أكاديمية كاوست
KAUST ACADEMY

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

Post-Quantum Secure Federated Intrusion Detection for Military UAV Networks

Khalid Alkhaldi, Anas Alshehri, Feras Aloufi,
Meshal Alshawi, Ammar Alkayyal, Dr. Muhammad Shahbaz Khan

khalid.alkhaldi@kaust.edu.sa, anas.shehri@kaust.edu.sa, feras.aloufi@kaust.edu.sa,
meshal.alshawi@kaust.edu.sa, ammar.alkayyal@kaust.edu.sa, engr.msk90@gmail.com

Abstract

Military UAV fleets face growing exposure to network-layer cyberattacks, while centralising the traffic data needed to train detection models creates both privacy risk and a single point of compromise. This project presents a post-quantum secure federated intrusion detection system for five UAVs coordinated by a central control centre. Each UAV trains a detection model locally on its own traffic and transmits only the resulting model updates, which the control centre aggregates by federated averaging into an improved global model. All updates are protected in transit using ML-KEM for key establishment, ML-DSA for authentication, and AES-GCM for confidentiality and integrity. The system improves attack detection across the fleet while keeping local data private and securing communications against tampering and future quantum-capable adversaries.

Date: August 24, 2026
Code: https://github.com/Khxlld/ids-fl-project
Project Page: https://github.com/Khxlld/ids-fl-project

Contents

1 Executive Summary **4**

2 Introduction **4**

2.1 Motivation . . . 4
2.2 Problem Statement . . . 5
2.3 Objectives . . . 5
2.4 Research and Evaluation Questions . . . 5
2.5 Contributions . . . 5
2.6 Scope and Assumptions . . . 6
Data and clients. . . . 6
Modelling. . . . 6
Privacy. . . . 6
Security. . . . 7
Deployment. . . . 7

3 Background and Related Work **7**

3.1 UAV Communication and Security Threats . . . 7
3.2 UAV-Specific Intrusion Detection . . . 7
3.3 Federated Intrusion Detection . . . 8
3.4 Federated Learning under Non-IID Data . . . 9
3.5 Secure and Post-Quantum Federated Learning . . . 9
3.6 Positioning . . . 10

4 System Design **10**

4.1 Use-Case Architecture . . . 10
4.2 Threat Model . . . 10
4.3 Security Requirements . . . 12
4.4 Software Architecture and Technology Stack . . . 12
4.5 Rationale for a Direct Federated Implementation . . . 12
Bit-level reproducibility. . . . 13
Transport substitution. . . . 13
Auditability. . . . 13

5 Dataset and Data Preparation **13**

5.1 UAVIDS-2025 . . . 13
5.2 Independent Audit . . . 13
5.3 Client-Selection Strategy and Partition Sizing . . . 14
5.4 Leakage Prevention . . . 15
5.5 Non-IID Client Analysis . . . 16

6 Intrusion Detection Method **16**

6.1 Features and Preprocessing . . . 16
6.2 Model Architecture . . . 16
6.3 Model Selection . . . 17

7 Federated Learning Method 17
7.1 Local Training and Aggregation . . . 17
7.2 Training Configuration and Optimisation Budgets . . . 18
Local epoch budgets are comparable across arms. . . . 18
Centralized training sees fewer epochs but more data per epoch. . . . 18
Per-client step counts differ. . . . 19
7.3 Checkpoint and Threshold Selection . . . 19
7.4 Baselines . . . 19

8 Post-Quantum Secure Model Exchange 19
8.1 ML-KEM-768: Key Establishment . . . 20
8.2 ML-DSA-65: Authentication . . . 20
8.3 HKDF-SHA-256: Key Derivation . . . 20
8.4 AES-256-GCM: Authenticated Encryption . . . 20
8.5 Message Protocol . . . 20
8.6 Threat-to-Control Matrix . . . 21

9 Implementation and Deployment 23
9.1 Containerised Services . . . 23
9.2 Control Centre . . . 23
9.3 UAV Clients . . . 23
9.4 Operator Dashboard . . . 23

10 Experimental Design 25

11 Results 25
11.1 Intrusion Detection Performance . . . 25
11.2 Client-Level Benefit . . . 27
11.3 Convergence . . . 27
11.4 Attack-Family Detection . . . 28
11.5 Post-Quantum Functional Equivalence . . . 28
11.6 Post-Quantum Overhead . . . 29
11.7 Security Probes . . . 29
11.8 Reproducibility Outcomes . . . 31

12 Discussion 31
12.1 Answering the Research Questions . . . 31
RQ1: does federation improve on independent training? . . . 31
RQ2: how close does federation come to centralized training? . . . 31
RQ3: does post-quantum protection cost detection performance or latency? 31
12.2 Operational Reading of the False-Positive Rate . . . 32
12.3 Client Heterogeneity as a Design Signal . . . 32
12.4 Convergence and Communication Cost . . . 32
12.5 What the Security Evidence Does and Does Not Show . . . 33
12.6 Interpreting the Result Honestly . . . 33

13 Limitations 33

Data and client realism. 33
Training partition size. 34
Statistical strength. 34
Schedule length. 34
Detection scope. 34
Privacy. 34
Security scope. 34
Deployment realism. 34

14 Future Work **34**
Strengthen the statistical claim. 34
Close the privacy gap. 35
Defend against malicious participants. 35
Address non-IID degradation directly. 35
Improve detection utility. 35
Harden the deployment path. 35
Validate on real platforms. 35

15 Conclusion **36**

References **37**

1 Executive Summary

This project develops a post-quantum secure intrusion detection system for a group of five military UAVs connected to a central control centre. Each UAV trains an intrusion detection model locally on its own network traffic, learning patterns of normal and malicious activity without exposing raw data. Only the trained model updates are transmitted to the control centre, where federated learning combines them into an improved global model. These updates are protected in transit by post-quantum cryptography, using ML-KEM for key establishment, ML-DSA for authentication, and AES-GCM for encrypted and integrity-protected transmission.

On a locked, source-disjoint test partition of 53,949 flows, the federated model achieved a macro-F1 of 0.9502, against 0.8969 for the mean of isolated per-client models and 0.9775 for a centralized model trained on pooled data. The cryptographic layer produced aggregates bit-identical to the unprotected run, at a maximum tensor difference of 0.0, and rejected all twelve probes of a controlled adversarial exercise without any reaching aggregation.

The main beneficiaries are defence organisations, military UAV operators, and organisations managing security-sensitive autonomous or unmanned systems. The solution can be developed as a cybersecurity software product or managed security service for secure UAV fleet monitoring and collaborative intrusion detection. Its main advantage is improved cyberattack detection across multiple UAVs while keeping local data private and protecting communication against tampering and future quantum-related security risks.

2 Introduction

2.1 Motivation

Unmanned aerial vehicles have moved from isolated reconnaissance platforms to networked fleets that share position, telemetry and mission data over wireless links. That connectivity is also the attack surface. Routing and availability attacks such as blackhole, flooding, Sybil and wormhole do not require physical access to an aircraft: an adversary who can inject or relay traffic inside the fleet network can silently drop mission data, exhaust constrained radio capacity, forge additional node identities, or tunnel traffic through a controlled path [1, 2]. In a defence context the consequence is not only degraded service but loss of situational awareness at the moment it matters most.

Machine learning offers a practical route to detecting these behaviours from flow statistics [3, 4], but the conventional training pipeline is a poor fit for UAV fleets. It requires every aircraft to ship raw network traffic back to a central site. This creates a bandwidth burden over links that are already constrained, concentrates operationally sensitive traffic into a single repository that becomes an attractive target, and makes the detection capability of the whole fleet dependent on one point of compromise.

A second concern is temporal. Model updates and control traffic captured today can be stored by an adversary and decrypted later, once a cryptographically relevant quantum computer becomes available. Because defence platforms have long service lives and their traffic retains sensitivity for years, a system protected only by classical public-key cryptography is exposed to this harvest-now-decrypt-later strategy. NIST finalised its first post-quantum standards in 2024 [5, 6], which makes migration a present engineering decision rather than a future one [7].

2.2 Problem Statement

This project addresses the following question: how can a fleet of UAVs collaboratively train a network intrusion detection model that is stronger than what any single aircraft could learn alone, without transmitting raw network traffic off the aircraft, and while protecting the exchanged model updates against interception, impersonation, tampering and replay by an adversary who may also possess future quantum capability?

The question decomposes into three requirements the system must satisfy simultaneously. First, detection quality: the collaboratively trained model must generalise to traffic from sources it never observed during training, and must improve on isolated local training. Second, data locality: raw flow records must remain on the originating node, with only model parameters leaving it. Third, communication security: every model message exchanged between an aircraft and the control centre must be confidential, authenticated, bound to its session context, and resistant to replay, using algorithms secure against quantum-capable adversaries.

2.3 Objectives

The project pursued six concrete objectives.

1. **O1.** Establish a verifiable, leakage-controlled experimental basis from the UAVIDS-2025 dataset, with held-out partitions provably disjoint from training by source address and feature signature.
2. **O2.** Construct a realistic non-identically distributed federated client structure that reflects the traffic heterogeneity expected across a UAV fleet rather than an artificially balanced split.
3. **O3.** Quantify the benefit of federated learning by comparing it against both a centralized upper bound and isolated per-client training under one frozen policy.
4. **O4.** Implement the federated protocol as a genuinely distributed system of isolated services rather than a single-process simulation.
5. **O5.** Secure the model exchange with NIST-standardised post-quantum primitives, and demonstrate that doing so does not perturb the learning outcome.
6. **O6.** Deliver an operator-facing interface that performs real inference against the frozen model and surfaces fleet state, detections and security events.

2.4 Research and Evaluation Questions

The objectives are evaluated against three research questions.

RQ1. Does federated learning improve intrusion detection compared with UAVs training independently?

RQ2. How closely can federated learning approach centralised training without exchanging raw client traffic?

RQ3. Can model updates be protected using ML-KEM, ML-DSA and AES-GCM without materially affecting detection performance or training latency?

Each question is answered directly in Section 11 and discussed in Section 12. Table 1 maps each question to the evidence that addresses it.

2.5 Contributions

This work makes the following contributions.

Table 1 Mapping from research questions to supporting evidence.

RQ	Primary evidence	Where reported
RQ1	Federated vs. five local-only models on locked test	Section 11.1, Table 12, Figure 10
RQ2	Federated vs. centralized on locked test; error structure	Section 11.1, Figure 11
RQ3	Plain vs. secure aggregate equivalence; crypto timings; probe outcomes	Sections 11.5–11.7

1. **A leakage-controlled federated benchmark on UAVIDS-2025.** An explicit row policy, global signature deduplication applied before partition assignment, and whole-source assignment produce training, validation and test partitions that are pairwise disjoint by flow identifier, source address and feature signature. The accompanying audit documents 81 internally inconsistent rows and 8,207 redundant behavioural duplicates that a naive split would have leaked.
2. **A three-way comparison under one frozen policy.** Centralized, isolated local-only and federated arms share architecture, initialisation, preprocessing and evaluation protocol, so the measured differences are attributable to data access rather than to configuration drift between arms.
3. **An application-layer post-quantum protocol for federated model exchange.** Rather than assuming a secure channel, the protocol binds the federated session context, namely run identifier, round, sender, recipient and frozen contract hash, into the authenticated associated data of every message, which allows misaddressed, stale and replayed updates to be rejected before decoding.
4. **Evidence that post-quantum protection is free in learning terms.** Secure and plain runs produce bit-identical aggregates at a maximum tensor difference of 0.0, establishing that the security layer can be adopted without any detection-performance trade-off.
5. **A reproducibility discipline with verifiable artefacts.** Every model and threshold decision was frozen and hash-locked before the test partition was opened, the lock was verified unchanged afterwards, two clean reruns reproduced all 27 development artefacts byte for byte, and an independent audit recomputed every reported metric from saved predictions.

2.6 Scope and Assumptions

The evaluation boundary is stated here so that later results are read correctly.

Data and clients.. UAVIDS-2025 is generated from a documented NS-3 simulation of a UAV swarm and is not operational or classified traffic. The five participants are logical clients derived from distinct source addresses, not verified physical aircraft, a distinction that matters under Sybil conditions where source identity may itself be forged. Held-out partitions are disjoint by source address and exact feature signature, but the dataset carries no scenario, run, capture-session or timestamp fields, so scenario-level and temporal independence cannot be asserted.

Modelling.. Detection is binary. The model separates normal from attack traffic and does not infer which of the four attack families is present at inference time, although family-level detection rates were measured offline. Results come from a single deterministic seed and a single split, so no confidence intervals are reported. The validation partition was reused for candidate, checkpoint and threshold selection, making it a development set rather than an untouched benchmark.

Privacy.. Raw records never leave a client during federated training. However, preprocessing statistics for imputation and scaling were fitted once on the pooled training partitions of all

five clients. This is a research-prototype convenience and is explicitly not a privacy-preserving federated statistics protocol. Neither secure aggregation nor differential privacy is implemented, and the control centre observes plaintext updates after decryption.

