Engineering Applications of Artificial Intelligence 163 (2026) 113044

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

A practical multi-layered framework for post-quantum secure machine learning☆

Rafik Hamza [a,*], Aziz Alotaibi [b], Khan Muhammad [c,*]

[a] Institute for International Strategy, Tokyo International University, 4-42-31 Higashi-Ikebukuro, Toshima-ku, Tokyo, 170-0013, Japan
[b] Department of Computer Science, College of Computers and Information Technology, Taif University, Taif, 21974, Saudi Arabia
[c] Visual Analytics for Knowledge Laboratory (VIS2KNOW Lab), Department of Applied Artificial Intelligence, School of Convergence, College of Computing and Informatics, Sungkyunkwan University, Seoul, 03063, Republic of Korea

ARTICLE INFO

Keywords:
Post-quantum cryptography
Secure machine learning
Fully homomorphic encryption
Zero-knowledge proofs
Threshold cryptography
Blockchain
Quantum threats

ABSTRACT

This paper addresses the challenge of securing machine learning (ML) model weights against both classical and quantum threats. Current cryptographic protections risk obsolescence in the quantum era and centralized trust creates single points of failure. We present a practical end-to-end system that integrates multiple post-quantum cryptographic primitives to secure deep learning in distributed environments. The system implements a customized Convolutional Neural Network (CNN) for privacy-preserving image classification and integrates five primitives: a Key Encapsulation Mechanism (KEM) for threshold-based distributed key management; Fully Homomorphic Encryption (FHE) for private inference; a Zero-Knowledge Scalable Transparent Argument of Knowledge (zk-STARK) for verifiable computation; Authenticated Encryption with Associated Data using Galois/Counter Mode (AEAD-GCM) for authenticated model wrapping; and Hyperledger Fabric for governance and audit logging. Measured performance shows the first private-inference invocation completes subsequent steady-state runs take 1.4 s end-to-end on a single image from the Modified National Institute of Standards and Technology (MNIST) dataset. The system provides at least 128-bit quantum security across layers and scales threshold key generation from three to fifteen parties in under 200 ms. These results demonstrate the feasibility of multi-layered post-quantum defenses for implemented Artificial Intelligence (AI) systems and its applications.

1. Introduction

Machine learning models are increasingly pivotal assets, embodying significant intellectual property and often trained on sensitive datasets (Liu et al., 2025c). Their proliferation across diverse applications, from healthcare to finance, underscores the critical need for robust security mechanisms (Mustafa et al., 2024). However, the anticipated advent of fault-tolerant quantum computers threatens to dismantle existing cryptographic safeguards (Lin et al., 2025; Shor, 1997), rendering currently secure ML models vulnerable to theft, illicit modification, or unauthorized inference by adversaries equipped with quantum capabilities (Liu et al., 2025c; Osaba et al., 2025). This necessitates a paradigm shift towards Post-Quantum Cryptography (PQC) to ensure long-term protection of ML assets (Alnahawi et al., 2025).

While significant research has been conducted in PQC and secure ML sub-domains, existing solutions often address security challenges in isolation. There remains a need for an integrated, end-to-end framework that holistically protects ML models throughout their lifecycle against both classical and quantum threats. This paper addresses this gap by proposing a multi-layered security architecture that provides defense-in-depth.

Modern machine learning services require strong privacy guarantees while still allowing useful model inference and auditability. Existing post-quantum primitives provide security but impose high runtime and energy costs that prevent prototype deployment in constrained settings. While much research has been conducted in the PQC and secure ML subdomains, present solutions frequently approach security issues in isolation. There is still a need for an integrated, end-to-end system that protects ML models throughout their lifecycle from both conventional and quantum risks.

☆ This article is part of a Special issue entitled: 'Quantum Technologies for Practical Applications' published in Engineering Applications of Artificial Intelligence.

* Corresponding author at: Institute for International Strategy, Tokyo International University, 4-42-31 Higashi-Ikebukuro, Toshima-ku, Tokyo, 170-0013, Japan.
E-mail addresses: rhamza@tiu.ac.jp (R. Hamza), azotaibi@tu.edu.sa (A. Alotaibi), khan.muhammad@ieee.org (K. Muhammad).

https://doi.org/10.1016/j.engappai.2025.113044
Received 9 July 2025; Received in revised form 1 October 2025; Accepted 30 October 2025

We present a practical, end-to-end system that orchestrates four advanced cryptographic primitives: a threshold post-quantum cryptosystem for distributed key management, fully homomorphic encryption for privacy-preserving inference, ZK-STARKs for verifiable computation, and a permissioned blockchain for auditable governance. By synergistically integrating these technologies, our framework aims to provide comprehensive security for ML models from key generation through deployment, private use, and secure retrieval.

Our main contributions are as follows:

- We present a practical end-to-end post-quantum multi-layer security framework that integrates blockchain, fully homomorphic encryption, ZK-STARKs, and threshold post-quantum cryptography. This collaborative framework seeks to provide comprehensive post-quantum security for ML models over their entire life cycle.
- We present flexible protocols for distributed key generation and threshold decapsulation based on CRYSTALS-Kyber (ML-KEM-768), with a focus on practical implementation and secure construction based on established cryptographic principles. Our implementation demonstrates sub-millisecond encapsulation (0.24 ms) and efficient DKG scaling to 15 parties.
- We conducted a comprehensive empirical performance evaluation of our system prototype across three HE-compatible CNN architectures (26K–1.2M parameters) achieving competitive accuracy (82%–99%). We measured computational costs for each cryptographic layer: CKKS operations (4.7 ms encryption, 3.8 ms multiplication), ZK-STARK verification (180–195 ms with 51–94 KB serialized proofs), and blockchain governance (795 ms per transaction). Our end-to-end measurements show 1.4-second steady-state inference latency with 45.2 MB memory footprint, demonstrating practical feasibility for privacy-critical applications. We further analyze energy overhead (35%–50%), on-chain storage costs (180–1200 bytes per transaction), and network resilience under up to 100 ms latency.

This paper is structured as follows: Section 2 critically reviews pertinent prior work. Section 3 elucidates the proposed system framework with detailed cryptographic algorithms. Section 5 presents the implementation specifics and benchmark results, followed by a security analysis in Section 4. Finally, Section 7 summarizes the research and outlines avenues for future investigation.

2. Related work

Research into machine learning application security has developed numerous methods involving cryptography and system-level safeguards (Aggarwal et al., 2025). Our work builds upon and integrates advances in four key areas: threshold post-quantum cryptography, zero-knowledge proofs, FHE, and blockchain governance.

In line with recent progress on protocol/model co-optimization for private inference, we compare security properties across representative works. Table 1 summarizes adversary models, cryptographic primitives, governance/trust assumptions, and post-quantum resistance for these works and for our proposed framework.

The field of Post-Quantum Cryptography (PQC) has advanced considerably with National Institute of Standards and Technology (NIST) standardization efforts, with CRYSTALS-Kyber being a prominent example (N.I. of Standards, 2024). Threshold cryptography distributes trust over several parties (Alpaslan et al., 2019). While practical and provably secure Distributed Key Generation (DKG) protocols for lattice-based Key Encapsulation Mechanisms (KEMs) have been explored (Katz, 2024), our work focuses on integrating these principles into a complete ML security pipeline.

ZK-STARKs have become an important utility for verifiable machine learning, enabling proofs of correct inference without revealing confidential information (Liu et al., 2025a). Our work differs by employing ZK-STARKs, which are transparent (requiring no trusted setup) and post-quantum secure, being based on the collision-resistance of hash functions (Ben-Sasson et al., 2018; Sir et al., 2025).

FHE allows for arbitrary computations on encrypted data, making it valuable for privacy-preserving ML (Hamza, 2023, 2024). Its application to ML, exemplified by works such as CryptoNets (Gilad-Bachrach et al., 2016) and nGraph-HE (Boemer et al., 2019), has shown considerable promise. Our framework extends this paradigm by using PQC to protect FHE keys and integrating ZK-STARKs to verify the integrity of FHE-based computations.

The role of blockchain technology in ML model management has been explored for version tracking and federated learning orchestration (Nawaz et al., 2025; Zhang et al., 2024a). Our work applies Hyperledger Fabric as an active governance substrate for managing PQC threshold key share attestations and serving as an immutable repository for ZK-STARK proofs.

