Privacy-Preserving Hierarchical Federated Learning over Data Spaces

Alexandros S. Kalafatelis
Dept. of Port Management and Shipping
National and Kapodistrian
University of Athens
Euboea, Greece
alexkalafat@uoa.gr

Vasileios Nikolakakis
Dept. of Port Management and Shipping
National and Kapodistrian
University of Athens
Euboea, Greece
vnikolak@uoa.gr

Nikolaos Tsoulakos
Laskaridis Shipping Co., Ltd
Athens, Greece
tsoulakos@laskaridis.com

Panagiotis Trakadas
Dept. of Port Management and Shipping
National and Kapodistrian
University of Athens
Euboea, Greece
ptrakadas@uoa.gr

***Abstract*—We present FLoDS, a Data Space native federated learning framework that binds European Data Space governance (identity, policy, audit) to the training loop. FLoDS introduces stewards, trusted intermediaries that perform hierarchical aggregation and attestation via signed, auditable round manifests, and composes client-level differential privacy with secure aggregation and CKKS-based encrypted steward-server fusion. Beyond single ecosystems, FLoDS is designed to operate within and across Data Spaces, enabling cross-silo learning where identity and policy context travels with model updates. Using a predictive maintenance scenario consistent with a Shipping Data Space, we evaluate under non-IID partitions and Byzantine gradient sign-flip attacks. FLoDS sustains accuracy and cross-client parity, where state-of-the-art baseline approaches fail, maintaining an R^2 of 0.94, and Jain's index of 0.97, with privacy-robustness cost accounting. These results indicate that stewarded, governance-aligned FL can deliver verifiable confidentiality and resilience with explicit costs for deployment within and across Data Spaces.**

***Index Terms*—data spaces, federated learning, differential privacy, homomorphic encryption, predictive maintenance**

I. Introduction

Data has become a strategic asset underpinning modern AI services. Its usefulness grows through reuse across organisations and contexts, yet in regulated, high-stakes domains (e.g., healthcare, finance, manufacturing, mobility, maritime) raw data cannot be freely centralized due to confidentiality, compliance, and intellectual-property constraints [1], [2]. Data Spaces (DS) address this tension by coupling governance (identity, usage control, audit) with compute-to-data execution over certified connectors and federation services, enabling value-added analytics without relinquishing custody of raw data [3], [4]. Despite this progress, the DS landscape remains fragmented, as multiple spaces overlap, and different connectors hinder interoperability and innovation across DS boundaries [5], [6].

To mitigate these limitations, recent DS blueprints highlight sector-agnostic, open specifications as the basis for cross-DS interoperability and for building higher-level services (e.g., AI-as-a-Service) [7]–[11]. However, current deployments often stop at sovereign data exchange and KPI dashboards and rarely bind governance roles (identity, policy, audit) to the learning loop itself. The few DS-native FL examples found in the literature typically assume a single coordinator within one DS, benign clients, and harmonized semantics, with limited research on aspects of robustness, cross-silo training, fairness between clients, and traffic overheads of privacy-enhancing technologies (PETs) [12].

To address these gaps, we introduce *Federated Learning over Data Spaces (FLoDS)*, a DS-native federated learning framework that maps DS governance to the training loop. FLoDS injects stewards in the hierarchical FL paradigm (i.e., trusted intermediaries between clients and the server), where clients apply per-client differential privacy (DP) and participate in steward-local secure aggregation (SecAgg). Stewards issue signed, hash-chained manifests (i.e., participants, controls, learning timings/traffic) and, when policy requires, encrypt steward to server fusion using Cheon-Kim-Kim-Song (CKKS) homomorphic encryption. Furthermore, from the server side, the orchestrator applies stabilized, globally clipped updates. The design makes privacy-robustness cost trade-offs auditable and applies within a single DS or across multiple DSs with minimal adaptation.

For concreteness, we instantiate FLoDS in a simulated Shipping Data Space scenario for predictive maintenance (PdM), forecasting Main Engine (ME) cylinder Exhaust Gas

This research initiative is supported by the European Union's Horizon Europe Framework Programme for Research and Innovation, under the Optimized Real-time maritime Border Surveillance with Intelligent satellite-drone Systems (ORBIS) project (Grant Agreement Project 101225659). Views and opinions expressed are, however, those of the authors only and do not necessarily reflect those of the ORBIS consortium and the European Union, which cannot be held responsible for them.

Temperature (EGT) across fleets, where company headquarters act as cross-silo clients, being also members of different DSs. Toward this end, the main contributions of this work include the following:

- DS-native learning architecture: a stewarded hierarchy that ties DS roles (identity, policy, audit) to aggregation and attestation, decoupling clients from the server and enabling intra- and cross-DS training;
- Integrated PET stack with budgets: client-level DP, SecAgg, and CKKS composed with server-side adaptivity and global update clipping, with configuration surfaces and per-round time/bandwidth accounting;
- Deployment-oriented evaluation: comparison between state-of-the-art baseline models under non-IID partitions and Byzantine stress in a Shipping DS consistent PdM scenario on real ship telemetry, reporting utility, cross-client fairness, and systems costs.

The remainder of this paper is organized as follows: Section II reviews DSs, compute-to-data execution, and security-driven FL applications. Section III presents the FLoDS architecture and protocols. Section IV details the experimental setup (dataset, preprocessing, model, FL strategies, metrics) and reports results. Section V concludes with findings, limitations, and directions for future work.

II. Background and Related Work

A. Data Spaces and Compute-to-Data Execution

European DS initiatives define identity, policy/usage control, and compute-to-data to enable sovereign analytics across organizations. The International Data Spaces (IDS) Reference Architecture Model and the Gaia-X trust framework specify certified connectors, brokers, and policy enforcement as first-class components for controlled exchange and in-place processing [13]–[15]. Sector pilots like the Mobility DS and the Seaport DS, show how connector-mediated exchange reduces bilateral integration and improves auditability, but they largely stop at sovereign data sharing and dashboards rather than collaborative model training under privacy and non-IID conditions [16], [17].

Within this substrate, FL is frequently proposed as a privacy-enhancing, data visiting mechanism in which models travel to silos and only updates leave the enclave. However, while several works argue FL is domain-agnostic, they highlight practical gaps, including semantic alignment across providers, orchestration complexity, and the need for intermediating roles beyond point-to-point coordination [14], [18]–[20]

Early FL examples on DS like [21] and [22], adopt a centralized FL topology (single server coordinating clients), assume benign IID clients and synchronous participation, and under-report robustness, fairness, and PET induced system costs. However, these models can suffer from negative transfer when tested under severe data heterogeneity among clients [23].

B. PET for FL Applications

From the privacy-security side, a practical FL stack layers complementary PETs. Secure aggregation (SecAgg) hides per-client updates from an honest-but-curious server and tolerates client dropouts [24]. To guard against manipulation, verifiable aggregation augments the protocol with homomorphic hashing, improving integrity at added protocol cost [25], [26]. Approximate homomorphic encryption (CKKS) enables packed arithmetic over ciphertexts for encrypted aggregation, but real deployments must still budget ciphertext inflation, key custody/rotation, and straggler sensitivity [27]. In parallel, client-level DP-SGD per-client clipping with Gaussian noise bounds each silo's influence and mitigates update inference, while Rényi DP and its subsampled variant provide tight, round-wise composition/accounting under varying participation [28], [29].

Orthogonal to PETs, the learning rule governs robustness and equity across silos. Byzantine-robust aggregators explicitly limit the effect of adversarial or outlying updates [30], [31]. Server-side choices-adaptive optimizers (FedAdam/FedYogi), proximal regularization (FedProx), and global update clipping-further stabilize training under heterogeneity [32]–[34]. However, a systematic evaluation that (i) binds DS governance roles (identity, policy, audit) to the learning loop via aggregation and attestation, (ii) quantifies robustness and fairness under non-IID data and Byzantine stress, and (iii) reports auditable time/traffic for concrete PET choices (SecAgg/CKKS/DP) is currently missing for DS applications.

C. Positioning of FLoDS

As illustrated in Fig. 1, centralized FL follows a flat client-server communication loop, where clients (e.g., edge devices, ships, etc.) send updates directly to a single coordinator while identity, policy, and audit remain outside the training path and per-client updates are typically visible to the server. Cross-silo FL in a DS retains this flat topology where the clients are organizational silos (e.g., a company HQ aggregating ship operational telemetry data) connected via certified DS connectors for compute-to-data execution. In practice, DS-oriented FL deployments still assume benign participants and synchronous rounds, and DS governance is not bound to the aggregation step, yielding limited confidentiality and non-standardized auditing of privacy budgets [35], [36].