Security.. The cryptographic layer is an academic application-layer prototype. It assumes the server host, provisioned trust store, cryptographic library and container runtime are trusted, and uses a controlled allowlist of demonstration identities rather than a production public key infrastructure. It protects application payloads but does not conceal HTTP metadata, does not guarantee availability, does not provide forward secrecy after session key compromise, and does not determine whether an authenticated client has submitted a poisoned update.

Deployment.. The containerised demonstration runs on x86 CPU containers whose resource limits are inspired by real edge hardware rather than emulating it. It does not reproduce ARM microarchitecture, accelerator behaviour, power draw, thermal throttling, radio links or flight dynamics, so no hardware performance or energy claim is derived from it.

3 Background and Related Work

This section reviews five areas that together define the space this project occupies: UAV-specific intrusion detection, federated intrusion detection, federated learning under non-identically distributed data, secure and post-quantum federated learning, and UAV communication security. It closes with a comparison positioning this work against the closest prior systems.

3.1 UAV Communication and Security Threats

Early analyses of UAV security established the threat surface that later detection work targets. Javaid et al. modelled the cyber-security threats of a UAV system across its communication, sensing and control subsystems [2], and Krishna and Murphy surveyed the vulnerabilities of unmanned aerial vehicles across hardware, firmware and network layers [1]. Both note that UAV networks inherit the weaknesses of mobile ad hoc networks, in particular the dependence on multi-hop forwarding by nodes that cannot be physically secured.

That dependence is what makes routing and availability attacks the dominant concern for fleets rather than individual aircraft. Blackhole and wormhole attacks subvert the forwarding path, Sybil attacks forge node identities to gain disproportionate influence over routing decisions, and flooding attacks exhaust the constrained radio and processing capacity of nodes. These four families are precisely the ones represented in the dataset used here [8]. Sharma et al. and subsequent work on UAV swarm communication emphasise that the dynamic topology of a swarm invalidates static perimeter defences, since the set of neighbours changes continuously in flight [9].

3.2 UAV-Specific Intrusion Detection

Choudhary et al. provided one of the first structured surveys of intrusion detection for networked UAVs, classifying approaches by detection strategy and by placement within the fleet [3]. Their central observation, that centralized IDS placement conflicts with the mobility and intermittent connectivity of UAV networks, remains the motivating tension for distributed approaches such as the one taken here.

Subsequent work has pursued higher detection accuracy through deeper models. Abu Al-Haija and Al Badawi demonstrated a high-performance deep learning IDS for networked UAVs [4], while Subbarayalu and Vensuslaus approached drone swarm intrusion detection through timed

probabilistic automata, exploiting the temporal regularity of legitimate swarm traffic [10]. More recently, Zhou et al. combined classical machine learning with graph neural networks to capture the relational structure of UAV network traffic, arguing that node-level features alone discard information about how traffic propagates through the swarm [11]. A recent survey by Ramadan et al. consolidates machine learning approaches to UAV intrusion detection and notes that the field remains constrained by dataset availability [12].

A distinct line of work fuses cyber and physical signals. Hassler et al. argue that purely network-derived features limit detection performance for UAVs, and demonstrate that combining cyber features with physical flight-dynamics telemetry improves results, releasing a public cyber-physical testbed dataset [13]. Their result is relevant to the present work as a boundary condition: the dataset used here provides no flight-dynamics channel, so this project detects attacks from network flow statistics alone and does not attempt cyber-physical fusion.

The dataset scarcity that Choudhary et al. and Ramadan et al. both identify has begun to ease. Whelan et al. released a UAV attack dataset covering GPS spoofing and jamming [14], and Zeng et al. released UAVIDS-2025, generated in NS-3.24 with an extended BOID mobility model, covering the four routing and availability attacks used here [8, 15]. UAVIDS-2025 is the dataset adopted for this project, and Section 5 documents an independent audit of it.

3.3 Federated Intrusion Detection

Federated averaging, introduced by McMahan et al., allows clients to train a shared model by exchanging parameter updates rather than raw data [16]. The approach suits intrusion detection because network traffic is often the most sensitive artefact an organisation holds, and because detection benefits from exposure to attack patterns no single participant observes.

Rahman et al. framed the design space directly, comparing centralized, on-device and federated intrusion detection for IoT and showing that the federated arm recovers much of the centralized benefit while avoiding data centralisation [17]. That three-way comparison is the structure adopted in this project's experimental design. Ferrag et al. released Edge-IIoTset explicitly to support both centralized and federated evaluation of IoT and IIoT intrusion detection [18], and it has become a common federated IDS benchmark; this project used it during an earlier familiarisation phase before moving to UAVIDS-2025, and the two sets of results are not comparable because they differ in data, client construction, preprocessing and evaluation conditions.

A substantial body of subsequent work refines federated IDS for constrained environments. Olanrewaju-George and Pranggono compared supervised and unsupervised deep learning models within a federated IoT IDS, finding that autoencoder-based detection transfers less cleanly across heterogeneous clients than supervised classification [19]. Ramesh et al. investigated optimisation of the federated IDS training loop for IoT deployments [20], and Bai et al. surveyed the broader design space of federated IoT security [21]. Nguyen addressed the resource-constrained case specifically, which is the operating regime of a UAV companion computer [22]. Guerra et al. provide a useful centralized baseline, comparing conventional machine learning techniques for IoT intrusion detection without federation [23].

Within the UAV domain specifically, federated intrusion detection is recent. Zeng et al. propose FedGraph-ID, a federated graph learning framework for UAV intrusion detection under adversarial settings [24], and earlier work combined federated learning with GAN-based augmentation for UAV networks [25]. These are the closest systems to the present work in application domain, and neither addresses the cryptographic protection of the model exchange itself.

3.4 Federated Learning under Non-IID Data

The dominant obstacle to federated performance is statistical heterogeneity. When client data is not independently and identically distributed, each client optimises towards a different local minimum during its local epochs, and averaging those divergent parameter sets yields a model that is not the minimiser any client would have reached. Zhao et al. characterised this effect quantitatively, showing accuracy degradation that grows with the earth-mover distance between client distributions [26].

Several algorithmic responses exist. FedProx adds a proximal term to the local objective, penalising deviation from the global model and thereby limiting client drift [27]. SCAFFOLD corrects drift explicitly using control variates that estimate and subtract the difference between local and global update directions [28]. FedNova normalises local updates by the number of local steps each client performs, removing the objective inconsistency that arises when clients hold different amounts of data [29]. MOON applies a contrastive loss at the representation level to keep local features aligned with the global model [30]. A recent survey by Lu et al. organises the field by type of heterogeneity and notes that most methods address only one type in isolation [31].

This project deliberately uses unmodified sample-count-weighted FedAvg rather than any of these corrections. The reason is measurement: the objective is to quantify the baseline federated benefit and cost under realistic heterogeneity, which requires that the aggregation rule not be a confounding variable. The non-IID structure is retained rather than balanced, with attack prevalence ranging from 50.64% to 90.08% and two clients holding no Sybil observations at all, and Section 12.3 discusses what the resulting per-client evidence implies for weighting schemes.

3.5 Secure and Post-Quantum Federated Learning

Keeping raw records local is a meaningful reduction in exposure, but parameter updates are derived from those records and are not inherently free of information about them. Bonawitz et al. addressed this with practical secure aggregation, allowing a server to compute the sum of client vectors without learning individual contributions, using pairwise masking and secret sharing with tolerance for client dropout [32]. Differential privacy provides a complementary bound on residual leakage at a measurable cost in utility. Neither is implied by federated averaging, and neither is implemented here.

The cryptographic primitives underpinning secure aggregation are themselves subject to quantum attack. Bonawitz et al.'s construction relies on Diffie-Hellman key exchange, which Shor's algorithm breaks. NIST finalised lattice-based replacements in 2024: FIPS 203 specifies ML-KEM for key encapsulation [5] and FIPS 204 specifies ML-DSA for digital signatures [6]. NIST IR 8547 sets out the transition timeline [7], and the Open Quantum Safe project provides liboqs, the reference implementation used in this work [33].

Work combining post-quantum cryptography with federated learning has focused almost exclusively on post-quantum secure aggregation. Zuo et al. adapted the double-masking secure aggregation protocol to lattice-based primitives [34], and subsequent schemes have pursued lower communication cost, including key-homomorphic pseudorandom function constructions for the cross-silo setting [35]. Byzantine-robust variants combine post-quantum secure aggregation with malicious-client detection for critical infrastructure [36].

This concentration on aggregation leaves a gap. Most federated learning research treats the transport layer as an implementation detail and assumes an authenticated secure channel exists, while most post-quantum engineering work targets general-purpose protocols such as TLS rather than the specific message pattern of a federated training round. The present work occupies that gap: it applies standardised post-quantum primitives directly to the federated model-exchange

protocol, binding the session context of a training round, namely run identifier, round number, sender, recipient and frozen protocol contract, into the authenticated associated data of every message. This binding is what allows the system to reject misaddressed, stale and replayed updates rather than merely encrypting them.

3.6 Positioning

Table 2 positions this project against the closest prior systems along the four dimensions that matter for the research questions.

Table 2 Positioning against closely related systems. PQ transport denotes post-quantum protection of the model-exchange channel itself, as distinct from post-quantum secure aggregation.

Work	UAV	Federated	PQ transport	Deployed services
Choudhary et al. [3]	yes	no	no	no
Hassler et al. [13]	yes	no	no	no
Zeng et al. [8]	yes	no	no	no
Rahman et al. [17]	no	yes	no	no
Ferrag et al. [18]	no	yes	no	no
Zeng et al. [24]	yes	yes	no	no
Bonawitz et al. [32]	no	yes	no	no
Zuo et al. [34]	no	yes	partial	no
This work	yes	yes	yes	yes

The distinguishing combination is the final row: a UAV-domain federated intrusion detector whose model exchange is protected end to end by NIST-standardised post-quantum primitives, implemented as isolated communicating services rather than a single-process simulation, and evaluated with the security layer shown not to perturb the learning outcome.

4 System Design

4.1 Use-Case Architecture

The deployment scenario is a fleet of five UAVs coordinated by a ground control centre. Each aircraft observes its own network traffic, trains locally, and transmits only model parameters. The control centre aggregates those parameters and returns an improved global model. Figure 1 shows the arrangement.

The system was built in five sequential stages, each producing artefacts the next consumes without modification. This ordering was chosen so that every claim made later traces to a frozen, hash-verified input rather than to a setting tuned after seeing its own result.

4.2 Threat Model

The adversary is assumed to be able to read, copy, delay, replay, alter or inject application messages between any client and the control centre, and to claim another client's identity. The adversary may also record traffic now for decryption later, once quantum capability becomes available.

The following are trusted: the server host, mounted runtime secrets, the cryptographic library, the container engine, and the provisioned trust store. Out of scope are host compromise, traffic analysis of message timing and size, denial of service, key exfiltration, and malicious behaviour by a legitimately authenticated client.

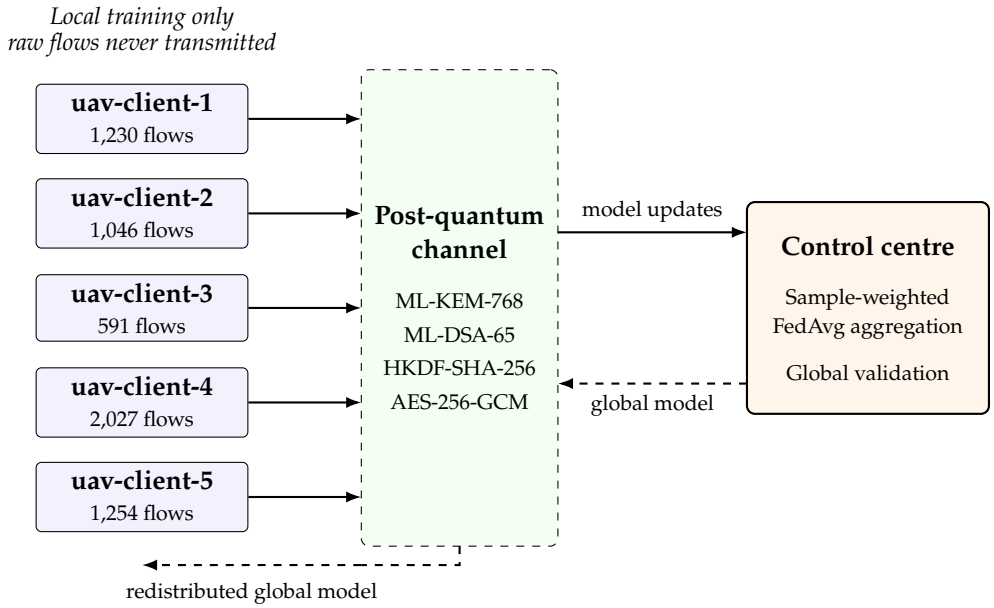


Figure 1 System architecture. Each logical UAV client trains on its own flow records and transmits only model parameters. All messages in both directions are protected by the post-quantum channel described in Section 8.5. Sample counts are the retained training rows after the row policy of Section 5.4.

