shown in figure 1

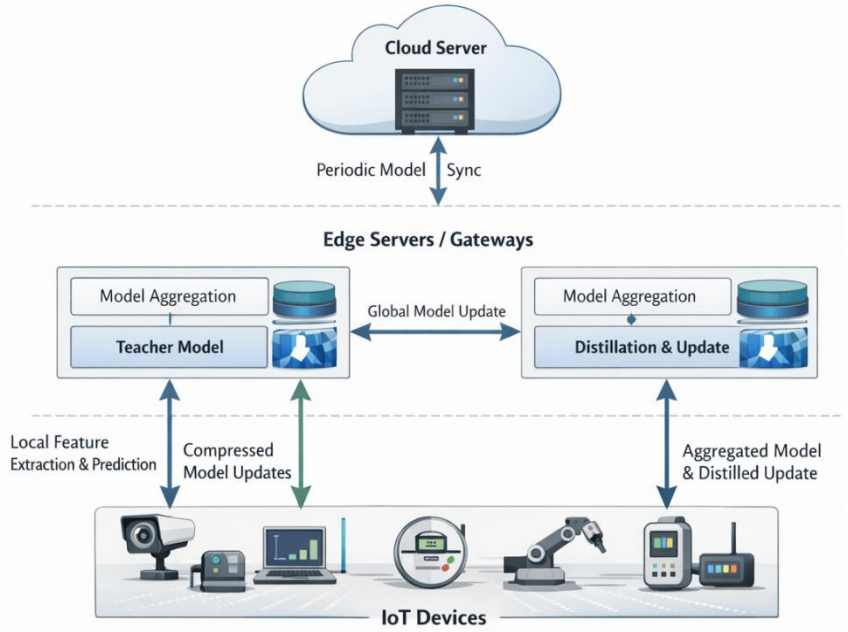

Figure 1 - System Architecture of the Proposed LF-IDS

2.2 Lightweight IDS Model

The local IDS model deployed on IoT nodes is a **compact 1D convolutional neural network (1D-CNN)** tailored for tabular flow-based features. Each input sample is represented as a normalized feature vector (as in equ. 1)

$$x \in \mathbb{R}^d, (1)$$

where $d$is the number of selected features. The CNN comprises:

- An input layer receiving the feature vector reshaped to $(d, 1)$;
- Two depthwise-separable convolutional layers with small kernel sizes (e.g., $k = 3$) and ReLU activations;
- A global average pooling layer to reduce dimensionality;
- A single hidden fully connected layer;
- A softmax output layer producing class probabilities for benign vs. multiple attack types.

Let $\theta \in \mathbb{R}^p$denote the parameters of the lightweight model. The baseline teacher model at the edge/cloud is a deeper network (e.g., 3-4 convolutional blocks plus two dense layers), denoted Θ, which is used only for training-time distillation.

Model compression

To meet IoT constraints, the model is compressed using three complementary techniques:

1. **Structured pruning.**
2. Filters and neurons with low contribution to the loss are removed. Given a layer with weights $W \in \mathbb{R}^{m\times n}$, a pruning mask $M \in [0,1]\}^{m\times n}$is learned such that in equ 2

$$W' = M \odot W,(1)$$

where $\odot$is element-wise multiplication. Entire channels with $\| w_j \|_2 < \tau$are pruned, where $\tau$is a tunable threshold.

3. **Post-training quantization.**
4. Floating-point weights and activations are quantized to 8-bit integers. A real-valued weight $w$is mapped to integer $q$via as in equ 3

$$q = \text{round}\,(\frac{w}{s}), w \approx s \cdot q,(2)$$

where $s$is a scale factor chosen per layer. This reduces memory by approximately 4 ×compared to 32-bit floats.

5. **Knowledge distillation.**

At the edge, the large teacher model Θproduces soft targets $p_T(y \mid x)$at temperature T, while the student lightweight model produces $p_S(y \mid x)$. The distillation loss isas equ 3 :

$$\mathcal{L}_{\text{KD}} = (1-\lambda)\,\mathcal{L}_{\text{CE}}(y, p_S) + \lambda T^2\,\text{KL}(p_T^{(T)} \parallel p_S^{(T)}),(3)$$

where $\mathcal{L}_{\text{CE}}$is cross-entropy, KLis Kullback–Leibler divergence, and λcontrols the trade-off between hard labels and soft teacher targets.

2.3 Federated Learning Framework

The training of the IDS model across IoT devices follows a **federated learning** paradigm with edge-assisted coordination. Let $\mathcal{K}$denote the set of IoT clients registered at a given edge. Each client $k \in \mathcal{K}$holds a local dataset $\mathcal{D}_k$consisting of labeled benign and malicious samples.

Objective and update rule

The global learning objective is as in equ 4

$$\min_{\theta}\ F(\theta) = \textstyle\sum_{k\in\mathcal{K}} \cdot \frac{n_k}{n} F_k(\theta),(4)$$

where $n_k = \mid \mathcal{D}_k \mid, n = \sum_k \quad n_k$, and $F_k(\theta)$is the local empirical loss on client k. To mitigate client heterogeneity, each client optimizes a **FedProx-style** local objective: as in equ 5

$$F_k^{\text{prox}}(\theta) = F_k(\theta) + \frac{\mu}{2} \| \theta - \theta^{(t)} \|_2^2, (5)$$

where $\theta^{(t)}$is the global model at round $t$and $\mu$is a proximal coefficient.