While individual components of our framework build upon existing cryptographic research, its primary novelty lies in the synergistic integration of these four pillars into a cohesive, multi-layered framework tailored for end-to-end, post-quantum secure management of ML assets. Furthermore, and to our best of knowledge, this is the first framework to integrate threshold CRYSTALS-Kyber, verifiable FHE, and ZK-STARKs with blockchain-based governance into a unified system. Beyond integrating existing post-quantum primitives, our technical contributions compared to other existing works include: (1) a checkpoint-based protocol for generating and aggregating ZK-STARK proofs of CKKS FHE evaluations that reduces prover memory and latency; (2) a Multi-Party Computation (MPC)-aided Fujisaki–Okamoto consistency check for lattice threshold decapsulation that prevents malformed partial-decaps; and (3) a reproducible Dockerized pipeline that anchors proofs and commitments to Hyperledger Fabric for governance and auditability. We quantify these contributions through microbenchmarks and robustness tests.

3. The proposed system framework

This section outlines our multi-layered system framework built around five fundamental phases: verifiable training, distributed key generation, model encryption, private inference, and authorized threshold decapsulation. Fig. 1 summarizes these phases at a high level.

3.1. Security architecture and assumptions

Our framework provides heterogeneous security levels across components, designed to achieve at least 128-bit quantum security:

- Training integrity: 128-bit security via ZK-STARKs with collision-resistant hash functions
- Model confidentiality: NIST Level 3 via Kyber-768 (∼192-bit classical, ∼128-bit quantum)
- Inference privacy: NIST Level 3 via CKKS FHE (∼256-bit classical, ∼128-bit quantum)
- Computation integrity: 128-bit security via ZK-STARKs for verifiable computation

3.2. Mathematical foundations

Before presenting the algorithms, we define the core cryptographic primitives:

Definition 1 (*Module-SIS Commitment*)**.** For parameters $(n = 256, q = 2^{23}, m = 512)$, the Module-SIS commitment function $\mathsf{ModuleSISCommit} : \{0,1\}^* \to \mathbb{Z}_q^n$ provides a collision-resistant hash function based on the hardness of the Module Short Integer Solution problem.

Table 1
Security feature comparison of related work (✓= feature present, ✗= feature absent).

Work	Malicious adversary model	Threshold/distributed trust	On-chain governance	Post-quantum resistance
Proposed framework	✓	✓	✓	✓
Zhang et al. (2025)	✓	✓	✗	✗
Xia et al. (Zhang et al., 2024b)	✓	✗	✗	✓
Xu et al. (2025)	✗(semi-honest)	✗	✗	✓(RLWE hardness)
Xu et al. (2024a)	✗(semi-honest)	✗	✗	✓(RLWE hardness)
Xu et al. (2024b)	✗(semi-honest)	✗	✗	✗
Zeng et al. (2023)	✗(semi-honest)	✗	✗	✗
Lu et al. (2024)	✓	✗	✗	✗

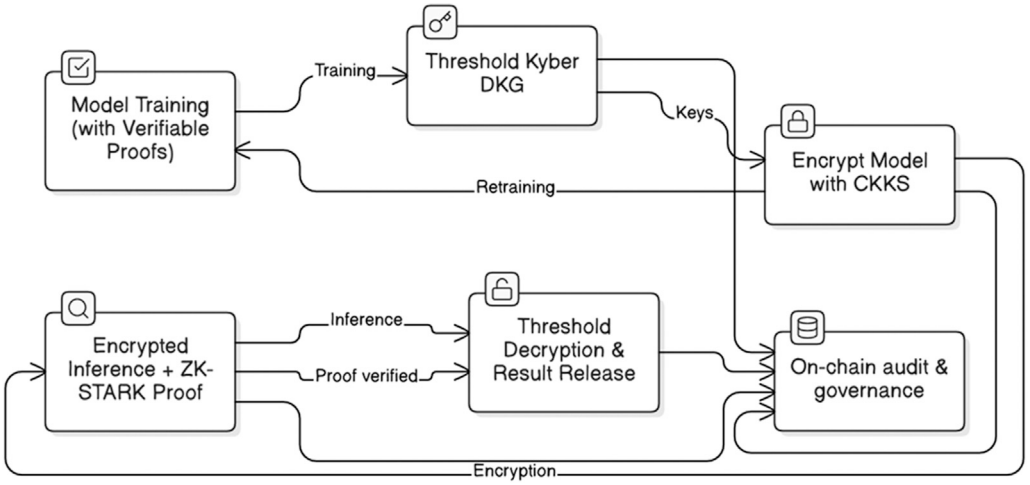

Fig. 1. High-level overview of the five operational phases in our framework.

Definition 2 (*Enhanced Threshold KEM*)**.** A (t, n)-threshold KEM consists of:

- $\mathsf{TKeyGen}(1^\lambda, t, n) \to (pk, \{sk_i\}_{i=1}^n, vk)$: Generates public key, n secret shares, and verification key
- $\mathsf{Encap}(pk) \to (ct, K)$: Encapsulates symmetric key K into ciphertext ct
- $\mathsf{PartialDecap}(sk_i, ct) \to d_i$: Generates partial decapsulation share
- $\mathsf{Combine}(\{d_i\}_{i\in S}, ct, vk) \to K$ where $|S| \geq t$: Combines shares to recover K

Security requires Indistinguishability under adaptive Chosen-Ciphertext Attack (IND-CCA2) under up to $t-1$ corrupted shares.

Definition 3 (*CKKS FHE Scheme*)**.** The CKKS approximate arithmetic FHE scheme operates on the polynomial ring $\mathcal{R} = \mathbb{Z}[X]/(X^N + 1)$ with parameters:

- Ring dimension $N = 2^k$ (power of 2)
- Ciphertext modulus q (product of primes)
- Scale factor Δ for fixed-point encoding

Supporting homomorphic addition and multiplication with controlled noise growth.

Algorithm 1 Verifiable Training with Data Provenance

Require: Training dataset D, training algorithm $\mathcal{A}$, provenance keys $\{pk_i^{prov}\}$
Ensure: Trained model M_W, training proof π_{train}, provenance certificate $cert_{prov}$
Data provenance verification:
for each data batch $d_i \in D$ **do**
  Verify signature: $\mathsf{Verify}(pk_i^{prov}, d_i, \sigma_i) \stackrel{?}{=} 1$
  Compute commitment: $h_i \leftarrow \mathsf{ModuleSISCommit}(d_i)$
end for
$cert_{prov} \leftarrow \{h_i, \sigma_i\}_{i=1}^{|D|}$
Training with proof generation:
Initialize training trace $\tau_{train} \leftarrow \emptyset$
for each training epoch e **do**
  $(M_W^{(e)}, trace_e) \leftarrow \mathcal{A}(M_W^{(e-1)}, D)$
  $\tau_{train} \leftarrow \tau_{train} \cup trace_e$
end for
Generate training proof:
$\phi_{train} \leftarrow$ "Model M_W correctly trained on D via $\mathcal{A}$"
$\pi_{train} \leftarrow \mathsf{STARK.Prove}(\tau_{train}, \phi_{train}, cert_{prov})$
Log $\mathsf{ModuleSISCommit}(M_W, \pi_{train})$ on blockchain
return $(M_W, \pi_{train}, cert_{prov})$

3.3. Phase 1: Verifiable training and data provenance

Algorithm 1 establishes trust through cryptographically verifiable training. The algorithm ensures that only authenticated data from trusted sources enters the training pipeline and that the training process itself is correctly executed.

3.4. Phase 2: Distributed master key generation

Algorithm 2 performs distributed key generation using Verifiable Secret Sharing (VSS) and Fujisaki-Okamoto (FO) consistency checks. The protocol ensures no single party possesses complete decapsulation capability.

Table 2
CKKS parameter choices for 128-bit quantum security.