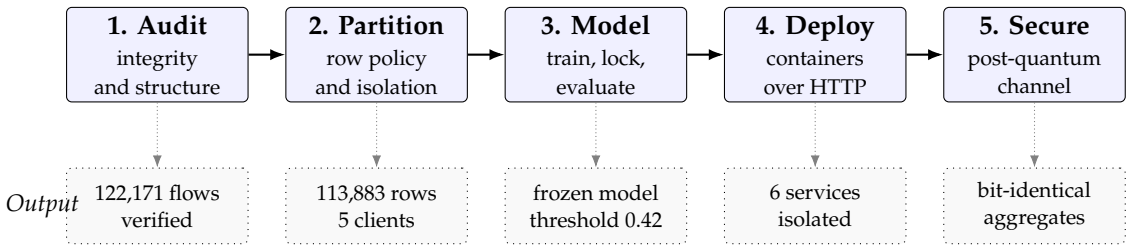


Figure 2 The five-stage methodology. Each stage consumes the previous stage's frozen, hash-verified artefacts without modifying them, so that every reported claim traces back to a locked input.

That last exclusion is significant and is revisited in Section 13. The system authenticates *who* may contribute but does not evaluate *what* they contribute, so an authenticated client submitting deliberately poisoned parameters passes every implemented check.

4.3 Security Requirements

Six requirements follow from the threat model and drive the protocol design of Section 8.

SR1 Confidentiality. Model parameters must not be readable by an observer of the channel, including one who records now and decrypts later.

SR2 Authentication. The control centre must establish that an update originates from a client it recognises, and the client must establish that it is speaking to the genuine control centre.

SR3 Integrity. Any modification to a message, whether to its payload or its routing and round metadata, must be detected and the message rejected.

SR4 Context binding. A message must be valid only within the run, session, round and sender-recipient pair for which it was produced.

SR5 Replay resistance. A previously valid message must not be accepted a second time.

SR6 Fail-safe rejection. A message failing any check must be rejected before its contents are decoded, and must not influence aggregation or leave the receiver in a state that rejects subsequent genuine messages.

4.4 Software Architecture and Technology Stack

The system is organised into four layers, summarised in Table 3.

Table 3 Technology stack by layer.

Layer	Technology	Role
Data	Python, pandas, NumPy	Audit, row policy, deduplication, partition construction, integrity assertions
Learning	PyTorch, scikit-learn	MLP definition, local training loops, explicit FedAvg aggregation, preprocessing and metrics
Transport	Python HTTP services, Docker Compose	Client and control-centre services, NumPy archive exchange, schema and tensor validation
Security	liboqs via OQS Python bindings, `cryptography`	ML-KEM-768, ML-DSA-65, HKDF-SHA-256, AES-256-GCM
Interface	HTTP and JSON adapter, web dashboard	Fleet state, live inference, round progress, security events

Communication between UAV clients and the control centre uses HTTP with JSON-encoded envelopes. Model parameters travel as NumPy archives loaded with pickle deserialisation disabled, which removes an arbitrary-code-execution path that a naive checkpoint exchange would introduce. Each service runs in its own container with a read-only root filesystem, dropped Linux capabilities and a minimal writable temporary filesystem.

4.5 Rationale for a Direct Federated Implementation

Federated averaging is implemented directly rather than through an established framework such as Flower or TensorFlow Federated. Three considerations drove this choice.

Bit-level reproducibility.. The central claim of Section 11.5 is that the secure and plain runs produce bit-identical aggregates at a maximum tensor difference of 0.0. Establishing that requires complete control over aggregation order, floating-point accumulation, optimizer state handling between rounds and random-number consumption. Framework abstractions manage these internally and may vary them across versions, which would have made an exact-equality claim unverifiable rather than merely difficult.

Transport substitution.. The project's security contribution requires replacing the transport layer beneath an otherwise unchanged learning protocol, so that any difference in outcome is attributable to the cryptography. Frameworks bundle transport with the training loop, typically over gRPC, which would have meant either modifying framework internals or accepting that the plain and secure arms differed in more than one respect.

Auditability.. The aggregation rule is roughly thirty lines of explicit code that a reviewer can read and independently reimplement. This mattered for the audit described in Section 11.8, in which every reported aggregate was recomputed from saved client updates and compared against the control centre's output.

The trade-off is real and worth stating: the implementation forgoes the client-sampling strategies, dropout handling, secure aggregation modules and scalability engineering that mature frameworks provide. For a five-client research system evaluated on reproducibility grounds this is an acceptable exchange, but a production deployment at fleet scale would be better served by a framework, and Section 14 treats this as future work rather than as a permanent design position.

5 Dataset and Data Preparation

5.1 UAVIDS-2025

The system uses UAVIDS-2025, a flow-level dataset of UAV swarm network traffic generated in the NS-3.24 simulator with UAV mobility modelled by an extended BOID algorithm [8]. The working copy was obtained from the official archive record [15] and verified against both an MD5 and a SHA-256 digest recorded in the project configuration before any processing; every subsequent stage re-checks those digests. The verified file contains 122,171 flows described by 22 published feature columns and one label column.

The label column carries five classes, mapped to a binary target as shown in Table 4.

Table 4 Mapping from published labels to the binary detection target.

Published label	Binary label
Normal Traffic	0
Blackhole Attack	1
Flooding Attack	1
Sybil Attack	1
Wormhole Attack	1

5.2 Independent Audit

A read-only audit checked schema conformance, data types, missing, infinite and negative values, constant and near-constant columns, exact duplicate rows, repeated feature vectors, label conflicts among repeated patterns, and per-source class distributions.

It found no missing, infinite or negative numeric cells and no exact full-row duplicates. It identified one constant column, `Protocol`, which records UDP for every flow. It flagged 81 rows as internally inconsistent under the published feature definitions, because `PacketDropRate` falls outside the interval $[0, 1]$ or `LostPackets` exceeds `TxPackets`. It also found 8,219 redundant rows once flows were compared on behavioural features alone, with no conflicting labels among the repeated groups.

The audit further established that `SrcAddr` is the strongest available client key, taking 176 distinct values of which 95 contain both binary classes. The dataset documentation does not map these addresses to distinct physical aircraft, and Sybil traffic may contain forged addresses, so the project adopted the explicit convention that they define source-based logical clients only.

5.3 Client-Selection Strategy and Partition Sizing

The resulting split allocates 6,148 rows to training, 53,786 to validation and 53,949 to test, so approximately 5.4% of retained data is used for training. This is far from the conventional 70/15/15 proportion and the reason is structural rather than arbitrary.

The constraint that drives it is source-disjointness. To measure whether a federated model generalises to aircraft it has never observed, every source address must belong entirely to one partition, and the held-out partitions must contain *different* sources from the training clients. Federated training additionally requires that training data be divided among exactly five clients, each corresponding to one logical UAV. Together these force the design: five source addresses are reserved as clients, and the remaining 171 sources become held-out data.

The consequence follows from the dataset's own structure. Source groups vary greatly in size, and the five reserved as clients hold between 591 and 2,027 rows each. No selection of five sources could have produced a conventional training proportion without either splitting sources across partitions, which would destroy the disjointness guarantee, or selecting only the five largest sources, which would have biased client composition towards whichever traffic patterns happen to be most numerous.

Three considerations make this trade acceptable.

First, the quantity that matters for the research questions is the comparison between arms, not the absolute score. Centralized, local-only and federated models all train on exactly the same 6,148 rows, differing only in how those rows are accessed. A larger training partition would raise all three scores without changing what the comparison measures.

Second, the small training partition makes the evaluation harder rather than easier. Both principal models are asked to generalise from roughly six thousand flows to nearly fifty-four thousand flows from entirely unseen sources, and the centralized model still reaches 0.9775 macro-F1. A generous training proportion would have made the generalisation claim weaker, not stronger.

Third, the large held-out partitions are a benefit in their own right. With 87 validation sources and 84 test sources, the reported metrics rest on a broad sample of source behaviour rather than on a handful of held-out groups, which reduces the risk that a result reflects the idiosyncrasies of a few sources.

The honest limitation is that a training partition of this size limits the capacity of model that can be fitted without overfitting, and constrains how much any arm can learn. This is why the architecture selected in Section 6.2 is deliberately small.

5.4 Leakage Prevention

Partitioning was performed deterministically by a single script that fails rather than proceeding whenever an integrity or isolation assertion is violated.

The row policy excludes the 81 internally inconsistent rows outright. Values were not clipped or imputed, because a row contradicting its own feature definitions cannot be repaired without inventing data. Feature signatures were then computed over the 15 approved model features before any partition was assigned. The script fails if any signature carries conflicting labels; none did. For each remaining group of identical same-label signatures, only the representative with the lowest `FlowID` is retained globally, removing 8,207 rows. Performing this deduplication before assignment is what prevents an identical flow from appearing in both a training and a held-out partition. The final accounting retains 113,883 unique rows.

Every remaining source group was assigned whole to either validation or test by a deterministic, class-aware greedy algorithm. The script then asserts that flow identifiers, source addresses and approved feature signatures are pairwise disjoint across every partition, and writes model-facing files containing only the 15 features plus the original and binary labels.

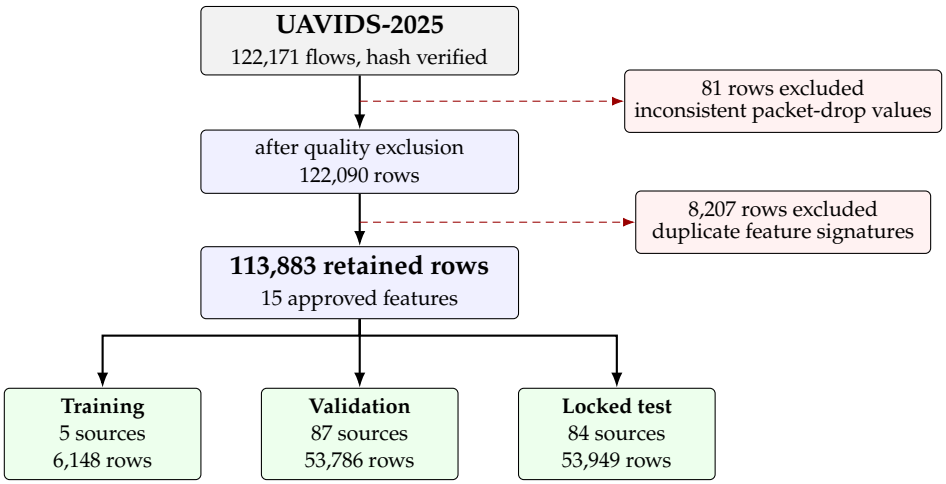


Figure 3 Row accounting and partition construction. Exclusions are applied globally before any source is assigned to a partition, which prevents an identical flow from appearing in both training and held-out data. No source address is ever split across partitions.

The 15 approved features are flow duration, transmitted and received packet counts, lost packets, transmitted and received byte counts, transmitted and received packet rates, transmitted and received byte rates, mean delay, mean jitter, throughput, packet drop rate and average hop count. Identifiers and shortcut fields were excluded deliberately: `FlowID`, `SrcAddr` and `DstAddr` as row, source and endpoint identities; `SrcPort` and `DstPort` because they take only two values, are identical to one another in every row and are strongly scenario-associated; `Protocol` because it is constant; and `MeanPacketSize` because it re-exposes the same port and application regime and is redundant with the packet and byte counts.

These controls establish source-address and feature-signature separation. They do not establish scenario, simulation-run, temporal, capture-session or destination-network independence, because the dataset does not carry the metadata that would be required to do so.

5.5 Non-IID Client Analysis

The five training clients are naturally non-identically distributed, and this structure was retained rather than balanced because uniform clients would misrepresent the deployment condition the system must handle. Table 5 gives the binary composition and Table 6 the attack-family composition.

Table 5 Training client distributions after row policy and deduplication.

Client	Rows	Normal	Attack	Attack %
uav-client-1	1,230	597	633	51.46
uav-client-2	1,046	264	782	74.76
uav-client-3	591	145	446	75.47
uav-client-4	2,027	201	1,826	90.08
uav-client-5	1,254	619	635	50.64
Total	**6,148**	**1,826**	**4,322**	**70.30**

Table 6 Attack-family coverage per training client. Two clients observe no Sybil traffic and one is heavily flooding-dominated.

Client	Blackhole	Flooding	Sybil	Wormhole
uav-client-1	present	present	present	present
uav-client-2	present	present	present	present
uav-client-3	present	present	present	present
uav-client-4	present	dominant	**absent**	present
uav-client-5	present	present	**absent**	present

Two properties of this structure matter for the results. Attack prevalence ranges from 50.64% to 90.08%, a spread that guarantees local optima will diverge during local training. More consequentially, two of the five clients hold no Sybil observations, so a model trained in isolation on either partition has never been shown that attack family and cannot be expected to detect it. This is precisely the condition federated learning is intended to address, and Section 11.2 measures how far it does so.