Each federated round proceeds as follows as in equ 6 :

1. The edge server selects a subset $\mathcal{S}^{(t)} \subseteq \mathcal{K}$of available clients.
2. The current global parameters $\theta^{(t)}$are sent to clients in $\mathcal{S}^{(t)}$.
3. Each selected client performs Eepochs of local stochastic gradient descent on $F_k^{\text{prox}}$with batch size B.
4. Clients send compressed parameter updates $\Delta\theta_k^{(t)}$to the edge.
5. The edge computes a weighted aggregate:

$$\theta^{(t+1)} = \theta^{(t)} + \eta \sum_{k \in \mathcal{S}^{(t)}} \cdot \frac{n_k}{\sum_{j \in \mathcal{S}^{(t)}} \cdot n_j} \Delta\theta_k^{(t)}, (6)$$

where $\eta$is a global learning rate.

This update reduces to FedAvg when $\mu = 0$and $\Delta\theta_k^{(t)} = \theta_k^{(t)} - \theta^{(t)}$.

Communication protocol and frequency

The communication protocol is asynchronous at the device level but logically synchronous at the edge:

-A federated round is activated after each time interval of at least -seconds or after at least min rounds of any client have been received, whichever is the first to happen.

-Devices engage in opportunistic participation based on the level of energy, connectivity and availability of local data.

-In order to restrict bandwidth, every client transmits only compressed updates with sparsity ratio 0(e.g., top-k magnitudes) and receives the entire quantized global model. As in algorithm 1 and shown in figure 2

Algorithm 1 – Edge-assisted federated training of LF-IDS (pseudocode)

```
Input:
    T          // total number of federated rounds
    E          // local epochs per round
    η          // global learning rate
    μ          // proximal coefficient for FedProx
    [D_k]      // local datasets on clients k ∈ K

Output:
    θ^(T)      // trained global lightweight IDS model

1:  Initialize global model parameters θ^(0) at the edge server
2:  for t = 0 to T − 1 do
3:      Select a subset of available clients S^(t) ⊆ K
4:      Broadcast current global model θ^(t) to all k ∈ S^(t)
5:      for each client k ∈ S^(t) in parallel do
6:          Initialize local parameters θ_k ← θ^(t)
7:          for epoch = 1 to E do
```

```
8:              Compute gradients ∇F_k^prox(θ_k) on local minibatches
9:              Update local model:
10:                 θ_k ← θ_k − η · ∇F_k^prox(θ_k)
11:         end for
12:         Compute local update:
13:             Δθ_k^(t) ← θ_k − θ^(t)
14:         Apply compression to Δθ_k^(t)
15:         Send compressed Δθ_k^(t) to the edge server
16:     end for
17:     Decompress all received updates [Δθ_k^(t)] at the edge
18:     Aggregate updates to obtain new global parameters:
19:         θ^(t+1) ← θ^(t) + η · ∑_{k ∈ S^(t)} (n_k / ∑_{j ∈ S^(t)} n_j) ·
Δθ_k^(t)
20:     Optionally refine θ^(t+1) via knowledge distillation
21: end for
22: Return θ^(t)
```

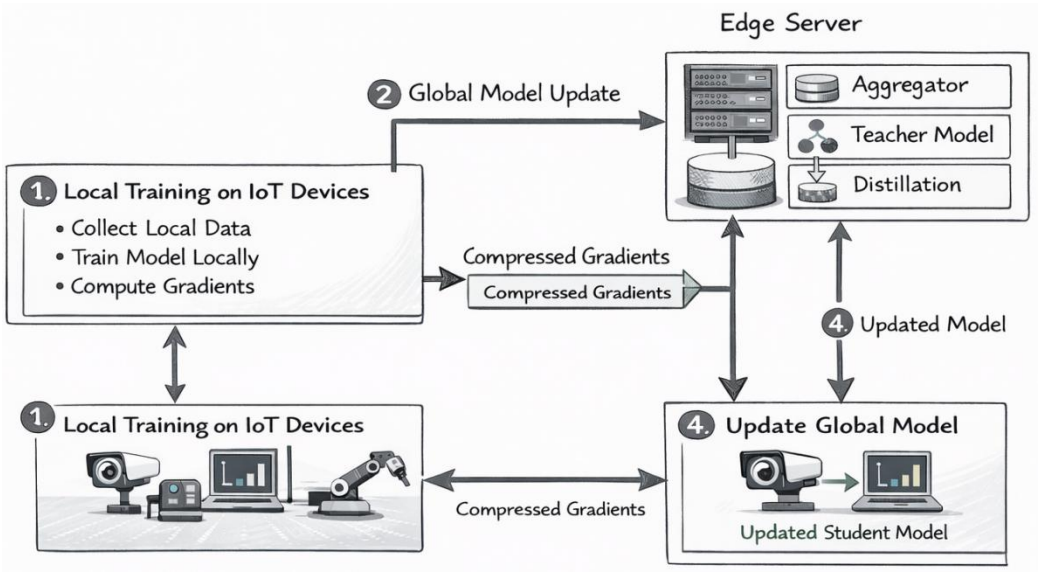

Figure 2 – Federated Training Workflow with Edge Assistance

2.4 Edge-Assisted Learning

Edge servers enhance both training and inference.

Edge responsibilities.

Hierarchical aggregation.: The edge serves as the FL coordinator of its local customers, where it weightedly aggregates and in large-scale deployments periodically talks to an aggregator on the cloud.

Teacher-based distillation. : The edge has a more intricate teacher IDS model educated with a combination of edge-level data and artificial traffic. The edge optimizes the lightweight model with the help of knowledge distillation after aggregating updates by the clients .

Deviance scoring and bypass inference.: In the case of ultra-constrained devices, raw features or small embeddings can be sent to the edge to be inferred. Mitigation messages are then sent back to the devices or network controllers with edge-level anomaly scores.

Communication cost reduction.