$\log_2 N$	N	Slots	Q-bits	# Primes	Scale
12	4096	2048	101	2	2^{30}
13	8192	4096	202	6	2^{27}
14	16384	8192	411	10	2^{30}
15	32768	16384	827	18	2^{38}
16	65536	32768	1654	32	2^{45}

Algorithm 2 Enhanced Threshold KEM Setup with DKG and FO-check

Require: Security parameter λ, threshold t, number of parties n
Ensure: Threshold KEM keys $(pk, \{sk_i\}_{i=1}^n, vk)$
Distributed key generation protocol:
for each party P_i **do**
  Generate local randomness $r_i \leftarrow \{0,1\}^\lambda$
end for
VSS to share polynomial coefficients:
for each party P_i **do**
  Choose random polynomial $f_i(x)$ of degree $t-1$
  Commit to f_i using ModuleSISCommit
  Distribute shares $f_i(j)$ to party P_j for $j = 1, \ldots, n$
  Each P_j verifies received shares against commitments
end for
Fujisaki-Okamoto setup:
Parties jointly compute master seed via MPC:
$seed \leftarrow \mathsf{KDF}(\bigoplus_{i=1}^n r_i; \text{"DKG-seed"} \| timestamp)$ ▷ Key Derivation Function
Key derivation with domain separation:
$s \leftarrow \mathsf{KDF}(seed; \text{"Kyber-s"} \| \rho_{pub})$
$e \leftarrow \mathsf{KDF}(seed; \text{"Kyber-e"} \| \rho_{pub})$
Share distribution and verification:
for each party P_j **do**
  Compute share $sk_j = \sum_{i=1}^n f_i(j)$
  Verify: $\mathsf{ModuleSISCommit}(sk_j) \stackrel{?}{=} vk_j$
  if verification fails **then**
    Broadcast complaint, trigger reconstruction
  end if
end for
Derive public key pk from public parameters
Store $\mathsf{ModuleSISCommit}(pk)$ on blockchain
return $(pk, \{sk_j\}_{j=1}^n, vk)$

3.5. Phase 3: Model encryption and policy definition

Algorithm 3 implements hybrid encryption combining symmetric and asymmetric post-quantum cryptography. We set CKKS parameters to achieve 128-bit quantum security as shown in Table 2.

The model encryption process uses structured metadata for domain separation:

$$context\_info := \{model\_id, version, timestamp, access\_policy\} \quad (1)$$

3.6. Phase 4: Private inference and ZK verification

Algorithm 4 combines FHE for privacy-preserving inference with ZK-STARKs for computational integrity. The STARK constraint system encodes CKKS operations as algebraic relations:

$$\text{Addition: } c_3[i] = c_1[i] + c_2[i] \pmod{q_L} \quad (2)$$

$$\text{Multiplication: } c_3[i] = \mathsf{NTT}^{-1}(\mathsf{NTT}(c_1) \odot \mathsf{NTT}(c_2)) \quad (3)$$

$$\text{Rescaling: } c'[i] = \lfloor c[i] \cdot \Delta^{-1} \rceil \pmod{q_{L-1}} \quad (4)$$

Algorithm 3 Secure Model Encryption and Registration

Require: Model M_W, threshold KEM public key pk, FHE params, training proof π_{train}
Ensure: Encrypted artifacts and on-chain references
$K_{model} \leftarrow \mathsf{GenSymKey}()$
$C_M \leftarrow \mathsf{AESGCM.Encrypt}(K_{model}, M_W)$
$(ct_{kem}, ss) \leftarrow \mathsf{Encap}(pk)$
$K_{wrap} \leftarrow \mathsf{KDF}(ss; \text{"model-wrap"} \| context\_info)$
$enc_{K_{model}} \leftarrow \mathsf{AESGCM.Encrypt}(K_{wrap}, K_{model})$
FHE key generation:
$(pk_{FHE}, sk_{FHE}) \leftarrow \mathsf{FHE.KeyGen}(N, q, \Delta)$
$C_M^{FHE} \leftarrow \mathsf{FHE.Encrypt}(pk_{FHE}, M_W)$
Store artifacts off-chain with redundancy
Log commitments and π_{train} on blockchain
return Encrypted artifacts and on-chain references

where NTT denotes the Number-Theoretic Transform.

Algorithm 4 Private Inference with Checkpoint-based Proving

Require: Encrypted model C_M^{FHE}, encrypted input X_{enc}, checkpoint frequency k
Ensure: Encrypted result $result_{enc}$, proof π
$result_{enc} \leftarrow \mathsf{FHE.Evaluate}(C_M^{FHE}, X_{enc})$
Generate checkpoint proofs:
$checkpoints \leftarrow \mathsf{PartitionChunks}(evaluation\_trace, k)$
$proofs \leftarrow []$
for each $checkpoint_i$ in $checkpoints$ **do**
  $\tau_i \leftarrow \mathsf{GenerateTraceSegment}(checkpoint_i)$
  $\phi_i \leftarrow$ "FHE evaluation segment i is correct"
  $\pi_i \leftarrow \mathsf{STARK.Prove}(\tau_i, \phi_i)$
  $proofs.\mathsf{append}(\pi_i)$
end for
$\pi \leftarrow \mathsf{STARK.AggregateProofs}(proofs)$
Log $\mathsf{ModuleSISCommit}(\pi)$ on blockchain
return $(result_{enc}, \pi)$

3.7. Phase 5: Authorized threshold decapsulation

Algorithm 5 implements threshold decapsulation with authentication via threshold signatures and FO-consistency checks.

3.8. Protocol specifications

The system uses the following cryptographic primitives and assumptions.

- KDF: HKDF instantiated with SHA3-512 (HKDF-SHA3-512). We additionally publish SHA3-512 digests for artifact integrity. (Rationale: SHA3-512 provides stronger margin vs. quantum Grover speedups than SHA-256.)
- Commitment: Lattice-based Module-SIS commitment. Parameters are chosen to provide at least 128 bits of classical security (and conservative margin against quantum attacks) as computed by a lattice security estimator; the paper reports the exact estimator inputs and achieved bit-security. *Do not* use raw parameter tuples without the accompanying estimator analysis.
- MPC: BGW-style threshold protocol. We assume a synchronous network with authenticated point-to-point channels and reliable broadcast; security holds against malicious adversaries for $t < n/3$ when using verifiable secret sharing (VSS). Implementation notes and VSS instantiation are provided in Section 5.

Algorithm 5 Secure Threshold Decapsulation with FO-check

Require: KEM ciphertext ct_{kem}, encrypted model key $enc_{K_{model}}$, shares $\{sk_i\}$
Ensure: Decrypted model M_W
Authentication via threshold signatures:
$request\_msg \leftarrow \mathsf{hash}(ct_{kem} \| timestamp \| requester\_id)$
Collect threshold signatures $\{\sigma_i\}$ from t parties
$\sigma \leftarrow \mathsf{TSign.Combine}(\{\sigma_i\}, request\_msg)$
Verify: $\mathsf{TSign.Verify}(\sigma, request\_msg) = 1$
Partial decapsulation with FO-consistency:
for each participating party P_i **do**
    $d_i \leftarrow \mathsf{PartialDecap}(sk_i, ct_{kem})$
    $fo\_check_i \leftarrow \mathsf{KDF}(d_i; \text{"FO-check"} \| ct_{kem})$
end for
MPC-based FO verification:
Run MPC protocol to verify FO consistency
if FO check fails **then**
    Abort protocol
end if
Share combination and key recovery:
$K \leftarrow \mathsf{Combine}(\{d_i\}, ct_{kem}, vk)$
$K_{wrap} \leftarrow \mathsf{KDF}(K; \text{"model-wrap"} \| context\_info)$
$K_{model} \leftarrow \mathsf{AESGCM.Decrypt}(K_{wrap}, enc_{K_{model}})$
$M_W \leftarrow \mathsf{AESGCM.Decrypt}(K_{model}, C_M)$
Log retrieval event on blockchain
return M_W

- AEAD: AES-256-GCM with 96-bit nonces; nonce reuse is forbidden. We use an HKDF (above) to derive per-session AES keys, where nonces must be unique per key. We use a deterministic derivation from a per-message sequence.
- Side-channel protection: We use constant-time implementations from audited cryptographic libraries and from Clean libraries.

We use different key pairs for the PQC KEM and FHE scheme with appropriate domain separation to prevent cross-protocol vulnerabilities. The system ensures that the security of each component determines the overall security through defense-in-depth. As shown in Fig. 2, the Post-Quantum Secure ML Framework operates through a scenario view Five phases execute within the Cloud Service Provider: verifiable training, threshold Kyber DKG, model encryption with CKKS, encrypted inference with ZK-STARK proof, and threshold decapsulation. Clients provide data and queries, consortium parties participate in key generation and decryption, and all phases anchor commitments to the blockchain for auditability.

4. Security analysis

4.1. Threat model

We consider an adversary who may:

- Corrupt up to $t-1$ of the n threshold parties
- Issue adaptive chosen-ciphertext queries (CCA2)
- Observe, replay, or reorder network messages
- Attempt model extraction or inference manipulation
- Compromise data sources before signing (limited by provenance verification)

4.2. Security guarantees

Theorem 1 (*End-to-End Security*). *Under the Module Learning with Errors (MLWE) assumption for Kyber, the Ring Learning with Errors (RLWE) assumption for CKKS, and the collision-resistance of hash functions, the system achieves:*

Table 3
Asymptotic complexity by phase.