Unlike previous DS works that emphasize sovereignty but treat learning as a black box, FLoDS generalizes hierarchical FL to the DS setting by introducing stewards, accredited intermediaries between clients and the server. This architecture ties DS governance to the learning loop, confines risk near the source, enables both intra- and cross-DS training, and makes training auditable. To our knowledge, the joint integration of (i) DS-native roles tied to aggregation and attestation, (ii) hierarchical stewardship, and (iii) measured robustness privacy-cost trade-offs under non-IID and Byzantine conditions has not been addressed in prior FL works.

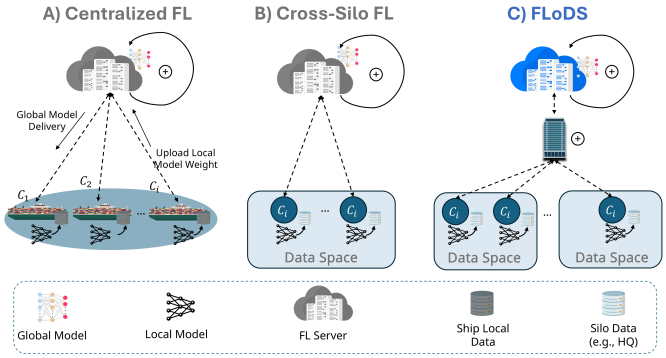

Fig. 1: FL topologies: (A) Centralized FL, where benign considered clients upload deltas directly to a single server, and governance and audit are external to training; (B) Cross-silo FL in a DS, where aggregation remains flat and assumes benign clients; and (C) FLoDS, where stewards interpose between clients and server to bind DS roles enabling auditable, governance-aligned training within and across DS.

III. FLoDS Framework Overview

We study collaborative model training among organizations that belong to one or more DS, a governed ecosystem where algorithms "visit" data at rest (compute-to-data) rather than replicating datasets across boundaries. In maritime settings, DS participants include ship operators, OEMs, terminal operators, port authorities, and related service providers. FLoDS is a hierarchical FL scheme that aligns with DS governance, enabling the training of a model for a common ML problem, across different DS participants, while addressing non-IID data distributions and operational constraints typically found in shipping.

A. System Model and Actors

Let $\mathcal{C} = \{1, ..., N\}$ denote clients (DS members), with each client i holding a private dataset D_i. Let $\mathcal{S} = \{1, ..., M\}$ denote stewards, accredited independent third-party organizations customary in maritime governance in this scenario (e.g., classification societies, recognized auditors, or flag/port-state control bodies). A central orchestrator (FLoDS server) coordinates rounds across stewards and maintains the global model but never receives raw data.

In detail, under FLoDS, clients compute locally, while Stewards sit at the DS aggregation boundary and (i) validate identities and participation policies; (ii) enforce privacy controls mandated by the DS; (iii) perform robust aggregation over their member updates; and (iv) publish signed, tamper-evident round records (participants, controls, timings, and validation metrics). The orchestrator fuses steward aggregates using a stabilized optimizer step. This placement mirrors DS practice, where neutral intermediaries administer identity and policy, and extends it into the learning loop.

Stewards introduce a neutral, auditable enforcement layer to the data-space ecosystem, delivering guarantees that a flat client-server FL topology cannot provide. Specifically, they are designed to enable:

TABLE I: NOMENCLATURE

Symbol	Description
$\mathcal{C}$, $\mathcal{S}$	Global sets of clients and stewards
$\mathcal{C}_s$	Clients attached to steward s
$\mathcal{C}_s^t$	Poisson-sampled clients at steward s active at round t
$\mathcal{A}_s^{(t)}$	Clients in $\mathcal{C}_s^t$ that responded to steward s at round t
$\mathcal{N}_i^t$	SecAgg neighbors of client i at round t
t	Round index
n_i, W_s, W	Local sample count of client i, sample mass at steward s, and total sample mass
$\mathbf{w}^t$, $\mathbf{w}_s^t$	Global model at round t, and steward s broadcast to its clients
$\boldsymbol{\theta}_i^t$	Client i's local model after its solve at round t
$\mathbf{u}_i^t$	Raw client delta $\mathrm{vec}(\boldsymbol{\theta}_i^t - \mathbf{w}_s^t)$
$\mathbf{u}_i^{t,\mathrm{clip}}$	Client delta after ℓ_2 clipping with bound C
$\tilde{\mathbf{u}}_i^t$	DP-sanitized delta after adding Gaussian noise
$\hat{\mathbf{u}}_i^t$	Masked delta uploaded by client i
$\mathbf{r}_{ij}^t$	Symmetric pairwise mask between clients i and j at round t
$s(i,j)$	Sign for mask cancellation
$\bar{\mathbf{u}}_s^t$	Steward s sample-weighted mean update
$\boldsymbol{\Delta}^t$, $\mathbf{g}^t$	Server fused update across stewards and clipped step used for the optimizer update
$\mathbf{m}_t$, $\mathbf{v}_t$	Yogi first- and second-moment accumulators at round t
$G_{\max}$	Server-side step clip bound
C, σ	Client clip bound and DP noise multiplier
q, T	Poisson subsampling rate and number of rounds
ε, δ	Target DP privacy budget and failure probability
ρ	Fraction of Byzantine (adversarial) clients per round
$K_{\min}$, τ	Per-steward minimum quorum for round progress; round timeout
δ_{fair}, J	Fairness stabilizer in Jain's index and resulting fairness score
$\mathsf{Enc}_{pk}(\cdot)$, $\mathsf{Dec}_{sk}(\cdot)$	CKKS encryption with public key and decryption with secret key

- Governance and liability: credible policy enforcement and accountable attestation to the DS ecosystem. An accredited third party (e.g., a classification society) validates participation and issues signed, per-round manifests that regulators and partners can verify without access to raw data.
- Separation of trust and keys: a separation between data ownership and cryptographic termination to the DS ecosystem. Privacy controls (e.g., SecAgg, CKKS parameters) and key handling are anchored at a neutral entity rather than at competitors or the central server.
- Statistical robustness near the source: first-line containment of faulty or adversarial updates to the DS ecosystem, limiting propagation of outliers and poisoning.
- Operational containment and simplicity: a consolidation point for compliance and protocol complexity to the DS ecosystem. Clients remain lightweight, while cross-organization cryptographic coordination, policy checks, and audit logging are handled where institutional capability already exists.
- Auditability and provenance: tamper-evident model provenance to the DS ecosystem. Signed round records bind participants, privacy settings, timings/traffic, and validation metrics, enabling post-hoc reconstruction of

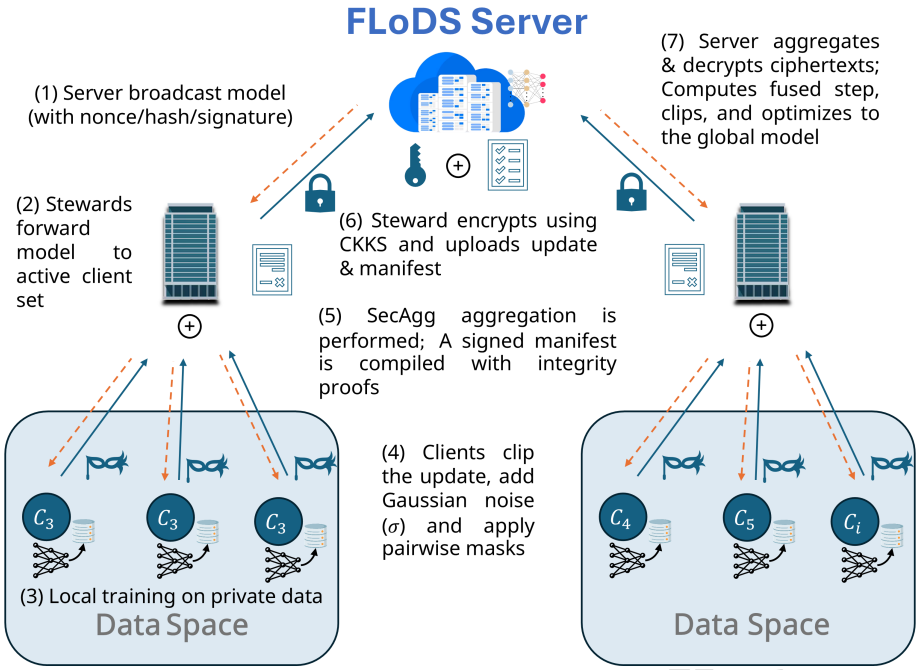

Fig. 2: FLoDS training across one or more Data Spaces. The server broadcasts the downstream model, clients via stewards train locally, clip and DP-noise their updates, while stewards run SecAgg, compile a signed manifest, and encrypt steward aggregates with CKKS. Finally, the server verifies, decrypts, fuses, clips, and applies the global step.

decisions without exposing sensitive data.