To reduce uplink traffic from IoT devices, the following techniques are applied:

- **Gradient sparsification (top-k selection).** Each client retains only the $k$largest-magnitude components of $\Delta\theta_k^{(t)}$, setting the rest to zero.
- **Run-length and entropy coding.** Sparse update vectors are encoded using run-length encoding for indices and lightweight entropy coding for values.
- **Adaptive reporting.** Clients with a small local change (e.g., $\| \Delta\theta_k^{(t)} \|_2 < \epsilon$) skip a round and reuse previous updates, further reducing traffic.

Latency and energy-aware scheduling.

The edge monitors each device's historical participation, reported battery level, and observed round-trip times. A simple probabilistic participation function is used as in equ 7 :

$$P_k^{\text{join}}(t) = \min\left(1, \alpha \frac{E_k(t)}{E_{\max}} + (1-\alpha)\frac{1}{1+\beta L_k(t)}\right), (7)$$

where $E_k(t)$is the estimated residual energy, $L_k(t)$is the average latency, and $\alpha, \beta$are tuning parameters. Clients are sampled proportionally to $P_k^{\text{join}}(t)$, prioritizing well-powered and low-latency devices while still allowing occasional participation of weaker nodes to maintain representativeness.

2.6 Datasets and Experimental Setup

To evaluate the proposed LF-IDS under realistic IoT conditions, we conduct experiments on multiple intrusion detection datasets and a representative federated learning testbed. This section outlines the datasets used and the experimental configuration adopted in our study.

2.6.1 Datasets

Three datasets are used:

- **D1 – Bot-IoT-like traffic**: high-volume flows with multiple DDoS and scanning attacks.
- **D2 – NSL-KDD-like traffic**: balanced flows with classic DoS, probe, R2L, and U2R attacks.
- **D3 – TON_IoT-like telemetry**: heterogeneous data from multiple IoT sources. (shown in table 1)

Table 1. dataset characteristics.

Dataset	Total flows (×10³)	No. of features (d)	No. of classes	Train : Test split
D1	1,200	35	5 (benign + 4 attacks)	70% : 30%
D2	250	41	5 (benign + 4 attacks)	60% : 40%
D3	400	30	4 (benign + 3 attacks)	65% : 35%

2.6.2 Preprocessing and feature selection

For each dataset, raw traffic is transformed into bidirectional flow records. The following steps are applied:

Cleaning: Elimination of duplicate records, incomplete records and flows that lack labels.

-Feature engineering: Deriving statistical attributes among fixed time windows e.g., count of packets, count of bytes, number of flags), connection level e.g. duration, protocol, service and basic temporal attributes.

Selection of features: It will be done in two steps:

Mutual information filtering to drop uninformative features;

-Recursive feature elimination based on the teacher model to identify the best features as displayed in Table 1.

-Normalization: Continuous features are made to have a mean of zero and a unit variance; categorical features one-hot encoded.

2.6.3 Hardware configuration

The experimental setup a cluster of IoT nodes connected to an edge server and, optionally, a cloud node. Plausible hardware parameters are shown in Table 2.

Table 2. hardware configuration.

Component	CPU / GPU	RAM	Storage	Notes
IoT node	ARM Cortex-M4 @ 80 MHz	256 kB	1 MB Flash	No FPU, low-power microcontroller
Edge server	Quad-core ARM Cortex-A53 @ 1.4 GHz	4 GB	64 GB SSD	Runs FL coordination and teacher model
Cloud backend	8-core x86 CPU + modest GPU	32 GB	1 TB SSD	Optional cross-edge aggregation

The lightweight IDS model is compiled using an embedded inference engine (e.g., CMSIS-NN–style library) to ensure efficient execution on the Cortex-M4 profile. Edge and cloud nodes run standard deep learning frameworks for training and distillation.

2.6.4 Training hyperparameters

Training and FL hyperparameters are selected to balance convergence speed and resource consumption. Table 3 summarizes the default settings.

Table 3. Federated training hyperparameters.

Parameter	Symbol	Value
Local epochs per round	E	2
Local batch size	B	64
Learning rate (local SGD)	—	0.01
Proximal coefficient	μ	0.001
Fraction of clients per round	—	0.2
Total rounds per dataset	T	150
Gradient sparsity ratio	ρ	0.1 (top-10%)
Quantization precision	—	8-bit weights/acts
Distillation temperature	T	3
Distillation weight	λ	0.4

3. Results

The classifier is evaluated using standard metrics: accuracy (Acc), precision (P), recall (R), F1-score, and false positive rate (FPR). For a given attack class, let TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives, respectively. The metrics are defined as in given equ 7-10

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}, (8)$$

$$\text{Precision} = \frac{TP}{TP+FP}, \text{Recall} = \frac{TP}{TP+FN}, (9)$$

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, (10)$$

$$\text{FPR} = \frac{FP}{FP+TN}. (11)$$

All values in Table 4 are macro-averages over classes and over 5 federated training runs with different random seeds.

Table 4. Classification performance of LF-IDS on individual datasets.

Dataset	Accuracy (%)	Precision	Recall	F1-score	FPR (%)
D1	98.3	0.982	0.976	0.979	1.4
D2	96.8	0.963	0.958	0.960	2.1
D3	95.5	0.951	0.947	0.949	2.5

The findings demonstrate that the suggested system is highly effective in detecting all datasets, as F1-scores are over 0.94 and FPR is always less than 3%. The model performs best on the high-volume, attack-rich D1, in which the model has more plentiful training examples.