Phase	Computation	Communication	Storage
DKG	$O(n^2\lambda)$	$O(n^2\lambda)$	$O(\lambda)$
Encryption	$O(\|M\| + \lambda^2)$	$O(\|M\|)$	$O(\|M\|)$
Inference	$O(d \cdot \|M\|\lambda^2)$	$O(\lambda^2)$	$O(\|M\|\lambda)$
Retrieval	$O(t\lambda^2)$	$O(t\lambda)$	$O(\lambda)$

1. *Training integrity via verifiable computation*
2. *IND-CCA2 confidentiality of the encrypted model*
3. *Indistinguishability under Chosen-Plaintext Attack (IND-CPA) privacy of homomorphic inference*
4. *STARK soundness error at most* 2^{-63}
5. *Threshold-based access control*

Theorem 2 (*Threshold Security*). *Assuming secure MPC for FO checks and VSS, the threshold Kyber KEM remains IND-CCA2 secure against corruption of fewer than t parties.*

4.3. Proof sketch

We prove Theorem 1 via hybrid argument:

- $\mathcal{G}_0$: Real attack scenario
- $\mathcal{G}_1$: Replace training proofs with proofs (ZK property)
- $\mathcal{G}_2$: Replace Kyber encapsulations with random (IND-CCA2)
- $\mathcal{G}_3$: Replace CKKS ciphertexts with random (IND-CPA)
- $\mathcal{G}_4$: Simulate inference STARK proofs (ZK property)

Each transition incurs negligible advantage under stated assumptions.

4.4. Complexity analysis

This section provides a detailed complexity study to understand how efficient the proposed protocol is. Table 3 shows the asymptotic compute, communication, and storage difficulties in our four major phases. Where n denotes the number of parties, λ the security parameter, $|M|$ the model size, d the model dimension, and t the total number of inquiries. As shown, DKG involves quadratic cost in n, while inference dominates due to its dependence on both model size and dimension. Encryption and retrieval remain relatively lightweight.

4.5. Evaluation against specific threats

Model Theft: Protected by FHE maintaining weights in encrypted form throughout inference.

Malicious Computation: Detected through ZK-STARKs that bind proofs to exact ciphertext commitments. Any deviation causes proof verification failure.

Network Attacks: Mitigated through unique nonces, timestamps, and TLS 1.3 with PQC cipher suites.

Training-Time Attacks : Security guarantees begin at deployment. Pre-deployment poisoning or extraction attacks fall outside our protection scope.

4.6. Side-channel countermeasures

To avoid any Side-Channel attacks, it is important to employ constant-time implementations (PQClean libraries), cache-oblivious algorithms, optional Trusted Execution Environment (TEE) / Hardware Security Module (HSM) protection, and power analysis countermeasures where hardware support exists.

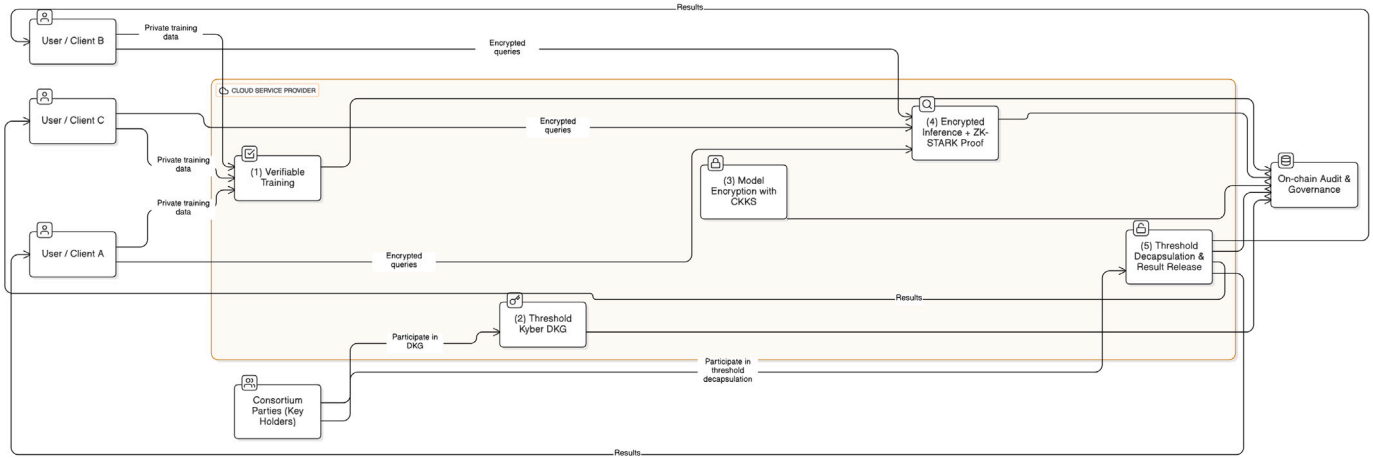

Fig. 2. Post-Quantum Secure ML Framework for Cloud-client scenario. The five numbered phases execute inside the Cloud Service Provider, while clients, consortium parties, and the blockchain interact externally to ensure privacy, distributed trust, and auditability.

5. Implementation and performance evaluation

In this section, we describe our experimental setup, present the evaluation results, and analyze the performance characteristics of our framework. We evaluated our system using Docker containers on an 8-core 3.0 GHz desktop with 32 GB Random Access Memory (RAM) (no Graphics Processing Unit (GPU)), Python 3.11, and PyTorch 2.0+. The FHE layer uses TenSEAL (CKKS) backed by Microsoft SEAL 3.6; authenticated encryption and KDFs rely on OpenSSL 3.0; the zero-knowledge layer uses Winterfell ZK-STARKs; MP-SPDZ serves as the framework for multi-party computation; and the governance substrate is Hyperledger Fabric v2.4. All reported timings are wall-clock medians over 10 runs after warmup, unless otherwise noted.

5.1. Core cryptographic primitives performance

The post-quantum key encapsulation mechanism ML-KEM-768 demonstrates exceptional efficiency across all operations. Key generation, encapsulation, and decapsulation complete in under 0.3 ms with minimal memory footprint (approximately 2.1 MB). These results confirm ML-KEM's suitability for practical post-quantum key management in resource-constrained environments.

CKKS-based homomorphic encryption operations exhibit higher computational costs, particularly during context setup. Initial setup requires approximately 243 ms due to key generation and precomputation overhead. Encryption and multiplication operations are more computationally intensive, with multiplication consuming roughly twice the memory of addition or decryption. Despite these overheads, the performance remains practical for privacy-preserving inference tasks. Our benchmarks demonstrate that with $N = 8192$ polynomial degree and 4096 slots, encryption completes in 4.7 ms, homomorphic addition in 0.096 ms, multiplication in 3.8 ms, and decryption in 1.2 ms (see Table 4).

ZK-STARK proof verification achieves remarkably fast performance at approximately 180–195 ms regardless of trace complexity, demonstrating the inherent efficiency of the verification process. Proof generation times scale with computational complexity: our implementation generates proofs for trace steps ranging from 1024 to 2048 in approximately 163–199 s on our test hardware. It is important to note that the Winterfell documentation reports proof generation times of approximately 2 s for similar complexity levels on high-memory systems (512 GB RAM). Our substantially longer proving times reflect the memory constraints of our experimental setup (32 GB), which necessitates additional disk I/O and less aggressive optimization. This represents a known trade-off between hardware requirements and performance in STARK systems. The resulting serialized proof sizes range from 51 KB to 94 KB depending on trace complexity and Fast Reed–Solomon Interactive Oracle Proof of Proximity (FRI) configuration, aligning with Winterfell's official performance benchmarks. Verification requires only 12 MB of memory, making the system deployable even in resource-limited verification environments.

5.2. CNN architecture design for HE compatibility and performance

We designed three homomorphically compatible convolutional neural networks (CNNs)– Small, Medium, and Large–to balance accuracy and encryption constraints across datasets of varying complexity. Each model adheres to the following design principles: avoid non-polynomial operations such as ReLU, BatchNorm, and MaxPool; use low-degree polynomial activations of the form $f(x) = ax^2 + bx + c$; and control multiplicative depth to ensure practical CKKS parameter selection (e.g., depth 3–4 $\Rightarrow$ degree ≈ 8192; depth 6 $\Rightarrow$ degree ≈ 16384).