Therefore, at round t, a Poisson-subsampled set $C_t \subseteq \mathcal{C}$ (rate q) trains locally on $\boldsymbol{w}^t$, forms a model delta, applies client-side DP, and sends a masked message for SecAgg to its Steward. Each steward i) verifies identity/policy at the DS boundary, ii) cancels pairwise masks so that only the sum of client deltas is revealed to the steward, and iii) returns a steward delta. The orchestrator averages steward deltas and applies a stabilized optimizer (e.g., SGD/Adam/Yogi). A signed, hash-chained manifest and fairness log are appended each round.

B. Learning Objective and Local Solve

We adopt sample-weighted empirical risk minimization [33], [34]:

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} L(\boldsymbol{w}) \triangleq \sum_{i=1}^{N} p_i\, f_i(\boldsymbol{w}), \quad f_i(\boldsymbol{w}) = \frac{1}{n_i} \sum_{(x,y)\in D_i} \ell(f_{\boldsymbol{w}}(x), y) \tag{1}$$

where, $p_i \geq 0$ with $\sum_i p_i = 1$ weight clients and ℓ is the supervised loss. Client $i \in C_t \cap \mathcal{C}_s$ receives $\boldsymbol{w}_s^t$ (steward broadcast) and solves a proximal local subproblem [33]:

$$\boldsymbol{\theta}_i^t \in \arg\min_{\boldsymbol{\theta}} \left\{ f_i(\boldsymbol{\theta}) + \tfrac{\mu}{2}\|\boldsymbol{\theta} - \boldsymbol{w}_s^t\|_2^2 \right\}, \quad \mu \geq 0 \tag{2}$$

where μ controls client drift under non-IID data, while at $\mu = 0$ reduces to FedAvg. The flattened parameter delta is given by

$$\boldsymbol{u}_i^t = \mathrm{vec}\left(\boldsymbol{\theta}_i^t - \boldsymbol{w}_s^t\right) \in \mathbb{R}^d \tag{3}$$

C. Client-Side Differential Privacy

We use the Gaussian mechanism on the ℓ_2-clipped update [28]

$$\boldsymbol{u}_i^{t,\mathrm{clip}} = \frac{\boldsymbol{u}_i^t}{\max\left(1, \frac{\|\boldsymbol{u}_i^t\|_2}{C}\right)}, \quad \|\boldsymbol{u}_i^{t,\mathrm{clip}}\|_2 \leq C, \tag{4}$$

$$\widetilde{\boldsymbol{u}}_i^t = \boldsymbol{u}_i^{t,\mathrm{clip}} + \mathcal{N}\left(\boldsymbol{0},\, \sigma^2 C^2 \mathbf{I}_d\right) \tag{5}$$

here C bounds per-round sensitivity and σ sets the noise scale. We compose privacy loss across T rounds under Poisson subsampling rate q using a moments accountant, and report per-client (ε, δ) at fixed δ [29], [37]. Furthermore, the steward records (C, σ) and pre/post-clip norms for audit. It should be noted that FLoDS treats each client's update as a single DP unit, i.e., the DP guarantee is at the client level, not the per-record level inside each silo.

D. Secure Aggregation at the Steward

To hide each client's DP-sanitized update $\widetilde{\mathbf{u}}_i^t \in \mathbb{R}^d$ while revealing only their sum, we adopt Bonawitz-style pairwise masking [24]. For each active pair (i, j) in round t, clients derive a symmetric seed via authenticated Elliptic-Curve Diffie-Hellman (ECDH), expand it with an HMAC-based Key Derivation Function (HKDF, SHA-256), and obtain a pseudorandom mask $\mathbf{r}_{ij}^t \in \mathbb{R}^d$ where $\mathbf{r}_{ij}^t = \mathbf{r}_{ji}^t$. We fix a sign convention $s(i, j) = +1$ if $i < j$, and $s(i, j) = -1$ otherwise. Client i uploads the masked message

$$\widehat{\mathbf{u}}_i^t = \widetilde{\mathbf{u}}_i^t + \sum_{j\in\mathcal{N}_i^t} s(i,j)\, \mathbf{r}_{ij}^t \tag{6}$$

to steward s, where N_i^t denotes the set of clients j with whom client i establishes a SecAgg pairwise mask at round t. Let $\mathcal{A}_s^{(t)}$ denote the set of responding clients for steward s at round t. Therefore, by summing over $\mathcal{A}_s^{(t)}$, each mask appears once with $+1$ and once with -1, so masks cancel and the steward learns only the sum of DP-sanitized updates, given by

$$\sum_{i\in\mathcal{A}_s^{(t)}} \widehat{\mathbf{u}}_i^t = \sum_{i\in\mathcal{A}_s^{(t)}} \widetilde{\mathbf{u}}_i^t \tag{7}$$

It should be noted that local ℓ_2 clipping and Gaussian noise are applied before masking. Furthermore, to account for potential dropouts, we adopt the standard Bonawitz recovery mechanism, where per-pair seed shares (Shamir t-of-n) are escrowed for the round so that, if a planned participant aborts, the steward reconstructs the missing $\mathbf{r}_{ij}^t$ to remove dangling masks without exposing any $\widetilde{\mathbf{u}}_i^t$ [24]. The steward compiles a signed manifest recording $\mathcal{A}_s^{(t)}$, protocol version, clip C, DP σ, and any dropout/rekey events for ex-post audit [24].

After secure aggregation, steward s holds the sum of DP-sanitized client updates, $\Delta_s^t = \sum_{i\in\mathcal{A}_s^{(t)}} \widetilde{\mathbf{u}}_i^t$, with $\mathcal{A}_s^{(t)}$ the responding clients at round t. In this work we use a weighted mean at the steward

$$\bar{\mathbf{u}}_s^t = \frac{\sum_{i\in\mathcal{A}_s^{(t)}} n_i\, \widetilde{\mathbf{u}}_i^t}{\sum_{i\in\mathcal{A}_s^{(t)}} n_i}$$

where n_i is the local sample count (equal weights if unspecified). The steward transmits $\bar{\mathbf{u}}_s^t$ (encrypted under CKKS) and a signed manifest that records the aggregation policy and weights (e.g., "mean, size-weighted"), clip norm C, DP noise σ, and $\mathcal{A}_s^{(t)}$.

FLoDS adopts the mean at the steward to isolate the effects of DP clipping/noise, SecAgg confidentiality, and server-side stabilization (global clipping with adaptive optimizer) on robustness and fairness. If however a steward fails quorum, its update is skipped, and the server re-normalizes weights over the remaining stewards.

E. Encrypted Steward-Server Communication

When policy requires confidentiality beyond steward scope, the steward further encrypts its fused update $\bar{\mathbf{u}}_s^t$ under CKKS [27] with the server's public key. With CKKS SIMD batching, a polynomial modulus degree N yields $N/2$ complex slots. We encode each steward aggregate $\bar{\mathbf{u}}_s^t \in \mathbb{R}^d$ elementwise into slots (packing real values in the complex field), so that homomorphic addition and plaintext-ciphertext scalar multiplication support weighted fusion, as expressed by

$$\mathsf{Enc}_{pk}\Big(\sum_{s=1}^{M} w_s\, \bar{\mathbf{u}}_s^t\Big) \;\approx\; \sum_{s=1}^{M} w_s\, \mathsf{Enc}_{pk}\big(\bar{\mathbf{u}}_s^t\big) \tag{8}$$

where the weights w_s are applied as plaintext scalars. In our deployment, SecAgg protects client-level confidentiality within a steward, while CKKS protects the steward aggregate in transit/at rest toward the server. The steward transmits the ciphertext together with its signed manifest, which records the active HE policy (parameter set and key epoch). The server verifies manifests and either (i) homomorphically fuses steward ciphertexts then decrypts, or (ii) decrypts per-steward and fuses in plaintext.

It should be noted that FLoDS restricts CKKS to addition and scalar multiplication (i.e., no rotations), which limits noise growth and keeps latency predictable. Furthermore, CKKS is used to provide confidentiality of $\bar{\mathbf{u}}_s^t$ against an honest-but-curious server, and not to mitigate adversarial values, as robustness aspects are handled by clipping/DP and the server's optimizer.

Each round also appends two signed, hash-chained logs, a manifest (i.e., configuration, participation, privacy modes, timings, traffic) and a fairness (i.e., storing per-client metrics and Jain's index).