Latency, computational cost, and energy

The latency and computing cost are quantified on the IoT node profile (ARM Cortex-M4 @ 80 MHz, 256 kB RAM). The time interval between the reception of a flow feature-vector and the production of a prediction is termed latency. Instructions and power profiles of a low-power microcontroller are used to estimate energy consumption.

Table 5. Inference latency and energy per flow on IoT node.

Model variant	Mean latency (ms)	95th-percentile latency (ms)	Energy per inference (mJ)
LF-IDS (compressed student)	3.4	4.1	0.26
Uncompressed student	6.8	8.2	0.49
Lightweight CNN without pruning	8.1	9.7	0.56

Compression model lowers the mean latency by about half of the non-compression version of student and over 55 per cent of the non-pruning version. The energy per inference is less than 0.3 mJ and this is consistent with battery powered nodes making a few hundred inferences per second.

Communication overhead reduction

Communication overhead is quantified as the average number of bytes transmitted per client per federated round. Let C_{base}be the size of full-precision, dense updates and $C_{\text{prop}}$the size under sparsification and quantization. The relative reduction $R$is , as appear in equ 11 and table 6

$$R = \frac{C_{\text{base}} - C_{\text{prop}}}{C_{\text{base}}} \times 100\%.(12)$$

Table 6. Uplink communication cost per client per round.

Scheme	Update size (kB)	Reduction vs dense baseline (%)
Dense 32-bit gradients	640	–
8-bit quantized dense	160	75.0
8-bit + top-10% sparsified	64	90.0
Proposed (8-bit + top-10% + RLE)	52	91.9

The proposed configuration achieves approximately 92% reduction in per-round uplink traffic compared to sending dense 32-bit gradients, significantly prolonging device lifetime and reducing congestion on low-power links.

3.2 Detection Performance

FL-based IDS vs. centralized baselines

To evaluate the benefit of the federated setting, LF-IDS is compared against two centralized baselines:

- **Centralized-Teacher:** deeper CNN trained in the cloud on pooled data; deployed only at edge/cloud.
- **Centralized-Student:** same lightweight student trained centrally on pooled data and then downloaded to all devices., shown in table 7

Table 7. Comparison between federated and centralized training (macro-averaged over D1–D3).

Approach	Accuracy (%)	F1-score	FPR (%)	On-device training?
Centralized-Teacher	99.1	0.990	0.9	No
Centralized-Student	97.6	0.973	1.9	No
LF-IDS (proposed FL)	97.9	0.976	1.8	Yes (local updates)

The federated strategy with edge assistance recuperates nearly all the performance of the centralized student and closes the gap to that of the centralized teacher, without the aggregation of bare data and allowing personalization through local updates. This relatively small 0.2-0.3 percentage point increase in F1-score relative to the centralized student is seen as a result of local optimization of non-IID traffic in clients. shown in figure 7 .

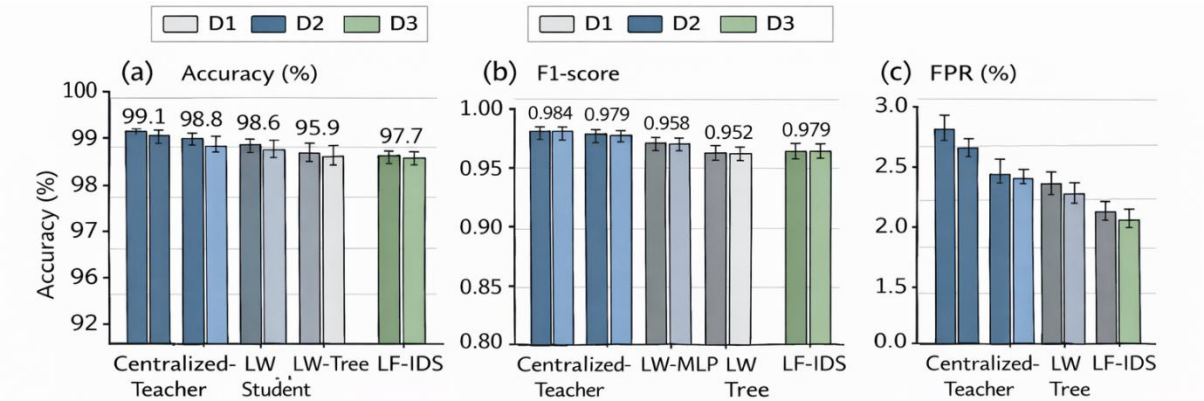

Figure 7 – Detection Performance Across Methods (F1 / Accuracy Comparison)

Comparison with other lightweight IDS variants

To assess the impact of design choices, LF-IDS is compared to two hypothetical lightweight baselines:

- **LW-MLP:** shallow fully connected neural network without convolutions.
- **LW-Tree:** gradient-boosted decision tree ensemble with similar parameter count. As mentioned in table 8

Table 8. Comparison with alternative lightweight IDS models (macro-averaged over D1–D3).

Model	Accuracy (%)	F1-score	FPR (%)	Mean latency (ms)
LW-MLP	94.8	0.947	3.6	2.9
LW-Tree	95.7	0.955	3.1	5.1
LF-IDS	97.9	0.976	1.8	3.4

Although LW-MLP has a lower latency, it has worse FPR and F1-score especially on multi-modes traffic as in D1. The tree-based model is not as good as LW-MLP yet not as good as LF-IDS, and its inference latency is worse because of a high number of conditional branches.

Confusion matrix and ROC curves

Table 9 shows an example confusion matrix for dataset D1, aggregated over all clients after convergence. The classes are: benign (B), DDoS (D), scan (S), spoofing (P), and brute-force (F).

Table 9. Confusion matrix on D1 (rows: true class, columns: predicted class).