6 Intrusion Detection Method

6.1 Features and Preprocessing

Median imputation and standard scaling were fitted on the 6,148 pooled training rows only, then applied unchanged to validation and test. Fitting on training data alone is what prevents held-out statistics from influencing the model. As noted in Section 2.6, this pooled fit does read all five clients' feature values centrally and is therefore not a privacy-preserving federated preprocessing protocol; Section 14 treats replacing it as a priority.

6.2 Model Architecture

The selected model is a multilayer perceptron with layer widths $15 \to 128 \to 64 \to 32 \to 1$, ReLU activations and dropout rates of 0.2, 0.1 and 0.0 across its hidden layers, trained with an unweighted binary cross-entropy loss computed from logits.

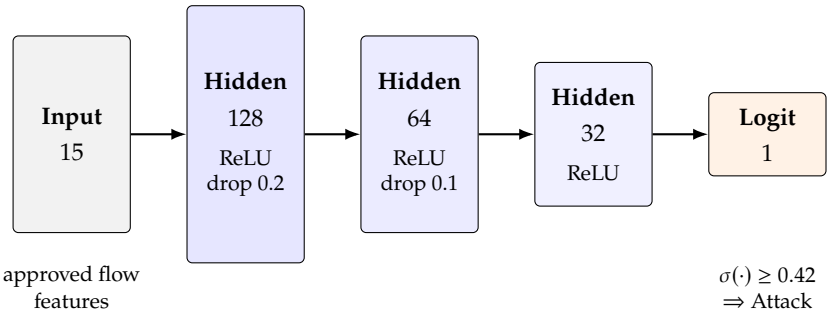


Figure 4 The binary detection model. The same architecture and initialisation are used for the centralized, local-only and federated arms, so that the three differ only in how training data is accessed. The decision threshold of 0.42 was selected on validation before the test partition was opened.

The architecture is deliberately small. With 6,148 training rows, a larger network would overfit, and the model must also be plausible for a resource-constrained companion computer of the class described in Section 9.3.

6.3 Model Selection

Four candidate architecture and loss combinations were compared using the training clients and validation partition only, with thresholds selected per candidate by validation macro-F1. Table 7 gives the outcome.

Table 7 Candidate comparison on validation.

Candidate	Loss	Thr.	Macro-F1	FPR	Selected
small_unweighted	unweighted	0.58	0.9744	0.0309	
small_balanced	weighted	0.40	0.9709	0.0444	
edge_style_unweighted	unweighted	0.46	**0.9775**	**0.0246**	yes
edge_style_balanced	weighted	0.30	0.9728	0.0391	

The selected candidate exceeded the smaller unweighted architecture by 0.0031 macro-F1. Applying global positive-class weighting reduced macro-F1 and raised the false-positive rate for both architectures, so the unweighted loss was retained.

7 Federated Learning Method

7.1 Local Training and Aggregation

Federated averaging is implemented directly in PyTorch with an explicit sample-count-weighted averaging function, for the reasons given in Section 4.5. The modelling stage runs as a deterministic single-process simulation with no network transport, which isolates the optimisation behaviour of federated averaging from any communication effect; network transport is introduced separately in Section 9.

The global model is initialised from the same seeded state used for the centralized and local-only baselines, so the three arms differ only in how training data is accessed. Each round proceeds as shown in Figure 5.

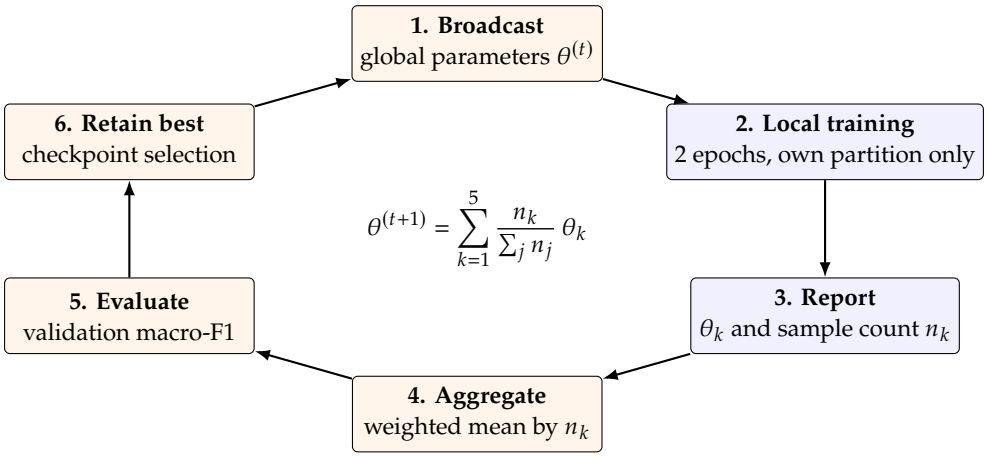


Figure 5 One federated averaging round. Clients receive the current global parameters, train locally for two epochs, and return parameters weighted by their retained sample count. The control centre never receives raw flow records.

7.2 Training Configuration and Optimisation Budgets

Table 8 gives the locked configuration shared by all three arms.

Table 8 Locked training configuration, common to all arms unless stated.

Setting	Value
Architecture	MLP 15−128−64−32−1
Activation, dropout	ReLU; 0.2, 0.1, 0.0
Loss	Unweighted BCE with logits
Optimizer	AdamW
Learning rate	0.001
Weight decay	0.0001
Batch size	128
Threshold search	0.10 to 0.90, step 0.01
Random seed	42
Reference device	CPU, single thread, deterministic

Because the three arms use different training schedules by construction, their optimisation budgets require explicit statement. Table 9 sets them out.

Three points follow, and they matter for interpreting the comparison in Section 11.

Local epoch budgets are comparable across arms.. Each federated client performs 60 local epochs in total, 2 per round across 30 rounds. Each local-only client is permitted up to 60 epochs under early stopping. The federated advantage reported later is therefore not attributable to federated clients receiving more passes over their own data than isolated clients; the budgets are matched by design.

Centralized training sees fewer epochs but more data per epoch.. The centralized model reached its best validation checkpoint at epoch 52, giving $52 \times \lceil 6148/128 \rceil = 2{,}548$ gradient steps. Each step

Table 9 Optimisation budget by arm. Steps are gradient updates at batch size 128. Federated per-client steps are computed as $\lceil n_k/128 \rceil \times 2$ epochs $\times$ 30 rounds.

Arm	Schedule	Data seen	Steps
Centralized	Up to 60 epochs, early stopping patience 10; best epoch 52	6,148 pooled rows	2,548
Local-only (per client)	Up to 60 epochs, early stopping patience 10	own partition only	$\leq 60\lceil n_k/128 \rceil$
Federated (per client)	30 rounds $\times$ 2 local epochs = 60 local epochs	own partition only	300 to 960
Federated (global)	30 aggregations	no direct data access	30

is computed on a batch drawn from the full pooled distribution, whereas a federated client's steps are drawn only from its own partition. The centralized advantage is therefore one of distributional coverage per step rather than of raw step count: federated clients collectively perform 3,000 steps, more than the centralized 2,548, and still reach a lower score.

Per-client step counts differ.. Because clients hold between 591 and 2,027 rows, per-client step counts range from 300 for client 3 to 960 for client 4. Standard FedAvg does not normalise for this, which is the objective inconsistency that FedNova addresses [29]. This project retains unmodified FedAvg so that the aggregation rule is not a confounding variable, and Section 12.3 discusses the consequences.

7.3 Checkpoint and Threshold Selection

The retained federated checkpoint is the round with the highest validation macro-F1, with ties broken by attack recall, then lower false-positive rate, then lower log loss. After checkpoint selection, the decision threshold is searched over 0.10 to 0.90 in steps of 0.01 using validation macro-F1. The selected federated checkpoint is round 30 at threshold 0.42; the centralized model uses threshold 0.46.

7.4 Baselines

Two baselines were trained under the identical policy. The *centralized* model has access to all five pooled training partitions and measures the benefit of pooling, serving as an upper bound. The *local-only* models are five separate models each restricted to a single client partition, measuring what an aircraft could achieve in isolation. Reporting both bounds the federated result from above and below, which is what allows RQ1 and RQ2 to be answered quantitatively rather than comparatively.

8 Post-Quantum Secure Model Exchange

The security layer wraps the verified HTTP protocol without changing its data, preprocessing, architecture, local training, sample counts or aggregation rule. Four primitives are used, all at

NIST security category 3, chosen as a defensible middle ground stronger than the category 1 variants while avoiding the larger handshake objects of category 5.

8.1 ML-KEM-768: Key Establishment

ML-KEM, standardised in FIPS 203 [5], is a lattice-based key encapsulation mechanism whose security rests on the Module Learning With Errors problem. At parameter set 768 it uses a 1,184-byte public key, produces a 1,088-byte ciphertext and yields a 32-byte shared secret. The control centre generates a fresh ephemeral key pair per session, so a shared secret compromised in one session does not expose another.

8.2 ML-DSA-65: Authentication

ML-DSA, standardised in FIPS 204 [6], provides digital signatures over the same lattice family. At parameter set 65 it uses a 1,952-byte public key and produces 3,309-byte signatures. Both parties sign their handshake contributions, which satisfies SR2 in both directions: the client verifies the control centre against a provisioned trust store, and the control centre verifies the client against an allowlist of recognised identities.

8.3 HKDF-SHA-256: Key Derivation

The 32-byte ML-KEM shared secret is expanded by HKDF-SHA-256 into two AES-256 keys and two four-byte nonce prefixes, one pair per direction. The salt and info parameters carry the session context, providing domain separation so that keys derived for one session, run or round cannot be valid in another. Distinct keys per direction prevent an attacker from replaying a client message back to the client as though it came from the server.

8.4 AES-256-GCM: Authenticated Encryption

Every post-handshake message is protected with AES-256-GCM under a unique 96-bit nonce formed from the direction's four-byte prefix and an eight-byte monotonic sequence number. GCM provides confidentiality and integrity together, and its associated-data facility is what implements context binding: the canonical message header is authenticated but not encrypted, so any modification to routing or round metadata invalidates the authentication tag.

8.5 Message Protocol

The handshake proceeds in four messages, shown in Figure 6. The client registers with its public key fingerprint. The server replies with a signed hello carrying a fresh ephemeral ML-KEM public key, binding the security version, run, session, client and server identifiers, the frozen contract hash, the algorithm names, a server challenge, the key hash, its own identity fingerprint and an expiry. The client verifies this against its trust store, encapsulates to the server's key, and returns the KEM ciphertext with a signature over the complete canonical handshake including its own challenge. The server verifies the client against its allowlist, decapsulates, derives the session keys, and confirms the session under AES-GCM.

Every subsequent message carries a header specifying security version, session, run, round, sender, recipient, message type, sequence number, nonce, contract hash, plaintext digest and model digest. Receivers require the exact next sequence number, which rejects replay before decryption is attempted. Invalid ciphertext does not advance receiver state, so a genuine message remains acceptable after a tampered copy has been discarded. Authentication and metadata validation complete before any JSON or tensor decoding, and rejected updates never reach the aggregator, satisfying SR6.

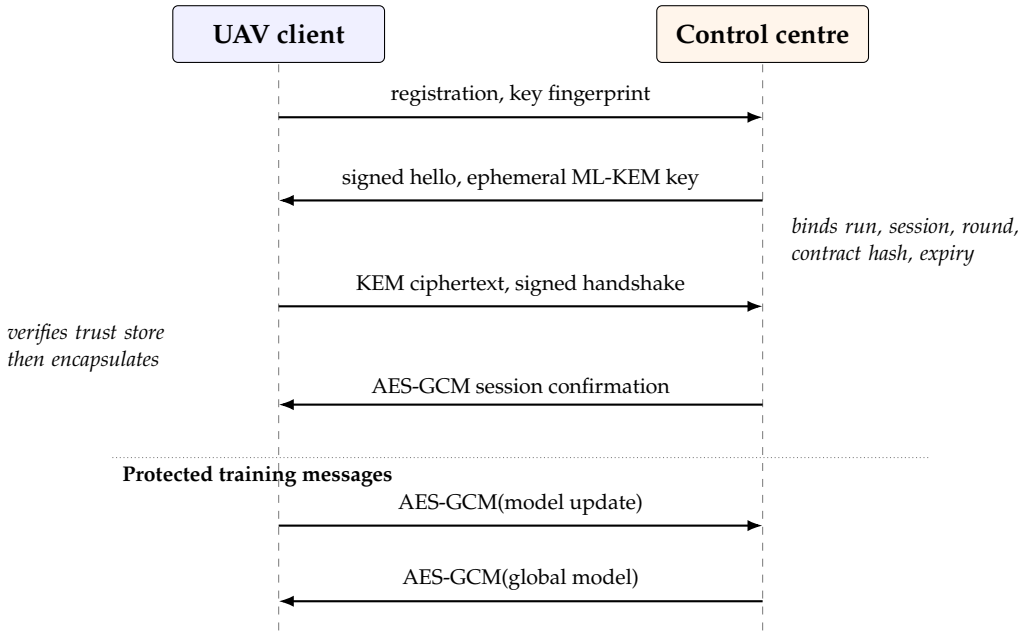


Figure 6 Post-quantum handshake and protected message exchange. HKDF derives a distinct AES-256 key and nonce prefix per direction from the ML-KEM shared secret. Every subsequent message authenticates its header as associated data, so tampering with round or routing metadata invalidates the tag, and strict sequence checking rejects replay before decryption is attempted.