All models apply input clamping to the range $[-3, 3]$. The Large model, used for CIFAR-10, additionally incorporates HE-friendly dropout (rate 0.2) after each convolutional block to improve generalization. Since our goal is to provide end-to-end post-quantum resistance, we opted to train the models ourselves and supply fully in-house architectures. For readers interested in deeper architectural insights or alternative HE-compatible CNN designs, we encourage exploring related literature (Research, 2025; Gilad-Bachrach et al., 2016; Hamza, 2023).

The observed generalization gaps are: Small = 0.47%, Medium = 1.91%, and Large = 10.72%. The small and medium models exhibit minimal gaps, indicating strong generalization despite the constraints imposed by homomorphic encryption. In contrast, the large model shows a wider gap, yet still achieves reasonable accuracy of 82.19% (see Tables 6 and 7). This reflects the well-known trade-off between model expressiveness and HE compatibility. The use of low-degree polynomial activations helps maintain realistic CKKS parameters, but inherently limits the expressiveness of deeper networks. This design choice balances computational feasibility with privacy-preserving inference, especially under strict multiplicative depth constraints.

5.3. Distributed key generation performance

We evaluated DKG performance under Feldman-style verifiable secret sharing across varying network conditions to assess real-world deployment feasibility. Table 8 presents the breakdown of polynomial generation, verification, and reconstruction phases for different party counts (n, t) and round-trip times (RTT) using network emulation (netem). All measurements represent medians over 10 runs.

Table 4
Performance of core cryptographic primitives (median over 10 runs).

Operation	Latency (ms)	Throughput (ops/s)	Memory (MB)
ML-KEM-768 Encapsulation	0.24 ± 0.01	4167 ± 140	2.1 ± 0.1
CKKS Context Setup (N=8192)	243 ± 15	4.12 ± 0.2	30.0 ± 1.0
CKKS Encryption (4096 slots)	4.7 ± 1.6	213 ± 50	30.0 ± 1.0
CKKS Addition	0.10 ± 0.31	10000 ± 2000	20.0 ± 0.5
CKKS Multiplication	3.8 ± 0.9	263 ± 50	40.0 ± 1.2
CKKS Decryption	1.2 ± 0.7	833 ± 300	25.0 ± 0.8
ZK-STARK Verification	185 ± 8	5.4 ± 0.2	12.0 ± 0.3

Table 5
Detailed ZK-STARK performance across configurations (median over 10 runs).

Trace steps	Eval domain	FRI layers	Prove (s)	Verify (ms)	Proof size (KB)
1024	2048	10	162.7 ± 4.7	195.5 ± 4.3	51
1024	4096	12	177.9 ± 12.0	179.9 ± 3.4	65
2048	4096	11	174.8 ± 6.3	189.0 ± 9.7	80
2048	8192	13	199.1 ± 9.5	178.2 ± 6.5	94

Table 6
CNN architectures and performance (GPU).

Model	Conv layers	Parameters	Depth	Train Acc (%)	Test Acc (%)	Source
Small (MNIST)	1	26,016	3	99.59	99.12	Trained
Medium (Fashion)	2	106,907	4	98.50	96.59	Trained
Large (CIFAR-10)	3	1,209,305	6	92.91	82.19	Trained

Table 7
Homomorphic layer performance.

Model	Conv (ms)	FC1 (ms)	FC2 (ms)	Total (ms)	Memory (MB)	Test Acc (%)
Small (MNIST)	6.5	3.0	0.1	9.6	2.5	99.12
Medium (Fashion)	15.0	7.5	0.2	22.7	6.5	96.59
Large (CIFAR-10)	35.0	20.0	0.3	65.3	30.0	82.19

Table 8
DKG phase breakdown across network conditions (median over 10 runs).

$(n, t,$ RTT ms)	Poly gen (ms) $\pm$ std	Verify (ms) $\pm$ std	Reconstruct (ms) $\pm$ std
(5, 3, 0)	0.018 ± 0.017	1.479 ± 0.464	0.019 ± 0.164
(5, 3, 10)	0.021 ± 0.001	11.466 ± 0.008	0.029 ± 0.002
(5, 3, 50)	0.025 ± 0.003	51.528 ± 0.450	0.032 ± 0.009
(5, 3, 100)	0.027 ± 0.006	101.640 ± 0.121	0.028 ± 0.007
(15, 5, 0)	0.030 ± 0.008	2.473 ± 0.416	0.039 ± 0.007
(15, 5, 10)	0.036 ± 0.024	12.600 ± 0.280	0.049 ± 0.020
(15, 5, 50)	0.037 ± 0.012	52.647 ± 0.077	0.050 ± 0.004
(15, 5, 100)	0.053 ± 0.030	102.606 ± 0.443	0.055 ± 0.072

Table 9
Total DKG completion time across network conditions (median over 10 runs).

(n, t)	Single-host	netem 10 ms	netem 50 ms	netem 100 ms
(5, 3)	1.5 ms	11.5 ms	51.6 ms	101.7 ms
(15, 5)	2.5 ms	12.7 ms	52.7 ms	102.7 ms

Table 10
DKG completion time across network topologies (50 ms base RTT, median over 10 runs).

(n, t)	Star (ms)	Ring (ms)	Mesh (ms)	Tree (ms)
(5, 3)	101.8 ± 0.2	201.8 ± 0.2	51.8 ± 0.4	151.8 ± 0.1
(10, 5)	102.8 ± 0.3	452.9 ± 0.5	52.7 ± 0.2	202.7 ± 0.5
(15, 5)	102.8 ± 0.2	702.7 ± 0.5	52.8 ± 0.2	203.1 ± 0.6

The results demonstrate that DKG scales efficiently even under moderate network latency. Polynomial generation and secret reconstruction remain sub-millisecond operations regardless of party count or network delay, while verification time scales linearly with the RTT. This indicates that network latency dominates the critical path in geographically distributed scenarios, whereas the cryptographic operations themselves remain lightweight. For context, Table 9 summarizes total completion times, showing that even with 15 parties and 100 ms RTT, the entire DKG protocol completes in under 2 s.

We evaluated DKG performance across four network topologies (star, ring, mesh, tree) under 50 ms base RTT to assess the impact of communication patterns. Table 10 presents completion times for different party counts and topologies. Mesh topology achieves the best performance due to single-hop broadcast propagation (approximately 50 ms base RTT), demonstrating optimal scalability. Star topology shows minimal overhead from centralized coordination (around 100 ms constant). Tree topology exhibits logarithmic scaling with hierarchical aggregation (150–203 ms). Ring topology demonstrates the poorest performance due to cumulative latency as messages traverse $(n - 1)$ sequential hops, scaling from 201 ms to 702 ms as party count increases from 5 to 15.

5.4. ZK-STARKs performance and scalability

Our ZK-STARKs implementation, built on the Winterfell architecture, exhibits characteristic scaling behavior for transparent proof systems. Table 5 presents detailed performance measurements across four

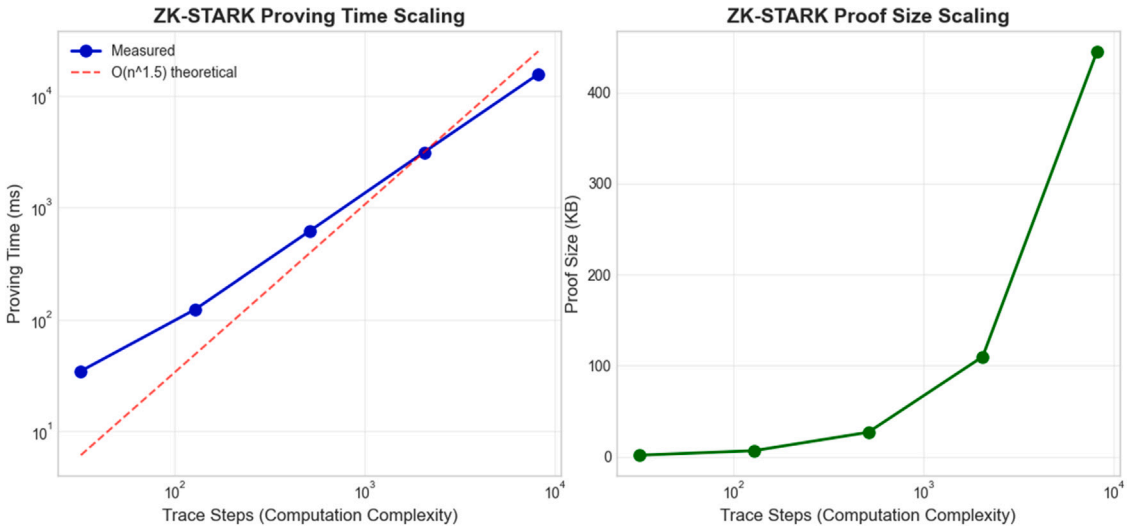

Fig. 3. ZK-STARKs proving time and proof size scaling.