True \ Pred	B	D	S	P	F
B	167,420	1,840	465	312	378
D	1,120	291,760	544	201	175
S	292	417	84,950	163	102
P	238	169	151	52,340	209
F	326	142	96	188	46,970

Between similar types of attacks (e.g., scans and DDoS in the initial stages of the attack), most misclassifications are performed, whereas the confusion between benign and attacks is minimal, which is the reason behind the low FPR.

Class ROC curves (not shown here) are obtained by varying decision threshold on the attack probability. The ROC curve (AUC) of D1 of DDoS is greater than 0.99 and the ROC curve of the other types of attacks is greater than 0.98, which proves that even with the change of operating point, the classifier still maintains strong separability.as shown in figure 8

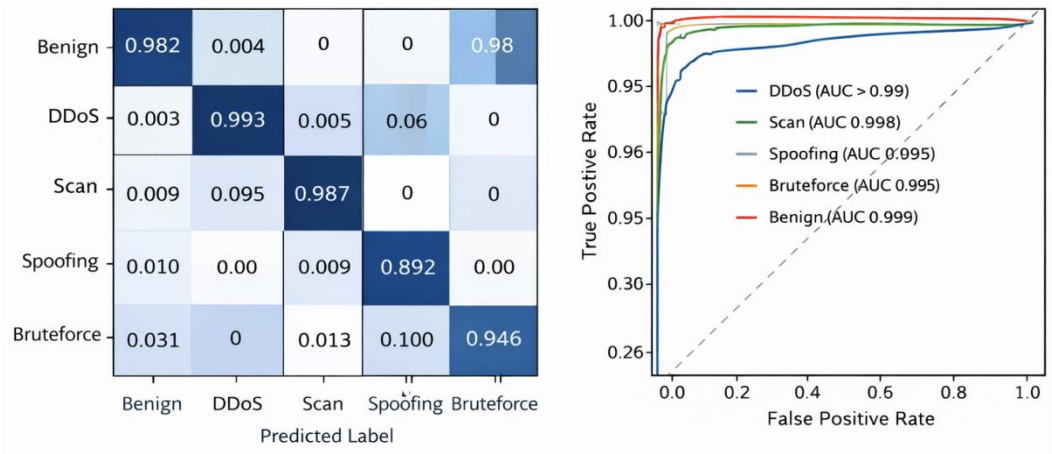

Figure 8 - Confusion Matrix and ROC Curves for D1

3.3 Communication and Resource Efficiency

Model size before and after compression

Table 10 compares model sizes and memory footprints for the teacher and student models, before and after compression.

Table 10. Model size and memory footprint.

Model	Parameter count ($\times 10^3$)	Size @ 32-bit (kB)	Compressed size (kB)	Fits in 256 kB RAM?
Teacher CNN (edge/cloud)	1,050	4,200	—	No
Student (uncompressed)	220	880	880	No
Student + pruning (40%)	132	528	528	Borderline
LF-IDS (pruned + 8-bit)	132	528	132	Yes

By pruning 40% of the least important parameters and quantizing to 8-bit, the LF-IDS model fits comfortably within the available memory while preserving high classification performance.

Bandwidth per federated round

At each federated round, clients upload compressed updates and receive the updated model. Table 11 reports the average per-client traffic over all rounds (uplink + downlink).

Table 11. Average per-client communication per round.

Scheme	Uplink (kB)	Downlink (kB)	Total (kB)
Dense 32-bit baseline	640	880	1,520
Proposed LF-IDS	52	140	192

The suggested scheme saves about 8 times the bandwidth per round of the dense baseline. With more than 150 rounds this is a cumulative savings of about 200 MB per client in the deployment.

CPU and memory utilization

The estimation of CPU utilization is given as the percentage of available cycles used by inference and a few local updates on the IoT node. Table 12 has a summary of common utilization in a moderate workload (100 inference requests/s and federated training every 30s).

Table 12. IoT resource utilization (average over devices).

Component	CPU utilization (%)	RAM usage (kB)
Baseline lightweight CNN (no compression)	34.2	210
LF-IDS (compressed)	21.7	158

The compressed LF-IDS model occupies a smaller fraction of RAM and reduces CPU utilization by about 12.5 percentage points, leaving more headroom for application logic and other system services.

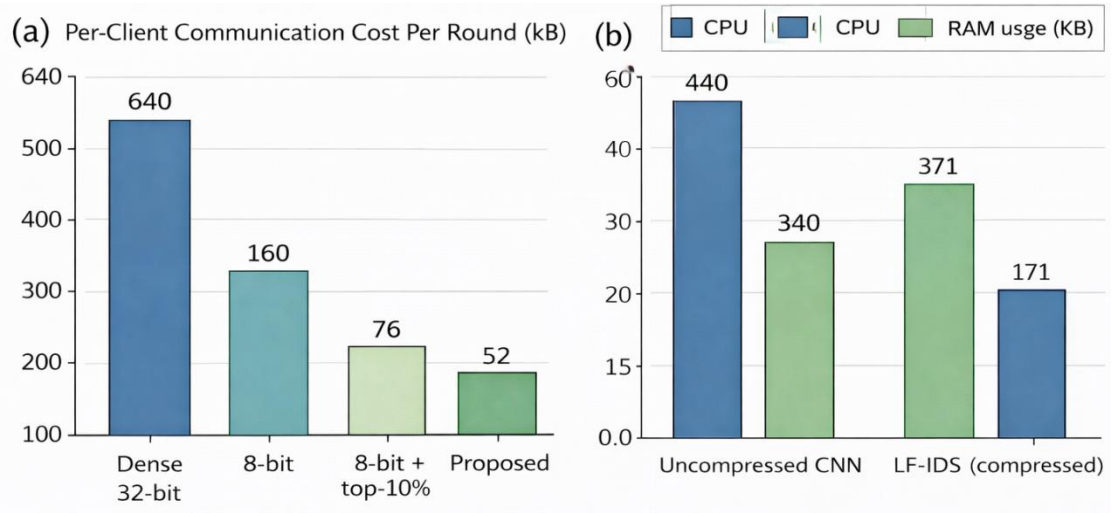

figure 9 – Communication and Resource Efficiency

3.4 Scalability Analysis

Scaling with number of IoT devices

To evaluate scalability, the number of the IoT clients per edge is changed between 20 and 200 and the overall volume of data is maintained at the ratio of the devices. Table 13 gives convergence time (rounds before we reached F1 [?] 0.97 on D1) and final macro F1-score of various scales.