F. Server Fusion and Stabilized Update

Let $W_s = \sum_{i\in\mathcal{A}_s^{(t)}} n_i$ be the sample mass at steward and $W = \sum_{s=1}^{M} W_s$. After decryption, the server fuses steward deltas and applies Yogi for the model update:

$$\boldsymbol{\Delta}_t = \sum_{s=1}^{M} \frac{W_s}{W}\, \bar{\mathbf{u}}_s^t \tag{9}$$

$$\mathbf{g}_t = \mathrm{clip}_{G_{\max}}(\Delta_t) \tag{10}$$

where $\mathrm{clip}_{G_{\max}}(\mathbf{x}) = \mathbf{x}\cdot\min\big(1, G_{\max}/\|\mathbf{x}\|_2\big)$. In detail, with Yogi update, the server initializes $m_0 = 0$ and $v_0 = 0$, with fixed hyperparameters $(\eta, \beta_1, \beta_2, \varepsilon)$, updating based on the following

$$\mathbf{m}_t = \beta_1\, \mathbf{m}_{t-1} + (1-\beta_1)\, \mathbf{g}_t \tag{11}$$

$$\mathbf{v}_t = \mathbf{v}_{t-1} - (1-\beta_2)\, \mathrm{sign}\big(\mathbf{v}_{t-1} - \mathbf{g}_t^{\odot 2}\big) \odot \mathbf{g}_t^{\odot 2} \tag{12}$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta\, \frac{\mathbf{m}_t/(1-\beta_1^t)}{\sqrt{\mathbf{v}_t/(1-\beta_2^t)} + \varepsilon} \tag{13}$$

where $\odot$ denotes elementwise multiplication, $g_t^{\odot 2} = g_t \odot g_t$, η is the server learning rate, (β_1, β_2) are momentum coefficients, and ε is a small constant for numerical stability. It should be noted that the client clip C bounds each $\mathbf{u}_i^{t,\mathrm{clip}}$ before masking (per-client sensitivity), whereas the server clip $G_{\max}$ bounds the fused cross-steward step Δ_t after SecAgg/CKKS.

IV. Performance Evaluation

A. Experimental Setup

All simulations were executed on a personal workstation equipped with an Intel® Core™ Ultra i9 275HX CPU (2.7 GHz), 32 GB RAM, and a 64-bit OS. Experiments were implemented using Python v3.11.0, PyTorch v2.10.0, and TenSEAL v0.3.16. No GPU acceleration was used for training, ensuring the reported timings reflect CPU-only operation.

B. Shipping Data-Space Validation Scenario and Dataset

We evaluate FLoDS in a realistic maritime environment with governed DSs and neutral stewards, reflecting how shipping organizations already collaborate under compliance and class rules. Toward this end, we simulate three DSs to exercise

Algorithm 1 FLoDS

Require: Initial global model $\mathbf{w}^0$, number of rounds T, client clip C, noise σ, subsampling rate q, server clip $G_{\max}$, and CKKS public key pk
Ensure: Final model $\mathbf{w}^T$, signed manifests and fairness logs

FOR $t = 0, \ldots, T-1$ DO
  Orchestrator: broadcast $\mathbf{w}^t$ to all stewards
  FOR each steward $s \in \mathcal{S}$ DO
    Sample active clients $\mathcal{C}_s^t \subseteq \mathcal{C}_s$ via Poisson subsampling at rate q
    Forward $\mathbf{w}_s^t \leftarrow \mathbf{w}^t$ to all $i \in \mathcal{C}_s^t$

  Client $i \in \mathcal{C}_s^t$ **at steward** s**:**
  Local solve $\Rightarrow \boldsymbol{\theta}_i^t$; set $\mathbf{u}_i^t \leftarrow \mathrm{vec}(\boldsymbol{\theta}_i^t - \mathbf{w}_s^t)$
  DP CLIP: $\mathbf{u}_i^{t,\mathrm{clip}} \leftarrow \mathbf{u}_i^t / \max\{1, \|\mathbf{u}_i^t\|_2 / C\}$
  DP NOISE: $\tilde{\mathbf{u}}_i^t \leftarrow \mathbf{u}_i^{t,\mathrm{clip}} + \mathcal{N}(\mathbf{0}, \sigma^2 C^2 \mathbf{I})$
  SECAGG MASK: for each $j \in \mathcal{N}_i^t$, derive a shared seed (ECDH→HKDF) and generate symmetric mask $\mathbf{r}_{ij}^t = \mathbf{r}_{ji}^t$
    Set $\hat{\mathbf{u}}_i^t \leftarrow \tilde{\mathbf{u}}_i^t + \sum_{j \in \mathcal{N}_i^t} s(i,j)\, \mathbf{r}_{ij}^t$ and send $\hat{\mathbf{u}}_i^t$ to steward s

  Steward s**:**
  Wait up to timeout τ; set $\mathcal{A}_s^{(t)} \leftarrow$ set of responding clients
  IF $|\mathcal{A}_s^{(t)}| < K_{\min}$ THEN
    Log quorum failure for round t; **continue** to next steward
  SECAGG AGGREGATE: run SecAgg dropout recovery (Alg. 2) on $\{\hat{\mathbf{u}}_i^t\}_{i \in \mathcal{A}_s^{(t)}}$ to obtain $\sum_{i \in \mathcal{A}_s^{(t)}} \tilde{\mathbf{u}}_i^t$
  Compute local mass $W_s \leftarrow \sum_{i \in \mathcal{A}_s^{(t)}} n_i$
  Sample-weighted mean: $\bar{\mathbf{u}}_s^t \leftarrow \frac{1}{W_s} \sum_{i \in \mathcal{A}_s^{(t)}} n_i\, \tilde{\mathbf{u}}_i^t$
  IF CKKS enabled THEN
    $\mathbf{c}_s^t \leftarrow \mathsf{Enc}_{pk}(\bar{\mathbf{u}}_s^t)$; send $\mathbf{c}_s^t$ and signed manifest to orchestrator
  ELSE
    Send $\bar{\mathbf{u}}_s^t$ and signed manifest to orchestrator

  Server (orchestrator):
  For each steward message: verify manifest signature and hash-chain; check policy fields
  Let $\mathcal{S}_{\mathrm{ok}}^t$ be stewards that satisfied quorum and passed verification
  Compute $W \leftarrow \sum_{s \in \mathcal{S}_{\mathrm{ok}}^t} W_s$
  IF CKKS enabled THEN
    Fuse ciphertexts $\mathbf{c}^t \leftarrow \sum_{s \in \mathcal{S}_{\mathrm{ok}}^t} \frac{W_s}{W}\, \mathbf{c}_s^t$; decrypt $\Rightarrow \boldsymbol{\Delta}^t$
  ELSE
    $\boldsymbol{\Delta}^t \leftarrow \sum_{s \in \mathcal{S}_{\mathrm{ok}}^t} \frac{W_s}{W}\, \bar{\mathbf{u}}_s^t$
  SERVER CLIP: $\mathbf{g}^t \leftarrow \mathrm{clip}_{G_{\max}}(\boldsymbol{\Delta}^t)$
  YOGI: update $(\mathbf{m}_t, \mathbf{v}_t)$; set $\mathbf{w}^{t+1} \leftarrow \mathbf{w}^t - \eta \dfrac{\mathbf{m}_t/(1-\beta_1^t)}{\sqrt{\mathbf{v}_t/(1-\beta_2^t)} + \varepsilon}$
  Append signed manifest and fairness logs for round t

Algorithm 2 SecAgg Dropout Recovery at Steward

Require: Target client set $\mathcal{C}_s^t$, timeout τ, quorum $K_{\min}$, pairwise HKDF seeds, and per-pair Shamir shares
Ensure: Sum-only DP-sanitized update $\sum_{i \in \mathcal{A}_s^{(t)}} \tilde{\mathbf{u}}_i^t$; manifest entries