Failures return a generic HTTP error and emit an event recording only the endpoint and a coarse category. Keys, signatures, ciphertext, nonces and tensor contents are never logged.

8.6 Threat-to-Control Matrix

Table 10 maps each threat in the model of Section 4.2 to the control that addresses it, the security requirement it satisfies, and the observed outcome in the controlled adversarial run of Section 11.7.

Table 10 Threat-to-control matrix. "Out of scope" entries are stated explicitly rather than omitted, since an unlisted threat could otherwise be read as an addressed one.

Threat	Security control	SR	Result
Eavesdropping, including harvest-now-decrypt-later	ML-KEM-768 key establishment, AES-256-GCM encryption	SR1	Protected
Message modification	AES-GCM authentication tag	SR3	Rejected
Client impersonation	ML-DSA-65 signature, identity allowlist	SR2	Rejected
Server impersonation	ML-DSA-65 signature, provisioned trust store	SR2	Rejected
Replay	Monotonic sequence number, session context	SR5	Rejected
Wrong-round or misaddressed message	Authenticated header metadata as associated data	SR4	Rejected
Nonce reuse	Per-direction nonce prefix, strict sequence check	SR5	Rejected
Malformed update	Schema, shape, dtype, finiteness and size validation	SR6	Rejected
Legitimate but malicious client	None implemented	—	Out of scope
Denial of service	None implemented	—	Out of scope
Traffic analysis of size and timing	None implemented	—	Out of scope
Host or key-store compromise	None implemented; assumed trusted	—	Out of scope

9 Implementation and Deployment

9.1 Containerised Services

The deployment stage moves the frozen protocol onto six services managed by Docker Compose: five UAV clients and one control centre. Each client mounts exactly one training file read-only; the control centre mounts none and holds only the validation partition. Container roots and all data mounts are read-only, containers run unprivileged with all Linux capabilities dropped, and each has a small temporary filesystem.

Model updates are exchanged as NumPy archives loaded with pickle deserialisation disabled, and are subject to strict version, shape, dtype, finiteness, hash, sample-count, round and size checks before acceptance.

9.2 Control Centre

The control-centre service exposes the registration, handshake, session and update endpoints, maintains per-client session state, validates each incoming update against the frozen contract, performs sample-count-weighted aggregation, evaluates the new global model on the validation partition, and returns the updated parameters. It holds no client training data at any point.

9.3 UAV Clients

Client resource limits are inspired by real edge hardware rather than emulating it, spanning roles from a small memory-constrained node to a stronger companion computer, as summarised in Table 11. The limits deliberately fall below each board's full capacity, because a deployed aircraft must also reserve resources for its operating system, flight software and sensors.

Table 11 Device-inspired container resource profiles.

Client	Inspiration	CPU	Memory
uav-client-1	Raspberry Pi 4 Model B	0.65	512 MiB
uav-client-2	NVIDIA Jetson Nano	0.55	512 MiB
uav-client-3	Raspberry Pi Zero 2 W	0.35	448 MiB
uav-client-4	NVIDIA Jetson Orin Nano	1.00	768 MiB
uav-client-5	NXP NavQPlus	0.45	512 MiB

9.4 Operator Dashboard

The operator dashboard consumes a single HTTP and JSON adapter and never loads checkpoints, datasets, preprocessing objects or cryptographic material directly. The adapter verifies the locked hashes of the two deployment artefacts, the frozen federated checkpoint and the fitted preprocessor, before loading them, and performs genuine inference with that model.

The interface exposes fleet state, live predictions with confidence, federated round progress, security events and alerts. When the containerised backend is unreachable, the adapter reports replay mode explicitly rather than silently substituting recorded telemetry, while inference continues against the real model. Because the model is binary, the interface displays only Normal or Attack and never asserts an attack family.

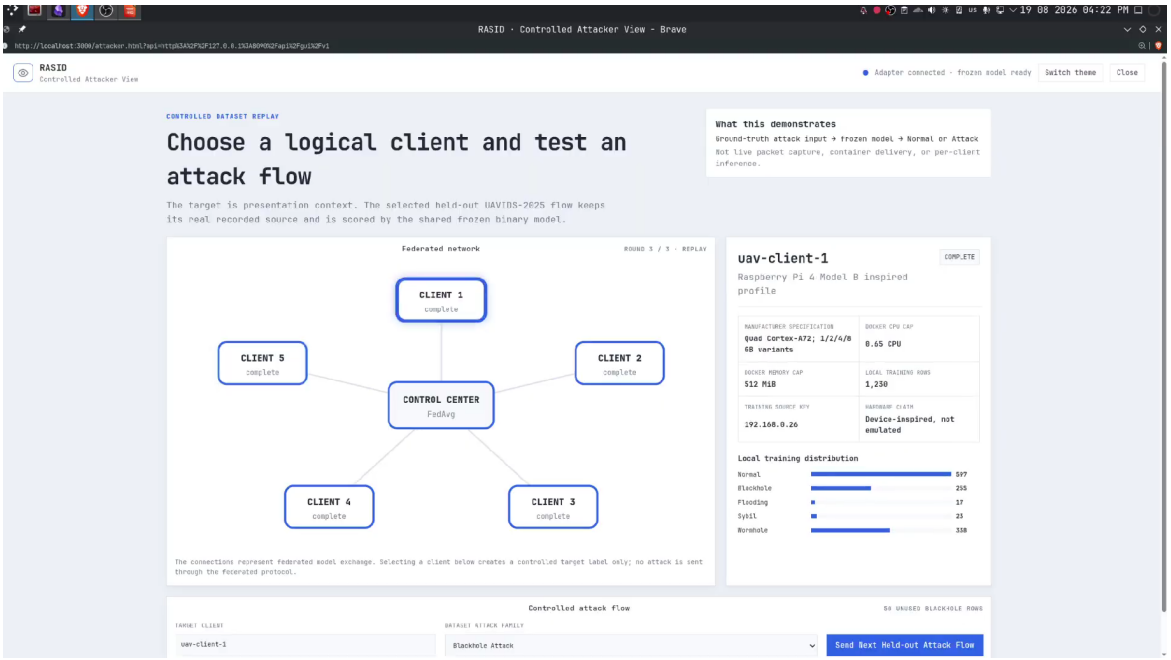


Figure 7 Operator dashboard, fleet overview. Each of the five UAV clients is shown with its current connection state, most recent prediction and model confidence, alongside the active federated round.

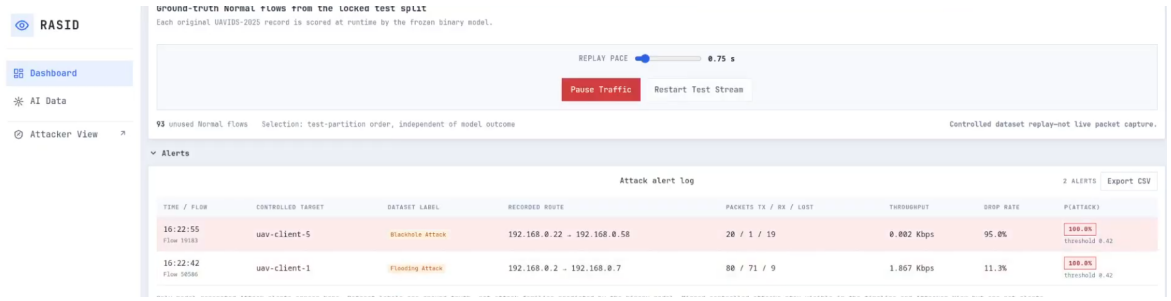


Figure 8 Detection view. Live inference output for a selected client, showing the binary classification, the model confidence against the locked decision threshold of 0.42, and recent detection history.

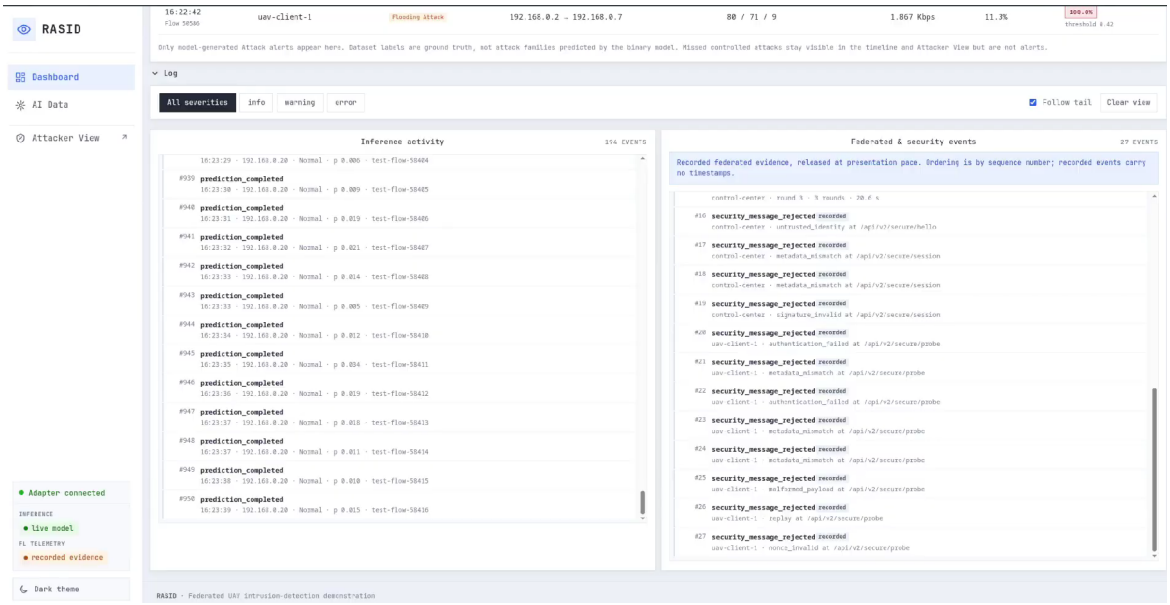


Figure 9 Security event view. Federated round progress alongside the security event log, including rejected messages and their coarse rejection category as recorded by the protocol described in Section 8.5.

10 Experimental Design

The evaluation protocol was designed so the test partition could influence no decision. The development stage refuses to load the test file at all. Every candidate, checkpoint and threshold decision was made on validation, after which the complete policy, including artefact hashes, was written to a locked configuration file. Only then was the test partition transformed with the already-fitted preprocessor and evaluated once, across all seven models. The hash of the locked configuration was confirmed unchanged after evaluation, establishing that no setting was revised in response to a test result.

Seven models are evaluated: one centralized, one federated, and five local-only. Reported metrics are accuracy, macro-F1, per-class precision, recall and F1, balanced accuracy, false-positive and false-negative rates, ROC-AUC, PR-AUC, log loss and full confusion counts. Macro-F1 is the primary selection metric because the partitions carry roughly 77% attack prevalence, under which plain accuracy would reward a model that neglects the normal class. Detection rates were also computed per original attack family, so weakness in a specific family is not concealed by the binary aggregate.

Determinism was enforced throughout: a fixed seed of 42, CPU execution with a single thread and deterministic algorithms enabled.

11 Results

11.1 Intrusion Detection Performance

Table 12 reports the single locked-test evaluation across all seven models, and Figure 10 presents the macro-F1 values visually.

Answering RQ1 directly: federated averaging reached macro-F1 0.9502, exceeding the local-only mean of 0.8969 by 0.0533 and exceeding the best individual local model, client 2 at 0.9444, by 0.0058.

Answering RQ2 directly: the federated model trails centralized pooling by 0.0273, recovering

Table 12 Locked-test results on 53,949 source-disjoint flows. Thresholds were frozen from validation before the partition was opened.