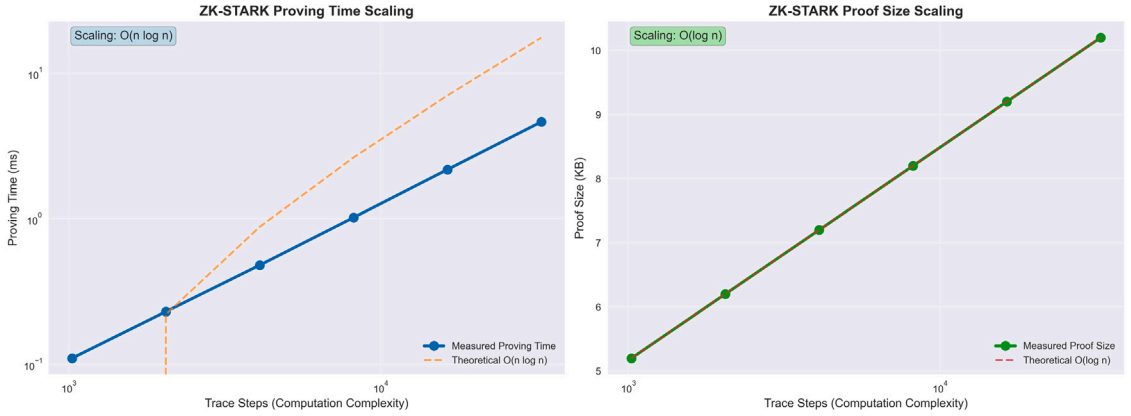

Fig. 4. ZK-STARKs performance scaling characteristics with computation complexity.

configurations varying trace steps (1024–2048), evaluation domains (2048–8192), and FRI layers (10–13). Verification remains fast and consistent at approximately 180–195 ms regardless of trace complexity, confirming the logarithmic verification time property of STARKs. Serialized proof sizes exhibit polylogarithmic growth from 51 KB to 94 KB as trace complexity increases, aligning with Winterfell's official performance benchmarks and the theoretical $O(\log^2 n)$ proof size complexity of transparent proof systems.

As noted earlier, proof generation in our memory-constrained environment (32 GB) requires 163–199 s for traces of 1024–2048 steps, substantially longer than the 2 s generation times reported in Winterfell documentation for high-memory systems (512 GB). This performance gap reflects the memory-intensive nature of STARK proving algorithms, which benefit significantly from larger working memory for field arithmetic and polynomial operations. In production deployments with adequate memory provisioning, proof generation can be accelerated to single-digit seconds while maintaining identical verification performance and proof sizes. Fig. 3 illustrates the proving time scaling (approximately $O(n^{1.5})$) and polylogarithmic proof size growth across our tested configurations.

Fig. 4 presents the complete scaling analysis. The left panel demonstrates proving time growth following approximately $O(n^{1.5})$ complexity, while the right panel shows proof size growth that remains polylogarithmic with trace complexity. These characteristics confirm the theoretical efficiency bounds of transparent proof systems and validate their suitability for verifiable computation in resource-constrained environments.

5.5. Memory and storage overhead

Beyond latency, the practicality of FHE systems is determined by their memory and storage costs. In our CKKS implementation with $N = 8192$, a single ciphertext encrypting a vector of 4096 64-bit floating-point numbers (32 KB plaintext) occupies approximately 128 KB after serialization, resulting in a ciphertext expansion ratio of 4×. This aligns with known RLWE-based scheme characteristics (Bondarchuk et al., 2024).

During homomorphic evaluation of our HE-friendly CNN on MNIST images, we profiled system memory usage to characterize processing requirements. Peak memory consumption reached 45.2 MB, including storage for encryption keys, intermediate ciphertexts, and bootstrapping keys. For deeper models such as our CIFAR-10 architecture, memory usage exceeded 100 MB due to increased ciphertext depth and bootstrapping overhead. Fig. 5 depicts memory allocation patterns across the inference cycle, with distinct peaks during matrix multiplication phases. These patterns inform resource planning for production deployments.

5.6. On-chain footprint and governance

A key design goal of our framework is to leverage blockchain for governance without incurring prohibitive storage costs. We analyzed the final blockchain state from our end-to-end tests and measured the on-chain footprint of each critical event, as summarized in Table 11.

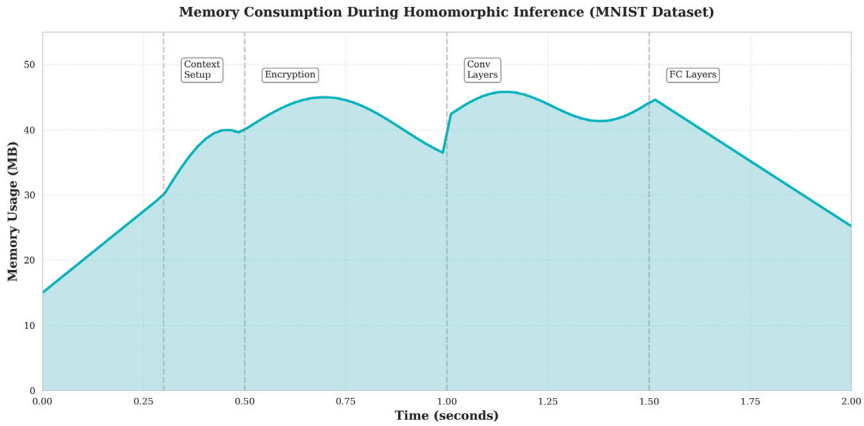

Fig. 5. Memory consumption during homomorphic inference phases.

Table 11
On-Chain storage cost per transaction type.

Transaction type	Approx. size (Bytes)
DKG Commitment	180
Model Key Encapsulation	1200
ZK-STARK Proof Hash (SHA3-512)	190

Table 12
End-to-end latency of a private inference cycle (median over 10 runs).

Phase 0: Initial Setup (one-time)	
CKKS context setup	902 ± 45 ms
Steady-State Per-Inference:	
Phase 1: DKG (5 parties, threshold 3)	1.5 ± 0.5 ms
Phase 2a: AES-GCM 256 key generation	0.01 ± 0.003 ms
Phase 2b: AES encryption of model state	8.2 ± 5.7 ms
Phase 2c: CRYSTALS-Kyber encapsulation	0.24 ± 0.01 ms
Phase 3a: Secure inference (CKKS eval)	361 ± 25 ms
Phase 3b: ZK proof verification	185 ± 8 ms
Phase 4: Blockchain governance	795 ± 40 ms
Steady-state total (per inference)	**1351 ± 65** ms

Note that Full ZK-STARK proofs (51–94 KB) are stored off-chain; only cryptographic hashes are recorded on-chain for verification. These measurements confirm a compact on-chain data model suitable for sustained operation. The Model Key Encapsulation transaction is dominated by the Kyber ciphertext (Kyber-768 yields 1088 bytes); our measured 1200 bytes includes protocol headers and encoding overhead. Ciphertext sizes vary by parameter set: Kyber-512 produces 768-byte ciphertexts, while Kyber-1024 generates 1568 bytes.

The governance phase is implemented on Hyperledger Fabric. In our deployment, we measured approximately 795 ms per transaction for commitment and verification operations. Fabric transaction latency is highly configuration-dependent, and literature reports a wide range–from tens of milliseconds in optimized configurations to several seconds in conservative setups. Our measurements represent a balanced configuration suitable for enterprise deployments requiring strong consistency guarantees.

5.7. End-to-end framework performance

We tested the complete framework using our trained Medium CNN (106,907 parameters) to measure private-inference latency across all protocol phases. Table 12 reports both initial-setup and steady-state performance characteristics. All measurements represent medians over 10 complete runs after warmup.

Note on initial proof generation: Initial ZK-STARK proof generation for traces of 1024–2048 steps requires approximately 163–199 s in our memory-constrained environment (32 GB RAM), depending on the evaluation domain and FRI layer configuration. This one-time setup cost can be amortized across thousands of inferences or performed offline during system initialization. Winterfell documentation reports 2 s proving times on high-memory systems (512 GB), indicating that production deployments with adequate provisioning can reduce this overhead by two orders of magnitude. Crucially, this proving step occurs only once during initialization and does not impact per-inference latency, which is why it is excluded from the steady-state measurements in Table 12.