Wait up to τ for masked uploads $\hat{\mathbf{u}}_i^t$; set $\mathcal{A}_s^{(t)} \leftarrow \{\, i \in \mathcal{C}_s^t :$ upload received $\}$
IF $|\mathcal{A}_s^{(t)}| < K_{\min}$ THEN
  Log quorum failure in manifest; **return**
$\mathcal{D} \leftarrow \mathcal{C}_s^t \setminus \mathcal{A}_s^{(t)}$ Dropout clients for this round
FOR each pair (i,j) with $i \in \mathcal{A}_s^{(t)}$, $j \in \mathcal{D}$ for which a mask $\mathbf{r}_{ij}^t$ was planned DO
  Request recovery shares for pair (i,j)
  Reconstruct symmetric mask $\mathbf{r}_{ij}^t$ via Shamir t-of-n
$\mathbf{S}_{\mathrm{masked}} \leftarrow \sum_{i \in \mathcal{A}_s^{(t)}} \hat{\mathbf{u}}_i^t$
$\mathbf{S}_{\mathrm{clean}} \leftarrow \mathbf{S}_{\mathrm{masked}} - \sum_{\text{recovered } (i,j)} s(i,j)\, \mathbf{r}_{ij}^t$ Compute sum-only update by subtracting recovered dangling masks from the sum of masked uploads
Set $\sum_{i \in \mathcal{A}_s^{(t)}} \tilde{\mathbf{u}}_i^t \leftarrow \mathbf{S}_{\mathrm{clean}}$
Append to manifest: $\mathcal{A}_s^{(t)}$, $|\mathcal{D}|$, recovered-pair counters, rekey flags, secagg bytes/timings, pre/post norms

Algorithm 3 Offline Manifest Verification

Require: Round-t manifest man_t, previous hash h_{t-1}, steward public key, DP accountant spec, server norm records, fairness-log handle
Ensure: PASS/FAIL flags for integrity, policy, and budget checks

1 **Integrity:** recompute hash-chain value h_t and verify digital signature on man_t; if either fails, return FAIL
2 **Policy:** check required fields and ranges for $(C, \sigma, q, T, G_{\max})$, HE parameters, $\mathcal{A}_s^{(t)}$, and recorded bytes/timings, if any violation, return FAIL
3 **DP budget:** with the configured subsampled moments accountant, recompute (ε, δ) from (C, σ, q, T) and compare to the values stored in man_t, if mismatched, return FAIL
4 **Server norms:** from server records, verify $\|\boldsymbol{\Delta}^t\|_2$ and $\|\mathbf{g}^t\|_2$ with $\mathbf{g}^t = \mathrm{clip}_{G_{\max}}(\boldsymbol{\Delta}^t)$ and $\|\mathbf{g}^t\|_2 \le G_{\max}$, if violated, return FAIL
5 **Fairness linkage:** verify fairness-log hash and, optionally, recompute summary metrics (e.g., Jain's index) from per-client entries, if inconsistent, return FAIL
6 **return** PASS

cross-DS learning, where DS operators (i.e., FL clients) include shipping companies, with operational data collected from their multiple fleets. Specifically, we evaluate FLoDS on operational data from a bulk carrier with a Main Engine (ME) rated at 12,009 HP operated by Laskaridis Shipping Ltd., previously used in [38]. The dataset comprises 59,619 high-frequency time-series records across 759 shipboard sensors. The prediction target is the ME cylinder exhaust gas temperature (EGT), a standard PdM target used to assess ME thermal and combustion state [39]–[43].

Preprocessing and Feature Engineering: To select the most correlated drivers of ME EGT, we performed Pearson screening. Nine variables were retained (Table II), prioritizing interpretable physical mechanisms (fuel delivery, air path, coolant/oil thermal states, and engine shaft dynamics). In detail, concerning the preprocessing, missing timestamps were forward/back-filled while rows with missing targets were dropped. All features use information available at or before time t. Starting from the nine original drivers, we add (i) a one-step lag of the target y_{t-1}, (ii) first differences of each driver $\Delta x_{j,t} = x_{j,t} - x_{j,t-1}$ to capture short-term changes, (iii) 12-step rolling means $\bar{x}_{j,t}^{(12)} = \frac{1}{12} \sum_{h=0}^{11} x_{j,t-h}$ to capture local trends, and (iv) a relative-time covariate (seconds since start), yielding 29 predictors, with no look-ahead. All numeric inputs are standardized. Concerning the model training and client formation, we reserved 20% as a global validation holdout (tail selection), while the remaining 80% formed the training pool for federated partitioning. In addition, a minimum-sample safeguard was set at 96 sequences, rebalancing extreme tails to avoid near-empty clients.

PdM Model: We use a compact feed-forward neural network with a lookback window of $L = 96$ time steps, ingesting all predictors and producing the EGT at time t. The network flattens the $96 \times D$ input, applies a fully connected layer with 128 units and ReLU activation, a dropout layer with rate 0.25, and a final linear unit for the scalar output. Training uses Adam with learning rate 1.5×10^{-4}, batch size 128, and six local epochs per federated round.

TABLE II: PREDICTOR VARIABLES USED TO FORECAST CYLINDER EGT

Feature	Rationale
ME Fuel Load (%)	Primary driver of combustion intensity and exhaust temperature
ME Fuel Oil Inlet Pressure (bar)	Determines atomization quality and combustion efficiency
ME Fuel Oil Inlet Temperature (°C)	Conditions fuel viscosity and spray formation at injection
Cylinder Scavenging Air Temperature (°C)	Sets oxygen supply conditions prior to ignition
Air Cooler Air Inlet Temperature (°C)	Influences intake-air density before cooling
Air Cooler Air Outlet Temperature (°C)	Direct intake-air temperature entering cylinders
Cylinder Cooling Fresh Water Outlet Temperature (°C)	Reflects cylinder heat-removal effectiveness
Piston Cooling Oil Outlet Temperature (°C)	Indicates piston crown heat flux and thermal loading
ME Rotational Speed (rpm)	Proxy for engine load, flow rates, and produced heat

FL Strategies and Hyperparameters: To evaluate the proposed FLoDS framework, we compared: (i) FedYogi [32] as the mean-based adaptive baseline and (ii) DP-FedAvg, extending FedAvg with per-client clipping and Gaussian mechanism [28]. Plain FedAvg [34] is not included as a canonical reference, as is known to be brittle under adversarial outliers [44], [45]. Server learning rates were 3×10^{-4} for FedYogi and FLoDS (Yogi server optimizer) and 1×10^{-4} for DP-FedAvg (SGD). Client-level DP was fixed across methods with $(C, \sigma, \delta) = (0.8,\ 0.6,\ 10^{-5})$ to ensure privacy aligned comparisons. The server also applies a global step clip (G_{max}) of 0.2 to the fused cross-stewards update, which is constant across runs. FLoDS further enabled secure aggregation and CKKS homomorphic encryption using polynomial degree 8192 and coefficient modulus $(60, 40, 40, 60)$, at scale 2^{40}. Clients received non-IID data via Dirichlet sampling with concentration $\alpha \in \{0.1, 0.3, 1.0\}$, as in [46], which induced controlled label/feature skew, while different client sweeps were also analyzed $\in \{8, 16, 24, 48\}$ to quantify systems scaling (round time and communication) with participation fraction set at 0.5. For each client sweep, stewards were fixed as follows, $8/3$, $16/4$, $24/5$, and $48/6$ (clients/stewards). It should be noted that all experiments were configured for a maximum of 80 FL rounds with early stopping, where validation was performed every 2 rounds and training halted after 4 consecutive validation checks without improvement (patience), i.e., after 8 rounds with no gain, while all results are averaged over 5 seeds.

Adversary Model: To evaluate the frameworks resilience, we simulate Byzantine gradient sign-flip attacks [47], under an honest-but-curious server. Specifically, in each round, a fraction $\rho \in \{0, 0.1, 0.3, 0.5\}$ of the *participating* clients (chosen uniformly at random, independently across rounds) replace their model update $\Delta \boldsymbol{w}$ with $-\Delta \boldsymbol{w}$ (unit scaling), before any masking/encryption and exactly after local training (and clipping/DP, when enabled). Attacked clients are resampled every round, while message sizes remain unchanged.

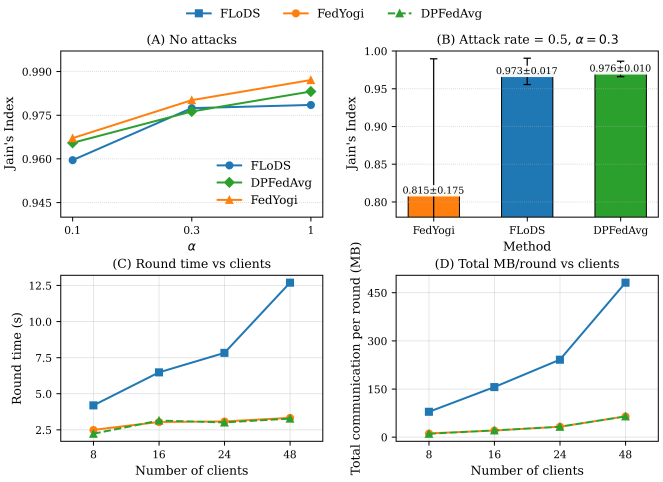

Fig. 3: Fairness and systems costs comparison. Panel (A) shows fairness across Dirichlet α without attacks; (B) shows fairness under ρ=0.5 (grad-flip) at α=0.3; (C) shows round time over the number of clients (CPU-only); and (D) shows the total communication per round.