Model	Thr.	Acc.	Macro-F1	Atk. prec.	Atk. recall	FPR
Centralized	0.46	0.9841	**0.9775**	0.9923	0.9871	0.0260
Federated FedAvg	0.42	0.9650	**0.9502**	0.9780	0.9767	0.0749
Local uav-client-1	0.28	0.9519	0.9295	0.9595	0.9791	0.1410
Local uav-client-2	0.49	0.9618	0.9444	0.9680	0.9831	0.1111
Local uav-client-3	0.90	0.9322	0.8977	0.9372	0.9778	0.2237
Local uav-client-4	0.48	0.9462	0.9256	0.9786	0.9512	0.0711
Local uav-client-5	0.22	0.8301	0.7872	0.9468	0.8268	0.1586
Local-only mean	–	–	0.8969	–	–	–

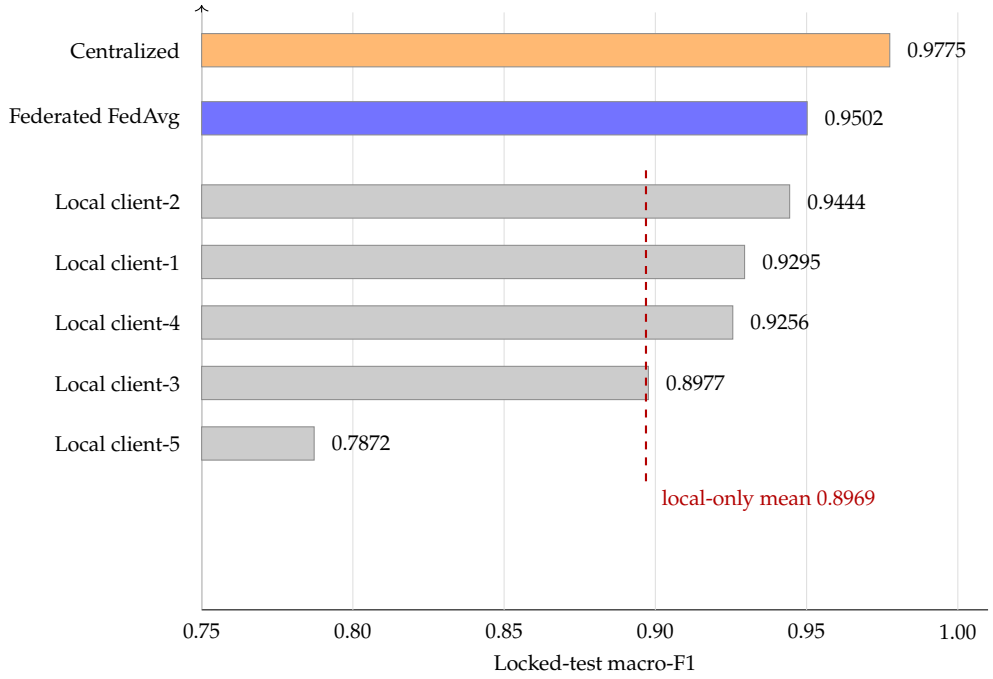


Figure 10 Locked-test macro-F1 across all seven models. Federated averaging sits above every individual local-only model and well above their mean, but below centralized pooling.

approximately two thirds of the gap between isolated and pooled training. The federated false-positive rate of 0.0749 is roughly three times the centralized rate of 0.0260.

Figure 11 localises where the deficit falls.

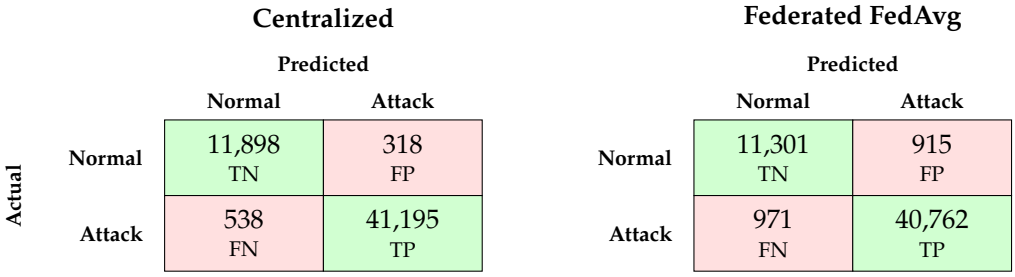


Figure 11 Locked-test confusion counts. The federated model's deficit relative to centralized falls mainly on the normal class: it raises false positives by a factor of 2.9 while false negatives rise by a factor of 1.8.

The federated model produces 915 false positives against 318 for centralized, and 971 false negatives against 538. False positives therefore rise by a factor of 2.9 while false negatives rise by only 1.8, so most of the federated deficit is on the normal class rather than on detection itself.

11.2 Client-Level Benefit

The five local-only models span a macro-F1 range of 0.1572, from 0.7872 for client 5 to 0.9444 for client 2. Table 13 contrasts each client's isolated result with the federated model it would receive instead.

Table 13 Per-client benefit of federation. Every client receives the same federated model, so the gain differs by what each client could achieve alone.

Client	Train rows	Local macro-F1	Federated macro-F1	Gain
uav-client-1	1,230	0.9295	0.9502	+0.0207
uav-client-2	1,046	0.9444	0.9502	+0.0058
uav-client-3	591	0.8977	0.9502	+0.0525
uav-client-4	2,027	0.9256	0.9502	+0.0246
uav-client-5	1,254	0.7872	0.9502	+0.1630

Every client gains, and the gain is largest where isolated training is weakest. Client 5 alone missed 7,227 attacks on the test partition against 971 for the federated model, despite holding a larger training partition than clients 2 and 3. Training partition size does not predict local performance: client 4 holds the largest partition and reaches only 0.9256, while client 2 reaches the best local result from roughly half as many rows. The stronger predictor is distributional coverage, and as Table 6 showed, clients 4 and 5 both lack Sybil observations entirely.

Selected thresholds also vary widely across the local models, from 0.22 to 0.90, indicating that isolated training produces models whose output probabilities are calibrated very differently from one another.

11.3 Convergence

Figure 12 shows the federated validation history across all 30 rounds.

The curve answers the question of whether round 30 was a genuine plateau. Macro-F1 rises sharply over the first four rounds, from an initial value near 0.43 to approximately 0.92, then enters a long shallow improvement phase, ending at 0.9517. Validation log loss falls steeply to

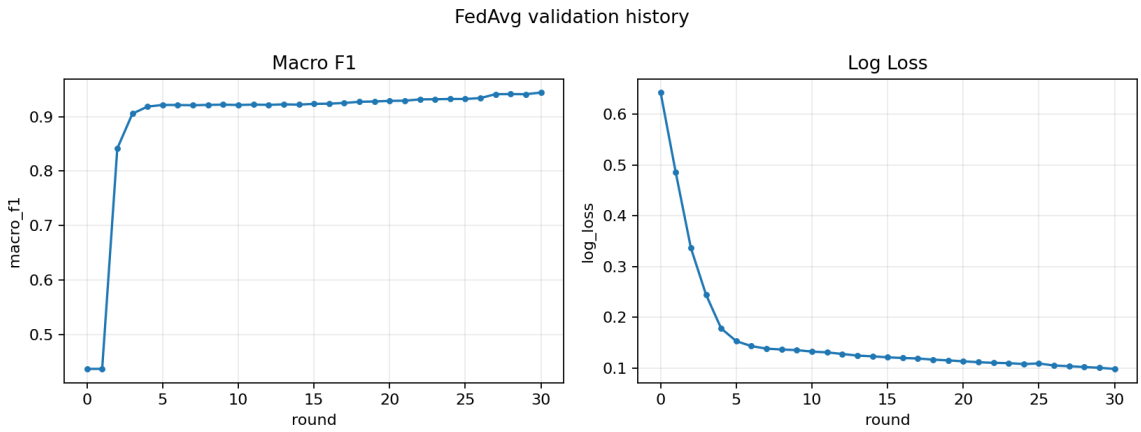


Figure 12 FedAvg validation history over 30 rounds. Macro-F1 rises steeply through round 4, plateaus near 0.92, and then improves slowly to 0.9517 by round 30. Validation log loss continues to decline gradually throughout.

roughly 0.15 by round 5 and then declines slowly but monotonically to approximately 0.10 at round 30.

The honest reading is that the model had substantially converged but was not fully saturated. The bulk of the achievable performance was obtained within five rounds, which is itself an operationally useful finding for a bandwidth-constrained fleet. However, both curves were still improving marginally at the point the configured schedule ended, so a longer schedule would likely have yielded a small additional gain. Because 30 rounds was fixed in the locked configuration before training, the selection of round 30 reflects the end of the schedule rather than a detected plateau, and Section 13 records this as a limitation.

11.4 Attack-Family Detection

Detection rates computed per original attack family show that the weakest family for both principal models is the wormhole attack, at recall 0.9587 for centralized and 0.9264 for federated. This weakness is invisible in the binary aggregate scores and is reported separately for that reason. Wormhole traffic is plausibly the hardest of the four families to separate on flow statistics alone, because a tunnelled path can preserve the volumetric characteristics of legitimate forwarding while altering only the topology through which traffic passes, and topology is not directly represented in the 15 approved features.

11.5 Post-Quantum Functional Equivalence

The central security result answers the first half of RQ3: adding the cryptographic layer changed nothing about the learning outcome. The plain and secure runs each completed three rounds with all five clients, and their aggregates were bit-identical. The maximum difference between plain and secure aggregate tensors was 0.0 against a tolerance of 10^{-7}, as was the maximum difference from an independent re-aggregation of the secure run. Final validation macro-F1 remained 0.9517 in both modes.

Independent recomputation of every aggregate agreed exactly with the control centre. Isolation checks confirmed that each client exposed exactly one training file, the control centre exposed none, all container roots and data mounts were read-only, all Linux capabilities were dropped, and the built image contained no dataset files.

11.6 Post-Quantum Overhead

Table 14 gives measured per-operation timings, answering the second half of RQ3.

Table 14 Measured cryptographic operation timings in milliseconds. Handshake operations occur once per client per session; AEAD operations occur per message.

Operation	Count	Mean	Median	Max
ML-KEM encapsulation	5	2.371	1.748	4.947
ML-KEM decapsulation	5	0.144	0.137	0.253
ML-DSA signing	5	0.495	0.350	0.922
ML-DSA verification	5	0.628	0.469	1.160
AES-GCM encryption	202	0.065	0.058	0.266
AES-GCM decryption	142	0.097	0.059	1.274

Every operation completes in single-digit milliseconds or less, against local training times measured in hundreds of milliseconds to several seconds per client per round. Cryptographic work is therefore three to four orders of magnitude below training cost in this workload.

Message envelopes expanded by 34.2% in both directions, from a mean of 65,596 to 88,043 bytes server-to-client and from 64,279 to 86,289 bytes client-to-server. This expansion is dominated not by cryptographic overhead proper but by base64 representation of binary archives and ciphertext inside JSON.

End-to-end runtimes did not increase. The secure coordinator completed in 27.58 seconds against 32.35 seconds for the plain run, and compose-to-terminal runtimes were 36.90 and 44.94 seconds respectively. These figures are host-specific and fall within normal run-to-run variation for this workload, so they support only the claim that cryptographic cost is not a dominant term, not a claim that security improves performance.

For reference, the containerised runs completed three rounds with all five clients contributing sample counts of 1,230, 1,046, 591, 2,027 and 1,254 for a total of 6,148 per round. Round one required 3.45 seconds against 1.19 and 1.24 seconds for rounds two and three, the difference being one-time library and optimizer warm-up under simultaneous constrained startup. Aggregation itself took between 1.67 and 2.05 milliseconds per round, and update archives measured 46.7 to 46.8 KiB.

11.7 Security Probes

A controlled attack run exercised twelve adversarial probes while genuine training continued. Table 15 details each probe, the manipulation applied, the expected and observed outcomes, and whether the message reached the aggregator.

Figure 13 aggregates these by category.

Between the tampering probes, a genuine message was sent and correctly accepted, and after the replay probe a further genuine message was accepted. This confirms that rejection does not corrupt receiver state: an invalid ciphertext does not advance the sequence counter, so legitimate traffic continues to flow after an attack attempt. Genuine training completed normally, and all three aggregate archives were byte-identical to the plain run, confirming that no rejected message reached or influenced aggregation.

Table 15 Controlled adversarial probes. All twelve were rejected, and none reached aggregation. Rejection categories are those emitted by the protocol.