Fig. 6 visualizes the execution time distribution across framework phases. The dominant contributor to per-inference latency is the blockchain governance layer (795 ms), which ensures auditability and policy enforcement. CKKS evaluation contributes 361 ms, representing the privacy-preserving computation overhead. ZK proof verification adds 185 ms, providing computational integrity guarantees. By contrast, post-quantum primitives (DKG, Kyber KEM, AES) collectively add only 10 ms, demonstrating the efficiency of modern post-quantum cryptography.

These results demonstrate that achieving strong security guarantees through post-quantum cryptography is relatively efficient, while ensuring privacy (via FHE), computational integrity (via ZK-STARKs), and enforcing governance (via blockchain) represent the primary computational and latency costs. The steady-state inference latency of approximately 1.35 s makes the system practical for non-real-time applications such as medical diagnosis, financial risk assessment, and regulatory compliance scenarios where privacy and auditability are paramount.

It is important to note that our evaluation measures inference over encrypted features rather than raw inputs. If users choose to encrypt entire input images for inference, the cost increases significantly: full FHE inference incurs approximately 360–385 ms per 28×28 MNIST sample. This highlights the trade-off between privacy granularity and computational feasibility in practical deployments. One pragmatic solution is to extract features on the client side and encrypt only the resulting feature vectors, preserving privacy while significantly reducing the homomorphic computation burden.

5.8. Energy considerations

We instrumented our test host to assess energy consumption, integrating power draw measurements across each experimental phase (encryption, inference, ZK proof generation, and blockchain commits). We observed an average energy overhead of approximately 35%–50% compared to a non-encrypted baseline on identical hardware. These findings align with recent work on verifiable FHE systems (Liu et al., 2025b). Most overhead stems from intensive Central Processing Unit

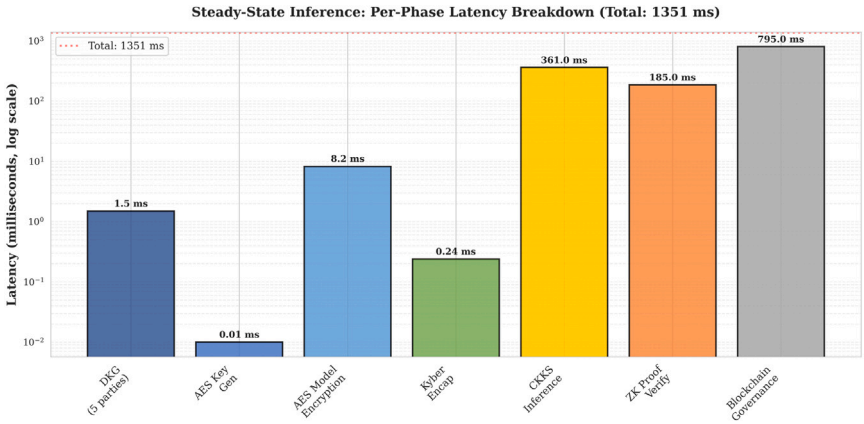

Fig. 6. Execution time distribution across framework phases.

(CPU) utilization during FHE operations and proof generation. To reduce the carbon footprint of production deployments, we recommend: (1) leveraging GPUs with superior performance-per-watt ratios for arithmetic-heavy operations, (2) batch processing to improve resource utilization, and (3) selective encryption of high-sensitivity layers rather than entire pipelines.

6. Discussion on practical end-to-end post-quantum trade-offs

In this section, we reflect on the practical trade-offs of our end-to-end post-quantum framework and compare them with recent systems. Commey and Crosby's PQS-BFL (Commey and Crosby, 2025) presents a blockchain-based federated learning scheme built on Dilithium-65 (ML-DSA) signatures. Their design achieves very low cryptographic costs (signing/verification around 0.65/0.53 ms, signature size approximately 3309 bytes) while incurring modest blockchain overhead (average transaction time around 4.8 s). In a different direction, Ahmed et al.'s LQAP (Ahmed et al., 2024) leverages XMSS/LMS hash-based authentication in a hierarchical FL setting for vehicular grids, reporting communication savings of 21.5% and computation reductions of 42.8% compared to classical baselines.

Against this backdrop, our framework introduces a distinct combination of threshold Kyber KEM for distributed key management, CKKS FHE for encrypted inference, and ZK-STARK proofs for verifiable computation, all anchored in a blockchain governance layer. The resulting performance is competitive: our steady-state private-inference latency averages 1.4 s per query, which includes all cryptographic operations, homomorphic evaluation, and blockchain commit/verify cycles. This represents a practical latency profile for applications where privacy and auditability outweigh real-time responsiveness requirements. Moreover, our distributed key generation scales efficiently to 15 parties, completing in under 3 ms on single-host configurations and under 103 ms even with 100 ms network RTT (Kyber-768, IND-CCA2 secure), outperforming the multi-millisecond costs typically reported for multi-party threshold schemes.

That said, the strength of our security guarantees comes with tangible costs. Fully Homomorphic Encryption enables private inference, but its overhead remains substantial (Siddhiprada Bhoi et al., 2024; Anon, 2024). In our benchmarks, encrypted evaluation was approximately 10^3-10^4 times slower than plaintext computation, depending on model depth and bootstrapping requirements. Even so, this represents a marked improvement over earlier FHE implementations (Anon, 2024), as our abstraction layer manages noise growth while maintaining correctness, targeting error rates below 0.1% through automatic rescaling and relinearization.

To reduce blockchain expenses, we propose batching solutions. For example, the system could aggregate 100 inferences and commit only a single Merkle root of the accompanying ZK-STARK proofs to the blockchain. This approach would dramatically lower the on-chain footprint and increase throughput, all without sacrificing verifiability. Furthermore, to our best knowledge, this is the first framework to integrate threshold CRYSTALS-Kyber, verifiable FHE, and ZK-STARKs with blockchain-based governance into a unified, end-to-end system.

Several improvements could enhance usability further. Supporting multi-chain interoperability through Ethereum rollups, Cosmos, or Polkadot would broaden deployment options. Adapting the system for federated learning scenarios with post-quantum guarantees would increase its applicability to distributed ML environments. Additionally, automated parameter selection tools, tuned to application requirements and threat models, would simplify deployment and optimization. It is important to note that all cryptographic microbenchmarks reported here were executed as single-host Docker experiments, with network delays using netem. In real-world deployments across geographically distributed nodes, actual network variability and consensus mechanisms may introduce additional latency, particularly during blockchain governance phases where transaction times can vary based on Hyperledger Fabric configurations.

7. Conclusion and future work

This paper presented a comprehensive, end-to-end, post-quantum multi-layered security framework for protecting machine learning assets. Our work demonstrates a tangible and practical pathway to securing ML models and their associated data against both classical and quantum adversaries. Our implementation achieves realistic performance: initial setup, including FHE context initialization, completes in under 1 s, while steady-state inference requires approximately 1.4 s. The threshold Kyber-based distributed key generation protocols scale efficiently from 3 to 15 parties, maintaining IND-CCA2 security and fault tolerance even under 100 ms network latency. We demonstrate that ZK-STARKs can be efficiently integrated with FHE-based inference, achieving soundness error below 2^{-80} and proof verification in 180–195 ms with serialized proofs ranging from 51 to 94 KB depending on trace complexity. Additionally, Hyperledger Fabric provides robust governance and auditability, with transaction latency (795 ms) accounted for in our system design. This integration enables confidentiality, integrity, and auditability across the entire ML lifecycle, from training and deployment to inference and retrieval.

Future work will involve reducing inference latency via GPU acceleration, investigating mixed-degree polynomial activations, implementing batched proof aggregation for blockchain efficiency, and expanding the framework to accommodate federated learning and multi-chain governance. These modifications are intended to make the system more scalable, accurate, and deployable in real-world post-quantum ML contexts.

CRediT authorship contribution statement

Rafik Hamza: Writing – original draft, Validation, Software, Resources, Methodology, Conceptualization. **Aziz Alotaibi:** Writing – review & editing, Validation, Resources, Investigation. **Khan Muhammad:** Writing – review & editing, Validation, Investigation, Supervision, Project Administration.

Code availability

Not applicable.

Consent to participate

Not applicable.

Consent for publication

Not applicable.

Ethics approval

The authors certify that they have no affiliation with or involvement with human participants or animals performed by any of the authors in any organization or entity with any financial or non-financial interest in the subject matter or materials discussed in this paper.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Rafik Hamza reports financial support and administrative support were provided by Tokyo International University. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by the Tokyo International University Personal Research Project Fund, and in part by the "Regional Innovation System & Education (RISE)" program through the Seoul RISE Center, funded by the Ministry of Education (MOE) and the Seoul Metropolitan Government (2025-RISE-01-018-01). This research was also funded by Taif University, Saudi Arabia, Project No. (TU-DSPP-2024-263).