Metrics: For each evaluation, we report standard regression error, cross-client fairness, and systems costs:

$$\mathrm{RMSE}(\hat{y}) = \sqrt{\frac{1}{|\mathcal{V}|}\sum_{t\in\mathcal{V}}(y_t - \hat{y}_t)^2}, \tag{14}$$

$$\mathrm{MAE}(\hat{y}) = \frac{1}{|\mathcal{V}|}\sum_{t\in\mathcal{V}}|y_t - \hat{y}_t|, \tag{15}$$

$$R^2(\hat{y}) = 1 - \frac{\sum_{t\in\mathcal{V}}(y_t - \hat{y}_t)^2}{\sum_{t\in\mathcal{V}}(y_t - \bar{y})^2}, \tag{16}$$

Fairness across clients is quantified with Jain's index $J \in (0, 1]$ computed on per-client utilities u_k derived from RMSE (higher is better):

$$u_k = \frac{1}{\delta_{fair} + \mathrm{RMSE}_k}, \qquad J = \frac{\left(\sum_{k=1}^{K} u_k\right)^2}{K\sum_{k=1}^{K} u_k^2}, \quad \delta_{fair} = 10^{-6}. \tag{17}$$

Systems metrics include round time (wall-clock seconds per round, CPU-only) and communication cost measured as total upload/download MB per round.

C. Performance Analysis

In this section, we report comparisons at a representative operating point (24 clients, participation f=0.5, Dirichlet α=0.3). This setting is mid-scale for our data size, avoids very small clients, and keeps wall-clock/communication budgets comparable across strategies, as larger/smaller sweeps (8, 16, 48 clients) are used for scalability trends.

Utility and fairness under benign non-IID scenarios: At the representative operating point (i.e., α=0.3, 24 clients, f=0.5), FedYogi attains the lowest error (RMSE 0.00793±0.00038, R^2=0.9661±0.0033) with high parity (J=0.9801±0.0113), DP-FedAvg is close (RMSE 0.00814±0.00015, R^2=0.9642±0.0014, J=0.9762±0.0131),

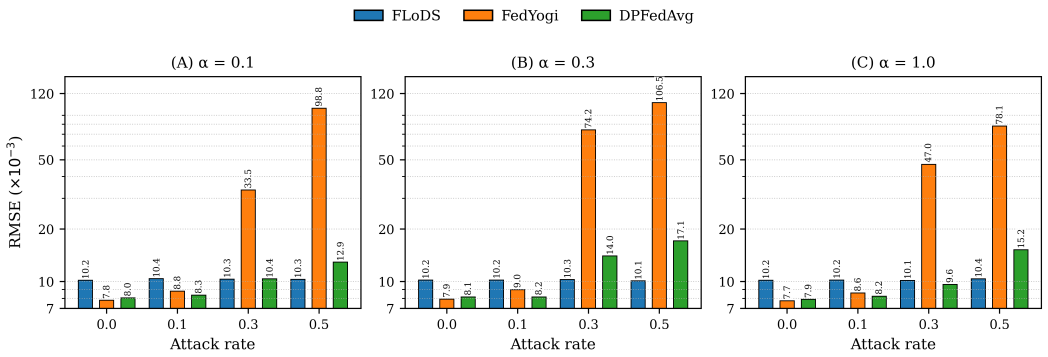

Fig. 4: Convergence and byzantine resilience. Panels show RMSE ($\times 10^{-3}$) against different attack rates ρ at three non-IID levels: (A) α=0.1, (B) α=0.3, (C) α=1.0, under 24 clients and participation set at f=0.5.

TABLE III: ACCURACY, FAIRNESS, AND SYSTEM COSTS AGAINST BYZANTINE RATES ($a = 0.3$)

Model	ρ	RMSE	MAE	R^2	Fairness	t_{round} (s)	Total (MB)
FedYogi	0.0	0.0079 ± 0.0004	0.0063 ± 0.0003	0.9661 ± 0.0033	0.9801 ± 0.0113	3.4123 ± 0.6664	32.7625 ± 2.1909
FedYogi	0.1	0.0090 ± 0.0018	0.0073 ± 0.0017	0.9551 ± 0.0120	0.9782 ± 0.0179	3.7907 ± 0.5733	30.6663 ± 2.6899
FedYogi	0.3	0.0742 ± 0.0383	0.0723 ± 0.0377	-2.6036 ± 3.7576	0.8985 ± 0.1728	4.3142 ± 0.5862	31.2338 ± 2.2957
FedYogi	0.5	0.1065 ± 0.0826	0.1039 ± 0.0813	-8.0675 ± 12.6009	0.8148 ± 0.1748	3.9371 ± 0.1528	31.6691 ± 3.1353
FLoDS	0.0	0.0102 ± 0.0019	0.0083 ± 0.0016	0.9425 ± 0.0222	0.9774 ± 0.0116	7.8253 ± 0.7889	241.3579 ± 13.6589
FLoDS	0.1	0.0102 ± 0.0021	0.0083 ± 0.0018	0.9421 ± 0.0246	0.9683 ± 0.0155	10.6057 ± 1.2952	228.3554 ± 25.9222
FLoDS	0.3	0.0103 ± 0.0019	0.0083 ± 0.0016	0.9418 ± 0.0219	0.9760 ± 0.0114	9.9445 ± 0.9907	233.2313 ± 17.1429
FLoDS	0.5	0.0101 ± 0.0018	0.0082 ± 0.0015	0.9437 ± 0.0207	0.9729 ± 0.0174	8.9209 ± 0.6309	236.4819 ± 23.4123
DP-FedAvg	0.0	0.0081 ± 0.0001	0.0065 ± 0.0001	0.9642 ± 0.0014	0.9763 ± 0.0131	3.0098 ± 0.6841	32.3220 ± 1.8292
DP-FedAvg	0.1	0.0082 ± 0.0003	0.0065 ± 0.0003	0.9641 ± 0.0029	0.9754 ± 0.0125	3.8815 ± 0.5042	30.5808 ± 3.4714
DP-FedAvg	0.3	0.0140 ± 0.0067	0.0122 ± 0.0066	0.8749 ± 0.1180	0.9806 ± 0.0090	3.7165 ± 0.4894	31.2338 ± 2.2957
DP-FedAvg	0.5	0.0171 ± 0.0080	0.0149 ± 0.0081	0.8076 ± 0.1930	0.9762 ± 0.0102	3.8056 ± 0.3877	31.6691 ± 3.1353

and FLoDS shows the expected small penalty from its privacy stack (RMSE 0.01018 ± 0.00189, R^2=0.9425 ± 0.0222, J=0.9774 ± 0.0116) (Table III). Varying the heterogeneity level across $\alpha \in \{0.1, 0.3, 1.0\}$ (Fig. 3A) yields a consistent pattern, where fairness improves mildly with larger α for all methods, and the ordering FedYogi $\gtrsim$ DP-FedAvg $\approx$ FLoDS holds across the range. In detail, the gaps are small at α=0.1 (on the order of $\sim$0.5-1.0 percentage points in J) and narrow further for $\alpha \geq 0.3$, where DP-FedAvg and FLoDS are effectively on the same level. Thus, under benign conditions FLoDS maintains high and uniform utility and parity across the tested heterogeneity levels.

Robustness to byzantine updates: As the attack rate increases, FedYogi collapses. Specifically, at ρ=0.5 it drops to a negative R^2 value with low J. DP-FedAvg degrades more gracefully but still loses accuracy at high attack while keeping parity high. In contrast, FLoDS preserves both accuracy and parity across $\rho \in \{0, 0.1, 0.3, 0.5\}$ with small variance (e.g., R^2=0.9425 ± 0.0222 at ρ=0 and 0.9437 ± 0.0207 at ρ=0.5, J between 0.9683 ± 0.0155 and 0.9774 ± 0.0116). These trends are visible in Fig. 3B, Fig. 4, and summarized in Table III.

System cost under benign conditions: With α=0.3 and f=0.5, all methods scale approximately linearly with the number of clients (Fig. 3C-D). At 24 clients, total per-round traffic is 32.76 ± 2.19 MB for FedYogi and 32.32 ± 1.83 MB for DP-FedAvg, versus 241.36 ± 13.66 MB for FLoDS. Round time shows a similar factor (3.41 ± 0.67 s and 3.01 ± 0.68 s for FedYogi/DP-FedAvg vs. 7.83 ± 0.79 s for FLoDS). Downlink is dominated by the model broadcast and is comparable across methods, while uplink expansion from secure aggregation and CKKS accounts for FLoDS's higher overhead.