Stage	Probe	Manipulation	Observed category	Expected	Reached agg.?
Registration	Untrusted identity	unrecognised key fingerprint	untrusted_identity	reject	No
Handshake	Wrong run	`run_id` altered in signed metadata	metadata_mismatch	reject	No
Handshake	Wrong identity	`client_id` changed to another client	metadata_mismatch	reject	No
Handshake	Invalid signature	final byte of ML-DSA signature flipped	signature_invalid	reject	No
Channel	Ciphertext tamper	final byte of AES-GCM ciphertext flipped	authentication_failed	reject	No
Channel	Wrong run	header `run_id` altered	metadata_mismatch	reject	No
Channel	Wrong round	header `round` set to 99	metadata_mismatch	reject	No
Channel	Wrong recipient	header `recipient` altered	metadata_mismatch	reject	No
Channel	Wrong message type	header `message_type` altered	authentication_failed	reject	No
Channel	Malformed payload	ciphertext replaced with invalid base64	malformed_payload	reject	No
Channel	Replay	previously accepted message resent	replay	reject	No
Channel	Duplicate nonce	nonce of an earlier message reused	nonce_invalid	reject	No

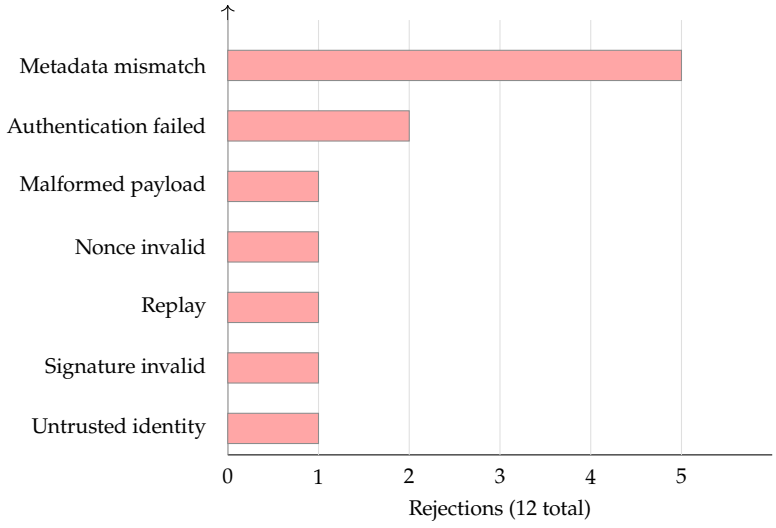


Figure 13 Safe rejections raised during the controlled adversarial run. Each category corresponds to a distinct check in the protected-message contract. Every probe was rejected before decoding, and none reached aggregation.

11.8 Reproducibility Outcomes

Two independent clean reruns of the development stage reproduced all 27 locked artefacts, tables, histories and plots byte for byte. An audit that recomputed every metric from saved per-row predictions reproduced all seven metric rows exactly, with matching discrete predictions and confusion counts. The hash of the locked configuration was unchanged before and after the final test evaluation.

12 Discussion

12.1 Answering the Research Questions

RQ1: does federation improve on independent training?. Yes, and by a margin that is operationally meaningful rather than marginal. Federated averaging reached 0.9502 macro-F1 against a local-only mean of 0.8969, and every one of the five clients gained. The comparison against client 5 is the sharpest illustration: that model reached only 0.7872 and missed 7,227 attacks on the test partition. An aircraft training in isolation on that traffic distribution would field a detector that fails on roughly one attack in six. The same aircraft participating in federated training receives a global model that misses 971.

The mechanism matters for interpreting this. Client 5's partition contains no Sybil traffic at all, so its isolated model has never been shown that attack family. The federated gain is therefore not primarily statistical efficiency from additional rows; it is coverage of attack behaviour a node never observes. Table 13 supports this: the gain is largest for the two clients whose distributions are most incomplete, and smallest for client 2, whose partition already covers all four families.

RQ2: how close does federation come to centralized training?. It recovers approximately two thirds of the gap, trailing by 0.0273 macro-F1. The residual deficit is the expected consequence of non-identically distributed clients: each client optimises towards its own local minimum during its two local epochs, and averaging divergent parameter sets produces a model that is not the minimiser any of them would have reached [26]. With attack prevalence spanning 50.64% to 90.08% and two clients missing an entire attack family, the divergence here is substantial.

The error structure localises the cost. False positives rise by a factor of 2.9 while false negatives rise by only 1.8, so most of the deficit falls on the normal class rather than on detection. Attack recall is nearly preserved at 0.9767 against 0.9871. This is the more favourable way for the gap to manifest: the federated model remains close to centralized at the job it exists to do, and pays mainly in specificity.

The optimisation budgets in Table 9 rule out a competing explanation. Federated clients collectively perform 3,000 gradient steps against the centralized model's 2,548, so the gap is not caused by federated training receiving less optimisation. It is caused by each federated step being computed on a partial view of the distribution.

RQ3: does post-quantum protection cost detection performance or latency?. No, on both counts, and the evidence is unusually clean. The secure and plain runs produced aggregates identical to the bit, at a maximum tensor difference of 0.0. This follows from the design: the cryptographic layer wraps the message contract without touching the training path, so it cannot perturb the arithmetic. The practical consequence is that the security properties can be evaluated independently of model quality, and adopting them costs nothing in detection.

The latency picture is equally clear. Handshake operations run once per client per session in single-digit milliseconds, and per-message AEAD operations average well under a tenth of a

millisecond, against local training of hundreds of milliseconds to seconds. Cryptographic work sits three to four orders of magnitude below training cost.

12.2 Operational Reading of the False-Positive Rate

Whether the federated trade is acceptable depends on operational context rather than on the metric. At 0.0749, roughly one normal flow in thirteen is flagged. In a high-volume link this generates an alert stream that will exhaust analyst attention and, in the familiar pattern, train operators to dismiss alerts.

The threshold of 0.42 was selected to maximise macro-F1, which weights both classes equally. An operator who values precision differently should reselect the threshold against an explicit cost model, and the calibration data to do so is already saved. This is a tuning decision rather than a limitation of the approach. That said, the point should not be overstated: raising the threshold to suppress false positives will reduce attack recall, and the 0.0749 figure is a real operational cost of federation in its current form, not merely an artefact of metric choice.

12.3 Client Heterogeneity as a Design Signal

The finding that training partition size does not predict local performance deserves emphasis. Client 4 holds the largest partition and performs worse than client 2, which holds roughly half as many rows. What separates them is distributional coverage rather than volume.

This has a direct consequence for how a real fleet should be composed. Adding aircraft that observe traffic similar to what the fleet already covers yields little, while a single participant contributing an unrepresented attack family may matter a great deal. Sample-count-weighted averaging, the standard FedAvg rule used here, weights by volume and is therefore indifferent to this distinction. Client 4 receives the largest aggregation weight despite holding no Sybil traffic and being flooding-dominated. A weighting scheme sensitive to distributional contribution rather than row count is a natural direction, and the per-client evidence saved by this project is the right starting point for evaluating one.

The spread of selected thresholds across local models, from 0.22 to 0.90, is a second signal. Models trained in isolation on different distributions produce output probabilities that are not comparable to one another. Any deployment that mixed locally trained models, or allowed aircraft to fall back to local training when disconnected, would need explicit recalibration rather than a shared threshold.

12.4 Convergence and Communication Cost

The convergence curve carries a practical implication beyond the question of whether round 30 was a plateau. Macro-F1 reached approximately 0.92 within four rounds, capturing the large majority of the eventual 0.9517. For a fleet communicating over constrained links, this suggests that a substantially shorter schedule would deliver most of the benefit at a fraction of the communication cost. At roughly 47 KiB per update per client per round, thirty rounds costs approximately 7 MiB of model traffic across the fleet, where five rounds would cost around 1.2 MiB for a model only marginally weaker.

The counterpoint is that both curves were still improving at round 30, so the schedule length was not validated as sufficient. A production system would be better served by an explicit convergence criterion than by a fixed round count.

12.5 What the Security Evidence Does and Does Not Show

The adversarial run supports a narrower claim than it might first appear to. Twelve probes across seven categories were rejected while genuine training continued, and aggregates remained byte-identical to the plain run. The interleaving of genuine messages between probes is a meaningful detail: it demonstrates that rejection is fail-safe rather than merely fail-closed, since an invalid ciphertext does not advance receiver state and legitimate traffic continues afterwards.

What this does not demonstrate is resistance to attacks nobody thought to write a probe for. The probes were written by the same team that wrote the checks, which is a structural limitation of self-designed adversarial testing. It also says nothing about an adversary who holds a legitimate identity: a client that authenticates correctly and then submits a deliberately poisoned update passes every check in this system and enters aggregation unimpeded. Given that FedAvg weights by self-reported sample count, such a client could additionally inflate its own influence.

The 34.2% message expansion is the one measured figure that would matter over a constrained radio link, but it is largely an artefact of encoding rather than cryptography: binary archives and ciphertext are base64-encoded inside JSON. The genuinely cryptographic component, the ML-DSA signature and ML-KEM ciphertext, is a few kilobytes against update archives of roughly 47 KiB.

12.6 Interpreting the Result Honestly

The overall outcome is favourable but mixed, and it is worth being precise about which claims the evidence supports.

The evidence supports the claim that sample-weighted federated averaging over these five non-identically distributed logical sources produces a detector better than isolated local training and somewhat worse than centralized pooling, under one fixed policy, on one deterministic seed and split. It supports the claim that the implemented post-quantum layer provides confidentiality, authentication, context binding and replay resistance for model messages under the stated trust model, at negligible computational cost and without altering aggregation.

The evidence does not support a claim of statistical superiority, because a single seed and split yield no confidence intervals. It does not establish privacy, because parameter updates are visible in plaintext to the control centre after decryption and because preprocessing statistics were fitted centrally across all five training partitions. It does not establish performance on physical UAV hardware. And it does not establish generalisation beyond source-address and feature-signature disjointness, because the dataset lacks the scenario and temporal metadata that stronger claims would require.

Stating these boundaries is not a weakness of the result. A system intended for defence use is more valuable when its evaluation boundary is explicit, because an operator can then reason about what additional evidence deployment would require.

13 Limitations

Data and client realism.. UAVIDS-2025 is generated from a documented NS-3 simulation and is neither operational nor classified traffic. The five participants are source-address groups rather than verified physical aircraft, a distinction especially delicate under Sybil conditions where source identity may itself be forged. Because the dataset carries no scenario, run, timestamp or capture-session fields, held-out partitions can be shown disjoint by source address and feature signature but not independent in any stronger sense. Rows originating from the same simulation scenario may appear in both training and test partitions without the partitioning logic being able

to detect it.

Training partition size.. Only 5.4% of retained data is used for training, for the structural reasons given in Section 5.3. While this makes the generalisation test harder rather than easier, it does limit the model capacity that can be fitted without overfitting, and a larger and more diverse training partition would likely raise all three arms.

Statistical strength.. All results come from a single deterministic seed and a single split, so no confidence intervals accompany any figure. The 0.0273 gap between federated and centralized training, and the 0.0533 gap between federated and local-only mean, are point estimates from one run. The validation partition was reused for candidate, checkpoint and threshold selection, making it a development set rather than an untouched benchmark, so validation figures are optimistic.

Schedule length.. The federated checkpoint was selected at round 30, the last round in the configured schedule. As Section 11.3 shows, both macro-F1 and log loss were still improving marginally at that point, so round 30 represents the end of the schedule rather than a detected plateau.

Detection scope.. The model is binary. It does not identify which of the four families is present, so an operator receives no indication of whether an alert reflects a blackhole, flooding, Sybil or wormhole condition, and these require materially different responses. Wormhole is the weakest family for both principal models, at federated recall 0.9264, a weakness the binary aggregate conceals.

Privacy.. Raw records never leave a client, but this is narrower than privacy. Preprocessing statistics were fitted centrally on the pooled training partitions of all five clients. The control centre observes plaintext parameter updates after decryption, and parameter updates are derived from local data and are not inherently free of information about it. Neither secure aggregation nor differential privacy is implemented.

Security scope.. The cryptographic layer is an academic prototype operating over a controlled allowlist of demonstration identities, without certificates, revocation, hardware-backed key storage, automated rotation or organisational identity proofing. It protects application payloads but leaves HTTP metadata observable, provides no forward secrecy after session key compromise, guarantees no availability, and cannot determine whether an authenticated client's update is benign. The adversarial evidence covers only the probe categories that were implemented, and those probes were authored by the same team that wrote the checks.

Deployment realism.. Container resource limits are inspired by real edge hardware rather than emulating it. The demonstration does not reproduce ARM instruction sets, accelerator behaviour, power draw, thermal throttling, radio links, sensor input or flight dynamics, so no hardware performance or energy claim follows from its timings. All measurements come from one host. The federated schedule also requires all five clients in every round, with no partial-participation or dropout recovery path, which is unrealistic for aircraft that lose connectivity.

14 Future Work

Strengthen the statistical claim.. The most immediate improvement requires no new capability: repeat the full experiment across multiple seeds and multiple source-disjoint splits, and report confidence intervals. This would establish whether the observed federated advantage is robust

or an artefact of one partition, and it is the first question a reviewer of these results should ask. Extending the schedule beyond 30 rounds, with an explicit convergence criterion replacing the fixed count, would separately resolve the plateau question raised in Section 11.3.

Close the privacy gap.. Two steps would substantiate the privacy framing the architecture implies. Replacing pooled preprocessing with a federated statistics protocol, in which imputation and scaling parameters are computed by aggregation rather than central access, removes the one stage that currently reads all clients' data. Adding secure aggregation [32] would prevent the control centre from observing individual updates. Because the transport is already post-quantum, the natural target is a post-quantum secure aggregation construction [34, 35] rather than the classical Diffie-Hellman-based original. Differential privacy on client updates would bound residual leakage, at a measurable cost in detection performance that should be characterised rather than assumed.

Defend against malicious participants.. The current system authenticates who may contribute but not what they contribute. Because aggregation is weighted by self-reported sample count, a single authenticated client submitting crafted parameters can influence the global model disproportionately. Robust aggregation rules, update-statistics screening and Byzantine-detection approaches from the federated IDS literature [36, 37] would address this, and the existing per-client evidence provides a baseline against which to measure any such defence.