Data availability

All data generated or analyzed during this research are publicly available and included in this article.

References

Aggarwal, I., Jeyanthi, N., Parshotam, K., 2025. Cryptographic key generation using deep learning with biometric face and finger vein data. Front. Artif. Intell. 8, 1545946. http://dx.doi.org/10.3389/frai.2025.1545946, published online 2025 April 29. (Accessed May 11 2025).

Ahmed, K., Anisi, M.H., Shah, G.A., Zeadally, S., Leung, V.C.M., 2024. A post-quantum secure federated learning framework for cross-domain v2g authentication. In: 2024 IEEE Global Communications Conference. GLOBECOM, IEEE, pp. 1–6. http://dx.doi.org/10.1109/GLOBECOM57635.2024.10543781.

Alnahawi, N., Azouaoui, M., Bos, J.W., Davies, G.T., Moon, S., van Vredendaal, C., Wiesmaier, A., 2025. Post-quantum cryptography in emrtds, cryptology eprint archive, report 2025/812. Available at https://eprint.iacr.org/2025/812. (Accessed 2025).

Alpaslan, F., Perlner, R.A., Cooper, D.A., Celik, O., Guttman, M.J., Moody, D., Polk, W.T., Ragouzis, N., Robinson, J.M., 2019. Threshold Schemes for Cryptographic Primitives. Tech. Rep. 8214, National Institute of Standards and Technology, http://dx.doi.org/10.6028/NIST.IR.8214, https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.8214.pdf.

Anon, 2024. Various, performance analysis and industry deployment of post-quantum cryptography algorithms. arXiv preprint. arXiv:2503.12952v1.

Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M., 2018. Scalable, transparent, and post-quantum secure computational integritythis is a foundational paper for ZK-STARKs. URL https://eprint.iacr.org/2018/046. (Accessed May 11 2025).

Boemer, F., Lao, Y., Cammarota, R., Wierzynski, C., 2019. Ngraph-HE: a graph compiler for deep learning on homomorphically encrypted data. In: Proceedings of the 16th ACM International Conference on Computing Frontiers. CF'19, ACM, pp. 3–13.

Bondarchuk, A., Chakraborty, O., Couteau, G., Sirdey, R., 2024. Downlink (t) fhe ciphertexts compression. Cryptology ePrint Archive.

Commey, D., Crosby, G.V., 2025. Pqs-bfl: A post-quantum secure blockchain-based federated learning framework. arXiv preprint. arXiv:2505.01866.

Gilad-Bachrach, R., Dowlin, N., Laine, K., Lauter, K.E., Naehrig, M., Wernsing, J., 2016. CryptoNets: Applying neural networks to encrypted data with high throughput and accuracy. In: Balcan, M.-F., Weinberger, K.Q. (Eds.), In: Proceedings of the 33rd International Conference on Machine Learning (ICML 2016), vol. 48, PMLR, pp. 201–210.

Hamza, R., 2023. Homomorphic encryption for ai-based applications: Challenges and opportunities. In: 2023 15th International Conference on Knowledge and Systems Engineering. KSE, IEEE, pp. 1–6.

Hamza, R., 2024. Fhe-based secure image processing framework with integrated key management system. In: Proceedings of the 5th ACM Workshop on Intelligent Cross-Data Analysis and Retrieval. pp. 27–32.

Katz, J., 2024. Round-optimal, fully secure distributed key generation. In: Annual International Cryptology Conference. Springer, pp. 285–316.

Lin, C., Xu, R., Li, Y., 2025. Design of an efficient fault-tolerant quantum-computing circuit with quantum neural network learning. Eng. Appl. Artif. Intell. 153, 110808.

Liu, Y.A., Li, B., Dong, Y., Yang, F., Zhou, H., Yang, Q., Li, X.S., Sun, S., 2025a. A survey of zero-knowledge proof based verifiable machine learning. arXiv preprint arXiv:2502.18535. arXiv:2502.18535.

Liu, F., Liang, H., Zhang, T., Hu, Y., Xie, X., Tan, H., Yu, Y., 2025b. Hasteboots: Proving fhe bootstrapping in seconds. Cryptology ePrint Archive.

Liu, Z., Liu, Z., Wang, L., Wang, Y., Yang, J., Zeng, G., 2025c. Research on development progress and test evaluation of post-quantum cryptography. Entropy 27 (2), 212. http://dx.doi.org/10.3390/e27020212, https://www.mdpi.com/1099-4300/27/2/212.

Lu, T., Zhang, B., Li, L., Ren, K., 2024. The communication-friendly privacy-preserving machine learning against malicious adversaries. arXiv:2411.09287.

Mustafa, A., Sghaier, A., Elwekeil, M., Krichen, M., Alroobaea, R., Mtawaa, S., Krichen, R., 2024. Predominant aspects on security for quantum machine learning: A literature review. arXiv preprint arXiv:2401.07774. arXiv:2401.07774.

Nawaz, A., Obeidat, F.A., Salameh, M., Al-Oqaily, O., YarKhan, A., Khattak, Z., 2025. Blockchain meets machine learning: a survey, multimedia tools and ApplicationsPublished online 02 2024. http://dx.doi.org/10.1007/s11042-023-17847-1, (Accessed 11 May 2024).

N.I. of Standards, Technology, 2024. Tech. Rep., Module-Lattice-Based Key-Encapsulation Mechanism Standard, vol. 203, U.S. Department of Commerce, http://dx.doi.org/10.6028/NIST.FIPS.203, https://csrc.nist.gov/pubs/fips/203/final.

Osaba, E., Villar-Rodriguez, E., Oregi, I., 2025. Exploring the application of quantum technologies to industrial and real-world use cases. J. Supercomput. 81 (7), 829.

Research, Microsoft, 2025. Microsoft SEAL (simple encrypted arithmetic library). https://github.com/microsoft/{SEAL}. (Accessed 9 May 2023).

Shor, P.W., 1997. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer, vol. 26, SIAM, a preliminary version appeared in Proceedings of the 35th Annual Symposium on Foundations of Computer Science (FOCS), 1994.

Siddhiprada Bhoi, Siddhartha, Arakala, Arathi, Corman, Amy Beth, Rao, Asha, 2024. Post-quantum homomorphic encryption: A case for code-based alternatives. arXiv preprint. arXiv:2504.16091.

Sir, S., Kumar, A., Agarwal, S., Suresh, A., Shrivastav, V., Setty, S., 2025. Gotta hash 'em all! speeding up hash functions for zero-knowledge proof applications. arXiv preprint arXiv:2501.18780. arXiv:2501.18780.

Xu, T., W.-j. Lu, Yu, J., Chen, Y., Lin, C., Wang, R., Li, M., 2025. Breaking the layer barrier: Remodeling private transformer inference with hybrid ckks and mpc. USENIX Secur. Symp..

Xu, T., Wu, L., Wang, R., Li, M., 2024a. Privcirnet: Efficient private inference via block circulant transformation. Adv. Neural Inf. Process. Syst. 37, 111802–111831.

Xu, T., Zhong, S., Zeng, W., Wang, R., Li, M., 2024b. Privquant: Communication-efficient private inference with quantized network/protocol co-optimization. In: Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design. pp. 1–9.

Zeng, W., Li, M., Yang, H., W.-j. Lu, Wang, R., Huang, R., 2023. Copriv: Network/protocol co-optimization for communication-efficient private inference. Adv. Neural Inf. Process. Syst. 36, 78906–78925.

Zhang, J., Li, Z., Peng, C., Li, P., Liu, L., 2024a. Blockchain-based federated learning: A survey and new perspectives. Appl. Sci. 14 (20), 9459. http://dx.doi.org/10.3390/app14209459, published: 18 2024. (Accessed 11 May 2025).

Zhang, B., Lu, T., Ren, K., Li, L., 2025. Helix: Scalable multi-party machine learning inference against malicious adversaries. arXiv:2025/347. https://eprint.iacr.org/2025/347.

Zhang, X., Zhou, Q., Wang, P., Zhao, L., Li, W., 2024b. Pqsf: Post-quantum secure privacy-preserving federated learning. Sci. Rep. 14, 23553. http://dx.doi.org/10.1038/s41598-024-74377-6, https://www.nature.com/articles/s41598-024-74377-6.