Robustness vs cost trade-off: At α=0.3, f=0.5, 24 clients, and ρ=0.5, FedYogi exhibits severe utility and parity loss, DP-FedAvg retains parity but with reduced accuracy, while FLoDS maintains both accuracy ($R^2 \approx 0.944$) and parity ($J \approx 0.973$) at the cost of significant higher per-round time and bandwidth. For deployments where resilience and client equity are first-order, the FLoDS framework provides stable performance under attack with predictable systems overhead.

Privacy accounting: It should be noted that because the DP implementation (C, σ, q, δ) is matched across DP-FedAvg and FLoDS, and we stop at the same early stopping round, both methods achieve the same privacy budget ($\epsilon \approx 10.7$). Performance differences therefore arise from optimization and robustness, not from ϵ.

D. Ablation Study

To analyze the effectiveness of FLoDS we conduct a detailed ablation study, where we isolate the contribution of each component at α=0.3, 24 clients, and f=0.5. It should be noted that all ablations use the stewarded, weighted aggregation path. Table IV reports mean $\pm$ std accuracy (RMSE/MAE/R^2), cross-client parity and systems cost at $\rho \in \{0, 0.3\}$.

Specifically, under a benign regime (ρ=0), the baseline FedYogi attains the lowest errors (RMSE, MAE, R^2) with high parity (J=0.9801 ± 0.0113). The addition of DP incurs a small

TABLE IV: ABLATION STUDY

Model	ρ	RMSE	MAE	R^2	Fairness	t_{round} (s)	Total (MB)
FedYogi	0.0	0.0079 ± 0.0004	0.0063 ± 0.0003	0.9661 ± 0.0033	0.9801 ± 0.0113	3.4123 ± 0.6664	32.7625 ± 2.1909
FedYogi	0.3	0.0742 ± 0.0383	0.0723 ± 0.0377	-2.6036 ± 3.7576	0.8985 ± 0.1728	4.3142 ± 0.5862	31.2338 ± 2.2957
+SecAgg	0.0	0.0102 ± 0.0019	0.0083 ± 0.0016	0.9426 ± 0.0222	0.9764 ± 0.0139	4.2059 ± 0.6716	37.1704 ± 2.1035
+SecAgg	0.3	0.0103 ± 0.0019	0.0083 ± 0.0016	0.9416 ± 0.0220	0.9751 ± 0.0109	5.0365 ± 0.5977	35.9188 ± 2.6401
+DP	0.0	0.0080 ± 0.0002	0.0065 ± 0.0001	0.9649 ± 0.0013	0.9771 ± 0.0132	3.2112 ± 0.6441	32.4103 ± 1.9446
+DP	0.3	0.0139 ± 0.0076	0.0121 ± 0.0058	0.8688 ± 0.1210	0.9810 ± 0.0091	3.8550 ± 0.5224	31.1228 ± 2.2346
+CKKS	0.0	0.0102 ± 0.0017	0.0083 ± 0.0014	0.9426 ± 0.0198	0.9768 ± 0.0143	10.8442 ± 1.2788	241.3579 ± 13.6589
+CKKS	0.3	0.0103 ± 0.0019	0.0083 ± 0.0016	0.9416 ± 0.0220	0.9744 ± 0.0121	9.7593 ± 1.4221	233.2313 ± 17.1429
FLoDS	0.0	0.0102 ± 0.0019	0.0083 ± 0.0016	0.9425 ± 0.0222	0.9774 ± 0.0116	7.8253 ± 0.7889	241.3579 ± 13.6589
FLoDS	0.3	0.0103 ± 0.0019	0.0083 ± 0.0016	0.9418 ± 0.0219	0.9760 ± 0.0114	9.9445 ± 0.9907	233.2313 ± 17.1429

accuracy penalty while keeping J high (0.9771 ± 0.0132). Furthermore, the +SecAgg and +CKKS variants exhibit similar utility and fairness, as expected, since confidentiality layers do not change the learning rule. In attack scenarios (ρ=0.3), the baseline variant appears to collapse ($R^2 = -2.6036 \pm 3.7576$, $J = 0.8985 \pm 0.1728$), while the addition of +DP showed to substantially mitigate the impact ($R^2 = 0.8688 \pm 0.1210$, $J = 0.9810 \pm 0.0091$). FLoDS maintains both utility and parity nearly unchanged in both scenarios of ρ.

Concerning the computational cost profiles for each variant, plaintext methods (baseline, +DP) use $\sim$32 MB/round and $\sim$3.2-3.4 s/round, +SecAgg adds modest overhead ($\sim$36-37 MB, $\sim$4.2-5.0 s), while +CKKS and FLoDS incur the expected encrypted-uplink expansion ($\sim$233-241 MB and $\sim$7.8-10.8 s), as downlink is comparable across methods.

V. Conclusion

In this work we presented FLoDS, a hierarchical federated learning framework that combines stewarded secure aggregation with client-level DP and CKKS encryption. On operational ship data within a Shipping DS scenario, FLoDS sustained both accuracy and cross-client parity under non-IID partitions and strong byzantine stress, against state-of-the-art baseline methods. At a representative regime (24 clients, f=0.5, α=0.3), FLoDS maintained $R^2 \approx 0.94$ and Jain's index ≈ 0.97 even at ρ=0.5, whereas a mean-based baseline collapsed in both utility and parity. We also quantified the systems trade-offs relative to plaintext methods, where adopting FLoDS incurred $\sim$ 2-3$\times$ longer round time and $\sim$ 7-8$\times$ higher total per-round traffic, with the increase dominated by encrypted uplink expansion, while downlink remained comparable.

Concerning the limitations and future research directions of this work, there are multiple pragmatic extensions to explore. In detail, our study focuses on gradient sign-flip as the adversary and a compact MLP on a single operational dataset with fixed DP settings. Broader threats and model families may surface additional trade-offs. Future work should explore (i) benchmarking against stronger robust aggregators and additional attacks (stealthy model poisoning, backdoors, sybil), (ii) integrating DS infrastructure end-to-end (e.g., Shipping DS), via IDS-compliant connectors to transmit model parameters to stewards with policy-aware endpoints and signed manifests, while quantifying protocol overheads (i.e., running the FLoDS client inside each provider's connector-protected environment, while steward nodes and the global coordinator act as separate participants that exchange the privacy-preserving model updates), (iii) studying asynchronous/partial participation and client selection under stewards, including their impact on fairness and convergence, (iv) scaling to wider benchmarks and architectures (e.g., temporal CNNs/transformers, multi-task FL), with energy reporting (e.g., CO_2/round) and (v) exploring tighter privacy accounting and adaptive (C, σ) schedules.

Overall, FLoDS enables a practical, deployment-oriented FL pipeline that sustains accuracy, cross-client parity, and confidentiality under non-IID data and Byzantine stress for DSs. To our knowledge, this is the first end-to-end framework in a Data Space setting to incorporate stewards with client-level DP, secure aggregation, and CKKS encryption in a single, auditable stack. The FLoDS design exposes clear robustness-privacy-cost trade-offs with predictable overheads. Addressing the outlined research directions will further strengthen FLoDS and extend its applicability across DSs and industrial deployments.

Acknowledgment

The authors would like to thank Laskaridis Shipping Co., Ltd., for data provision.

References

[1] K. Liu, S. Hu, S. Z. Wu, and V. Smith, "On privacy and personalization in cross-silo federated learning," *Advances in neural information processing systems*, vol. 35, pp. 5925–5940, 2022.

[2] C. Huang, J. Huang, and X. Liu, "Cross-silo federated learning: Challenges and opportunities," *arXiv preprint arXiv:2206.12949*, 2022.

[3] S. Scerri, T. Tuikka, I. L. de Vallejo, and E. Curry, "Common european data spaces: Challenges and opportunities," *Data Spaces: Design, Deployment and Future Directions*, pp. 337–357, 2022.

[4] A. Martella, C. Martella, and A. Longo, "Designing data spaces: Navigating the european initiatives along technical specifications," *arXiv preprint arXiv:2503.15993*, 2025.

[5] G. Enes, "The common european data space (s). a governance model "the european way"," in *International Conference a Digital Europe for Citizens, Data governance, Digital Markets, Digital Services*. Springer Nature Switzerland Cham, 2023, pp. 35–71.