Address non-IID degradation directly.. This project deliberately used unmodified FedAvg so the aggregation rule would not confound the measurement. With that baseline established, the obvious next step is to evaluate drift-correcting alternatives against it: FedProx's proximal term [27], SCAFFOLD's control variates [28], or FedNova's normalisation for unequal local step counts [29], which is directly relevant given the 300 to 960 per-client step range reported in Table 9. A weighting scheme sensitive to distributional contribution rather than row count, motivated by Section 12.3, is a further direction.

Improve detection utility.. Extending from binary detection to attack-family classification would give operators actionable alerts, and the family labels required are already retained in the partition files. The wormhole weakness is the natural first target, and graph-based representations that capture forwarding topology rather than per-flow statistics [11, 24] are a promising route, since wormhole attacks alter topology while preserving volumetric characteristics.

Harden the deployment path.. Realistic operation requires tolerating clients that arrive late, drop out or rejoin, which the present all-or-nothing schedule does not. On the security side, the controlled allowlist would need to become a real identity lifecycle with certificates, revocation and rotation, and the plain HTTP transport should be complemented by production TLS in a post-quantum or hybrid configuration rather than replaced by application-layer protection alone. Replacing base64-in-JSON with a binary transport would remove most of the measured 34.2% message expansion. At fleet scale, migrating to an established federated framework would supply the client sampling and dropout handling this implementation deliberately forgoes, as discussed in Section 4.5.

Validate on real platforms.. The strongest remaining gap is the absence of physical hardware. Running the existing containerised protocol unchanged on ARM companion computers of the class the profiles were drawn from would produce the first genuine measurements of on-device training time, energy draw and thermal behaviour, and would test the protocol over real radio links rather than a Docker bridge network. Capture of live traffic from such a testbed would also

provide the scenario and temporal metadata whose absence currently bounds what the evaluation can claim.

15 Conclusion

This project set out to determine whether a fleet of UAVs can collaboratively train a network intrusion detection model without transmitting raw traffic, and whether the resulting model exchange can be protected against adversaries including future quantum-capable ones. Both questions were answered affirmatively within a clearly bounded evaluation.

On RQ1, sample-count-weighted federated averaging across five non-identically distributed logical clients reached a locked-test macro-F1 of 0.9502, exceeding the mean local-only baseline of 0.8969 by 0.0533. Every client gained, and the gain was largest where isolated training was weakest: the weakest client's own model missed 7,227 attacks where the federated model missed 971. Federation's contribution is coverage of attack behaviour that individual nodes never observe, rather than statistical efficiency from additional rows.

On RQ2, the federated model trailed centralized pooling by 0.0273, recovering roughly two thirds of the gap between isolated and pooled training. The deficit falls predominantly on the normal class, with attack recall nearly preserved at 0.9767 against 0.9871. Matched optimisation budgets rule out training volume as the explanation; the cause is each federated step being computed on a partial view of the distribution.

On RQ3, the post-quantum layer built on ML-KEM-768, ML-DSA-65, HKDF-SHA-256 and AES-256-GCM delivered confidentiality, authentication, context binding and replay resistance for every model message without altering aggregation in any measurable way. Secure and plain runs produced bit-identical aggregates at a maximum tensor difference of 0.0. Cryptographic operations cost single-digit milliseconds for per-session handshakes and well under a millisecond per message, three to four orders of magnitude below local training cost. A controlled adversarial run rejected all twelve probes across seven categories while genuine training completed and aggregates remained unchanged.

The project's methodological contribution is as substantial as its numerical one. Every model and threshold decision was frozen from validation and hash-locked before the test partition was opened, the lock was verified unchanged afterwards, clean reruns reproduced all 27 development artefacts byte for byte, and an independent audit recomputed every reported metric from saved predictions. This discipline is what allows the results above to be stated as measurements rather than as claims.

Equally, the evaluation boundary is explicit. The clients are logical source partitions rather than verified aircraft, the data is simulated rather than operational, the results rest on one seed and one split without confidence intervals, preprocessing was fitted centrally and is therefore not privacy-preserving, and the cryptographic layer is an academic prototype rather than production infrastructure. The system demonstrates that post-quantum secure federated intrusion detection is achievable and that its security cost is not the obstacle. Establishing that it is deployable requires the multi-seed validation, secure aggregation, malicious-client defences and physical-hardware testing set out in Section 14.

The result relevant to a defence organisation weighing this architecture is narrow but useful: collaborative detection across a fleet recovers most of the benefit of pooling traffic centrally, without moving raw records off the aircraft, and protecting the exchange with standardised post-quantum cryptography costs essentially nothing in either detection performance or runtime.

References

[1] C. G. Leela Krishna and Robin R. Murphy. "A review on cybersecurity vulnerabilities for unmanned aerial vehicles". In: *2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. IEEE. 2017, pp. 194–199.

[2] Ahmad Y. Javaid et al. "Cyber security threat analysis and modeling of an unmanned aerial vehicle system". In: *2012 IEEE Conference on Technologies for Homeland Security (HST)*. IEEE. 2012, pp. 585–590.

[3] Gaurav Choudhary et al. "Intrusion detection systems for networked unmanned aerial vehicles: A survey". In: *2018 14th International Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE. 2018, pp. 560–565.

[4] Qasem Abu Al-Haija and Ahmad Al Badawi. "High-performance intrusion detection system for networked UAVs via deep learning". In: *Neural Computing and Applications* 34.13 (2022), pp. 10885–10900.

[5] National Institute of Standards and Technology. *Module-Lattice-Based Key-Encapsulation Mechanism Standard*. Tech. rep. FIPS 203. U.S. Department of Commerce, 2024. DOI: 10.6028/NIST.FIPS.203.

[6] National Institute of Standards and Technology. *Module-Lattice-Based Digital Signature Standard*. Tech. rep. FIPS 204. U.S. Department of Commerce, 2024. DOI: 10.6028/NIST.FIPS.204.

[7] Dustin Moody et al. *Transition to Post-Quantum Cryptography Standards*. Tech. rep. NIST IR 8547 (Initial Public Draft). National Institute of Standards and Technology, U.S. Department of Commerce, Nov. 2024. DOI: 10.6028/NIST.IR.8547.ipd.

[8] Qingli Zeng, Abdalrahman Bashir, and Farid Nait-Abdesselam. "UAVIDS-2025: A Benchmark Dataset for Intrusion Detection in UAV Networks Using Machine Learning Techniques". In: *2025 IEEE Conference on Communications and Network Security (CNS)*. IEEE. 2025, pp. 1–9. DOI: 10.1109/CNS66487.2025.11194990.

[9] Harrison Kurunathan et al. "Machine learning-aided operations and communications of unmanned aerial vehicles: A contemporary survey". In: *IEEE Communications Surveys and Tutorials* (2024).

[10] Venkatraman Subbarayalu and Maria Anu Vensuslaus. "An intrusion detection system for drone swarming utilizing timed probabilistic automata". In: *Drones* 7.4 (2023), p. 248.

[11] Wei Zhou, Bo Li, and Xiang Zeng. "Intrusion detection model of UAV system based on machine learning and neural network". In: *Frontiers in Physics* 13 (2026), p. 1660104. DOI: 10.3389/fphy.2025.1660104.

[12] Raghdah Saem Ramadan, Sarah Aljehani, et al. "Machine learning approaches to intrusion detection in unmanned aerial vehicles (UAVs)". In: *Neural Computing and Applications* (2024). DOI: 10.1007/s00521-024-10306-y.

[13] Samuel Chase Hassler, Umair Ahmad Mughal, and Muhammad Ismail. "Cyber-Physical Intrusion Detection System for Unmanned Aerial Vehicles". In: *IEEE Transactions on Intelligent Transportation Systems* 25.6 (2024), pp. 6106–6117.

[14] Jason Whelan et al. *UAV Attack Dataset*. IEEE DataPort. 2020. DOI: 10.21227/00dg-0d12.

[15] Qingli Zeng, Abdalrahman Bashir, and Farid Nait-Abdesselam. *UAVIDS-2025: A Benchmark Dataset for Intrusion Detection in UAV Networks Using Machine Learning Techniques*. Zenodo. Version v1. May 2025. DOI: 10.5281/zenodo.15336998.

[16] Brendan McMahan et al. "Communication-Efficient Learning of Deep Networks from Decentralized Data". In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*. 2017, pp. 1273–1282.

[17] Sawsan Abdul Rahman et al. "Internet of things intrusion detection: Centralized, on-device, or federated learning?" In: *IEEE Network* 34.6 (2020), pp. 310–317.

[18] Mohamed Amine Ferrag et al. "Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning". In: *IEEE Access* 10 (2022), pp. 40281–40306. DOI: 10.1109/ACCESS.2022.3165809.

[19] Babatunde Olanrewaju-George and Bernardi Pranggono. "Federated learning-based intrusion detection system for the internet of things using unsupervised and supervised deep learning models". In: *Cyber Security and Applications* 3 (2025), p. 100068. DOI: 10.1016/j.csa.2024.100068.

[20] R. Ramesh et al. "An optimal federated learning-based intrusion detection for IoT environment". In: *Scientific Reports* (2025). DOI: 10.1038/s41598-025-93501-8.

[21] Y. Bai et al. "Enhancing IoT Security via Federated Learning: A Comprehensive Approach to Intrusion Detection". In: *IET Information Security* (2025). DOI: 10.1049/ise2/8432654.

[22] Michel Nguyen. *Federated Learning-Based Intrusion Detection System for IoT Networks in Resource-Constrained Environments*. TechRxiv preprint. 2025. DOI: 10.36227/techrxiv.176403418.87468767/v1.

[23] Jorge Luiz Guerra, Carlos Catania, and Eduardo Veas. "A Comparative Analysis of Machine Learning Techniques for IoT Intrusion Detection". In: *arXiv preprint arXiv:2111.13149* (2021).

[24] Qingli Zeng, Yinjin Fu, and Farid Nait-Abdesselam. "FedGraph-ID: A Federated Graph Learning Framework for Intrusion Detection in UAV Networks Under Adversarial Settings". In: *IEEE INFOCOM 2026 – IEEE Conference on Computer Communications*. IEEE. 2026.

[25] Qingli Zeng, Semire Olatunde-Salawu, and Farid Nait-Abdesslam. "Federated learning and GAN-augmented intrusion detection system for UAV networks". In: *2024 IEEE 10th International Conference on Collaboration and Internet Computing (CIC)*. IEEE. 2024, pp. 50–59.

[26] Yue Zhao et al. "Federated Learning with Non-IID Data". In: *arXiv preprint arXiv:1806.00582* (2018).

[27] Tian Li et al. "Federated Optimization in Heterogeneous Networks". In: *Proceedings of Machine Learning and Systems (MLSys)*. Vol. 2. 2020, pp. 429–450.

[28] Sai Praneeth Karimireddy et al. "SCAFFOLD: Stochastic Controlled Averaging for Federated Learning". In: *Proceedings of the 37th International Conference on Machine Learning (ICML)*. 2020, pp. 5132–5143.

[29] Jianyu Wang et al. "Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 33. 2020, pp. 7611–7623.

[30] Qinbin Li, Bingsheng He, and Dawn Song. "Model-Contrastive Federated Learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 10713–10722.

[31] Daniel M. Lu et al. “Non-IID data in Federated Learning: A Survey with Taxonomy, Metrics, Methods, Frameworks and Future Directions”. In: *arXiv preprint arXiv:2411.12377* (2024).

[32] Keith Bonawitz et al. “Practical Secure Aggregation for Privacy-Preserving Machine Learning”. In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 2017, pp. 1175–1191. DOI: 10.1145/3133956.3133982.

[33] Open Quantum Safe Project. *liboqs: C Library for Quantum-Safe Cryptographic Algorithms*. Available: https://openquantumsafe.org/liboqs/. 2025.

[34] Ruozhou Zuo et al. “Post-quantum Privacy-Preserving Aggregation in Federated Learning Based on Lattice”. In: *Cyberspace Safety and Security (CSS)*. Springer, 2022, pp. 314–326. DOI: 10.1007/978-3-031-18067-5_23.

[35] Yijian Lin et al. “Efficient Post-Quantum Cross-Silo Federated Learning Based on Key Homomorphic Pseudo-Random Function”. In: *Mathematics* 13.9 (2025), p. 1404.

[36] Milad Rahmati and Nima Rahmati. “Byzantine-Robust Federated Learning Framework with Post-Quantum Secure Aggregation for Real-Time Threat Intelligence Sharing in Critical IoT Infrastructure”. In: *arXiv preprint arXiv:2601.01053* (2026).

[37] Naila Latif, Wenping Ma, and Hafiz Bilal Ahmad. “Advancements in securing federated learning with IDS: a comprehensive review of neural networks and feature engineering techniques for malicious client detection”. In: *Artificial Intelligence Review* 58 (2025), p. 91. DOI: 10.1007/s10462-024-11082-w.