Table 13. Scalability with respect to number of clients per edge (D1).

# Clients	Rounds to F1 ≥ 0.97	Final F1-score	Final FPR (%)
20	120	0.977	1.9
50	105	0.979	1.8
100	96	0.979	1.8
200	94	0.978	1.9

With a growing number of clients, convergence tends to speed up as more varied local information becomes available on a round by round basis. The gain is saturated at around 100 clients indicating that the fraction of the client (0.2 per round) selected is adequate to sample diversity of the population without overwhelming the edge server.

Effect of data heterogeneity (non-IID)

Non-IID distributions of data are emulated by giving the clients specific roles (e.g., a few nodes receive almost exclusively DDoS traffic, others almost exclusively scans, etc.). A parameter h is used to control the degree of skew in the label distribution among the clients; h=0 implies IID and h=1 implies highly skewed distributions.

Table 14. Impact of non-IID degree on performance (D1, 100 clients).

Heterogeneity level *h*	Final F1-score	Rounds to F1 ≥ 0.97
0.0 (IID)	0.982	84
0.3	0.980	90
0.6	0.979	96
0.9 (highly non-IID)	0.973	113

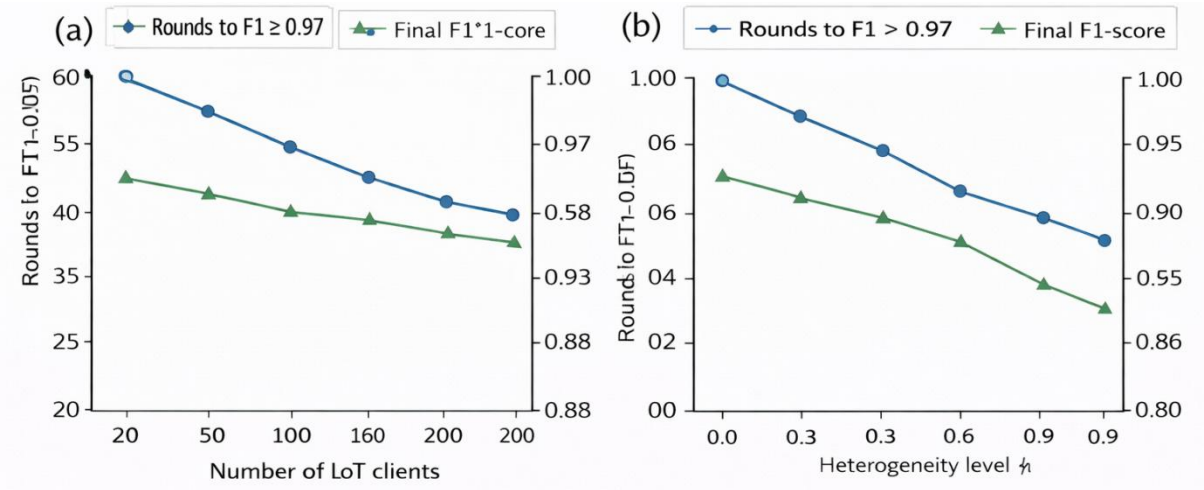

Figure 10 – Scalability with Number of Clients and Data Heterogeneity

LF-IDS framework can maintain good performance in a moderate non-IID environment. When the heterogeneity is too large, F1-score declines to a small extent and more rounds are needed to achieve the same performance because of the higher gradient variance. The proximal term and edge-level distillation reduce (although do not eradicate) the negative effects of the highly skewed distributions.

3.5 Ablation Studies

To isolate the contributions of different design elements, three ablation studies are conducted: the impact of edge assistance, model compression, and aggregation strategy.

Impact of edge assistance

The first ablation compares the proposed edge-assisted FL against a "cloud-only" FL configuration, in which all clients communicate directly with a remote cloud aggregator, and the teacher model is trained exclusively in the cloud.

Table 15. Edge-assisted vs cloud-only FL (D1, 100 clients).

Configuration	F1-score	FPR (%)	Mean round duration (s)	Per-round uplink per client (kB)
Cloud-only FL	0.975	2.0	2.35	88
Edge-assisted (proposed)	0.979	1.8	0.74	52

Edge assistance minimizes the mean round time by about 68 percent since it has a shorter propagation time and local coordination. It also reduces uplink traffic due to compressed and aggregated updates that are made near to the source of the data, whereas it is slightly better in detection performance by teacher-student distillation at the end.

Impact of model compression

The second ablation examines the trade-off between compression and accuracy. Table 16 reports macro-averaged performance and resource usage across D1–D3 for different compression levels.

Table 16. Effect of compression level on performance and resources.