[6] M. Gabellini, L. Civolani, M. Ronchi, L. D. Naldi, and A. Regattieri, "Data spaces in manufacturing and supply chains: A review and insights from european initiatives," *Applied Sciences*, vol. 15, no. 11, p. 5802, 2025.

[7] S. Bader, J. Pullmann, C. Mader, S. Tramp, C. Quix, A. W. Müller, H. Akyürek, M. Böckmann, B. T. Imbusch, J. Lipp *et al.*, "The international data spaces information model–an ontology for sovereign exchange of digital content," in *International Semantic Web Conference*. Springer, 2020, pp. 176–192.

[8] A. Braud, G. Fromentoux, B. Radier, and O. Le Grand, "The road to european digital sovereignty with gaia-x and idsa," *IEEE network*, vol. 35, no. 2, pp. 4–5, 2021.

[9] J. Chandra and S. K. Navneet, "Policy-driven ai in dataspaces: Taxonomy, explainability, and pathways for compliant innovation," *arXiv preprint arXiv:2507.20014*, 2025.

[10] "Towards scalable data space interoperability and federation—discussion paper," Dec. 2024. [Online]. Available: https://coe-dsc.nl/wp-content/uploads/2024/12/Towards-Scalable-Data-Space-Interoperability-and-Federation.pdf

[11] J.-P. Soininen, D. Alonso, T. Guggenberger, L. King, H. Korhonen, K. Kuikkaniemi, G. Laatikainen, M. Nuutinen, D. J. Regeczi, S. Rogotis, and J. Suksi, "Data spaces' synergies," 2024, [Online]. Available: https://dssc.eu/space/DSSE/758350768/Data+Spaces

[12] J. P. S. Piest, W. Datema, D. R. Firdausy, and H. Bastiaansen, "Developing and deploying federated learning models in data spaces: Smart truck parking reference use case," in *International Conference on Enterprise Design, Operations, and Computing*. Springer, 2023, pp. 39–59.

[13] B. Otto, M. t. Hompel, and S. Wrobel, "International data spaces: Reference architecture for the digitization of industries," *Digital transformation*, pp. 109–128, 2019.

[14] M. Bacco, A. Kocian, S. Chessa, A. Crivello, and P. Barsocchi, "What are data spaces? systematic survey and future outlook," *Data in Brief*, vol. 57, p. 110969, 2024.

[15] D. Appelt, P. Kraemer, A. Reiberg, and A. Smolen, "Data trusts, data intermediation services and gaia-x," 2023.

[16] C. Doulkeridis, G. M. Santipantakis, N. Koutroumanis, G. Makridis, V. Koukos, G. S. Theodoropoulos, Y. Theodoridis, D. Kyriazis, P. Kranas, D. Burgos *et al.*, "Mobispaces: An architecture for energy-efficient data spaces for mobility data," in *2023 IEEE International Conference on Big Data (BigData)*. IEEE, 2023, pp. 1487–1494.

[17] D. Sarabia-Jacome, C. E. Palau, M. Esteve, and F. Boronat, "Seaport data space for improving logistic maritime operations," *Ieee Access*, vol. 8, pp. 4372–4382, 2019.

[18] I. Čilić, A. Lackinger, P. A. Frangoudis, I. P. Žarko, A. Furutanpey, I. Murturi, and S. Dustdar, "Reactive orchestration for hierarchical federated learning under a communication cost budget," in *2025 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN)*. IEEE, 2025, pp. 1–7.

[19] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and trends® in machine learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[20] L. Liu, J. Zhang, S. Song, and K. B. Letaief, "Client-edge-cloud hierarchical federated learning," in *ICC 2020-2020 IEEE international conference on communications (ICC)*. IEEE, 2020, pp. 1–6.

[21] R. Gehrer, S. Dumss, F. Gast, W. Wünschel, F. Schwill, M. Šoša, S. Zhou, G. H. Ristow, T. Gharagyozyan, C. Heistracher *et al.*, "Euprogigant: A decentralized federated learning approach based on compute-to-data and gaia-x," *Procedia CIRP*, vol. 128, pp. 710–715, 2024.

[22] M. Bacco, M. Di Leo, A. Kona, M. Santoro, and P. Mazzetti, "Federated learning for data spaces: a privacy-enhancing strategy based on data visiting," in *2024 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor)*. IEEE, 2024, pp. 592–597.

[23] M. Zhang, K. Sapra, S. Fidler, S. Yeung, and J. M. Alvarez, "Personalized federated learning with first order model optimization," *arXiv preprint arXiv:2012.08565*, 2020.

[24] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "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.

[25] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin, "Verifynet: Secure and verifiable federated learning," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 911–926, 2019.

[26] X. Guo, Z. Liu, J. Li, J. Gao, B. Hou, C. Dong, and T. Baker, "V eri fl: Communication-efficient and fast verifiable aggregation for federated learning," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 1736–1751, 2020.

[27] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *International conference on the theory and application of cryptology and information security*. Springer, 2017, pp. 409–437.

[28] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.

[29] I. Mironov, "Rényi differential privacy," in *2017 IEEE 30th computer security foundations symposium (CSF)*. IEEE, 2017, pp. 263–275.

[30] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Advances in neural information processing systems*, vol. 30, 2017.

[31] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *International conference on machine learning*. Pmlr, 2018, pp. 5650–5659.

[32] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečnỳ, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," *arXiv preprint arXiv:2003.00295*, 2020.

[33] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

[34] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[35] B. Yurdem, M. Kuzlu, M. K. Gullu, F. O. Catak, and M. Tabassum, "Federated learning: Overview, strategies, applications, tools and future directions," *Heliyon*, vol. 10, no. 19, 2024.

[36] A. S. Kalafatelis, N. Nomikos, A. Giannopoulos, and P. Trakadas, "A survey on predictive maintenance in the maritime industry using machine and federated learning," *Authorea Preprints*, 2024.

[37] Y.-X. Wang, B. Balle, and S. P. Kasiviswanathan, "Subsampled rényi differential privacy and analytical moments accountant," in *AISTATS 2019*, 2019.

[38] A. S. Kalafatelis, A. Pitsiakou, N. Nomikos, N. Tsoulakos, T. Syriopoulos, and P. Trakadas, "Fluid: Dynamic model-agnostic federated learning with pruning and knowledge distillation for maritime predictive maintenance," *Journal of Marine Science and Engineering*, vol. 13, no. 8, p. 1569, 2025.

[39] M. Cheliotis, I. Lazakis, and G. Theotokatos, "Machine learning and data-driven fault detection for ship systems operations," *Ocean Engineering*, vol. 216, p. 107968, 2020.

[40] Z. Ji, H. Gan, and B. Liu, "A deep learning-based fault warning model for exhaust temperature prediction and fault warning of marine diesel engine," *Journal of Marine Science and Engineering*, vol. 11, no. 8, p. 1509, 2023.

[41] A. S. Kalafatelis, N. Nomikos, A. Giannopoulos, G. Alexandridis, A. Karditsa, and P. Trakadas, "Towards predictive maintenance in the maritime industry: A component-based overview," *Journal of Marine Science and Engineering*, vol. 13, no. 3, p. 425, 2025.

[42] A. S. Kalafatelis, N. Stamou, A. Dailani, T. Theodoridis, N. Nomikos, A. Giannopoulos, N. Tsoulakos, G. Alexandridis, and P. Trakadas, "A lightweight predictive maintenance strategy for marine hfo purification systems," in *European, Mediterranean, and Middle Eastern Conference on Information Systems*. Springer, 2024, pp. 88–99.

[43] B. Liu, H. Gan, D. Chen, and Z. Shu, "Research on fault early warning of marine diesel engine based on cnn-bigru," *Journal of Marine Science and Engineering*, vol. 11, no. 1, p. 56, 2022.

[44] M. P. Uddin, Y. Xiang, M. Hasan, J. Bai, Y. Zhao, and L. Gao, "A systematic literature review of robust federated learning: Issues, solutions, and future research directions," *ACM Computing Surveys*, vol. 57, no. 10, pp. 1–62, 2025.

[45] E. Kritharakis, D. Jakovetic, A. Makris, and K. Tserpes, "Robust federated learning under adversarial attacks via loss-based client clustering," *arXiv preprint arXiv:2508.12672*, 2025.

[46] Y. Zhang and Y. Zhang, "Fedpdc: Federated learning for public dataset correction," *arXiv preprint arXiv:2302.12503*, 2023.

[47] X. Xie, C. Hu, H. Ren, and J. Deng, "A survey on vulnerability of federated learning: A learning algorithm perspective," *Neurocomputing*, vol. 573, p. 127225, 2024.