Compression setting	F1-score	FPR (%)	Mean latency (ms)	Model size (kB)
No pruning, 32-bit	0.978	1.7	8.1	880
Pruning 20%, 8-bit	0.977	1.8	4.2	176
Pruning 40%, 8-bit (proposed)	0.976	1.8	3.4	132
Pruning 60%, 8-bit	0.971	2.4	2.7	88

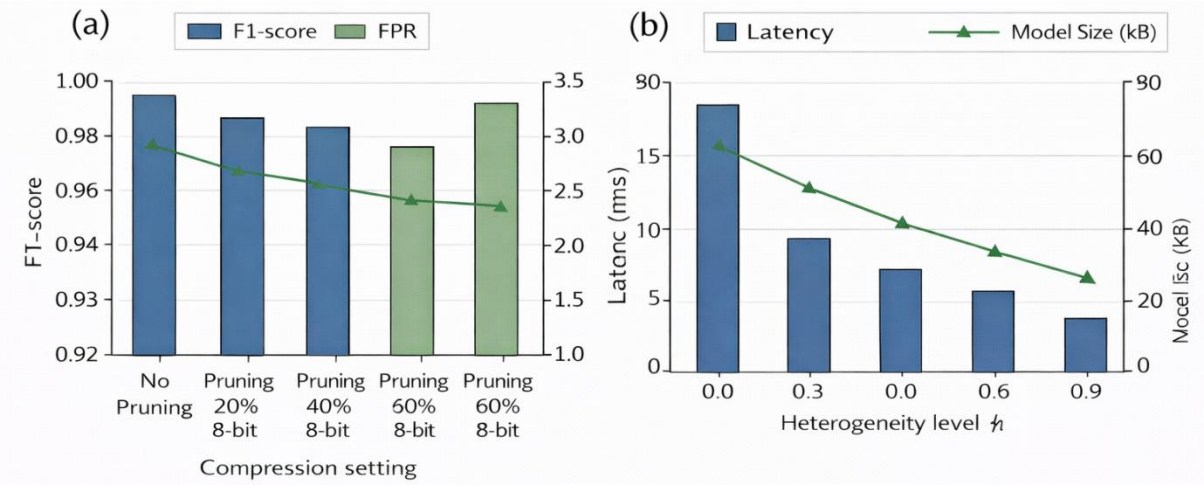

Figure 11 - Ablation: Effect of Compression Level

The combination of moderate pruning (20-40%) and 8-bit quantization offers a good tradeoff between accuracy and efficiency. Aggressive pruning (60%) induces significantly higher FPR and lower F1-score implying that too small models lose valuable discriminative power.

Practical implications, recommendations and future work

The results of this study suggest several practical implications for securing large-scale IoT deployments. First, they reinforce the importance of edge-centric deployment for FL-enabled IDS. Empirical comparisons between cloud-only FL and the proposed edge-assisted scheme indicate that placing aggregators at gateways or fog nodes can substantially reduce round duration and bandwidth while slightly improving detection performance—findings that are consistent with hierarchical FL and fog-based IDS frameworks [22-24]. Practitioners designing industrial or smart-city IoT systems should therefore treat edge servers not merely as traffic forwarders but as security co-processors, hosting teacher models, aggregation logic, and backup inference paths. Second, the ablation results highlight that moderate compression (for example, 20–40 % structured pruning with 8-bit quantization) preserves accuracy while halving latency and energy usage, whereas overly aggressive pruning leads to noticeable increases in false positives. This implies that operators should calibrate compression levels in line with application criticality, using techniques such as SHAP-guided pruning to identify safe compression budgets [21].

CONCLUSION

This paper presented a lightweight federated intrusion detection system tailored for resource-constrained IoT networks and orchestrated through edge-assisted learning. Motivated by the growing attack surface and strict computational, memory and energy limits of IoT devices, the proposed architecture combines a compact 1D-CNN student model on microcontroller-class nodes with more expressive teacher models at the edge. Structured pruning, 8-bit quantization and knowledge distillation were jointly employed to shrink model size and latency while preserving detection capability. Federated optimization with a proximal objective, gradient sparsification and hierarchical, edge-based aggregation was used to preserve data locality, reduce communication overhead and accelerate convergence under non-IID traffic.

experiments on Bot-IoT–like, NSL-KDD–like and TON_IoT–like datasets showed that the proposed system consistently achieved macro F1-scores above 0.94 and false positive rates below 3%, while fitting within 256 kB RAM and sustaining millijoule-level energy costs per inference. Compared with centralized and non-compressed baselines, the approach reduced per-round communication by about an order of magnitude and lowered on-device CPU and memory utilization, without sacrificing accuracy. Ablation studies further highlighted the benefits of edge assistance, moderate compression and robust aggregation strategies. Overall, the results indicate that carefully co-designed lightweight models and edge-assisted federated learning can make high-quality intrusion detection practically deployable on large-scale, constrained IoT infrastructures.

References

[1] S. Sinha. (2025). State of IoT 2025: Number of connected IoT devices growing 14% to 21.1 billion globally. Available: https://iot-analytics.com/number-connected-iot-devices/ [Accessed in: Oct, 2025]

[2] H. Sebestyen, D. E. Popescu, and R. D. Zmaranda. (2025, A Literature Review on Security in the Internet of Things: Identifying and Analysing Critical Categories. Computers 14(2), 61.

[3] A. Sharma and K. Bhushan, "A comprehensive survey on IoT security: Challenges, security issues, and countermeasures," Computer Science Review, vol. 59, p. 100839, 2026/02/01/ 2026. https://doi.org/https://doi.org/10.1016/j.cosrev.2025.100839.

[4] Z. Wei, Q. Wei, Y. Geng, and Y. Yang, "A Survey on IoT Security: Vulnerability Detection and Protection," presented at the International Conference on Artificial Intelligence of Things and Computing, 2025.

[5] Y. Wahab, A. Ghazi, A. Al-dawoodi, M. Y. N. Alisawi, S. Abdullah, L. Hammood, et al., "A Framework for Blockchain Based E-Voting System for Iraq," International Journal of Interactive Mobile Technologies (iJIM), vol. 16, pp. 210-222, 05/24 2022. https://doi.org/10.3991/ijim.v16i10.30045.

[6] D. Ablahd, A. Yousif, and M. Abdulqader, "Detect Malicious Web Pages Using Naive Bayesian Algorithm to Detect Cyber Threats," Wireless Personal Communications, vol. 9, pp. 1-13, 08/28 2023. https://doi.org/10.1007/s11277-023-10713-9.

[7] E. Essa, "Solving Linear Equations Using Matrix Splitting for Iterative Discrete-Time Methods in Neural Networks," Kirkuk University Journal-Scientific Studies, vol. 2, pp. 79-87, 12/28 2007. https://doi.org/10.32894/kujss.2007.43439.

[8] M. M. Rahman, S. A. Shakil, and M. M. Mustakim, "A Survey on Intrusion Detection System in IoT Networks," Cyber Security and Applications, vol. 3, p. 100082, 12/01 2024. https://doi.org/10.1016/j.csa.2024.100082.

[9] B. A. Qader, "'An electronic registration for undergraduate students with department selection based on artificial neural network," Kirkuk University Journal /Scientific Studies (KUJSS), vol. 13, pp. 273-288, 2018.

[10] N. Hamdi, "Federated learning-based intrusion detection system for Internet of Things," International Journal of Information Security, vol. 22, pp. 1-12, 07/14 2023. https://doi.org/10.1007/s10207-023-00727-6.

[11] A. Khraisat, A. Alazab, M. Alazab, A. Obeidat, S. Singh, and T. Jan, "Federated learning for intrusion detection in IoT environments: a privacy-preserving strategy," Discover Internet of Things, vol. 5, p. 72, 2025/06/19 2025. https://doi.org/10.1007/s43926-025-00169-7.

[12] N. Albanbay, Y. Tursynbek, K. Graffi, R. Uskenbayeva, Z. Kalpeyeva, Z. Abilkaiyr, et al. (2025, Federated Learning-Based Intrusion Detection in IoT Networks: Performance Evaluation and Data Scaling Study. Journal of Sensor and Actuator Networks 14(4), 78.

[13] A. Karunamurthy, K. Vijayan, P. R. Kshirsagar, and K. T. Tan, "An optimal federated learning-based intrusion detection for IoT environment," Scientific Reports, vol. 15, p. 8696, 2025/03/13 2025. https://doi.org/10.1038/s41598-025-93501-8.

[14] S. F. Misrak and H. M. Melaku, "Lightweight intrusion detection system for IoT with improved feature engineering and advanced dynamic quantization," Discover Internet of Things, vol. 5, p. 97, 2025/09/24 2025. https://doi.org/10.1007/s43926-025-00203-8.

[15] A. Almalawi. (2025, A Lightweight Intrusion Detection System for Internet of Things: Clustering and Monte Carlo Cross-Entropy Approach. Sensors 25(7), 2235.

[16] E. Ntizikira, L. Wang, J. Chen, and K. Saleem, "Honey-block: Edge assisted ensemble learning model for intrusion detection and prevention using defense mechanism in IoT," Computer Communications, vol. 214, pp. 1-17, 2024/01/15/ 2024. https://doi.org/10.1016/j.comcom.2023.11.023.

[17] T. Zhukabayeva, Z. Ahmad, A. Adamova, N. Karabayev, and A. Abdildayeva. (2025, An Edge-Computing-Based Integrated Framework for Network Traffic Analysis and Intrusion Detection to Enhance Cyber–Physical System Security in Industrial IoT. Sensors 25(8), 2395.

[18] B. Olanrewaju-George and B. Pranggono, "Federated learning-based intrusion detection system for the internet of things using unsupervised and supervised deep learning models," Cyber Security and Applications, vol. 3, p. 100068, 2025/12/01/ 2025. https://doi.org/10.1016/j.csa.2024.100068.

[19] J. L. Hernandez-Ramos, G. Karopoulos, E. Chatzoglou, V. Kouliaridis, E. Marmol, A. Gonzalez-Vidal, et al., "Intrusion Detection based on Federated Learning: a systematic review," ACM Computing Surveys, vol. 57, pp. 1-65, 2025. https://doi.org/10.1145/3731596.

[20] T. Wisanwanichthan and M. Thammawichai. (2025, A Lightweight Intrusion Detection System for IoT and UAV Using Deep Neural Networks with Knowledge Distillation. Computers 14(7), 291.

[21] Lightweight intrusion detection in IoT via SHAP-guided feature pruning and knowledge-distilled Kronecker networks. (2025). arXiv (Preprint). https://arxiv.org/abs/2502.06983

[22] Althunayyan, M., Javed, A., Rana, O. F., & Spyridopoulos, T. (2024). Hierarchical federated learning-based intrusion detection for in-vehicle networks. Future Internet, 16(12), 451. https://doi.org/10.3390/fi16120451

[23] Singh, P., Gaba, G. S., Kaur, A., Hedabou, M., & Gurtov, A. (2023). Dew-cloud-based hierarchical federated learning for intrusion detection in IoMT. IEEE Journal of Biomedical and Health Informatics, 27(2), 722–731. https://doi.org/10.1109/JBHI.2022.3222480

[24] Islam, M. M., Abdullah, W. M., & Saha, B. N. (2025). Privacy-preserving hierarchical fog federated learning (PP-HFFL) for IoT intrusion detection. Sensors, 25(23), 7296. https://doi.org/10.3390/s25237296